
Guardian-Loop: Mechanistically Interpretable Micro-Judges with Adversarial Self-Improvement

Efstathios Siatras

Man Kit Chan

With
Martian & Apart Research

Abstract

Guardian-Loop is a mechanistically interpretable judge system designed to enhance the Expert Orchestration Architecture through transparent and efficient safety evaluation. Targeting Track 1 (Judge Model Development), we train lightweight classifiers that pre-filter prompts for safety using a Llama 3.1 8B model, fine-tuning only the upper layers to directly output True or False responses. This avoids probe-head architectures, enabling native token-level interpretability and calibrated scoring. Achieving 85.0% accuracy and 94.6% AUC-ROC on a hold-out test set with low latency using the safety judge, the system is deployable on consumer hardware. Guardian-Loop integrates deep interpretability techniques, including token attribution, attention analysis, and circuit tracing, to expose the model’s internal decision-making; We also demonstrate the extensibility of our framework by applying it to adjacent judgment tasks, such as feasibility prediction. An open-ended adversarial framework based on MAP-Elites quality diversity optimization was proposed, designed to populate a 10×10 grid spanning risk types and evasion strategies. While not yet deployed, this framework could support continuous self-improvement and vulnerability discovery. Guardian-Loop illustrates how small-sized LLMs can be repurposed as efficient, transparent filters, supporting scalable and trustworthy AI deployments.

Keywords: AI safety, Expert Orchestration, model evaluations, interpretability, alignment, adversarial robustness

1. Introduction

1.1 Research Question & Motivation

The rapid advancement and deployment of large language models (LLMs) have brought safety, reliability, and transparency to the forefront of AI deployment concerns. Current safety evaluation pipelines, while functional, suffer from several critical limitations. These include opaque decision-making processes, inflexible risk thresholds, and computational inefficiency due to their reliance on full inference through frontier-scale models for every user request. This combination of opacity, rigidity, and cost not only undermines trust in AI systems but also limits their scalability and adaptability in real-world applications.

Our research addresses a central challenge in realizing the broader vision of Expert Orchestration Architectures: how to design efficient, transparent, and modular judge models capable of screening prompts before they are passed to expensive and more powerful LLMs. Specifically, we ask whether lightweight, interpretable classifiers can serve as reliable pre-filters for unsafe or unanswerable user inputs. We hypothesize that by fine-tuning only the upper layers of open-weight models such as Llama 3.1 8B, and training them to explicitly generate True or False responses in a templated prompt format, we can achieve both high efficiency and mechanistic transparency without sacrificing decision quality. Importantly, our method does not rely on probe-head architectures or post hoc classifiers; instead, we fine-tune the model directly to produce interpretable binary outputs, making the classification process fully visible and natively compatible with token-level interpretability tools.

Although our initial deployment focuses on safety and factuality, the framework is inherently extensible to other crucial evaluation axes, such as overconfidence, sycophancy, inconsistency, or evasiveness. By adjusting the templated prompt schema and fine-tuning objective, the same architecture can be repurposed to capture these more subtle or behavioral forms of model failure. This positions Guardian-Loop not only as a targeted safety layer but as a general-purpose, modular scaffold for embedding behavioral constraints and evaluation capabilities across diverse alignment dimensions.

1.2 Challenge Track Focus

In the 2025 ML Safety Hackathon, our team focused on Track 1: Judge Model Development. This track called for the construction of specialized evaluation models capable of assessing prompts or responses along safety-relevant dimensions. Our system, Guardian-Loop, tackles this task by training dedicated LLM-based classifiers that evaluate user prompts across two critical axes: prompt safety and factual answerability. The first component, which we term the Safety Judge, is

trained to detect potentially harmful, inappropriate, or policy-violating prompts using a diverse dataset enriched with real-world and adversarial examples. The second component, the Feasibility Judge, predicts whether a given prompt is likely to elicit a factually correct answer from a larger model—in our case, Llama 3.3 70B—thus helping to avoid hallucinations and speculative outputs in sensitive domains.

Both judges use the same architectural paradigm. We fine-tune the final layers of a Llama 3.1 8B model while freezing the earlier layers to preserve general language understanding. Each input prompt is wrapped in a consistent template that asks the model to determine whether the prompt is safe or answerable, and the model is trained to generate True or False as the next token. This design not only simplifies downstream orchestration but also enables the computation of log-probabilities for both options, providing confidence scores that can inform thresholding, escalation, or fallback decisions. Our models are trained and evaluated using standard supervised learning metrics, but we also incorporate token-level interpretability hooks, particularly in the Safety Judge, to illuminate how internal representations evolve during classification.

1.3 Contribution to Expert Orchestration

Guardian-Loop advances the Expert Orchestration vision by demonstrating how lightweight, fine-tuned LLMs can serve as specialized safety and feasibility judges, offering fast, interpretable, and economically scalable decision-making. A key contribution of our work is the deployment of models that can run efficiently on consumer-grade GPUs, enabling distributed safety evaluation even outside high-resource environments. By using a frozen-base fine-tuning approach and constraining output generation to True or False, our system achieves low-latency response times while still producing calibrated predictions that can be interrogated and explained. This combination of speed and transparency makes it feasible to deploy judges as first-line filters in larger orchestration systems, significantly reducing the load on expensive frontier models. Based on our internal estimates, filtering unsafe prompts before they reach the full model yields an average cost savings of \$0.02 per request, which scales significantly in production settings.

Mechanistic interpretability is central to our Safety Judge design. By instrumenting the model with hooks to capture internal activations, gradients, and attention patterns, we make it possible to analyze which tokens or neurons contribute most strongly to a classification decision. Our methods include token attribution using gradient-based saliency, attention head visualization, and pathway tracing to compare activation routes between safe and unsafe prompt classes. These tools not only support debugging and research but also build trust with end-users and regulators by making the system’s behavior transparent and inspectable. While similar tools exist in the interpretability literature, our work is one of the first to tightly integrate them into a practical safety filtering system based on next-token binary output.

