QueryStop: Dynamic Stop Signals for Efficient Streaming Inference

Michelle Shu
Cornell Tech
ms3682@cornell.edu

Jingyao Li
CUHK
jingyao.li@link.cuhk.edu.hk

Pengguang Chen
SmartMore
akuxcw@gmail.com

Shu Liu
SmartMore
liushuhust@gmail.com

Jiaya Jia
HKUST
jia@ust.hk

Ramin Zabih
Cornell Tech & Google
ramin.zabih@cornell.edu

Abstract

As language models are increasingly deployed in long-context and streaming workflows, deciding when to stop context processing is central to inference efficiency. While recent research has shown that a subset of attention heads encodes sufficiency signals, these markers are typically detected via static, task-agnostic global features. In this paper, we demonstrate that stop signals are inherently query-relevant and introduce QUERYSTOP, an attention-aggregation method that systematically identifies query-focused attention heads whose hidden features effectively align with the input query. Driven primarily by query-focused heads, this tailored ensemble yields high-fidelity representations that significantly improve sufficiency prediction. Across four benchmarks on eleven datasets, our method improves sufficiency stop prediction by 3.52% F1 on average. Under comparable token reduction, it raises Sufficiency Reach Rate (SRR), our proposed stopping-adequacy metric, from 87.12% to 94.73% and yields a 7.09% average gain in downstream task accuracy over the prior baseline. Collectively, these results show that QUERYSTOP effectively eliminates unnecessary context processing while simultaneously improving downstream answer quality.

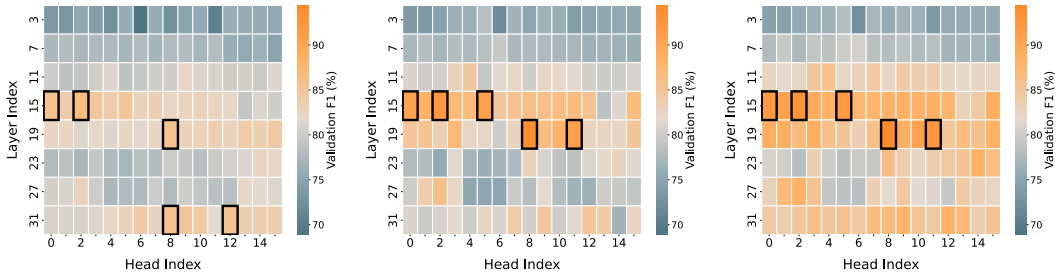
1 Introduction

Modern language-model systems are increasingly being developed as interactive agents that acquire information through search, web navigation, and external tool use rather than relying only on a static context provided upfront [26, 3, 48, 32, 54]. In these web-search, browser-agent, and tool-use settings, a model must iteratively interact with an external environment, issue search and browsing actions, incorporate newly returned evidence, and ultimately produce an answer. Information therefore arrives incrementally, and the amount required to answer a query is unknown in advance.

The open-ended nature of streaming evidence exposes a limitation of many existing context-efficiency techniques, including retrieval-augmented generation [17], prompt compression [8, 9], and token pruning or merging methods [53, 40], as they require the incoming context to be reduced to a retained prefix before they can be applied. Choosing this prefix introduces a fundamental trade-off, where truncating too early may exclude evidence necessary for a correct answer, while retaining too much context incurs avoidable latency and computation and may dilute access to relevant information in long contexts [49, 21]. This trade-off highlights the need for an efficient *stop signal*: an indicator that the evidence observed so far is sufficient for answering.

Prior work has shown that hidden representations in the attention heads of large language models encode latent sufficiency signals, allowing lightweight classifiers to predict whether the critical

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Global (b) Query (c) Combined

Figure 1: Visualization of probe performance across different hidden representations, measured by the F1 score for binary sufficiency prediction on the Long benchmark using Qwen 3.5 9B. Higher F1 scores from mid-layers of (b) indicate that query-focused features from mid-layer heads are more informative.

information needed to answer a query has been processed [45]. However, existing methods treat these markers as static and rely on broad, task-agnostic aggregation, which risks smoothing out highly localized, answer-bearing information. We argue that localized, query-aligned representations encode a substantially stronger sufficiency signal. This is evident in Figure 1, where, on a long-context dataset comprising both single-hop and multi-hop tasks, probes trained on query-aligned features extracted from a concentrated subset of mid-layer heads achieve much higher F1 scores on sufficiency stop point prediction than their global-averaging counterparts. Therefore, we introduce QUERYSTOP, an attention-aggregation method that systematically identifies query-focused attention heads whose hidden features effectively align with the input query. By constructing a tailored ensemble driven dominantly by these query-focused representations, our method yields high-fidelity features that improve sufficiency prediction by a large margin. This design aligns with established models of incremental information seeking [30, 25], which posit that while maintaining a broad awareness of the overall evidence stream is helpful, effective stopping decisions are primarily governed by the discovery of highly relevant, query-specific cues.

We evaluate QUERYSTOP across four benchmark groups covering eleven datasets: Long [45], Needle-in-a-Haystack [12], 2WikiMultiHopQA [5], and Passage Retrieval [13]. These benchmarks span localized evidence retrieval, multi-hop reasoning, synthetic needle localization, and heterogeneous domain-specific retrieval over trivia, e-commerce, and legal contexts. Across these settings, QUERYSTOP consistently improves stop-signal prediction over the prior probing baseline, with an average gain of **3.52% F1**. To assess whether better stopping also preserves downstream utility, we compare methods under matched token-reduction operating points and report *Sufficiency Reach Rate* (SRR), our proposed metric for measuring whether a stop point reaches the ground-truth sufficiency position. At comparable token reduction, our method improves task accuracy by **7.09%** and raises SRR from **87.12%** to **94.73%**. By reliably identifying the exact point of information sufficiency without requiring additional generation overhead, QUERYSTOP establishes a highly effective, deployable solution for dynamic context management in streaming environments.

2 Related Work

Context efficiency under streaming evidence. Large language models are increasingly deployed as interactive agents that acquire external evidence incrementally through search, web navigation, and tool use [26, 48, 32, 54, 3]. This streaming setting exposes a limitation of existing long-context efficiency methods. Retrieval-augmented generation [17], prompt compression [8, 9], and token merging or pruning [53, 40] primarily optimize processing *after* a retained input has already been determined. Relying on arbitrary truncation or hard token limits exposes a severe trade-off: stopping arbitrarily early may omit necessary evidence, while over-accumulating marginal evidence increases latency and degrades the agent's ability to deduce a correct answer [49, 21]. This highlights the necessity of a dynamic stop signal to govern context intake.

Trained stopping mechanisms and external evaluators. Recent work has explored learned controllers for adaptive information acquisition. InfMem [39], for example, introduces an active memory control loop for long-context agents, training a controller to decide when to retrieve additional in-

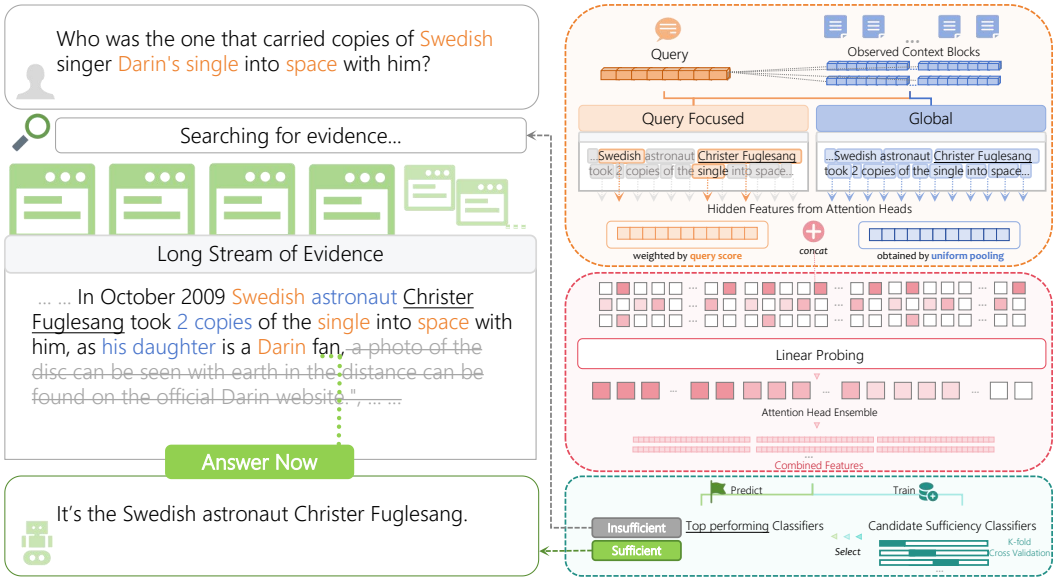


Figure 2: **Overview of the QueryStop pipeline.** As a long stream of context is processed, we evaluate block-level checkpoints. At each checkpoint, we apply linear probing to an ensemble of specialized attention heads to extract query-focused feature representations. These representations are then used to train a binary classifier, and the top-performing model is selected to trigger the dynamic context stop signal. Note that concatenation with a global representation is an optional supplement.

document evidence, update memory, or stop once sufficient evidence has been consolidated. Related RAG-based systems such as Stop-RAG [28], ChatR1 [22], and Self-RAG [1] learn to coordinate retrieval, generation, or self-reflection during inference. While these methods are well suited to adaptive retrieval over fixed corpora, indexed document collections, and bounded-memory environments, their controllers typically require reinforcement learning, instruction tuning, or reflection-token training; hence, they are not directly applicable to incrementally revealed evidence prefixes.

Stopping generation versus explicit self-prompting. Another line of work studies early stopping for internal reasoning, including chain-of-thought and long-reasoning models [34, 2]. These methods reduce unnecessary reasoning tokens by monitoring answer entropy [15], confidence dynamics [6], or answer convergence and stability [24]. Our setting concerns context intake rather than generation: the goal is to determine when the observed external evidence is sufficient, before the full context stream has been consumed. A natural training-free baseline for this setting is *self-prompting* [11, 1, 33], where the model is repeatedly asked whether the accumulated context is sufficient. Generating explicit self-assessments during context intake, however, introduces additional inference cost and, as our experiments show, can produce unstable stopping behavior and suboptimal answer quality.

Lightweight stop signals via internal representations. A more efficient alternative to generation-heavy methods is predicting sufficiency directly from the model's internal representations. Prior work [45] shows that attention-head representations encode latent sufficiency signals, allowing lightweight classifiers to halt processing without additional text generation. We demonstrate that such stop signals are inherently query-relevant and introduce an attention-aggregation method that systematically identifies query-focused attention heads. By forming a tailored ensemble of query-focused attention heads, QUERYSTOP extracts high-fidelity representations that significantly improve sufficiency prediction. This results in a highly accurate, deployable stop signal that correctly identifies the exact point of information sufficiency, preserving downstream utility and maximizing Sufficiency Reach Rate (SRR) in streaming environments.

3 Methodology

Our method dynamically controls information intake by identifying when a language model has accumulated enough evidence to answer a given query. Our approach is divided into three primary

stages (Figure 2): streaming context processing, attention head ensemble construction, and sufficiency signal prediction.

