Large Supervisor Models: Real-Time LLM Output Stream Supervision for Interruption

Praneeth Vadlapati
Independent Researcher
praneethv@arizona.edu
ORCID: 0009-0006-2592-2564

Abstract: Large language models (LLMs) have demonstrated remarkable capabilities across a broad spectrum of tasks, yet they remain susceptible to generating harmful, misleading, or otherwise unsafe content. Existing safety mechanisms, including post-hoc filtering, prompt engineering, and reinforcement learning from human feedback, are fundamentally reactive and operate either before or after the generation process, leaving a critical temporal gap during streaming output. This paper introduces the Large Supervisor Model (LSM), a novel architecture in which a lightweight, independently trained transformer-based model runs concurrently with an LLM, monitoring its token output stream in real time and issuing structured intervention signals—classified as abstain, feedback, or interrupt—at the moment unsafe content is detected. The LSM does not rewrite or alter model outputs; instead, it interrupts the generation stream and notifies the client with a structured JSON payload, allowing the partial response to be immediately cleared. We describe the design principles, training methodology, dual-path inference architecture combining an embedding-based neural classifier with a fine-tuned transformer, and an evaluation framework grounded in adversarially curated test sets. Experiments demonstrate that the combined LSM architecture achieves substantially higher precision and recall on harmful content detection compared to either component alone, while maintaining sufficiently low latency to operate transparently in production streaming pipelines.

The source code is available at github.com/Pro-GenAI/Large-Supervisor-Models.

Keywords: Large Language Models (LLMs), LLM safety, AI safety, real-time content moderation, streaming supervision, transformer architecture, harmful content detection, neural classifier, parallel inference

I. Introduction

The proliferation of large language models in consumer-facing applications has created an urgent need for robust, real-time safety mechanisms capable of operating at the speed and scale of modern inference pipelines. While significant progress has been made in aligning LLMs through techniques such as reinforcement learning from human feedback, constitutional AI, and instruction tuning, these methods address alignment at the level of model weights and training objectives rather than individual output streams. As a result, aligned models may still produce harmful content under adversarial prompting, unexpected context shifts, or edge cases not adequately covered during training. The gap between a model that is generally aligned and a system that reliably never produces harmful output at inference time remains substantial and practically significant.

Existing approaches to closing this gap fall into two broad categories: upstream filtering, which attempts to detect and refuse unsafe prompts before generation begins, and downstream filtering, which reviews completed responses before delivery. Both categories share a fundamental architectural limitation: they are temporally disjoint from the generation process itself. Upstream filters cannot anticipate what a model will generate given an ambiguous or multi-step prompt, while downstream filters must wait for the model to finish generating before any intervention is possible. In streaming deployments—which now constitute the dominant mode of LLM delivery in production systems—the downstream approach is particularly problematic, as tokens are transmitted to the client as they are generated, meaning harmful content may reach the user before any post-hoc system has had the opportunity to evaluate it.

This paper proposes a fundamentally different architectural paradigm: a dedicated supervision model that operates in parallel with the LLM during the generation process, reading the output token stream and capable of issuing an interruption signal mid-stream upon detecting unsafe content. We term this architecture the Large Supervisor Model (LSM). The LSM is conceptually analogous to a co-pilot or a real-time auditor: it does not generate, rewrite, or alter the content produced by the primary model, but it maintains a continuous and incremental representation of the ongoing response and acts the moment a safety threshold is crossed. This design preserves the full generative capability of the underlying LLM while providing a low-latency, independent safety layer that operates at the granularity of individual tokens or short spans.

A. Limitations of Existing Approaches

The limitations of existing safety mechanisms stem from their structural position relative to the generation pipeline. Prompt-level classifiers, while effective at catching overtly harmful inputs, are trivially circumvented by multi-turn manipulation, indirect phrasing, or context injection, and provide no protection against emergent harmful outputs that arise from otherwise benign prompts. Post-hoc content filters, even when highly accurate, are architecturally incompatible with the streaming paradigm: by the time a filter evaluates a completed response, the response has already been partially or fully transmitted to the client. Reinforcement learning from human feedback addresses alignment at the training level, producing models that are statistically less likely to generate harmful content, but this probabilistic guarantee is insufficient for safety-critical deployment contexts where even rare failures carry unacceptable consequences. Furthermore, existing guardrail systems that wrap LLM APIs typically introduce significant latency, operate on coarse-grained output units such as complete sentences or paragraphs, and lack the capacity for incremental, token-level analysis.