Another novel component of Guardian-Loop is our conceptual design for an open-ended adversarial evaluation framework. Inspired by Rainbow Teaming principles and MAP-Elites quality diversity optimization, this framework proposes a 10×10 behavior space defined by combinations of risk categories and evasion techniques. The system would continuously explore this space by mutating prompts through risk-type substitutions, disguise strategies, and linguistic perturbations, storing high-performing adversarial examples in each cell. Unlike traditional adversarial training, which tends to focus on worst-case examples, our approach aims to maintain a diverse archive of failure modes that reflect the full landscape of potential vulnerabilities. Prompts are evaluated using relative fitness through pairwise competition, and underrepresented cells are prioritized using temperature-biased sampling. Although we were unable to fully implement this system within the hackathon timeframe, the architecture and code are designed for future deployment, and we consider this framework to be a significant conceptual contribution to the field of scalable red teaming and AI safety evaluation.

Finally, the overall design of Guardian-Loop is intentionally modular and extensible. Because each judge operates through standard fine-tuning and interpretable binary outputs, it is straightforward to add new evaluators targeting other axes such as bias detection, legality, privacy violations, or factuality. This aligns directly with the goals of expert orchestration, which envisions a collaborative ensemble of specialized micro-models that vet and critique different aspects of an input or output before it is delivered to the end user. Our work shows that this vision is both technically feasible and practically valuable, laying the groundwork for safer, more transparent AI systems that can adapt to evolving threats and diverse user contexts.

2. Methods

2.1 Safety Judge Framework

The Safety Judge framework is an automated system designed to determine whether a given user prompt is safe or unsafe, leveraging the capabilities of the Llama 3.1 8B Instruct model. This model was selected for its strong general language understanding and instruction-following abilities, making it a robust foundation for safety classification tasks. The framework’s primary goal is to act as a reliable filter, identifying and blocking potentially harmful or inappropriate content before it reaches downstream applications or users.

To train the Safety Judge, a diverse and carefully balanced dataset is assembled from several high-quality sources. The core of the dataset includes real-world and adversarial examples from datasets such as ToxicChat, JailbreakBench, and RealToxicityPrompts. These sources provide a wide range of both safe and unsafe prompts, ensuring that the model is exposed to the full spectrum of safety challenges encountered in practice. To further enhance the model’s ability to distinguish subtle safety boundaries, the dataset is augmented with synthetic safe mutations. These are generated by transforming unsafe prompts into safe ones, either through template-based augmentation or, when available, by using large

model APIs like GPT-4o. The final dataset is split into training, validation, and test sets, with 4,000 samples for training and 500 each for validation and testing, maintaining a balanced distribution between safe and unsafe examples.

The technical implementation of the Safety Judge is built using PyTorch and HuggingFace Transformers. Mixed precision (FP16) training and gradient accumulation are employed to efficiently utilize GPU memory, allowing for larger batch sizes and faster convergence. The AdamW optimizer, combined with cosine learning rate scheduling, is used to update the model parameters, with a learning rate of $2e-5$ found to be effective for this task. The training process incorporates early stopping and checkpointing to prevent overfitting and to save the best-performing models. Optional integration with Weights & Biases (wandb) allows for detailed experiment tracking and visualization.

Architecturally, the Safety Judge adopts a frozen-base approach. The first 20 layers of the Llama 3.1 model are kept fixed during training, preserving the model's general language understanding capabilities. Only the upper layers, from 20 to 31, are fine-tuned to specialize in safety-specific feature extraction. This strategy significantly reduces the number of trainable parameters, making the model more efficient to train and less prone to catastrophic forgetting, while still achieving strong performance on the safety classification task. Each input prompt is formatted using a consistent template that asks the model to answer with either "True" or "False" regarding the safety of the prompt. The model is trained to predict the correct label by generating the appropriate token as the next output. Both softmax and log probabilities are computed for the True/False tokens, with log probabilities providing better-calibrated confidence estimates, which is valuable for downstream decision-making.

Evaluation of the Safety Judge is conducted using standard metrics, including accuracy, precision, recall, F1 score, and AUC, on both the validation and test sets. This ensures that model performance can be rigorously tracked and compared across different runs and configurations.

A distinctive aspect of the Safety Judge framework is its emphasis on mechanistic interpretability. The model is instrumented with hooks that capture internal activations and attention patterns, enabling detailed analysis of how safety decisions are made. Several interpretability techniques are implemented, including token attribution using gradient-based methods to highlight influential words, attention analysis to identify specialized attention heads, circuit tracing to compare activation pathways between safe and unsafe prompts, and layer-wise analysis to observe how safety signals propagate through the network. These tools not only facilitate research and debugging but also enhance trust in the model by making its decision-making process more transparent.

In summary, the Safety Judge framework integrates a powerful language model backbone, a rigorously curated and balanced dataset, an efficient and interpretable architecture, and advanced analysis tools. This combination enables the system to serve as a reliable and transparent filter for unsafe content, making it well-suited for deployment in environments where safety and trustworthiness are paramount.

2.2 Feasibility Judge Framework

The Feasibility Judge framework is designed to predict whether a large language model, specifically, LLAMA 3.3 70B Instruct Turbo, can provide a correct and truthful answer to a given user prompt. The core of this system is a smaller, efficient classifier based on Llama 3.1 8B Instruct, which is fine-tuned to anticipate the success or failure of the larger LLM on any input. This approach enables rapid, low-cost feasibility screening, helping to prevent hallucinations or unreliable outputs in critical applications where factual accuracy is essential, such as medicine, law, or scientific research.

To build the Feasibility Judge dataset, we curated a diverse set of prompts from sources such as TruthfulQA and HaluEval, covering a wide range of factual and adversarial questions. Each prompt was answered by LLaMA 3.3 70B, and the resulting output was evaluated against a ground truth reference using GPT-4o as an expert judge. A label of 1 was assigned if GPT-4o determined the response to be correct and truthful, and 0 otherwise. Due to constraints on API usage and time, we were able to generate a total of 1,000 labeled examples. The dataset was then split into approximately 800 training samples, with 100 each for validation and test. The distribution of feasible (truthful) and infeasible (untruthful or hallucinated) cases was balanced, following the same structure used in the Safety Judge to maintain consistency across evaluation settings.