3.1 Streaming Context Processing

Let M denote a pretrained language model, q a query, and I an incoming stream of external information. We partition the incoming context into an ordered sequence of non-overlapping information blocks, $I = s_1 \| s_2 \| \cdots \| s_N$, where N represents the maximum information stream the model can handle, and $\|$ denotes concatenation. Let $I_k = s_1 \| \cdots \| s_k$ denote the cumulative context up to block k. Our objective is to find the smallest $k \leq N$ such that the task performance using I_k matches the full context, i.e., $M(q, I_k) \approx M(q, I)$.

To approximate this optimal k dynamically, we define a feature extraction function f and train a sufficiency predictor S_p. At each checkpoint i, the predictor evaluates the features of the accumulated information blocks, $f(I_i)$, to produce a sufficiency score. Processing halts once this score exceeds a predefined threshold τ, such that $S(I_i) = 1$ if $S_p(f(I_i)) \geq \tau$, and 0 otherwise.

If the sufficiency score $S(I_i) = 1$, context intake terminates and I_i is passed to the model for final generation. Because the blocks are non-overlapping, the KV cache from previously processed blocks is inherently reused when a new block arrives, avoiding redundant computation over the shared prefix.

3.2 Attention Head Ensemble Construction

As outlined in the prior section, our sufficiency predictor S_p takes in features extracted from the accumulated context I_i at each timestep i to produce a sufficiency score. Inspired by information foraging theory [30], which posits that effective navigation and stopping decisions are driven by an "information scent," we identify internal query-key (QK) interactions as a naturally occurring scent of relevance within large language models, which have recently proven successful in optimizing various downstream tasks [18, 50, 36]. We therefore propose modeling the feature function f through an ensemble of specialized *query-focused heads* as follows.

Query-Focused Heads: Given a query q and an information stream I, we define U as the set of query token positions. For any specific attention head, we aggregate the interactions between the query vectors $Q_u \in \mathbb{R}^{d_h}$ ($u \in U$) and the context key vectors $K_t \in \mathbb{R}^{d_h}$ in the current prefix I_i. Exploiting the linearity of the dot product, the relevance score r_t for token t is reduced to:

$$r_t = \frac{1}{|U|} \sum_{u \in U} \frac{\langle Q_u, K_t \rangle}{\sqrt{d_h}} = \frac{\langle \bar{Q}, K_t \rangle}{\sqrt{d_h}}, \tag{1}$$

where $\bar{Q} = \frac{1}{|U|} \sum_{u \in U} Q_u$ represents the average query vector. These scores are then normalized via a softmax function over the valid token positions T_i within I_i (with temperature G) and are used to weight the hidden representations $o_t \in \mathbb{R}^{d_h}$ of each token to obtain the final query-focused feature:

$$x_i^q = \sum_{t \in T_i} \alpha_t o_t, \quad \alpha_t = \frac{\exp(r_t/G)}{\sum_{t' \in T_i} \exp(r_{t'}/G)} \tag{2}$$

Intuitively, this formulation acts as a dynamic query-relevant filter, where α_t selectively magnifies the hidden states of tokens that semantically align with the user's intent. Each attention head, therefore, yields a distinct query-focused feature x_i^q. To assess their informativeness, we train independent logistic regression probes via cross-validation to predict sufficiency scores based solely on this x_i^q representation. Figure 1 provides an example of the predictive performance (F1 score) of these individual heads. We observe that only a small, specialized subset of heads, predominantly in the middle layers, strongly encodes sufficiency via query-focused features.

Supplementary Global Heads: While query-focused features excel at pinpointing localized evidence, complex tasks such as multi-hop reasoning may require synthesizing distributed information and a sense of overall document structure. We therefore optionally incorporate global heads in our ensemble to maintain a summary of the overall accumulated state. Global heads are identified through a parallel process. For each head, we extract a global contextual feature by averaging the hidden representations over the current prefix, $x_i^g = \frac{1}{|T_i|} \sum_{t \in T_i} o_t$. And we similarly apply linear probing to x_i^g to evaluate its predictive capacity.

Ensemble Construction: During the ensemble stage, we choose the top-K_h performers with the highest predictive capacity as our dedicated query heads. To optionally supplement the query-focused attention head ensemble with global context information, we concatenate the query and global features into a single representation $[x_i^q; x_i^g]$ for each candidate head. A unified linear probe is then trained on this combined feature space. This approach allows the probe to autonomously weigh the relative importance of query alignment versus global context for every head. By letting the data drive the fusion, the ensemble naturally adapts to diverse task requirements.

3.3 Sufficiency Predictor Training and Inference

Training: Our final sufficiency predictor S_p operates on the collective signal of the attention head ensemble by concatenating individual feature representations from each into a single, unified feature vector $f(I_i)$ for the current prefix. Similar to prior work [45], we frame sufficiency prediction as a binary classification task managed by lightweight linear probes and train a collection of K_c independent lightweight classifiers $\{S_j\}_{j=1}^{K_c}$ on $f(I_i)$ using cross-validation on a held-out validation set to compute the overall sufficiency predictor $S_p(f(I_i)) = \frac{1}{K_c}\sum_{j=1}^{K_c} S_j(f(I_i))$, which is the average output of these top K_c selected classifiers.

Inference Time: During inference, the dynamic stopping process is highly optimized and introduces negligible overhead by exclusively utilizing existing internal states. Since the query remains fixed for a given task, the average query vector $\bar{Q}$ is computed only once at the beginning of the interaction and cached for reuse. As the language model processes the streaming context block by block, the hidden states and key vectors K_t are naturally generated as a byproduct of the standard forward pass. At each checkpoint i, the system simply routes these inherent hidden features from the attention head ensemble into our lightweight classifier to calculate the sufficiency score $S_p(f(I_i))$. If this score meets or exceeds the predefined threshold τ, context intake is immediately halted. The model then bypasses any remaining information blocks and proceeds directly to generating the final answer, achieving highly accurate dynamic cutoffs while minimizing latency and eliminating wasted computation.

4 Experiments

Benchmarks. We evaluate our method across four benchmark groups encompassing 11 distinct datasets: **Long** [45] combines single-hop and multi-hop long-context tasks. It includes SQuAD [31], Natural Questions [14], and Code Understanding based on PCSD [38] for localized evidence retrieval, alongside HotpotQA [47], MUSIQUE [37], and Multi-hop Key-Value Retrieval [51] for reasoning over distributed evidence. **Needle-in-a-Haystack** [12] embeds a target statement within distractor contexts of up to 32K tokens, abstracting the challenge of isolating minute evidence among large amounts of irrelevant context. **2WikiMultiHopQA** [5] evaluates multi-hop reasoning over Wikipedia and Wikidata, mirroring web-style information seeking where queries require integrating scattered evidence. Finally, **Passage Retrieval** from PolEval 2022/23 [13] provides a highly heterogeneous, practical retrieval setting featuring open-domain trivia, e-commerce FAQs (Allegro.pl), and Polish legal provisions. These benchmarks collectively stress-test the model's capacity for evidence localization, multi-document reasoning, and retrieval from heterogeneous contexts.

Evaluated Methods. To explicitly test our hypothesis regarding attention head specialization, we evaluate two variants of our proposed method: 1) **QueryStop-Q.** This variant uses an attention head ensemble that relies exclusively on query-focused features (x^q). This isolates the performance of "query heads" without the context of the global state. 2) **QueryStop-C:** It supplements the query representation with additional global information ($[x^q; x^g]$), allowing the linear probe to autonomously weigh query-focused relevance against global contextual states to select the optimal ensemble.

We also evaluate three stop prediction baselines: 1) **Dynamic Context Cutoff (DCC):** The probing framework introduced by Xie et al. [45], which acts as our primary baseline, predicts sufficiency using only global average features (x^g), conceptually mirroring an ensemble restricted entirely to "global heads." 2) **Self-Prompt-MultiRound (SP-M)** [45] repeatedly prompts the model with a "yes/no" sufficiency question at each timestep. Because the explicit meta-prompt it uses breaks the contiguous streaming state, the model cannot reuse its KV cache and must re-process the accumulated context from scratch at every step, requiring $k(k+1)/2$ redundant block reads for k steps. 3) **Self-Prompt-SingleRound (SP-S)** is our optimized single-round variant, which avoids this quadratic

Table 1: F1, accuracy, and recall of binary classifier for stop signal across models, methods, and datasets. **Bold** indicates the best performance, and underline indicates the second best.

Model	Method	Long			2WikiMQA			Passage Retrieval			Needle in Haystack		
		F1	Acc	Rec	F1	Acc	Rec	F1	Acc	Rec	F1	Acc	Rec
Qwen 3.5 2B	SP-S	37.6	49.6	27.4	45.6	55.2	32.9	33.3	49.6	21.4	19.8	48.0	11.4
	SP-M	41.3	55.9	28.0	56.9	61.7	44.4	49.2	58.1	34.4	62.1	54.5	66.1
	DCC	89.2	87.6	92.3	80.6	80.1	80.3	83.6	82.1	82.1	92.5	91.9	92.7
	QueryStop-Q	91.1	89.9	94.2	**85.5**	84.8	86.3	87.2	86.1	85.3	**98.1**	97.9	98.5
	QueryStop-C	**92.3**	91.2	94.6	82.5	82.0	82.3	**87.8**	86.6	87.5	98.0	97.8	98.3
Qwen 3.5 4B	SP-S	27.6	53.4	16.1	44.8	55.8	31.4	33.2	51.8	20.3	67.1	54.2	82.8
	SP-M	48.3	62.1	32.0	67.4	70.2	54.1	87.8	87.0	79.3	19.0	42.3	12.0
	DCC	92.4	91.4	94.4	84.5	83.8	85.5	87.6	86.2	87.6	96.0	95.7	95.2
	QueryStop-Q	**96.5**	96.0	98.0	**87.6**	86.9	89.2	89.8	88.9	88.7	**99.8**	99.7	99.7
	QueryStop-C	96.4	96.0	97.7	85.7	84.8	88.4	**90.7**	89.8	90.2	99.7	99.7	99.5
Qwen 3.5 9B	SP-S	52.2	60.8	38.8	77.3	70.3	88.7	53.5	58.2	40.8	70.1	59.9	83.4
	SP-M	82.5	81.7	77.6	67.5	69.8	55.1	**92.7**	91.9	87.4	34.2	51.0	22.6
	DCC	92.6	91.7	94.2	87.6	86.8	90.3	88.1	86.8	88.2	95.7	95.3	95.7
	QueryStop-Q	**96.2**	95.8	97.5	90.0	89.6	91.0	91.4	90.6	90.6	**99.4**	99.3	99.2
	QueryStop-C	**96.2**	95.7	97.7	**90.2**	89.8	90.6	91.2	90.4	90.0	99.0	98.9	98.9

overhead by maintaining a contiguous KV cache and appending a lightweight sufficiency suffix, switching to answer generation only when sufficiency is detected.

Experiment Setup and Metrics. We instantiate our experiments on the Qwen 3.5 family, using the 2B, 4B, and 9B variants.[1] Qwen 3.5 uses a hybrid architecture that combines linear-attention and standard self-attention layers. Aligned with the mechanistic interpretability literature [41, 23, 29, 50], we focus exclusively on standard self-attention mechanisms, and select the top $K_h = 5$ attention heads by validation F1, evaluate 8 types of lightweight binary classifiers, and retain the top 4 classifiers for the final ensemble.