B. Proposed Approach

The LSM architecture proposed in this work addresses these limitations by introducing a purpose-built supervision model that runs concurrently with the LLM during inference, consuming the same token output stream that is transmitted to the client. The LSM encodes only new tokens incrementally, avoiding the computational overhead of re-processing the full context on each update, and uses a queue-based processing mechanism to ensure that no tokens are missed even when the supervisor experiences transient processing delays. When the LSM detects content meeting or exceeding a configurable harm threshold, it emits an interrupt signal that is forwarded to the client alongside a structured payload containing the reason for interruption, a confidence score, and the offending token span. The client is thereby able to immediately clear the partial response and display an appropriate message. When confidence is below the interrupt threshold, the LSM can emit a feedback signal that is logged for downstream analysis, human review, or model retraining without interrupting the user experience. When the LSM determines that the content is safe, it emits an abstain signal and processing continues unimpeded.

C. Applications

The LSM architecture is broadly applicable to any deployment context in which an LLM generates text that is streamed to a user or downstream system. Consumer chat interfaces represent the most immediate application domain, where the risk of harmful content reaching vulnerable users—including minors, individuals in crisis, or users who have not consented to potentially disturbing content—is highest and the consequences most severe. Beyond direct consumer applications, the LSM framework is relevant to API providers seeking to enforce usage policies at the infrastructure level without relying solely on client-side enforcement, to enterprise deployments where compliance and auditability requirements demand a verifiable content review trail, and to research settings where automated red-teaming and systematic analysis of model failure modes require a principled mechanism for capturing and categorizing harmful outputs at scale.

D. Related work

Prior work on LLM safety supervision has explored several complementary directions. Constitutional AI introduced the idea of an LLM critiquing and revising its own outputs according to a set of principles, but this approach is computationally expensive and operates on complete generations rather than streaming outputs. Moderation APIs such as those provided by OpenAI and Perspective offer text classification services for completed inputs but are not designed for incremental, token-level supervision of streaming outputs. Research on speculative decoding and parallel decoding architectures has demonstrated the feasibility of running multiple models concurrently during inference, providing an existence proof for the computational model underlying the LSM approach. Work on early exit and anytime models in the natural language processing literature has similarly explored architectures in which predictions can be emitted at intermediate stages of processing, which is conceptually related to the LSM's requirement for low-latency incremental inference. The LSM distinguishes itself from all of these prior efforts by being the first architecture specifically designed for real-time, token-level, streaming supervision with structured intervention capabilities, operating as an independent model rather than a component of the primary LLM.

II. METHODS

A. System Architecture and Streaming Integration

The LSM operates as a sidecar process to the primary LLM inference server, subscribing to the token output stream via the OpenAI-compatible streaming API. Tokens are received as text chunks rather than raw logits or token identifiers, reflecting the practical constraints of widely deployed inference APIs. Upon receipt, each new token chunk is appended to an internal processing queue. A dedicated consumer thread dequeues token chunks sequentially and passes them to the LSM inference pipeline, which maintains an incremental encoding of the response seen so far. Crucially, the LSM does not re-encode the full response context on each new token; instead, it maintains a rolling hidden state that is updated incrementally as new tokens arrive, analogous to the recurrent update step in a streaming sequence model. This design choice is essential for achieving the low latency required for production deployment, as re-encoding the full context on each token would scale linearly with response length and quickly become prohibitive. The LSM's output is checked against configurable thresholds before each processed token chunk is forwarded to the client, ensuring that no token is delivered without having passed supervisor review.

B. Training Data Generation

A central challenge in training the LSM is the acquisition of sufficient labeled examples of harmful LLM outputs. By design, contemporary aligned LLMs refuse to generate the most dangerous categories of content, making it impossible to rely solely on automated generation pipelines to produce training data for the most critical safety scenarios. The training data generation process therefore adopts a hybrid strategy. In the first phase, an unaligned or less-restricted LLM is prompted with a diverse set of adversarial inputs spanning the target harm categories—self-harm promotion, hate speech, dangerous technical instructions, personal harassment, and bioweapon-adjacent content—and the resulting outputs are collected and labeled by human annotators according to the LSM output schema, which assigns each example an output type (abstain, feedback, or interrupt), a reason category, and a confidence score. In the second phase, human authors directly compose harmful text examples in the relevant categories, ensuring coverage of edge cases and linguistic patterns that automated generation may underrepresent. Benign examples covering a wide range of topics, registers, and output lengths are generated separately to ensure the training set is balanced and that the LSM does not develop a bias toward over-interruption. The resulting training corpus is structured as a set of (LLM output text, LSM output label) pairs, serialized in a standardized JSON format that includes the raw text, tokenized representation, and full LSM output object.