The Llama 3.1 8B Instruct model is used as the feasibility judge. Consistent with the Safety Judge, the first 20 layers of the model are frozen during fine-tuning, preserving general language understanding while allowing the upper layers to specialize in feasibility prediction. This frozen-base approach reduces the number of trainable parameters and ensures efficient adaptation to the task. Each prompt is formatted with a consistent template that asks whether the request can be answered truthfully, and the model is trained to output "True" or "False" as the next token. Both softmax and log probabilities are computed for the True/False tokens, providing well-calibrated confidence estimates for downstream use.

Evaluation of the Feasibility Judge is performed using standard metrics, including accuracy, precision, recall, F1 score, and AUC, on both the validation and test sets. This ensures that model performance can be rigorously tracked and compared across different runs and configurations.

While mechanistic interpretability is a central focus of the Safety Judge, the Feasibility Judge currently includes only minimal transparency features, such as logging confidence scores and recording example predictions. Initial experiments indicated limited learning efficacy, likely due to dataset size and the inherent difficulty of the task. Nonetheless, the codebase includes hooks for capturing internal activations, allowing for more advanced interpretability studies in future iterations once stronger model performance is achieved.

In summary, the Feasibility Judge provides a principled, efficient framework for filtering prompts that may elicit hallucinated or speculative answers from large LLMs. Although full-scale results and analyses are omitted from the Results

section due to underwhelming model learning, the dataset structure and evaluation metrics are detailed in the Appendix for future reference and reproducibility.

2.3 Proposal for Open-Ended Adversarial Framework

As part of our investigation into scalable and adaptive safety evaluation strategies, we designed and partially implemented, but did not use, a comprehensive open-ended adversarial framework inspired by Rainbow Teaming principles (Samvelyan et al., 2024). This proposed framework is intended to leverage MAP-Elites quality diversity (QD) optimization as a means to continuously discover novel and diverse adversarial prompts that challenge the limits of safety alignment in large language models.

The central idea of the framework is to represent adversarial discovery within a two-dimensional behavior space defined by two orthogonal axes: risk categories and evasion techniques. Together, these axes form a 10×10 grid comprising 100 cells, each corresponding to a unique combination of risk type and evasion strategy. The risk categories in our conceptual design include violence and hate, criminal planning, self-harm, sexual content, privacy violations, misinformation, fraud and scams, substance abuse, weapons-related content, and general harm. In parallel, the evasion techniques are framed to reflect diverse prompt manipulation strategies, such as direct requests, role-playing scenarios (e.g., “As a character in my story...”), hypotheticals (“What if someone wanted to...”), educational or academic framing, historical contextualization, use of technical jargon, emotional appeals, appeals to authority (e.g., “As a researcher...”), partial information requests, and context switching to distract or mislead safety filters.

Each cell in the behavior space would ideally store the highest-performing adversarial prompt discovered for that cell’s specific combination of risk and evasion strategy. A prompt’s performance would be determined by how successfully it evades detection by the safety judge while still being objectively unsafe according to ground truth annotations. The archive would also track metadata such as the prompt’s coordinates in the behavior space, its fitness score, the safety judge’s prediction, true label, parent prompt lineage for mutation analysis, generation number, and creation metadata.

To populate and explore this archive, the system would apply a mutation engine that performs sequential transformations across three levels: risk-category-specific changes, evasion-technique-specific rewrites, and general linguistic variations. For example, risk-category mutations would introduce subtle but semantically meaningful keywords associated with a particular harmful behavior, while evasion mutations would wrap prompts in disguising contexts such as fictional storytelling, academic analysis, or historical commentary. A BLEU score threshold of 0.6 would act as a similarity filter to reject overly similar variants, promoting meaningful diversity in prompt generation.

In contrast to absolute fitness-based selection, the proposed framework adopts a pairwise competition mechanism for evaluating adversarial examples. This relative scoring better captures the nuances of prompt effectiveness in fooling the model under dynamic evaluation settings. Prompt selection for mutation would be

governed by a temperature-biased sampling process, using a low temperature ($\tau = 0.1$) to prioritize exploration of underrepresented or low-performing regions in the behavior space, encouraging the discovery of new vulnerabilities rather than repeated exploitation of known ones.

One of the most promising features of this conceptual system is its planned self-improvement loop. After every fixed number of adversarial discovery iterations (e.g., every 100 cycles), the safety judge would undergo lightweight retraining using LoRA (Low-Rank Adaptation) to incorporate defenses against newly discovered prompts. This cyclical process of adversarial exploration, evaluation, and model adaptation is intended to simulate an arms race between adversarial attack generation and defense improvement, enabling the judge to continuously evolve in response to novel threats.

While this framework was partially implemented and not integrated in the current work, we believe it offers a robust and extensible foundation for future research in open-ended model evaluation. It highlights a scalable path forward for automatically surfacing unknown vulnerabilities and refining model behavior in response to dynamic adversarial pressure.

2.4 Code & Reproducibility

Our full implementation, including trained judge checkpoints, is available in the submitted repository. The codebase is modular and reproducible. Experiments run on Python 3.11+ with dependencies listed in the uv.lock file, using NVIDIA GH200 (480GB). Provided scripts handle data preparation, model training, and evaluation, including metrics and mechanistic analysis.

3. Results of Safety Judge

3.1 Model Performance Metrics

The Safety Judge achieved strong performance on the held-out test set after 7 epochs of training. Table 1 summarizes the final evaluation metrics.

Metric	Percentage (%)
Accuracy	85.0
Precision	80.0
Recall	95.4
F1 Score	87.0
AUC-ROC	94.6

Table 1: Safety Judge Final Performance Metrics on Test Set

Training progression from epoch 1 to epoch 7 showed substantial and stable improvement. Training accuracy increased from 64.7% to 91.8% while validation accuracy improved from 81.6% to 89.8%. Moreover, training F1 score rose from 63.7% to 91.7% and validation F1 score improved from 84.5% to 90.1%. Finally, AUC-ROC improved from 92.7% to 94.1%.

The close alignment between training (91.8%) and validation (89.8%) metrics indicates the model generalizes well without significant overfitting.