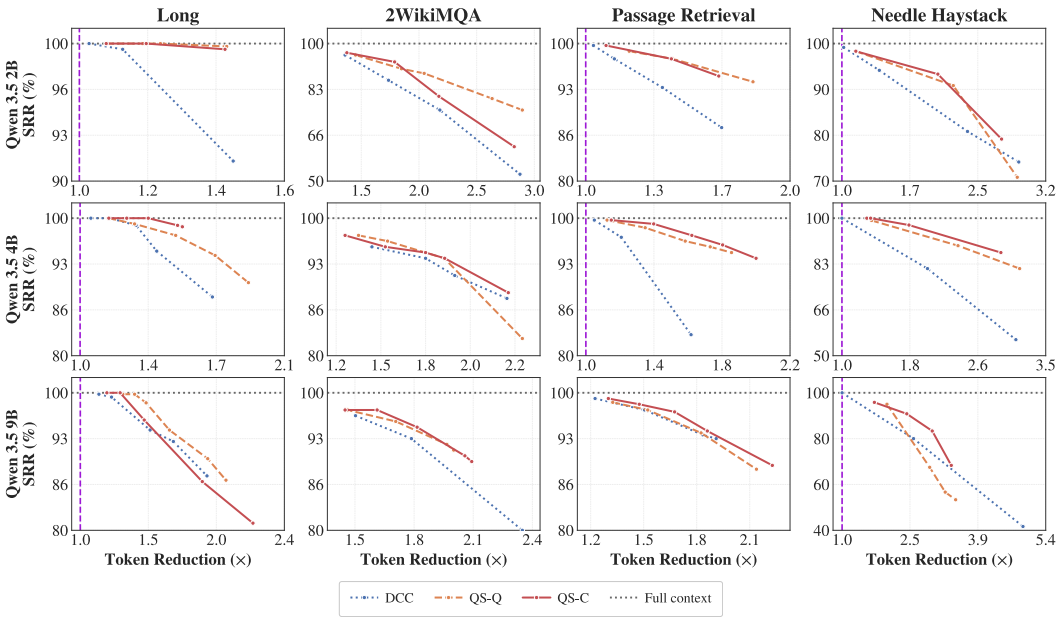
While the information blocks s can theoretically vary in length, utilizing fixed-size blocks is an efficient and practical setup for LLM deployment. Therefore, for all methods, we divide incoming contexts into cumulative prefixes at 10% intervals. Our supervision target is the earliest evaluated checkpoint containing the complete benchmark-annotated evidence chain. Note that this provides an evidence-grounded reference boundary, which need not coincide with model-specific answerability: a model may answer correctly before this boundary or fail even after all annotated evidence is available. Accordingly, we interpret sufficiency-prediction metrics as agreement with the annotation-based criterion and evaluate downstream answer accuracy separately to assess the practical consequences of stopping. Further analysis of this distinction and an ablation on checkpoint granularity are provided in Section A.6 and Section 5, respectively.

We primarily assess the performances of stop signal methods across 1) **Stopping Point Precision:** We report the binary classifier's **F1**, **Accuracy**, and **Recall@90** (Table 1) to evaluate its precision in identifying the exact sufficiency points. 2) **Mature Stopping Rate:** We introduce the **Sufficiency Reach Rate (SRR)**, which measures the percentage of samples for which the predicted stop point successfully reaches or surpasses the gold-standard evidence location. By evaluating SRR alongside the **Token Reduction Factor (T.RF)** (the multiplier of context effectively skipped), we can quantify a method's ability to avoid premature termination while still maintaining highly efficient context reduction. 3) **Downstream Task Quality:** We analyze the downstream task **Accuracy** of the generated answers, measured through model-based evaluation using GPT-5.4 Mini.

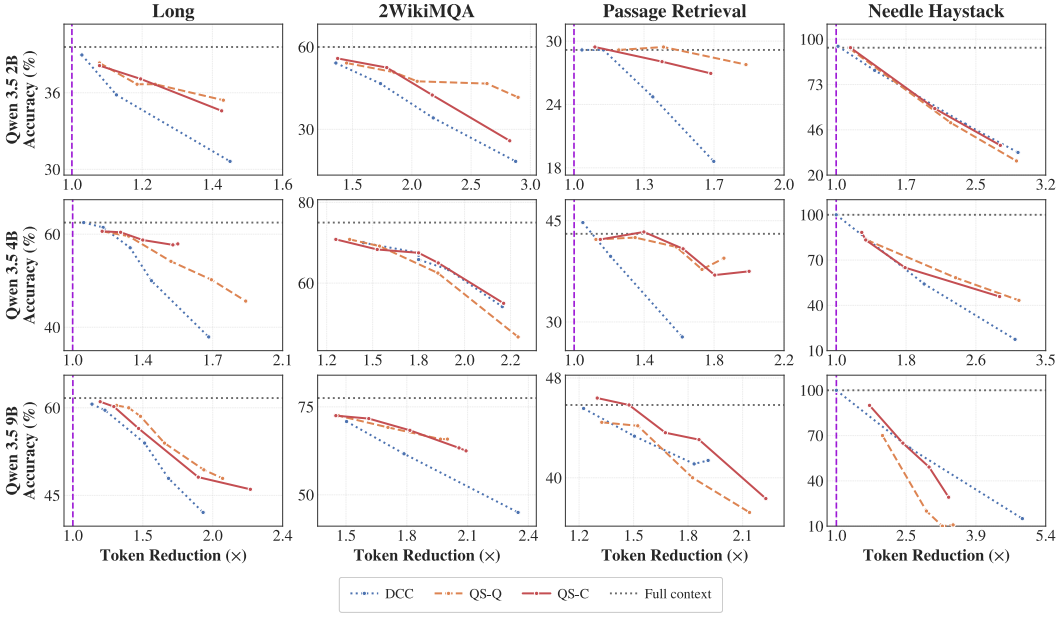
4.1 Results

Sufficiency Prediction Superiority. Table 1 demonstrates that explicitly routing sufficiency predictions through query-focused attention features (QUERYSTOP-Q) drastically improves boundary detection across tasks of varying dataset complexity. QUERYSTOP-Q consistently outperforms the DCC baseline and largely eclipses the self-prompting (SP) methods across all model scales. We also note that the results indicate that supplementing these query-focused features with global context (QUERYSTOP-C) yields only marginal, fluctuating differences. This illustrates that the global

[1] Preliminary evaluations included Llama 3.2 and Mistral v3 family (Section A.5), but both exhibited low answer accuracy even at full context. While QueryStop consistently predicts sufficiency scores with higher precision (F1 score), we also evaluate whether early stopping preserves full context utility, so these models are excluded for which the full context baseline is not reliable.



(a) Sufficiency Reach Rate (SRR) vs. Token Reduction Factor (RF).



(b) Answer Accuracy vs. Token Reduction Factor (RF).

Figure 3: Utility-efficiency trade-offs. (a) Sufficiency Reach Rate (SRR) measures whether the predicted stopping point reaches the gold sufficiency point. (b) Accuracy measures downstream answer correctness under different Token Reduction Factors (RF).

representation serves merely as an ancillary enhancement, as the query-aligned features alone reliably capture the core latent stopping signal.

Maintaining Utility and Adequacy. We plot the accuracy and SRR at multiple confidence thresholds, and our query-aware representations generally yield superior answer accuracy and SRR across token compression scales (Figures 3b and 3a). Table 2 further quantifies this at aligned Token Reduction Factor (T.RF) operating points, where QUERYSTOP-Q outperforms the DCC baseline at similar context savings. For example, on the Needle in Haystack task (Qwen 3.5 4B), an aggressive 3.10× reduction causes DCC to stop prematurely before reading evidence (dropping to 55.8% SRR and 17.5% accuracy). At a higher token reduction factor of 3.14×, QUERYSTOP-Q successfully

Table 2: Comparison of accuracy and stopping adequacy (SRR) at similar Token Reduction Factor (T.RF) operating points. Our method obtains better metrics at similar token reduction level.

Model	Method	Long			2WikiMQA			Passage Retrieval			Needle in Haystack		
		T.RF	Acc	SRR	T.RF	Acc	SRR	T.RF	Acc	SRR	T.RF	Acc	SRR
Qwen 3.5 2B	DCC	**1.43**	30.6	91.5	1.74	46.7	86.7	1.14	29.2	97.8	**2.35**	50.0	80.8
	QueryStop-Q	1.41	**35.4**	**99.8**	**1.85**	50.8	90.8	**1.21**	29.2	98.9	2.20	50.8	90.8
	QueryStop-C	1.41	34.6	99.6	1.79	**52.5**	**93.3**	1.10	**29.4**	**99.7**	2.03	**59.2**	**93.3**
Qwen 3.5 4B	DCC	1.40	50.0	95.2	1.45	**70.0**	95.8	1.62	27.8	83.1	3.10	17.5	55.8
	QueryStop-Q	1.50	54.2	97.5	**1.54**	69.2	**96.7**	1.59	**41.1**	96.7	**3.14**	43.3	81.7
	QueryStop-C	**1.53**	**57.9**	**98.8**	1.53	68.3	95.8	**1.63**	40.8	**97.5**	2.92	**45.8**	**87.5**
Qwen 3.5 9B	DCC	1.89	42.1	87.9	1.79	61.7	93.3	1.51	43.3	97.5	**2.53**	61.7	80.0
	QueryStop-Q	**1.90**	**49.4**	**90.4**	1.71	**69.2**	**95.8**	**1.52**	44.2	97.5	1.96	**70.0**	**95.0**
	QueryStop-C	1.86	48.1	87.1	**1.82**	68.3	95.0	1.48	**45.8**	**98.3**	2.39	65.0	90.8

Table 3: Sufficiency prediction ablation study on alternative hidden representation strategies. Our query-aware ensembles (QueryStop-Q, QueryStop-C) consistently outperform baseline aggregation strategies on complex reasoning tasks. **Bold** and underline indicate the best and second-best F1 scores.