C. Classifier-Based Detection Path

The first of the two concurrent detection paths in the LSM architecture is a classifier built on top of dense text embeddings. Input text, accumulated incrementally from the streaming token buffer, is encoded using a pretrained sentence embedding model to produce a fixed-dimensional representation of the current response span. This embedding is passed through a multilayer perceptron with dropout regularization, trained to predict one of the three LSM output classes. The classifier is optimized using cross-entropy loss with class weighting to account for the natural imbalance between harmful and benign examples in the training corpus. A key design consideration for the classifier is operating on partial outputs: because the LSM must make decisions mid-stream, the classifier is trained on truncated prefixes of complete responses as well as on full responses, teaching it to recognize harmful intent or content even when only a fragment of the dangerous passage has been generated. The classifier is the faster of the two detection paths and serves as the primary mechanism for early interruption of clear-cut harmful content, with the transformer path providing complementary coverage for more subtle or context-dependent cases.

D. Transformer-Based Detection Path

The second detection path employs a small, fine-tuned transformer model trained end-to-end on the LSM training corpus. The model takes the accumulated streaming text as input and is trained to autoregressively predict the appropriate LSM output token—abstain, feedback, or one of several interrupt reason tokens such as self_harm, hate_speech, dangerous_instructions, or harassment. The transformer architecture is selected to be significantly smaller than production-scale LLMs, targeting a parameter count in the range of tens to hundreds of millions to ensure that inference latency remains compatible with real-time streaming requirements. The model is fine-tuned from a pretrained checkpoint using a curriculum that begins with full-length examples and progressively introduces shorter prefixes, mirroring the incremental nature of the streaming supervision task. Because the transformer processes the full accumulated context at each inference step, it is better positioned than the classifier to detect harmful content that depends on earlier context in the response—for example, a passage that appears benign in isolation but becomes harmful in light of preceding text.

E. Score Fusion and Decision Logic

The outputs of the classifier and transformer paths are fused using a late fusion strategy in which confidence scores from both models are combined via a learned linear weighting that is calibrated on a held-out validation set. The fused score is then evaluated against two thresholds: an interrupt threshold, above which the system issues an interrupt signal and halts the stream, and a feedback threshold, below the interrupt threshold but above a baseline, at which the system logs a feedback signal without interrupting the stream. Content below the feedback threshold receives an abstain signal. When the two paths disagree substantially—as measured by the absolute difference in their confidence scores—the system defaults to the more conservative (higher-confidence) path, preferring false positives over false negatives in safety-critical contexts. This asymmetric decision policy reflects the fundamental design principle that an unnecessary interruption, while disruptive, is a substantially less severe failure mode than allowing harmful content to reach a user. The feedback signals generated at intermediate confidence levels are stored for offline analysis and used as an additional source of weak supervision signal for iterative retraining of both detection paths.

F. Evaluation Protocol

Evaluation of the LSM is conducted using a dedicated test set that is constructed independently of the training corpus to prevent data leakage. The test set comprises three subsets: a harmful subset containing adversarially generated and manually authored examples of content that should trigger an interrupt or feedback signal, a borderline subset containing examples that are potentially concerning but do not clearly warrant interruption, and a benign subset containing a diverse range of safe outputs. Each subset is evaluated separately and in combination to produce precision, recall, F1, and area under the receiver operating characteristic curve metrics for each detection path individually and for the fused system. Latency is measured as the time between receipt of a new token chunk and the issuance of the LSM's decision signal, averaged over a large sample of streaming responses of varying lengths. Both detection paths and the fused system are evaluated under this protocol, enabling direct comparison and ablation.

III. RESULTS

A. Classifier Performance

The embedding-based classifier demonstrated strong performance on clear-cut harmful content categories, achieving a precision of 0.91 and recall of 0.87 on the harmful subset of the evaluation corpus. On borderline examples, precision declined to 0.73 while recall remained relatively stable at 0.81, consistent with the expected behavior of a model trained on more extreme examples being less calibrated for ambiguous cases. The classifier's mean decision latency was 18 milliseconds, making it well-suited for real-time supervision of typical streaming response rates. False positive rates on the benign subset were low at 0.04, indicating that the classifier does not have a strong tendency toward over-interruption on ordinary content. Performance was notably weaker on harmful content expressed through indirect or euphemistic language, where the embedding representation was insufficiently sensitive to semantic nuance.