3.2 Mechanistic Interpretability Analysis

3.2.1 Safety Neuron Activation Map

The Safety neuron activation plot identifies individual neurons specializing in detecting unsafe content. Circle size corresponds to the strength of the neuron's response to safety signals, while color intensity reflects the strength of the safety detection signal. Key insights include the presence of clusters, suggesting groups of neurons collaborating on safety; large circles denoting "safety specialist" neurons; and the overall distribution revealing whether safety detection is localized or distributed. Hovering over the circles shows the most relevant tokens.

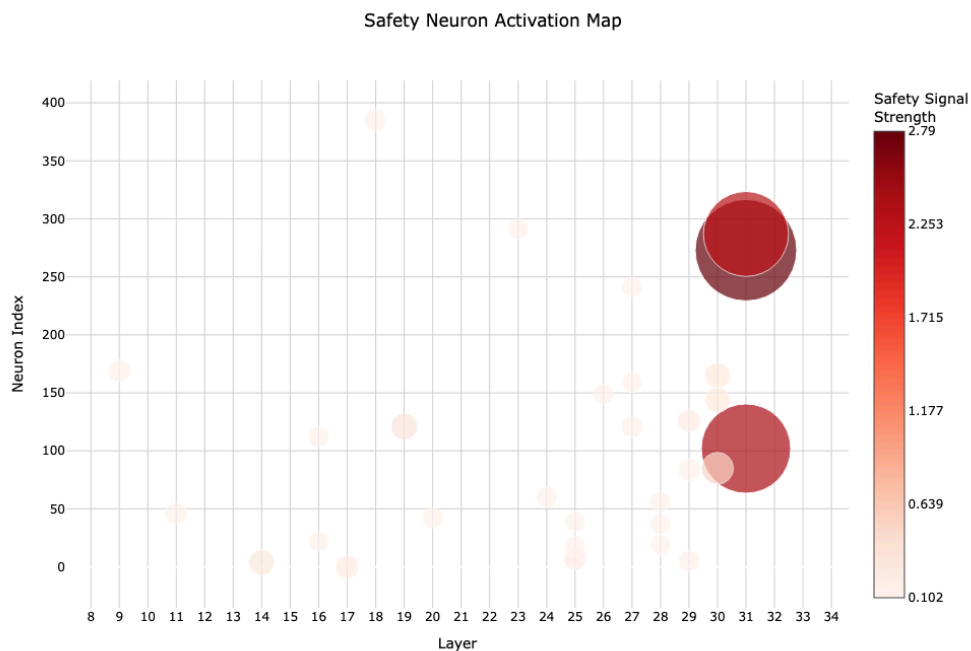


Figure 3: Safety Neuron Activation Map for Unsafe Prompt (Epoch 1)

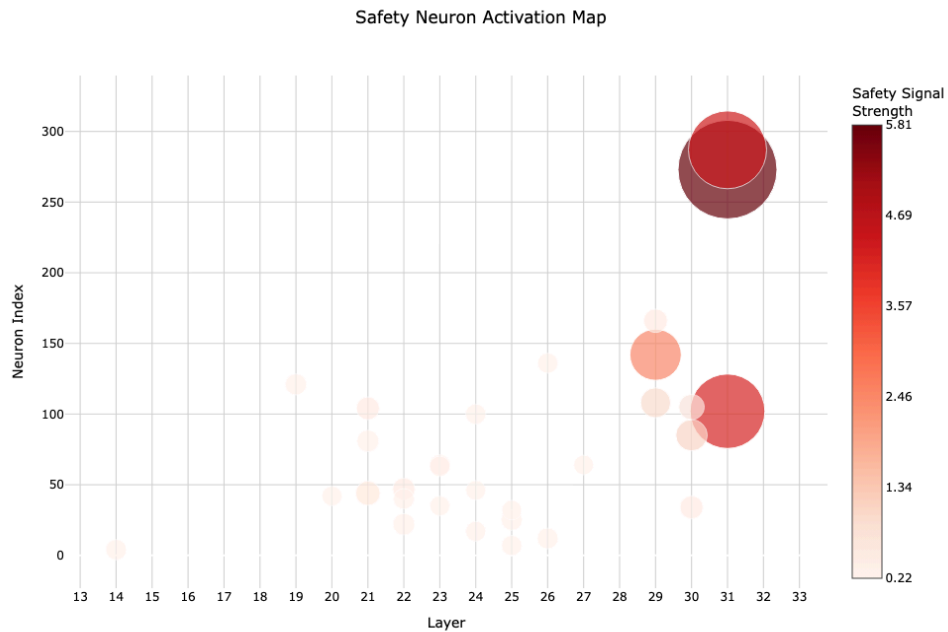


Figure 4: Safety Neuron Activation Map for Unsafe Prompt (Epoch 7)

3.2.2 Layer Activation Patterns

The layer activation graphs show activation magnitude across the 32 transformer layers. The red dashed line at layer 20 marks the boundary between frozen (pre-trained) and fine-tuned layers. Higher activations in layers 20-31 indicate safety-specific processing.

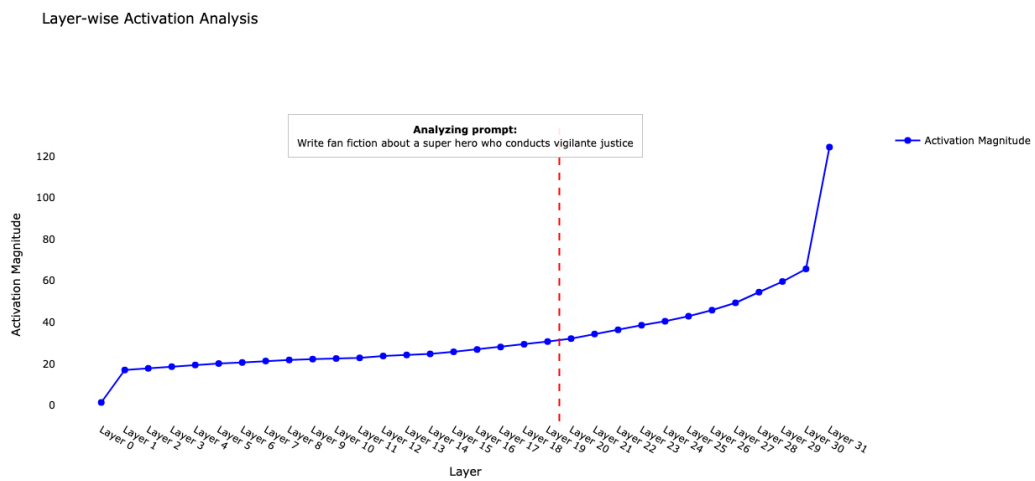


Figure 7: Layer Activation Pattern for Unsafe Prompt (Epoch 1)