Model	Method	Long			2WikiMQA			Passage Retrieval			Needle in Haystack		
		F1	Acc	Rec	F1	Acc	Rec	F1	Acc	Rec	F1	Acc	Rec
Qwen 3.5 2B	Entropy	85.8	83.9	88.3	76.1	76.2	73.4	81.6	80.0	80.6	86.6	85.5	86.7
	Inv-Entropy	63.2	65.5	53.8	66.1	66.9	62.4	68.3	67.2	63.8	77.2	76.4	74.0
	Global-Only (DCC)	89.2	87.6	92.3	80.6	80.1	80.3	83.6	82.1	82.1	92.5	91.9	92.7
	Weighted-Sum	90.6	89.5	92.4	82.5	81.9	82.6	86.6	85.4	85.5	96.2	95.9	96.4
	QR-style Scoring	91.3	90.2	93.5	84.0	83.5	83.9	86.6	85.4	85.2	**99.0**	98.9	99.1
	Query-Only (QueryStop-Q)	91.1	89.9	94.2	**85.5**	84.8	86.3	87.2	86.1	85.3	98.1	97.9	98.5
	Query-Global (QueryStop-C)	**92.3**	91.2	94.6	82.5	82.0	82.3	**87.8**	86.6	87.5	98.0	97.8	98.3
Qwen 3.5 4B	Entropy	91.4	90.2	94.8	80.8	80.2	80.5	83.7	82.2	82.8	92.1	91.4	93.0
	Inv-Entropy	71.2	70.1	67.0	66.7	65.6	66.6	67.2	66.1	63.0	70.9	70.3	66.9
	Global-Only (DCC)	92.4	91.4	94.4	84.5	83.8	85.5	87.6	86.2	87.6	96.0	95.7	95.2
	Weighted-Sum	94.6	94.0	96.5	85.4	84.5	87.4	90.0	88.8	90.6	99.5	99.5	99.1
	QR-style Scoring	96.1	95.7	97.0	84.6	84.0	85.2	90.1	89.2	88.5	99.8	99.7	99.7
	Query-Only (QueryStop-Q)	**96.5**	96.0	98.0	**87.6**	86.9	89.2	89.8	88.9	88.7	**99.8**	99.7	99.7
	Query-Global (QueryStop-C)	96.4	96.0	97.7	85.7	84.8	88.4	**90.7**	89.8	90.2	99.7	99.7	99.5
Qwen 3.5 9B	Entropy	91.6	90.5	93.5	84.8	83.5	89.2	85.2	83.9	83.7	92.1	91.2	95.2
	Inv-Entropy	75.6	74.1	72.8	70.7	69.4	71.5	70.8	69.5	66.9	72.3	71.7	68.4
	Global-Only (DCC)	92.6	91.7	94.2	87.6	86.8	90.3	88.1	86.8	88.2	95.7	95.3	95.7
	Weighted-Sum	95.8	95.3	96.8	89.7	89.2	91.1	90.6	89.7	90.1	98.8	98.7	98.0
	QR-style Scoring	95.9	95.4	97.1	89.4	89.0	89.4	**91.7**	90.8	91.9	**99.5**	99.4	99.8
	Query-Only (QueryStop-Q)	**96.2**	95.8	97.5	90.0	89.6	91.0	91.4	90.6	90.6	99.4	99.3	99.2
	Query-Global (QueryStop-C)	96.2	95.7	97.7	**90.2**	89.8	90.6	91.2	90.4	90.0	99.0	98.9	98.9

prevents this unsafe early stopping, maintaining an 81.7% SRR and boosting downstream accuracy by nearly 2.5× (43.3%). Consistent with our binary classification findings, appending global context (QUERYSTOP-C) offers comparable but not strictly necessary downstream utility. We also note that prompt-based baselines (SP-M, SP-S) are highly unstable. SP-M's $\mathcal{O}(k^2)$ read complexity frequently causes net computational waste (T.RF $< 1.0\times$), while SP-S is sensitive to prompt design and suffers from either overly aggressive or conservative stopping across different model sizes. To maintain clarity in the main figures, detailed results for these methods are deferred to Appendix A.4.

5 Ablation Studies

Hidden Representations. To isolate the effectiveness and efficiency of our query-focused method, we compare QUERYSTOP with several alternative hidden representation strategies (Table 3). We first consider **Token Entropy**, which has been used to regulate overthinking [20, 10, 7] and predict output length [44]. Applying entropy to latent sufficiency, however, introduces a theoretical ambiguity: should the representation prioritize tokens with high entropy (indicating uncertainty and high information gain) or low entropy (indicating high confidence and stability)? We therefore evaluate both standard Entropy-weighted and Inverse-Entropy-weighted variants. Second, we evaluate a **Weighted-Sum** aggregation as an alternative to our concatenated approach. This combines the query-focused and global hidden states via $\gamma \cdot x_i^q + (1-\gamma) \cdot x_i^g$, where we set $\gamma = 0.5$. Finally, we benchmark against **QR-style Scoring** [50], an alternative query-scoring strategy adapted to our sufficiency-prediction setting. This strategy uses a finer-grained, per-query-token operation: it applies softmax separately for each query token and then averages the resulting query-to-context attention weights, albeit at a higher runtime cost (Table 7). In contrast, QUERYSTOP averages query–key scores before softmax, enabling computation from a cached mean query vector.

As shown in Table 3, standard Entropy consistently outperforms Inverse-Entropy across all model scales and datasets, suggesting that tokens yielding high information gain are far more indicative of sufficiency than high-confidence tokens. Nevertheless, both entropy-based baselines fall short of

Table 4: Threshold stability metrics averaged across datasets. **Accuracy Sensitivity (S_{acc}) $\downarrow$**: the local rate of change in downstream accuracy under adjacent threshold perturbations; lower values indicate that answer quality is less dependent on precise threshold tuning. **T.RF Sensitivity (S_{TRF}) $\downarrow$**: the corresponding local variation in token reduction, capturing whether small threshold changes lead to abrupt shifts in computational cost. **Stable Width (W_ϵ) $\uparrow$**: the width of the threshold interval over which accuracy remains within ϵ of the method's best observed accuracy. $W_{.05}$ and $W_{.10}$ correspond to 5% and 10% accuracy tolerances, respectively. Larger values indicate a wider reliable operating region and therefore a lower tuning burden.

Method	Qwen 3.5 2B				Qwen 3.5 4B				Qwen 3.5 9B			
	$S_{\mathrm{acc}}\downarrow$	$S_{\mathrm{TRF}}\downarrow$	$W_{.05}\uparrow$	$W_{.10}\uparrow$	$S_{\mathrm{acc}}\downarrow$	$S_{\mathrm{TRF}}\downarrow$	$W_{.05}\uparrow$	$W_{.10}\uparrow$	$S_{\mathrm{acc}}\downarrow$	$S_{\mathrm{TRF}}\downarrow$	$W_{.05}\uparrow$	$W_{.10}\uparrow$
DCC	1.10	5.01	0.11	0.14	2.25	6.94	0.10	0.14	3.46	15.07	0.13	0.18
QueryStop-Q	**0.72**	**3.21**	**0.17**	**0.23**	1.05	4.49	0.09	0.13	**0.80**	**2.92**	**0.13**	**0.25**
QueryStop-C	0.78	3.80	0.15	0.17	**0.83**	**3.29**	**0.16**	**0.22**	0.98	3.30	**0.13**	0.23

Table 5: Effect of checkpoint granularity on Long using Qwen 3.5-4B QueryStop-Q. Classification metrics are percentages; test chunks are cumulative prefixes.

Segmentation	Test Chunks	Acc.	Prec.	Rec.	F1
5 equal-token	2,400	95.8	96.0	97.2	96.6
10 equal-token	4,800	96.0	94.9	98.0	96.5
20 equal-token	9,600	96.2	95.5	97.4	96.5
Natural (≤ 20)	4,164	96.1	95.5	96.3	95.9

query-aware strategies. QR-style Scoring remains competitive and achieves slightly higher F1 in some retrieval-oriented settings, while QueryStop-Q and QueryStop-C achieve higher F1 on Long and 2WikiMQA. Accounting for the additional runtime cost, these findings indicate that averaging query–key scores before softmax provides an effective efficiency–accuracy trade-off for sufficiency prediction. Further discussion of our QUERYSTOP heads and QR-style Scoring is in Section A.2.

Checkpoint Granularity. To assess sensitivity to segmentation, we independently reselect heads and retrain the Qwen 3.5-4B QueryStop-Q probe on Long using 5, 10, or 20 equal-token chunks on 2,400 total examples, as well as natural boundaries: paragraphs/documents for QA, records for KV retrieval, and logical blocks for code. Natural segmentation is capped at 20 checkpoints and yields 8.7 test checkpoints per example on average. Across these settings, F1 remains between 95.9% and 96.6% (Table 5), suggesting that sufficiency classification is stable across the evaluated checkpoint granularities and boundary types.

6 Analysis

Stability of Confidence Thresholds. Dynamic cutoff methods rely on a confidence threshold τ to determine when the observed context is sufficient, thereby controlling the trade-off between token reduction and answer quality. We observe that the baseline DCC stopping methods are often highly sensitive to small changes in τ, which is also a noted limitation in [45]. In Figure 3b, on the Qwen 3.5 9B Needle-in-a-Haystack task, for example, a minor threshold decrease from 0.78 to 0.72 drastically alters both the computational budget and downstream answer quality. As shown in Table 4, DCC becomes increasingly sensitive to threshold selection as model size increases. Formal definitions of the metrics are provided in Section A.1.

Long-Context Compression. Although primarily designed for dynamic in-stream stopping, QUERYSTOP also serves as an effective long-context compression method. We compare it against long context compression baselines (LLMLingua, LLMLingua-2, and LongLLMLingua [8, 9, 27]). Note that these methods demand a determined retained input and lack the capabilities required for streaming applications. Results show QUERYSTOP achieves comparable downstream answer quality and token reduction efficiency. Detailed performance metrics are in Section A.3.

Mechanistic Preference for Query-Aligned Heads. Recent works show that Large Language Models process context through highly specialized attention heads [41, 50, 42]. We also investigate the routing preferences of our hybrid QueryStop-C method. Recall QueryStop-C autonomously weighs the relative importance of query-aligned versus global-context features; hence, it naturally

Table 6: Mechanistic overlap of the top $K_h = 5$ attention heads selected by the hybrid QueryStop-C method. Values indicate the number of QueryStop-C selected heads that overlap with those chosen independently by the pure Query method (QueryStop-Q) and the Global baseline (DCC). "Both" indicates heads that were highly ranked by all three approaches. The ensemble consistently favors Query-aligned heads.

Model	Long			2Wikimqa			Passage Retrieval			Needle Haystack			Average		
	Query	Global	Both	Query	Global	Both	Query	Global	Both	Query	Global	Both	Query	Global	Both
Qwen 3.5 2B	5	3	3	3	4	3	4	3	3	5	4	4	**4.25**	3.50	3.25
Qwen 3.5 4B	4	2	2	4	3	2	4	3	3	4	3	3	**4.00**	2.75	2.50
Qwen 3.5 9B	5	3	3	4	1	1	4	3	2	5	2	2	**4.50**	2.25	2.00
Average	4.67	2.67	2.67	3.67	2.67	2.00	4.00	3.00	2.67	4.67	3.00	3.00	**4.25**	2.83	2.58

Table 7: End-to-end runtime and memory usage on Long using Qwen 3.5-4B. Memory denotes peak allocated GPU memory.

Method	Latency (s) Median / p95	Requests/s	Memory (GiB)
Full context	9.94 / 19.07	0.079	18.76
DCC	8.98 / 17.52	0.099	14.14
QueryStop-Q	9.29 / 17.54	0.095	15.40
QR-style Scoring	16.01 / 21.66	0.065	16.00

selects the most effective overall ensemble. We analyze the top $K_h = 5$ attention heads chosen by QueryStop-C and calculate their overlap with the independent selections of the pure query method (QueryStop-Q) and the global baseline (DCC). As detailed in Table 6, QueryStop-C exhibits a consistent, overwhelming preference for query-aligned attention heads across all datasets and model scales. On average, 4.25 out of the 5 heads selected by QueryStop-C overlap with the pure query method, compared to only 2.83 for the global baseline. This suggests that determining context sufficiency is essentially a query-dependent routing task, not a global semantic property. When the hybrid QueryStop-C model is allowed to freely weight all representations, it naturally converges on these specialized, query-aware heads to extract informative stopping features, actively discarding the majority of noisy, unspecialized global representations.