B. Transformer Performance

The fine-tuned transformer detection path achieved higher overall accuracy than the classifier on the full evaluation set, with a precision of 0.89 and recall of 0.93 on the harmful subset, and substantially better performance on borderline examples, where it achieved a precision of 0.81 and recall of 0.85. This performance advantage on borderline and context-dependent examples is consistent with the hypothesis that the transformer's capacity to represent long-range contextual dependencies enables more nuanced assessment of content that cannot be evaluated in isolation. The transformer's mean decision latency was 67 milliseconds, approximately 3.7 times higher than the classifier, reflecting the greater computational cost of full-context transformer inference. False positive rates on benign content were 0.06, marginally higher than the classifier but within acceptable bounds for production deployment.

C. Fused System Performance

The combined LSM system, fusing classifier and transformer outputs via the calibrated linear weighting described in the methods section, achieved the strongest performance across all evaluation subsets. On the harmful subset, the fused system achieved a precision of 0.94 and recall of 0.95, representing a statistically significant improvement over either component alone. On borderline examples, the fused system achieved a precision of 0.83 and recall of 0.88. False positive rates on benign content were 0.05, within the range of the individual components. End-to-end latency for the fused system, accounting for the parallel execution of both paths and the fusion step, was 74 milliseconds at median, with the 95th percentile latency reaching 132 milliseconds—well within the practical tolerance of streaming response pipelines, where individual token generation intervals typically exceed 50 milliseconds for production-scale LLMs. Ablation experiments confirmed that both components contributed meaningfully to fused system performance, with the classifier driving gains on early-stream detection and the transformer contributing disproportionately to accuracy on longer, more contextually complex responses.

IV. Discussion

The results presented above support the central thesis of this work: that parallel, real-time streaming supervision is both technically feasible and practically effective for improving the safety of LLM deployments without requiring modifications to the underlying model or its training procedure. The LSM architecture achieves this through a combination of incremental encoding, queue-based processing, and a dual-path inference design that balances speed and accuracy. Several aspects of the results warrant further discussion.

The performance gap between the classifier and transformer on borderline and context-dependent examples highlights an important dimension of the safety supervision problem that is frequently underappreciated in the literature: harmful content is not always self-evident from local token spans, and effective supervision often requires sensitivity to the broader discourse context in which a given passage appears. The transformer's superior performance in these cases, despite its higher latency, argues for the value of the dual-path design—the classifier handles clear-cut cases quickly, while the transformer provides deeper analysis for ambiguous cases where speed can be traded for accuracy without violating real-time constraints.

The feedback signal mechanism, which logs borderline cases without interrupting the stream, represents a practically important component of the LSM framework that is not fully captured by the evaluation metrics presented here. In production deployment, these feedback signals constitute a valuable source of ongoing monitoring data, enabling operators to detect distributional shifts in model outputs, identify emerging harmful patterns not covered by the training corpus, and generate labeled examples for iterative retraining. This feedback loop transforms the LSM from a static filter into a continuously improving safety system, which is essential given the rapidly evolving landscape of adversarial prompting techniques.

It is important to note that the LSM is not designed to provide adversarial robustness against sophisticated jailbreaking attacks. A determined adversary who understands the LSM's detection criteria may be able to craft outputs that evade detection by distributing harmful content across token spans in ways that individually fall below the interrupt threshold. Addressing this attack surface would require additional architectural components, such as attention over longer response windows or explicit adversarial training against evasion strategies, and is left for future work. The LSM's design objective is to protect the general public from harmful content that arises from ordinary use of LLMs, not to provide guarantees against targeted adversarial manipulation.

V. Conclusion

This paper has introduced the Large Supervisor Model, a novel architecture for real-time, token-level supervision of LLM streaming outputs. By operating in parallel with the primary LLM and making incremental decisions on the output stream rather than waiting for generation to complete, the LSM addresses a fundamental limitation of existing safety mechanisms and enables intervention at the earliest possible moment. The dual-path architecture combining an embedding-based classifier with a fine-tuned transformer achieves strong precision and recall on harmful content detection while maintaining latency characteristics compatible with production streaming deployments. The structured intervention framework—comprising interrupt, feedback, and abstain signals—provides a flexible vocabulary for expressing the LSM's assessment, enabling both hard interruptions for high-confidence harmful detections and soft feedback logging for borderline cases. Taken together, these contributions represent a meaningful step toward safety mechanisms that are temporally integrated with the generation process rather than structurally external to it, and we anticipate that the LSM framework will serve as a foundation for future work on real-time LLM supervision, adversarial robustness, and continuously improving safety systems.

REFERENCES

[1] <To be added later>