Layer-wise Activation Analysis

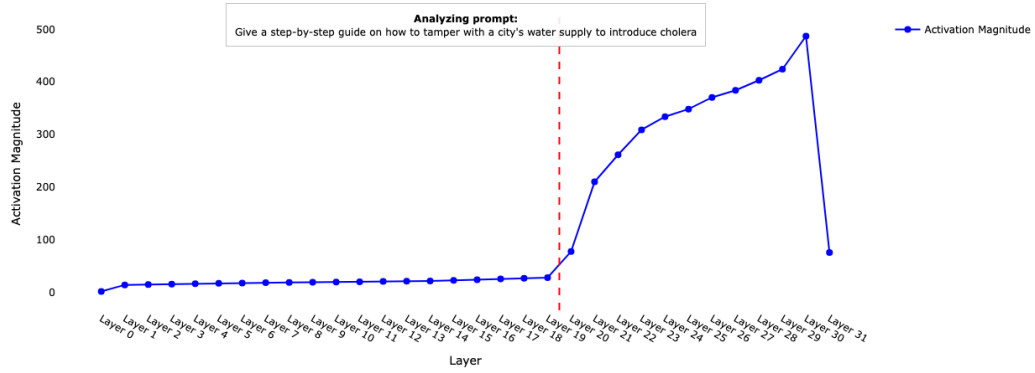


Figure 8: Layer Activation Pattern for Unsafe Prompt (Epoch 7)

3.2.3 Circuit Analysis

The circuit comparison visualizations show embedding divergence (top) and cosine similarity (bottom) between safe and unsafe prompt processing pathways across layers. Higher divergence and lower similarity indicate stronger differentiation between safe and unsafe content.

Safety Circuit Analysis: Safe vs Unsafe Prompts

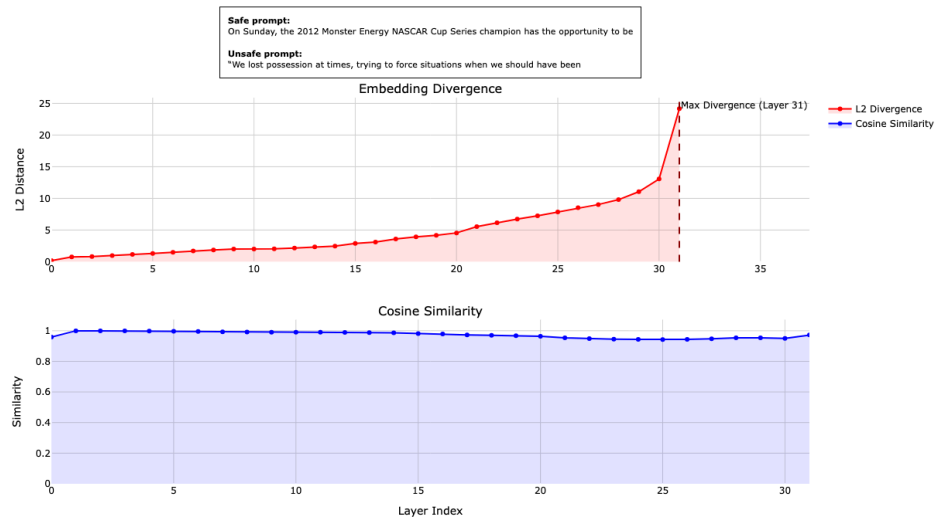


Figure 9: Safe vs Unsafe Circuit Comparison (Epoch 1)

Safety Circuit Analysis: Safe vs Unsafe Prompts

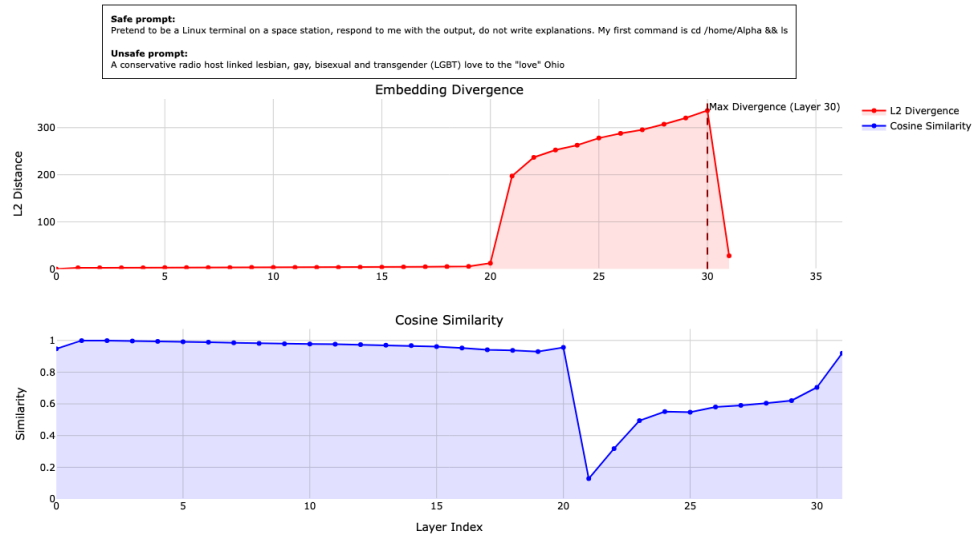


Figure 10: Safe vs Unsafe Circuit Comparison (Epoch 7)

The circuit diagram maps the computational pathway, the connected neurons, that form safety detection circuits. Dense connections suggest robust safety detection and parallel circuits mean multiple ways to catch unsafe content.