Runtime and Memory Efficiency. We profile Qwen 3.5-4B on Long, measuring end-to-end execution including checkpoint processing, feature extraction, classification, and final answer generation. As shown in Table 7, QueryStop-Q reduces median and p95 latency by approximately 6.5% and 8.0%, respectively, compared with full-context inference. It also reduces peak allocated memory by 17.9% and increases throughput by 20.3%. DCC remains slightly faster and more memory-efficient, while QueryStop-Q provides the stronger sufficiency prediction and downstream answer quality.

We additionally compare against QR-style Scoring, which incorporates finer-grained, per-query-token post-softmax query-to-context scoring within our sufficiency-prediction pipeline. Its higher latency illustrates the computational cost of finer-grained scoring relative to QueryStop-Q's aggregation.

7 Conclusion

We present QueryStop, a query-focused probing method for dynamic stopping in long-context inference. Our results and analysis show that our query-focused attention heads provide a strong and dominant signal for detecting when answer-relevant evidence has been reached. Across four benchmark groups and eleven datasets, QUERYSTOP improves stop-signal classification and yields better accuracy-efficiency trade-offs than prior probing and self-prompting baselines. These findings suggest that query-conditioned internal features offer a practical and efficient signal for reducing unnecessary context processing while preserving answer quality.

Acknowledgments and Disclosure of Funding

This work was conducted during the first author's paid research internship at SmartMore, which also provided computational resources for the experiments.

References

[1] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024.

[2] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do not think that much for 2+3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2025.

[3] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, volume 36, pages 28091–28114, 2023.

[4] In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt cache: Modular attention reuse for low-latency inference. In *Proceedings of Machine Learning and Systems*, 2024.

[5] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, 2020.

[6] Parsa Hosseini, Sumit Nawathe, Mahdi Salmani, Meisam Razaviyayn, and Soheil Feizi. Early stopping for large reasoning models via confidence dynamics. *arXiv preprint arXiv:2604.04930*, 2026.

[7] Jiameng Huang, Baijiong Lin, Guhao Feng, Jierun Chen, Di He, and Lu Hou. Efficient reasoning for large reasoning language models via certainty-guided reflection suppression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[8] Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376. Association for Computational Linguistics, 2023.

[9] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 1658–1677. Association for Computational Linguistics, 2024.

[10] Tianyi Jiang, Yi Bin, Yujuan Ding, Kainian Zhu, Fei Ma, Jingkuan Song, and Heng Tao Shen. Explore briefly, then decide: Mitigating llm overthinking via cumulative entropy regulation. *arXiv preprint arXiv:2510.02249*, 2025.

[11] Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992. Association for Computational Linguistics, 2023.

[12] Greg Kamradt. Needle in a haystack: Pressure testing llms. `https://github.com/gkamradt/LLMTest_NeedleInAHaystack`, 2023. GitHub repository.

[13] Łukasz Kobyliński, Maciej Ogrodniczuk, Piotr Rybak, Piotr Przybyła, Piotr Pęzik, Agnieszka Mikołajczyk, Wojciech Janowski, Michał Marcińczuk, and Aleksander Smywiński-Pohl. Poleval 2022/23 challenge tasks and results. In *Proceedings of the 18th Conference on Computer Science and Intelligence Systems*, volume 35 of *Annals of Computer Science and Information Systems*, pages 1243–1250, 2023.

[14] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[15] Yassir Laaouach. HALT-CoT: Model-agnostic early stopping for chain-of-thought reasoning via answer entropy. In *ICML 2025 Workshop on Models of Human Feedback for AI Alignment*, 2025.

[16] Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. Remarkable robustness of llms: Stages of inference? *Advances in Neural Information Processing Systems*, 38:130050–130083, 2026.

[17] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474, 2020.

[18] Jingyao Li, Han Shi, Sitong Wu, Chuanyang Zheng, Pei Chen, Zhenguo Li, Xin Jiang, Hong Xu, and Jiaya Jia. QuickLLaMA: Query-aware inference acceleration for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 508–528. Association for Computational Linguistics, 2025.

[19] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

[20] Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. Making slow thinking faster: Compressing LLM chain-of-thought via step entropy. In *The Fourteenth International Conference on Learning Representations*, 2026.

[21] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[22] Simon Lupart, Mohammad Aliannejadi, and Evangelos Kanoulas. ChatR1: Reinforcement learning for conversational reasoning and retrieval augmented question answering. *arXiv preprint arXiv:2510.13312*, 2025.

[23] Youmi Ma and Naoaki Okazaki. From interpretability to performance: Optimizing retrieval heads for long-context language models. *arXiv preprint arXiv:2601.11020*, 2026.

[24] Minjia Mao, Bowen Yin, Yu Zhu, and Xiao Fang. Early stopping chain-of-thoughts in large language models. *arXiv preprint arXiv:2509.14004*, 2025.

[25] Gary Marchionini. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.

[26] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[27] Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. *arXiv preprint arXiv:2403.12968*, 2024.

[28] Jaewan Park, Solbee Cho, and Jay-Yoon Lee. Stop-RAG: Value-based retrieval control for iterative RAG. *arXiv preprint arXiv:2510.14337*, 2025.

[29] Rongcan Pei, Huan Li, Fang Guo, and Qi Zhu. VERA: Identifying and leveraging visual evidence retrieval heads in long-context understanding. *arXiv preprint arXiv:2602.10146*, 2026.

[30] Peter Pirolli and Stuart Card. Information foraging. *Psychological review*, 106(4):643, 1999.

[31] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

[32] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551, 2023.

[33] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[34] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

[35] Rei Taniguchi, Yuyang Dong, Makoto Onizuka, and Chuan Xiao. Adaptive layer selection for layer-wise token pruning in LLM inference. *arXiv preprint arXiv:2601.07667*, 2026.

[36] Yuxing Tian, Fengran Mo, Zhiqi Huang, Weixu Zhang, and Jian-Yun Nie. Learning to route queries to heads for attention-based re-ranking with large language models. *arXiv preprint arXiv:2604.24608*, 2026.

[37] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[38] Yao Wan, Zhou Zhao, Min Yang, Guandong Xu, Haochao Ying, Jian Wu, and Philip S. Yu. Improving automatic source code summarization via deep reinforcement learning. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pages 397–407, 2018.

[39] Xinyu Wang, Mingze Li, Peng Lu, Xiao-Wen Chang, Lifeng Shang, Jinping Li, Fei Mi, Prasanna Parthasarathi, and Yufei Cui. InfMem: Learning system-2 memory control for long-context agent. *arXiv preprint arXiv:2602.02704*, 2026.

[40] Zheng Wang, Boxiao Jin, Zhongzhi Yu, and Minjia Zhang. Model tells you where to merge: Adaptive kv cache merging for llms on long-context tasks. *arXiv preprint arXiv:2407.08454*, 2024.

[41] Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *International Conference on Learning Representations*, 2025.

[42] Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. DuoAttention: Efficient long-context LLM inference with retrieval and streaming heads. In *International Conference on Learning Representations*, 2025.

[43] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

[44] Huanyi Xie, Yubin Chen, Liangyu Wang, Lijie Hu, and Di Wang. Predicting LLM output length via entropy-guided representations. In *The Fourteenth International Conference on Learning Representations*, 2026.

[45] Roy Xie, Junlin Wang, Paul Rosu, Chunyuan Deng, Bolun Sun, Zihao Lin, and Bhuwan Dhingra. Knowing when to stop: Efficient context processing via latent sufficiency signals. *Advances in Neural Information Processing Systems*, 2025.

[46] Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*, 2024.

[47] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018.

[48] Shunyu Yao, Jeffrey Zhao, Dian Yu, et al. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

[49] Yilun Yao, Shan Huang, Elsie Dai, Zhewen Tan, Zhenyu Duan, Shousheng Jia, Yanbing Jiang, and Tong Yang. ARC: Active and reflection-driven context management for long-horizon information seeking agents. *arXiv preprint arXiv:2601.12030*, 2026.

[50] Wuwei Zhang, Fangcong Yin, Howard Yen, Danqi Chen, and Xi Ye. Query-focused retrieval heads improve long-context reasoning and re-ranking. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 23791–23805, Suzhou, China, 2025. Association for Computational Linguistics.

[51] Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ∞Bench: Extending long context evaluation beyond 100k tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 15262–15277, 2024.

[52] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[53] Yiwu Zhong, Zhuoming Liu, Yin Li, and Liwei Wang. Aim: Adaptive inference of multi-modal llms via token merging and pruning. *arXiv preprint arXiv:2412.03248*, 2024.

[54] Shuyan Zhou, Frank F. Xu, Hao Zhu, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

8 Limitations and Future Work.

While QUERYSTOP demonstrates significant improvements in dynamic context management, we acknowledge a few boundary conditions. First, while it aligns with the vast majority of real-world pipelines, our method assumes a fixed user query during the streaming phase, as it relies on computing a query representation $\bar{Q}$ prior to processing the context stream. Like all dynamic cutoff mechanisms, QUERYSTOP requires defining a confidence threshold τ to navigate the speed-accuracy tradeoff. Nevertheless, as evidenced by our confidence threshold measurement across multiple benchmarks, QUERYSTOP degrades far more gracefully than existing alternatives, providing more stable and safe operating points regardless of the user's specific efficiency requirements.

Future Work. While this study establishes a foundational mechanism for latent sufficiency-boundary detection, it represents only a first step toward optimizing streaming context management. Since QUERYSTOP derives its signal from internal states already available during inference, it is highly modular and can be combined with orthogonal long-context efficiency techniques. A promising direction is to integrate dynamic stopping with static prompt/token compression methods such as LLMLingua, LongLLMLingua, and LLMLingua-2 [8, 9, 27], KV-cache retention or eviction methods such as StreamingLLM, H_2O, and SnapKV [43, 52, 19], and prompt-caching systems that reuse attention states across prompts [4]. We hypothesize that combining stream-aware truncation with these complementary paradigms can yield compounding computational savings, enabling lower-latency and more resource-efficient retrieval and streaming architectures.

Appendix

A Ablation Details **17**

A.1 Formal Definitions of Stability Metrics 17

A.2 QueryStop Heads against the broader family of specialized attention heads 17

A.3 QueryStop as a Long Context Efficiency Method 18

A.4 Self Prompting Downstream Results 19

A.5 Model Coverage and Additional Evaluation 19

A.6 Annotated Evidence and Model-Specific Answerability 20

B Dataset Splits, Model Selection, and Evaluation Protocol **20**

B.1 Train/Validation/Test Construction 20

B.2 Selection Pipeline and Data Usage 21

B.3 Probe Head Selection 21

B.4 Classifier Family and Ensemble Selection 22

B.5 Threshold Selection 22

B.6 Final Evaluation 22

C Prompt Templates Used in the Pipeline **23**

C.1 Standard Probe-Based Pipeline 23

C.2 Self-Prompting Baselines 23

C.3 Self-Prompt Variant Prompts 25

C.4 Downstream Answer Evaluation Prompt 25

D GPU Requirements **26**

A Ablation Details

A.1 Formal Definitions of Stability Metrics

To systematically quantify the brittleness of dynamic cutoff methods across our benchmark suite, we formally define the three stability metrics evaluated in Section 6:

- **Accuracy Sensitivity** (S_{acc}) $\downarrow$: Measures the local rate of change in downstream task accuracy relative to confidence threshold perturbations. It is calculated as the expected value of the adjacent gradient:

$$S_{acc} = \mathbb{E}\left[\frac{|\Delta \text{Accuracy}|}{|\Delta \tau|}\right] \tag{3}$$

  A lower value indicates that answer quality remains stable even if the threshold τ is imperfectly tuned.