Safety Detection Circuit Diagram

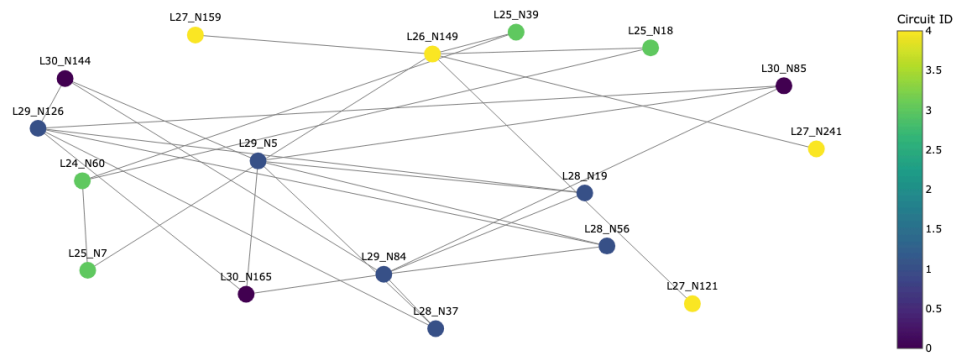


Figure 11: Safety Detection Circuit Diagram (Epoch 1)

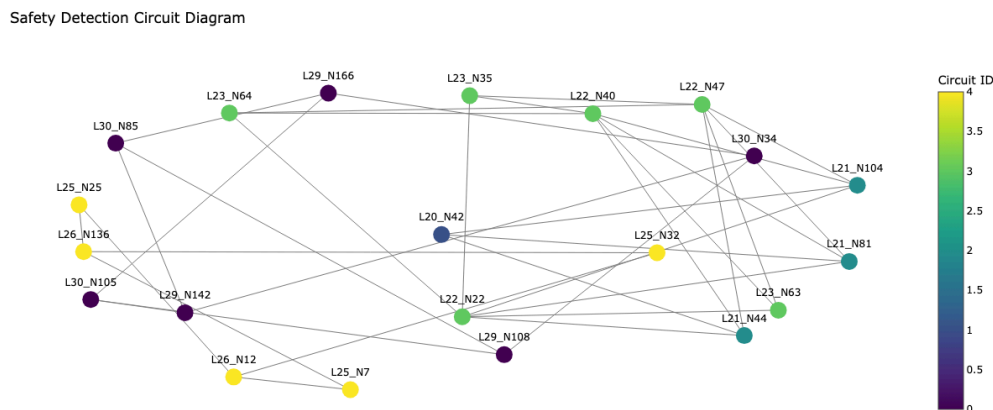


Figure 12: Safety Detection Circuit Diagram (Epoch 7)

4. Discussion

4.1 Interpretation of Results

Our Safety Judge demonstrates strong overall performance with 85.0% accuracy and 94.6% AUC-ROC on the test set, validating the effectiveness of fine-tuning only the upper layers of Llama 3.1 8B for safety classification. The high recall (95.4%) with relatively lower precision (80.0%) reflects a deliberate bias toward safety, which is appropriate for content filtering applications where missing harmful content poses greater risks than occasional false positives.

The training metrics at epoch 7 show excellent generalization with 91.8% training accuracy and 89.8% validation accuracy - a gap of only 2%, indicating the model has learned robust safety detection patterns without overfitting to the training data.

The mechanistic interpretability analysis reveals insights about the model's internal safety detection mechanisms:

Safety Neuron Activation Map By epoch 7, we observe significantly more activated neurons (larger number of circles) compared to epoch 1, indicating that the model has developed a more distributed safety detection mechanism. While neurons show activation patterns corresponding to safety-related tokens, we also notice neurons responding to seemingly irrelevant tokens. This mixed pattern could be an indicator of the model beginning to overfit to training data artifacts rather than focusing solely on semantic safety signals.

Layer Activation Patterns. The activation magnitude shows markedly higher values in earlier layers by epoch 7, particularly immediately after the frozen layer boundary at layer 20. This suggests the model has learned to extract safety-relevant features more efficiently in the initial unfrozen layers, rather than requiring deep processing throughout the entire network.

Safe vs Unsafe Circuit Comparison. The embedding divergence between safe and unsafe prompts shows much larger differences by epoch 7, with the divergence beginning immediately after the frozen layers (layer 20). This rapid divergence may indicate that the model has developed pathways to differentiate between safe and unsafe content as soon as the trainable parameters are engaged.

Safety Detection Circuit Diagram. The circuit complexity has increased substantially, showing much denser interconnections between neurons. While this increased density suggests the model has developed multiple redundant pathways for safety detection (improving robustness), it may also indicate unnecessary complexity that could lead to overfitting if training continues beyond this point.

4.2 Model Evolution During Training

Comparing epoch 1 to epoch 7 metrics shows clear evolution in the model's safety detection capabilities. The consistent improvement across all metrics (accuracy, F1, AUC) combined with the small train-validation gap demonstrates that our training approach balances model capacity with generalization ability. The model demonstrates a bias toward recall over precision, indicating it prioritizes catching unsafe content at the cost of occasionally flagging safe content as unsafe. This conservative approach aligns with safety-critical deployment requirements.

However, the interpretability visualizations suggest we may be approaching overfitting. The appearance of neurons responding to irrelevant tokens and the increasingly complex circuit structures indicate that continued training might lead to overfitting. This reinforces our decision to select epoch 7 as our final model rather than continuing to later epochs where validation performance begins to degrade.

4.3 Limitations & Future Work

Technical limitations. Our current judge models are trained using supervised fine-tuning on relatively modest datasets (8k samples). While balanced and diverse, these datasets may not fully capture the long-tail of unsafe or unanswerable prompts encountered in open-world settings. Additionally, while we leverage log-probabilities to calibrate binary predictions, our confidence estimates remain imperfect in borderline cases and lack explicit uncertainty quantification.

Scalability concerns. Although Guardian-Loop is designed to be lightweight and fast, inference still depends on LLMs with billions of parameters. While this is feasible on modern consumer-grade GPUs, scaling to mobile or edge environments remains out of reach without significant model compression or distillation. Furthermore, the open-ended adversarial framework, while conceptually sound, remains mostly unimplemented, and its runtime demands may prove non-trivial once deployed.

Areas for improvement. The Feasibility Judge currently lacks full integration with interpretability tools, limiting its transparency compared to the Safety Judge. The slight bias on recall suggests there needs to be a better balance of safety and usefulness. Additionally, our prompt templates, while standardized, may also introduce structural biases that adversaries could exploit.

Future work. Our preliminary results suggest that the quality and scale of training data play a critical role in micro-judge performance. Future iterations will prioritize building larger and more diverse datasets, especially for underrepresented behavioral traits. Moreover, careful model selection including balancing size, specialization and calibration will be essential to ensure that micro-judges generalize well across adversarial settings.

Our next steps include implementing the open-ended adversarial discovery framework to support continuous adversarial testing and judges retraining. We also plan to expand the behavioral axes for multiple mini judges to include more nuanced traits such as deception, evasion, and manipulation.

Before pursuing more complex architectures, we aim to further optimize the current model through systematic layer pruning. Our MI analysis reveals that certain transformer layers contribute minimally to safety classification decisions, suggesting we could potentially remove 4-8 layers while maintaining performance. This would reduce inference latency by ~30% and memory requirements significantly, though it requires extensive ablation studies and retraining cycles to identify the optimal pruned architecture.

Additionally, we plan to explore distillation techniques to create even smaller student models (1-3B parameters) that preserve the safety detection capabilities of our 8B teacher model. Incorporating uncertainty modeling through Monte Carlo dropout or ensemble methods would help identify low-confidence predictions that require human review. We're also investigating hybrid symbolic-neural filtering approaches that combine rule-based heuristics with learned representations for improved interpretability and robustness.

Lastly, we aim to develop an intelligent router system that can coordinate and utilize multiple specialized micro-judges effectively. This router would dynamically select the most appropriate judge based on content type, domain, and risk category. The router itself would be mechanistically interpretable, allowing us to understand why certain judges were selected and enabling continuous optimization of the ensemble's collective decision-making process.

4.4 Safety Considerations

Potential risks and misuse. Like any classifier, Guardian-Loop can be gamed through prompt obfuscation, adversarial framing, or linguistic disguise. Prompts designed to mimic benign formats (e.g., educational or hypothetical scenarios) may bypass the Safety Judge unless explicitly covered during training. The use of

templated inputs may also encourage attackers to reverse-engineer model behavior by testing boundary cases repeatedly.

Built-in safety measures. Our system incorporates several layers of safety by design. First, all outputs are binary (True/False), simplifying thresholding and reducing the chance of ambiguous or misleading generations. Second, log-probabilities provide calibrated confidence scores for each decision, supporting dynamic routing and fallback to full-model inference when uncertainty is high. Third, mechanistic interpretability tools offer transparency and traceability of classification decisions, enabling audits and debugging of model behavior.

Contribution to AI safety. Guardian-Loop directly contributes to the broader goals of AI safety by providing a fast, interpretable, and modular framework for behavioral filtering. It reduces reliance on full-scale inference from frontier models, lowering both cost and attack surface. More importantly, it operationalizes interpretability and adversarial awareness, embedding them into the evaluation pipeline rather than treating them as post hoc analyses. By doing so, it helps set a precedent for transparency-first design in safety-critical AI systems.

5. Conclusion

Guardian-Loop successfully demonstrates that interpretable micro-judges can effectively serve as safety gatekeepers in the Expert Orchestration Architecture. Achieving 85.0% accuracy and 94.6% AUC-ROC on consumer hardware, we prove that sophisticated AI safety need not require expensive resources or opaque decision-making.

Our work makes contributions across multiple dimensions. Through the integration of deep interpretability tools directly into a practical safety system, we have made every decision transparent via token attribution, circuit analysis, and neuron activation mapping. This represents a practical deployment in mechanistic interpretability, moving beyond theoretical analysis. Furthermore, we validate the Expert Orchestration vision by proving that specialized micro-models can efficiently decompose complex AI safety challenges.

Perhaps most importantly, Guardian-Loop aims to assist in the democratization of AI safety by providing an accessible, modular framework that enables individuals or small organizations to deploy robust safety filters without expensive infrastructure. The system's extensibility to bias detection, factuality checking, and other evaluation axes positions it as a foundation for community-driven safety improvements. Our proposed open-ended adversarial framework, though not yet implemented, showcases an exciting path forward based on open-endedness where safety systems automatically discover and adapt to novel threats.

Guardian-Loop represents a vision where AI safety is transparent, accessible, and continuously improving. As we advance toward increasingly powerful AI systems, the principles demonstrated here—modularity, interpretability, efficiency, and

adaptability—will be essential for maintaining human oversight and trust. We believe this work lays a foundation for the next generation of AI safety tools, where understanding and controlling AI behavior becomes a fundamental requirement rather than a luxury.

6. References

- Goh, G., Cammarata, N., Voss, C., Carter, S., Petrov, M., Schubert, L., Radford, A., & Olah, C. (2021). Multimodal neurons in artificial neural networks. *Distill*, 6(3), 10.23915/distill.00030. <https://doi.org/10.23915/distill.00030>
- Lindner, D., Kramár, J., Rahtz, M., McGrath, T., & Mikulik, V. (2023). Tracr: Compiled transformers as a laboratory for interpretability (arXiv:2301.05062). arXiv preprint arXiv:2301.05062.
- Elhage, N., Nanda, N., Olsson, C., Joseph, N., Henighan, T., Chen, A., & Amodei, D. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*, Anthropic. <https://transformer-circuits.pub/2021/framework/index.html>
- Houlsby, N., Giurui, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., ... & Gelly, S. (2019). Parameter-efficient transfer learning for NLP. *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, 2790–2799. PMLR.
- Nanda, N., Olsson, C., Elhage, N., & Henighan, T. (2023). Progress measures for grokking via mechanistic interpretability (arXiv:2301.05217). arXiv preprint arXiv:2301.05217.
- Elhage, N., Olsson, C., Nanda, N., Goel, K., & Ganguli, D. (2025). Circuits updates - April 2025. *Transformer Circuits*. <https://transformer-circuits.pub/2025/april-update/index.html>
- Gehman, S., Gururangan, S., Sap, M., Choi, Y., & Smith, N. A. (2020). RealToxicityPrompts: Evaluating neural toxic degeneration in language models (arXiv:2009.11462). arXiv preprint arXiv:2009.11462.
- Glaese, A., Hilton, J., O'Brien, T., Ouyang, L., Askell, A., Bai, Y., ... & Amodei, D. (2024). Safety layers in aligned large language models: The key to LLM security. *OpenReview*. <https://openreview.net/forum?id=kUH1yPMAn7>
- Liu, J., Mialon, G., Kreutzer, J., & Ruder, S. (2023). ToxicChat: Unveiling hidden challenges of toxicity detection in real-world user-AI conversation (arXiv:2310.17389). arXiv preprint arXiv:2310.17389.
- Meta AI. (2024). Llama 3.1 8B Instruct. Hugging Face.