- **T.RF Sensitivity** (S_{TRF}) $\downarrow$: Measures the volatility of the computational budget. It is defined similarly to capture the local rate of change in the Token Reduction Factor:

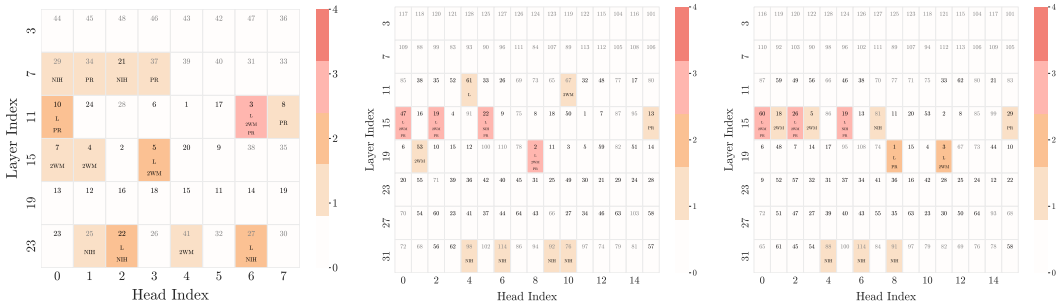
$$S_{TRF} = \mathbb{E}\left[\frac{|\Delta \text{T.RF}|}{|\Delta \tau|}\right] \tag{4}$$

  A lower value ensures that minor threshold adjustments do not cause massive, unpredictable spikes in token processing.

- **Stable Width** (W_ϵ) $\uparrow$: Quantifies the continuous range of thresholds that yield an accuracy within a small margin ϵ of the absolute peak performance. Let the set of stable thresholds be $T_\epsilon = \{\tau : \text{Accuracy}(\tau) \geq \max(\text{Accuracy}) - \epsilon\}$. The stable width is defined as:

$$W_\epsilon = \max(T_\epsilon) - \min(T_\epsilon) \tag{5}$$

  A wider range implies the method is highly robust and requires minimal task-specific hyperparameter tuning.



(a) Qwen 3.5 2B (b) Qwen 3.5 4B (c) Qwen 3.5 9B

Figure 4: Layer-head frequency heatmaps of selected QueryStop heads (QS-Q) across the evaluation benchmarks. The x-axis represents the attention head index and the y-axis represents the network layer. Bold numbers overlaid within the active clusters indicate the absolute ranking of top Query-Focused Retrieval Heads (QRHeads) identified using the methodology of Zhang et al. [50], illustrating the structural overlap between dynamic stopping and retrieval mechanisms.

A.2 QueryStop Heads against the broader family of specialized attention heads

Are QueryStop Heads Inherently "Query Retrieval" Heads? Given this strong preference for query-aligned features, we investigate how our selected QueryStop heads relate to Query-Focused Retrieval Heads (QRHeads), a sparse subset of attention heads known to dominate long-context information extraction [50]. QRHeads are identified using query-to-context attention scores, which mirror our representation weighting strategy, but with two differences. First, *computational efficiency*: our method caches the mean query vector to accelerate pre-softmax scoring during streaming inference, whereas QRHeads scoring require exact post-softmax token-level attention mass, which captures finer

token-specific information at a higher computational cost. Second, *downstream identification*: QRHeads are explicitly selected via their performance on ground-truth document retrieval tasks, whereas QueryStop heads are selected purely for their utility in predicting context sufficiency boundaries.

To investigate the relationship between these mechanisms, we replicated the QRHead identification strategy on the standard attention layers of the Qwen 3.5 family. We then overlaid the absolute rankings of the identified QRHeads onto our QueryStop frequency heatmaps (Figure 4), aggregating the attention heads selected across our four benchmarks. This analysis yielded two primary discoveries:

1) Sufficiency Evaluation Transcends Pure Retrieval: While top-ranked QRHeads (e.g., ranks 1, 3, and 5 in the 9B model) land squarely within the dominant QueryStop clusters, the overlap is not absolute. As established in the literature, retrieval heads constitute a very sparse subset (approximately 3-5%) of total attention heads [50]. However, several top-tier QRHeads (e.g., ranks 2 and 4) are completely ignored by the QueryStop ensemble. We also see the majority of QueryStop heads for non-NIH datasets fall within the top 50% of QR-scored heads, but outside the top 5%. This partial overlap mechanistically demonstrates that predicting the sufficiency stopping point is a more complex operation than pure retrieval. While the StopSignal task clearly utilizes retrieval pathways as a core routing component, it also heavily relies on distinct, non-retrieval attention head roles to evaluate whether the retrieved information is actually sufficient to construct a complete answer.

2) Task-Specific Routing Across Network Depths: QueryStop head selection itself also exhibits distinct, task-specific layer localization. For complex, multi-hop reasoning tasks such as Long, Passage Retrieval, and 2WikiMQA, which require gathering and synthesizing evidence across multiple text segments, the selected QueryStop heads cluster heavily in the middle layers of the network. This mirrors findings demonstrating that multi-step reasoning and contextual synthesis predominantly occur within the middle of Transformer architectures [46]. Conversely, for the Needle-in-a-Haystack (NIH) task, which requires pinpointing a single, explicit fact against a massive background of pure noise, the QueryStop heads localize almost entirely in the deepest attention layers of the network. This supports recent studies on the stages of LLM inference [16, 35], which propose a framework where the final, deepest layers of the network are heavily specialized for exact token extraction and final answer preparation, functioning separately from the robust, multi-pathway reasoning that occurs in the middle. This suggests that determining context sufficiency is not a monolith; models dynamically route sufficiency signals to different depths depending on the cognitive complexity of the task.

A.3 QueryStop as a Long Context Efficiency Method

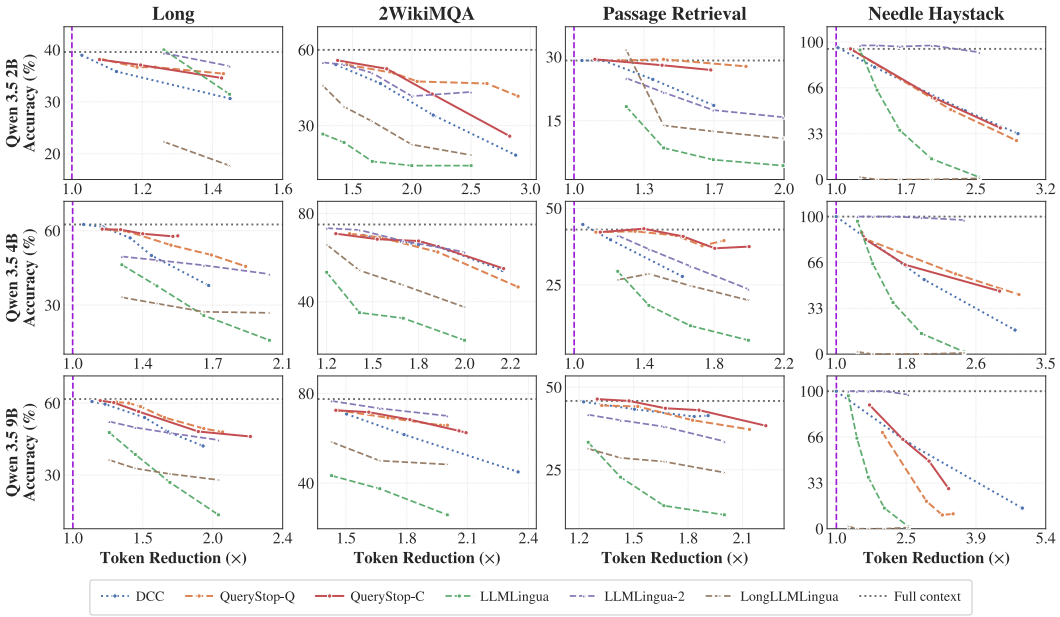


Figure 5: **Accuracy vs. Token R.F** compared to long context compression methods

While the primary objective of QUERYSTOP is to provide a dynamic stop signal for streaming inputs, the underlying query-aware representations can also be naturally adapted for general long-context compression. To evaluate its efficacy in this orthogonal task, we benchmark QUERYSTOP against state-of-the-art static context compression models, namely LLMLingua, LLMLingua-2, and LongLLMLingua [8, 27, 9], reporting downstream answer accuracy as a function of the Token Reduction Factor (T.RF).

Note that the Lingua models reduce token footprints by compressing an entire, fully instantiated long context simultaneously. Because they lack a dynamic truncator, they cannot evaluate continuous token streams or predict latent sufficiency points, making them structurally inapplicable to our primary streaming problem. Nevertheless, when evaluated purely on the orthogonal task of static context compression, we find that QUERYSTOP achieves highly competitive downstream answer quality across all four benchmarks, effectively matching the performance of dedicated compression baselines without requiring specialized compression training.

A.4 Self Prompting Downstream Results

Table 8: Accuracy, Sufficiency Reach Rate (SRR), and Token Reduction Factor (Token R.F) for Qwen 3.5 self-prompting variants. Accuracy and SRR are reported as percentages. Token R.F is computed as $\mathrm{mean}(\mathrm{total_chunks}/\mathrm{read_chunks})$; for SP-M, $\mathrm{read_chunks} = k(k+1)/2$, where k is the stopping chunk, and for SP-S, $\mathrm{read_chunks} = k$.

Dataset	Method	Qwen 3.5 2B			Qwen 3.5 4B			Qwen 3.5 9B		
		Acc	SRR	T.RF	Acc	SRR	T.RF	Acc	SRR	Token R.F
Long	SP-M	34.0	84.8	0.58	51.0	97.3	0.77	51.2	77.7	1.60
	SP-S	16.5	77.7	2.98	24.4	99.0	1.25	21.7	84.6	1.98
Passage Retrieval	SP-M	19.2	90.0	0.75	40.3	96.9	1.77	38.6	97.5	1.81
	SP-S	5.6	88.3	1.81	10.8	97.8	1.73	7.2	82.5	2.23
2WikiMQA	SP-M	48.3	84.2	1.38	61.7	90.8	1.20	67.5	87.5	0.78
	SP-S	14.2	85.8	2.74	14.2	90.0	2.42	12.5	40.8	4.10
Needle Haystack	SP-M	43.3	45.8	3.33	68.3	83.3	1.10	85.8	91.7	0.95
	SP-S	20.8	94.2	1.42	4.2	25.0	7.96	9.2	30.8	4.96

Table 8 provides a more detailed view of the downstream behavior of the self-prompting baselines across model scales and tasks. The multi-round variant (SP-M) often incurs substantial computational overhead because its repeated evaluations lead to cumulative reading cost, producing Token Reduction Factors (T.RF) below 1.0 in some settings (e.g., 0.58 on Long for Qwen 2B and 0.78 on 2WikiMQA for 9B). This suggests that, despite its stronger stopping judgments in certain cases, SP-M may reduce inference efficiency relative to standard full-context processing. In contrast, the single-round variant (SP-S) avoids this cumulative overhead and can achieve higher T.RF, but its downstream performance is less stable. For example, on Needle in Haystack with Qwen 4B, SP-S reaches a 7.96$\times$ T.RF but obtains only 25.0% SRR and 4.2% accuracy, indicating that its early stopping decisions can be overly aggressive. Overall, these results suggest that prompt-based stopping methods struggle to consistently balance efficiency and utility across settings.