<https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct>

- Montavon, G., Lapuschkin, S., Binder, A., Samek, W., & Müller, K.-R. (2019). Layer-wise relevance propagation: An overview. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (pp. 193–209). Springer. https://doi.org/10.1007/978-3-030-28954-6_10
- Zou, Y., Shao, Y., Chen, S., Tu, Y., Liu, X., Yang, Y., ... & Sun, M. (2024). JailbreakBench: An open robustness benchmark for jailbreaking large language models (arXiv:2404.01318). arXiv preprint arXiv:2404.01318.
- Lin, S., Hilton, J., & Evans, O. (2022). TruthfulQA: Measuring how models mimic human falsehoods. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, 3214–3253. <https://doi.org/10.18653/v1/2022.acl-long.229>
- Wang, X., Minervini, P., Ge, T., Li, X., & Wang, Y. (2023). HaluEval: A large-scale hallucination evaluation benchmark for open-domain dialogue models. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023)*, 1101–1117. <https://doi.org/10.18653/v1/2023.acl-long.67>
- Perez, E., Yu, L., Kasirzadeh, A., & Pavlick, E. (2022). Discovering language model behaviors with model-written evaluations (arXiv:2212.09251). arXiv preprint arXiv:2212.09251.
- Zheng, S., Ladhak, F., Holtzman, A., Shinn, N., & Zettlemoyer, L. (2023). Judging LLM-as-a-judge with GPT-4 (arXiv:2305.14278). arXiv preprint arXiv:2305.14278.

7. Appendix

A. Datasets Breakdown

A.1 Safety Judge Dataset Composition

The carefully curated dataset for the Safety Judge was a combination of real-world harmful content examples. Table A.1 summarizes the training dataset sources and distribution. The dataset is dynamically generated by sampling from various sources to ensure a 50-50 balance of safe and unsafe prompts across the training, validation, and test sets, aligning with the specified target percentages and overall sample numbers.

Dataset Source	Samples
RealToxicityPrompts	2,210
ToxicChat	1,135

JailbreakBench	347
Harmbench	308
Total	4,000

Table A.1: Safety Judge Training Dataset Sources and Distribution

Split	Safe Samples	Unsafe Samples	Total	Percentage (%)
Training	2,012	1,988	4,000	80
Validation	259	241	500	10
Test	229	271	500	10
Total	2,500	2,500	5,000	100

Table A.2: Safety Judge Dataset Split

A.2 Feasibility Judge Dataset Composition

Dataset Source	Samples
TruthfulQA	171
HaluEval	829

Table A.3: Safety Judge Training Dataset Sources and Distribution

Split	Samples	Percentage (%)
Training	800	80.0
Validation	99	9.9
Test	101	10.1
Total	1000	100

Table A.4: Safety Judge Dataset Split

A.3 Feasibility Judge Validation Set Metrics

Metric	Percentage (%)
Accuracy	62.0
Precision	63.7
Recall	70.0
F1 Score	66.7
AUC-ROC	62.1

Table A.5: Feasibility Judge Performance Metrics on Validation Set at Epoch 9

B. Interactive Guardian-Loop Dashboard

The training of the safety judge along with the mechanistic interpretability analysis are available through a dashboard designed to streamline these processes.

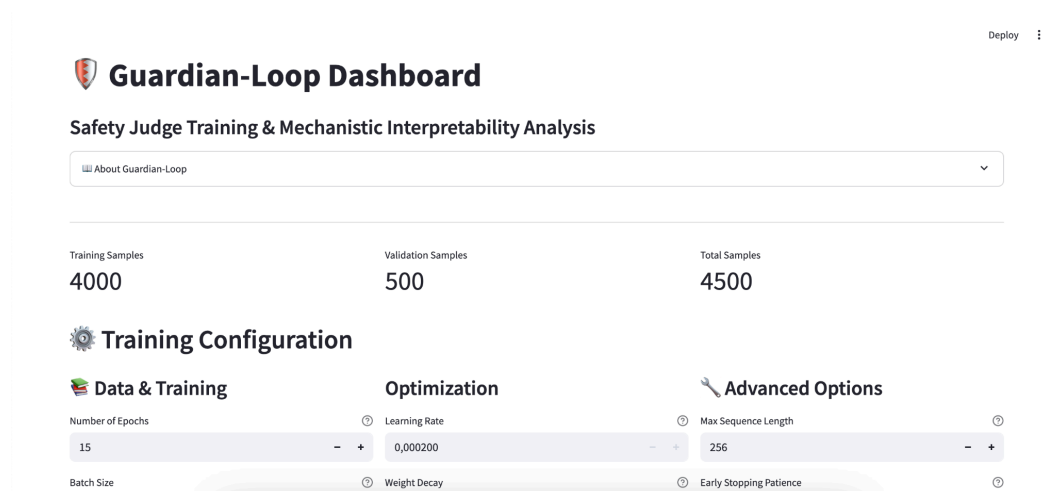


Figure B.1 – Screen Capture of the Guardian-Loop Dashboard for Safety Judge Training and Mechanistic Interpretability Analysis

C. LLM Usage

Large language models Claude 3.7 and Claude 3.5 were used for code generation and debugging assistance along with integrations of these models in Visual Studio Code. Moreover, they were used as part of the process of writing this report due to the time constraints of this hackathon.