A.5 Model Coverage and Additional Evaluation

Our main evaluation focuses on the Qwen 3.5 family at three model scales: 2B, 4B, and 9B. This choice enables controlled comparisons across model scales while maintaining reliable full-context downstream performance. Preliminary evaluations also included models from the Llama 3.2 and Mistral v3 families, but these models exhibited low downstream answer accuracy even when provided with the full context. Because our evaluation examines both sufficiency prediction and the preservation of full-context answer quality under early stopping, low full-context accuracy limits the informativeness of downstream utility comparisons. We therefore exclude these models from the main evaluation and report their preliminary results here for completeness.

Table 9 presents sufficiency prediction F1 and downstream answer accuracy on the Long benchmark for Llama-3.2-1B, Ministral-3-3B, and an additional model, Qwen2.5-14B. Both QueryStop variants consistently improve sufficiency prediction F1 over DCC across all three models. This indicates that

Table 9: Preliminary evaluation on additional models using the Long benchmark. Sufficiency prediction F1 and downstream answer accuracy are reported as percentages. Boldface indicates the best result within each model and metric group, including ties.

	Sufficiency F1 (%)			Answer Accuracy (%)		
Model	DCC	QueryStop-Q	QueryStop-C	DCC	QueryStop-Q	QueryStop-C
Llama-3.2-1B	85.1	**87.4**	87.2	3.5	5.0	**5.6**
Qwen2.5-14B	92.0	**97.9**	97.4	29.5	**31.7**	29.9
Ministral-3-3B	89.8	94.2	**94.3**	**4.0**	3.5	**4.0**

Table 10: Agreement between the annotation-based evidence boundary and the earliest gold-correct checkpoint on Long ($n = 480$). Exact denotes matching checkpoints; Earlier and Later indicate that the first gold-correct answer occurs before or after the annotation-based boundary, respectively. All percentages use the full evaluation set as the denominator, except Exact (answerable), which is restricted to answerable examples.

Exact (all)	Exact (answerable)	Earlier	Later	Correct at annotated boundary
43.5%	70.4%	8.8%	9.6%	52.1%

the benefit of query-aware representations extends beyond the Qwen 3.5 models used in the main evaluation, including settings where downstream answer quality is limited.

Downstream accuracy remains low for Llama-3.2-1B and Ministral-3-3B across stopping methods, consistent with their poor full-context performance observed in the preliminary evaluation. Although QueryStop improves sufficiency prediction on these models, this improvement does not necessarily overcome limitations in their underlying ability to answer the benchmark questions. These results motivate our focus on models with reliable full-context performance when assessing whether early stopping preserves downstream utility.

A.6 Annotated Evidence and Model-Specific Answerability

To examine the relationship between annotated evidence completion and model-specific answerability, we generate answers at all ten cumulative checkpoints for Qwen 3.5-4B on 480 Long examples. We compare the earliest checkpoint containing the complete annotated evidence chain with the first checkpoint producing a gold-correct answer. An example is considered answerable in this analysis if at least one evaluated checkpoint produces a gold-correct answer.

Among answerable examples, the annotation-based boundary matches the earliest gold-correct checkpoint in 70.4% of cases (Table 10). Earlier correct answers may arise from redundant evidence, parametric knowledge, guessing, or alternative reasoning paths. Conversely, reasoning or generation errors may prevent a correct answer even after the annotated evidence is complete. In other words, the benchmark label for "enough information has been observed" does not always match when the model actually has enough information to answer correctly. However, this is a limitation of the supervision formulation rather than evidence that QueryStop underperforms an existing method.

B Dataset Splits, Model Selection, and Evaluation Protocol

This appendix section specifies the data used at each stage of probe construction and evaluation. The key point is that the held-out test split is not used to select heads, choose classifier families, choose the top ensemble members, or tune the stopping threshold. Test data is used only for final reporting.

B.1 Train/Validation/Test Construction

Each benchmark group is represented as one or more task files, where each file contains raw examples with a query, context, answer field, and gold evidence annotations. In the main experiments, the benchmark groups are:

- **Long**: `code`, `multihop_kv`, `multihop_qa`, and `singlehop_qa`.
- **Passage Retrieval**: `trivia`, `allegro`, and `legal`.
- **2WikiMQA**: `2wikimqa`.
- **Needle-in-a-Haystack**: `needle_haystack`.

For each task, we sample 600 raw examples, the size of **Long** in [45]. Raw examples are split before any chunk-level examples are constructed. We first allocate 80% of raw examples to a development split and 20% to a held-out test split. The development split is then split again into 80% training and 20% validation. Therefore, the effective raw-example split is:

$$\text{train} : 65\%, \qquad \text{validation} : 15\%, \qquad \text{test} : 20\%.$$

After the raw-example split, each context is converted into cumulative context prefixes. In the main setup, we use percentage-based chunking with 10 chunks per context, corresponding to prefixes at 10%, 20%, ..., 100% of the context. Each prefix becomes one binary sufficiency example. A prefix is labeled sufficient if its endpoint reaches or passes the end position of the annotated gold evidence; otherwise it is labeled insufficient. Because chunking happens after the raw-example split, prefixes derived from the same raw example cannot appear in different train, validation, and test splits.

For benchmark groups with multiple task files, task representation is balanced after chunking. For each split, we find the minimum number of chunk-level examples across tasks and retain that many examples per task. Training and validation chunks are randomly sampled and shuffled; test chunks are retained in a fixed task-local order up to the shared minimum count. This prevents larger task files from dominating probe training or classifier selection.

B.2 Selection Pipeline and Data Usage

Table 11 summarizes which split is used for each selection or evaluation step. No row uses test performance for model selection.

Table 11: Data usage for probe selection, classifier selection, threshold selection, and final evaluation. The held-out test split is reserved for final evaluation and is not used for selecting heads, classifiers, ensemble members, or threshold values.

Stage	Data used	Criterion
Select top attention heads	Train + validation	Train one logistic probe per head on train; rank heads by validation F1.
Choose among base classifier families	Training split only	Stratified 5-fold cross-validation on train; rank by mean CV accuracy.
Select top 4 ensemble members	Training split only	Select the four highest-ranked base classifier families from training CV.
Fit final soft-voting ensemble	Training split only	Refit selected ensemble members on all training examples.
Report validation diagnostics	Validation split	Validation accuracy and AUC are diagnostic only, not test reporting.
Tune threshold τ, when tuning is enabled	Validation split only	Choose τ from validation probabilities, e.g., best validation F1 or target precision.
Default reported sufficiency classification	Test split	Evaluate the fixed probe, fixed ensemble, and fixed τ.
Downstream early-stopping evaluation	Test split	Report task accuracy, token reduction, and SRR for fixed or pre-specified thresholds.

B.3 Probe Head Selection

Probe selection is performed at the attention-head level. For each candidate head, we extract the configured head feature from every training and validation prefix.

Each candidate head is evaluated independently with a lightweight logistic regression probe. The probe is trained on the training split and evaluated on the validation split. Heads are ranked by

validation F1, and the top k heads are selected for the final sufficiency predictor. In the main experiments, we use $k = 5$. The test split is not used in this step. The selected heads are fixed after this validation-based selection step. The experiment output records the selected layer and head indices and their validation F1 scores.

B.4 Classifier Family and Ensemble Selection

After the top heads are selected, the feature vector for a prefix is formed by concatenating feature vectors from the selected heads. We then train a binary classifier ensemble to predict whether the current prefix is sufficient.

The current implementation has seven base classifier families. The classifier chart is shown in Table 12. The eighth row is the final ensemble, not an additional base classifier eligible for top-4 selection. Thus, top-4 ensemble members are selected from the seven base classifier families.

Table 12: Classifier families used by the sufficiency probe. The first seven rows are base classifier candidates. The final row is the soft-voting ensemble constructed after selecting the top four base classifiers by training-set cross-validation.

Name	Main hyperparameters	Training objective / loss
`xgboost`	`n_estimators=200`, `learning_rate=0.05`, `max_depth=6`, `gamma=0.1`, `subsample=0.8`, `colsample_bytree=0.8`, `scale_pos_weight=2`, `eval_metric="auc"`	XGBoost binary logistic objective, evaluated by AUC.
`lightgbm`	`n_estimators=200`, `learning_rate=0.05`, `max_depth=6`, `num_leaves=31`, `feature_fraction=0.8`, `bagging_fraction=0.8`, `bagging_freq=5`, `class_weight="balanced"`, `objective="binary"`, `metric="auc"`	Binary log-loss objective, evaluated by AUC.
`catboost`	`iterations=200`, `depth=6`, `learning_rate=0.05`, `l2_leaf_reg=3`, `border_count=128`, `bagging_temperature=1`, `random_strength=1`, `eval_metric="AUC"`	CatBoost binary classification default, effectively log loss unless overridden; evaluated by AUC.
`gradient_boosting`	`n_estimators=200`, `learning_rate=0.05`, `max_depth=6`, `min_samples_split=5`, `min_samples_leaf=2`, `subsample=0.8`, `max_features="sqrt"`	sklearn `GradientBoostingClassifier` default `log_loss`.
`mlp`	Hidden layers $(128, 64, 32)$, ReLU, `alpha=0.0001`, adaptive learning rate, `max_iter=3000`, `early_stopping=True`, `validation_fraction=0.1`	Neural-network cross-entropy / log loss with L2 regularization via `alpha`.
`svm`	RBF kernel, `C=1.0`, `gamma="scale"`, `probability=True`, `class_weight="balanced"`, `cache_size=2000`	C-SVC margin objective; probabilities are calibrated afterward.
`logistic`	`C=1.0`, L2 penalty, `solver="lbfgs"`, `max_iter=2000`, `class_weight="balanced"`	Logistic / cross-entropy loss with L2 regularization.
`ensemble`	Soft `VotingClassifier` over the top `num_top_classifiers=4`; weights are descending, e.g., $[1.3, 1.2, 1.1, 1.0]$ for four members.	No independent loss; combines member predicted probabilities.

Feature preprocessing replaces missing values, clips outliers using an interquartile-range rule, and standardizes features. Candidate base classifiers are compared using stratified 5-fold cross-validation on the training split only. The ranking criterion is mean cross-validation accuracy. We select the top four base classifier families by this training-CV score, construct a soft-voting ensemble with descending weights, and fit the final ensemble on the full training split. Test examples are not used for classifier-family ranking, top-4 selection, or ensemble fitting.

B.5 Threshold Selection

The classifier outputs a probability $p_\theta(y = \text{sufficient} \mid x)$. The stopping threshold τ determines when a prefix is predicted sufficient:

$$\hat{y} = 1\left[p_\theta(y = \text{sufficient} \mid x) \geq \tau\right].$$

For the default binary sufficiency evaluation, $\tau = 0.5$ is fixed by configuration and is not selected using test performance. For downstream early-stopping curves, thresholds are preset values evaluated across all methods.

B.6 Final Evaluation

We evaluate the learned stopping signal in two stages. First, we evaluate binary sufficiency prediction on held-out chunk-level test examples. For each prefix, the fixed ensemble outputs a probability that the prefix is sufficient. We report F1, accuracy, precision, recall@90, and confidence statistics

overall and per task. Second, we evaluate downstream early stopping. We report Token Reduction Factor, Sufficiency Reach Rate, and generated-answer accuracy. Token Reduction Factor measures how much context processing is reduced relative to full-context inference. Sufficiency Reach Rate (SRR) measures the percentage of examples for which the predicted stopping point reaches or passes the ground-truth sufficiency position. Answer Accuracy is evaluated using the gpt-5.4-mini model across all generated answers.

C Prompt Templates Used in the Pipeline

This appendix section documents the prompt templates used by the standard probe-based pipeline and by the self-prompting baselines. The standard probe-based method and the self-prompting baselines use different prompts for different purposes. In particular, the probe-based method does not ask the model to verbally decide sufficiency; it extracts internal features from a query–context prompt and uses a learned classifier. By contrast, the self-prompting baselines explicitly ask the model to output `[[YES]]` or `[[NO]]`.

C.1 Standard Probe-Based Pipeline

Query–Context Prompt for Feature Extraction. This prompt is used to compute hidden-state or attention-head features for the sufficiency probe.

```
Please provide a response to the query based only on the given context:

[QUERY]: query

[CONTEXT]: context
```

For Qwen 3.5 models, the same user content is wrapped with the tokenizer's chat template when available, using `add_generation_prompt=True` and `enable_thinking=False`. This changes the model-specific chat serialization but not the user-facing content shown above.

Answer-Generation Suffix. When the probe-based early-stopping pipeline generates a final answer, the retained prefix is followed by the answer suffix:

```
Your answer:
```

Thus the non-chat serialized answer prompt is:

```
Please provide a response to the query based only on the given context:

[QUERY]: query

[CONTEXT]: retained_context

Your answer:
```

C.2 Self-Prompting Baselines

The self-prompting baselines use the language model itself as a sufficiency judge. Context is divided into cumulative 10% prefixes. At each step, the model produces a hard sufficiency decision, parsed as `[[YES]]`, `[[NO]]`, or unknown.

C.2.1 Multi-Round Self-Prompt

The multi-round self-prompt baseline rebuilds a full cumulative prompt at every step. At step k, the model sees the question and the cumulative context prefix through block k. The sufficiency prompt is:

```
Given the following context and question, determine if the context contains enough
↪  information needed to answer the question.

[QUESTION]: question
```

```
[CONTEXT]: context_prefix

Your response should strictly ONLY consist of '[[YES]]' if context is enough, or
↪  '[[NO]]' if context is not enough. Omit any other output.

Your response:
```

The model is queried with this prompt for each cumulative prefix until it returns `[[YES]]`. If no prefix receives `[[YES]]`, the full context is retained. After prefix selection, the final answer is generated with the same answer prompt format used by the standard pipeline:

```
Please provide a response to the query based only on the given context:

[QUERY]: question

[CONTEXT]: retained_context

Your answer:
```

For Qwen 3.5 models, the sufficiency and answer prompts are wrapped as chat-template user messages when supported.

C.2.2 Single-Round Self-Prompt

The single-round self-prompt baseline uses a shared query–context prefix and switches suffixes depending on whether the model is being asked for sufficiency or for the final answer. This is designed for incremental execution with a reusable prefix/context cache.

Shared Prefix. The shared prefix is:

```
You are given a question and context:

[QUESTION]: question

[CONTEXT]: context_prefix
```

Sufficiency-Check Suffix. At each context step, the sufficiency suffix is appended to the shared prefix:

```
Given the context and question above, determine if the context contains enough
↪  information needed to answer the question.

Your response should strictly ONLY consist of '[[YES]]' if context is enough, or
↪  '[[NO]]' if context is not enough. Omit any other output.

Your response:
```

Answer Suffix. When the model returns `[[YES]]`, the sufficiency suffix is discarded and the answer suffix is appended to the same retained shared prefix:

```
Please provide an answer to the query above based only on the given context above.

Your response:
```

The important difference from multi-round self-prompting is structural. Multi-round self-prompting rebuilds and reprocesses a complete cumulative sufficiency prompt at every step. Single-round self-prompting keeps the query and observed context as a shared prefix, appends a sufficiency suffix for each decision, discards the suffix branch after the decision, and finally appends an answer suffix to the retained prefix.

Prefix variant	Template
neutral_question_context (original)	You are given a question and context. [QUESTION]: {question} [CONTEXT]: {context}
answer_from_given	Please answer using only the provided question and context: [QUESTION]: {question} [CONTEXT]: {context}
use_following	Use only the following question and context to provide a response: [QUESTION]: {question} [CONTEXT]: {context}
based_solely	Provide a response based solely on the given question and context: [QUESTION]: {question} [CONTEXT]: {context}
grounded_only	Ground your response only in the given question and context: [QUESTION]: {question} [CONTEXT]: {context}

Table 13: Single-round self-prompt prefix variants used in prompt-sensitivity runs. Line breaks are omitted in the table for compactness.

Suffix variant	Sufficiency-check wording
original	Given the context and question above, determine if the context contains enough information needed to answer the question. Respond only with [[YES]] if context is enough, or [[NO]] otherwise.
same_meaning_1	Given the question and context above, decide whether the context provides sufficient information to answer the question. Respond only with [[YES]] if sufficient, or [[NO]] otherwise.
same_meaning_2	Based on the question and context above, determine whether there is enough information in the context to answer the question. Respond only with [[YES]] if enough information is present, or [[NO]] otherwise.
same_meaning_3	Evaluate whether the context above contains the information needed to answer the question above. Respond only with [[YES]] if the needed information is present, or [[NO]] otherwise.
less_strict_1	Determine if the context contains enough evidence to answer the question, even if the answer requires combining facts. Respond only with [[YES]] or [[NO]].
less_strict_2	Determine if the context contains enough information to answer the question. Do not be overly conservative; respond [[NO]] only when the answer cannot be inferred from the provided context.
less_strict_3	Can the question above be answered using only the context above? Respond only with [[YES]] if it can, or [[NO]] if it cannot.

Table 14: Single-round self-prompt sufficiency suffix variants used in prompt-sensitivity runs. All variants preserve the same required output format: [[YES]] or [[NO]] only.

C.3 Self-Prompt Variant Prompts

The default single-round prompt can be varied by changing either the shared prefix or the sufficiency suffix. These variants are used only in prompt-sensitivity analyses; the main single-round setup uses the original prefix and suffix above.

C.4 Downstream Answer Evaluation Prompt

After generation, model answers are evaluated by a separate judge prompt. This prompt is used for downstream answer correctness evaluation, not for training the sufficiency probe.

```
You are an expert model evaluator specializing in natural language understanding.
↪ Your task is to determine if a model's answer is correct by comparing it with
↪ the provided gold answers, accounting for valid paraphrasing and alternate
↪ expressions of the same answers.

[QUESTION]
question
[/QUESTION]

[GOLD_ANSWERS]
correct_answers
[/GOLD_ANSWERS]

[MODEL_ANSWER]
model_answer
[/MODEL_ANSWER]

Evaluation criteria:
- Answer must convey the same core meaning as gold answers
- Partial matches should be marked incorrect
- Additional correct information beyond gold answers is acceptable
- Empty or off-topic responses are incorrect
```

```
Your response should strictly ONLY consist of '[[YES]]' if model answers question
↪  correctly, or '[[NO]]' if model answers question incorrectly. Omit any other
↪  output.

Your response:
```

The judge output is converted into an answer-correctness score. In match mode, generated text containing a yes decision is scored as correct and generated text containing a no decision is scored as incorrect. In probability mode for local judge models, the probabilities of `[[YES]]` and `[[NO]]` continuations are compared.

D GPU Requirements

While our total computational cluster consisted of 8 $\times$ A100 GPUs, the resource footprint for replicating individual experiments is highly accessible. As shown in Table 15, most standard evaluation runs easily fit on a single A100 GPU. The hardware requirement is primarily dictated by the backbone model size and the maximum context length of the specific benchmark, as our learned probes and classifier ensembles introduce negligible memory overhead compared to the cost of loading and running the base language model. The QRHead ablation studies to identify and rank QRHeads in models require more parallel computation and memory due to the dense extraction of exact post-softmax token-level attention weights and the long retrieval task the authors originally used to score the heads [50].

Experiment Type	Model Scale	Datasets	Hardware Requirement
Standard QUERYSTOP	Qwen 3.5 2B	All Datasets	1 $\times$ A100
	Qwen 3.5 4B	All Datasets	1 $\times$ A100
	Qwen 3.5 9B	All Datasets	2 $\times$ A100
QRHead Ranking	Qwen 3.5 2B	All Datasets	2 $\times$ A100
	Qwen 3.5 4B	All Datasets	2 $\times$ A100
	Qwen 3.5 9B	All Datasets	4 $\times$ A100

Table 15: Minimum hardware requirements for individual experiment runs. The learned probes and classifier ensembles introduce negligible overhead compared to the base model weights and context KV-cache.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the abstract covers the main contribution of our methods as well as the results reported in the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, limitations are discussed in results sections as well as ablation studies. A limitation section is also created.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The backbone models and benchmark datasets are publicly available; downstream answer evaluation uses an API-based judge. The settings are provided in the paper to fully replicate the results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We have not yet publicly released our experimental code or processed evaluation data. The original benchmark datasets are publicly available and cited in the paper.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Training data splits, hyperparameters and other details regarding probes are in appendix and supplementary material.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not report formal error bars or statistical significance tests. Since the evaluations are conducted over fixed benchmark splits and deterministic model outputs, we report per-dataset and per-model results rather than repeated-seed variance estimates.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: GPU information for the run is provided in supplementary and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that this research complies with its requirements.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This paper studies a general-purpose inference-efficiency method and does not include a dedicated discussion of its potential societal impacts.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This submission does not release new pretrained generative models or scraped datasets that would require additional safeguards for responsible release.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original sources of the benchmark datasets used in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer:[N/A]

    Justification: This submission does not release new datasets, model checkpoints, or software assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer:[N/A]

    Justification: This study uses existing benchmark datasets and does not involve crowdsourcing or human-subject experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This study does not involve human-subject research, so IRB approval is not applicable.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Yes. We used GPT-5.4 Mini as part of the method evaluation for downstream answer generation quality, which is documented in the evaluation setup. The authors verified all outputs and take responsibility for the paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.