Confidence-Driven Multi-Scale Model Selection for Cost-Efficient Inference

Bo-Wei Chen,[1] **Chung-Chi Chen,**[2] **An-Zi Yen**[1]
[1] Department of Computer Science, National Yang Ming Chiao Tung University, Taiwan
[2] Artificial Intelligence Research Center, AIST, Japan
h7a4n1k.cs12@nycu.edu.tw, c.c.chen@acm.org, azyen@nycu.edu.tw

Abstract

Large Language Models (LLMs) have revolutionized inference across diverse natural language tasks, with larger models performing better but at higher computational costs. We propose a confidence-driven strategy that dynamically selects the most suitable model based on confidence estimates. By assessing a model's confidence in handling the task and response accuracy, tasks that are likely to be solved correctly are retained, while more uncertain or complex cases are delegated to a larger model, ensuring reliability while minimizing computation. Specifically, we evaluate a model's likelihood of knowing the correct answer and the probability that its response is accurate. Experiments on the Massive Multitask Language Understanding (MMLU) benchmark show that our approach achieves accuracy comparable to the largest model while reducing computational costs by 20% to 40%. When applied to GPT-4o API calls, it reduces token usage by approximately 60%, further improving cost efficiency. These findings indicate the potential of confidence-based model selection to enhance real-world LLM deployment, particularly in resource-constrained settings such as edge devices and commercial API applications.

1 Introduction

As large language models (LLMs) become increasingly integrated into real-world applications, their deployment poses significant challenges in terms of computational cost and efficiency (Bender et al., 2021; Chen et al., 2023). While offering significant capability, an LLM query often requires forwarding a 10 to 100-billion-scale model. Moreover, with prompt engineering techniques such as in-context learning (Brown et al., 2020) or chain-of-thought reasoning (Wang et al., 2023), reasoning tasks can demand additional tokens, resulting in inputs spanning thousands of tokens. Therefore, running a large-scale model for every query is impractical, particularly for businesses, individual developers, and resource-constrained edge devices. For example, on smartphones, local models offer faster response times due to reduced latency and smaller parameter sizes, but they often sacrifice accuracy and performance. Conversely, querying API with a more capable model (i.e., GPT-4o (Hurst et al., 2024)) ensures higher-quality responses but introduces trade-offs such as network latency and increased computational expenses. Given the variance in model capabilities and the differing complexity of tasks, selecting the most cost-effective model that can still deliver an adequate response has become a critical research problem. Prior approaches have explored selective routing of queries, but an optimal balance between efficiency and accuracy remains an open challenge.

To address this, we propose a confidence-based model selection framework that optimizes efficiency while maintaining accuracy. Our approach begins by querying the smallest LLM and escalates to larger models if the initial model lacks confidence in its prediction. Inspired by Kadavath et al. (2022), we measure two facets of confidence: (1) $P(T)$ (Probability of Truth), which quantifies a model's confidence in the truthfulness of a given statement, and (2) $P(IK)$ (Probability of "I Know"), which measures the likelihood that a model can correctly answer a given query, as shown in Figure 1. By strategically orchestrating models of varying sizes, we achieve better cost-effectiveness while maintaining performance benchmarks on the Massive Multitask Language Understanding (MMLU) dataset (Hendrycks et al., 2021). We also analyze performance on an out-of-distribution (OOD) dataset to verify robustness.

In sum, our contributions are as follows. (1) A confidence-driven model selection strategy is proposed to reduce computational cost while main-

Findings of the Association for Computational Linguistics: EACL 2026, pages 1760–1770
March 24-29, 2026 ©2026 Association for Computational Linguistics

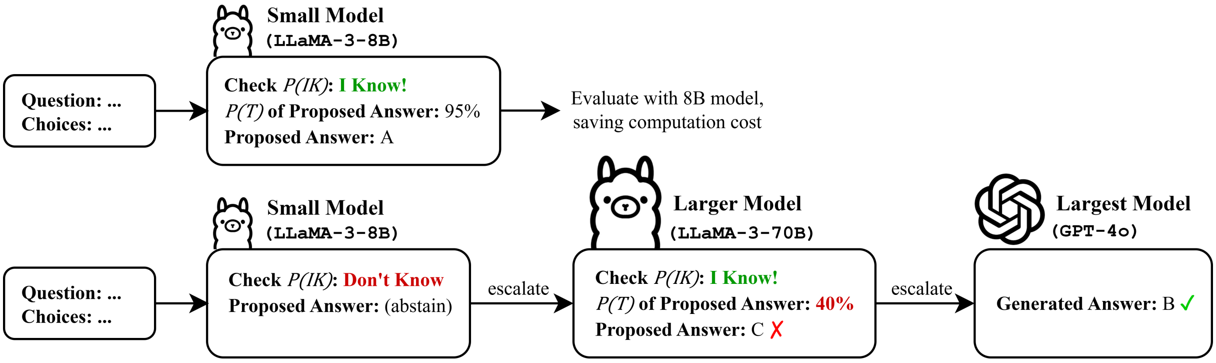

Figure 1: An illustration of our strategy.

taining accuracy.[1] (2) When applied to commercial APIs such as GPT-4o, the method reduces token costs with minimal performance degradation. (3) We demonstrate the generalizability of our approach to OOD datasets, showing its applicability beyond the training distribution.

2 Methodology

The core challenge in effectively selecting LLMs to balance cost and capability lies in measuring the confidence of smaller LLMs and using it as the basis for escalation decisions. Researchers (Kadavath et al., 2022; Lin et al., 2022; Kuhn et al., 2023; Chen and Mueller, 2024) have investigated whether a model can express precise confidence levels in their responses and assess their ability. Previous work (Chen et al., 2023) uses LLM cascading with a trained scoring function based on model outputs, but neglects internal model information, such as token probabilities and hidden states, which could improve confidence estimation and decision making. (Kadavath et al., 2022) show that language models produce well-calibrated probabilities when selecting among multiple explicit choices, with strong empirical results across benchmarks. These results suggest that the confidence estimates of the model can serve as a reliable uncertainty signal. Inspired by their work, we measure LLM confidences with $P(T)$ and $P(IK)$ of LLMs, where the former can be estimated without auxiliary models or additional training.[2]

2.1 Model Selection Process

Suppose the user has access to k LLMs, denoted as $M_1, M_2, \ldots, M_k$, sorted in ascending order by the number of parameters, where the largest model is M_k. Instead of always using M_k's output, our method first assesses $P(IK)$ for smaller models to determine whether they can confidently solve the query. Next, to verify the correctness of the response, we evaluate the model's $P(T)$ score to determine its confidence in the candidate answer. Specifically, if $P(IK)$ or $P(T)$ for M_i fall below a predefined threshold, the query is passed to the next larger model, M_{i+1}. Otherwise, M_i is used to answer the query, thereby reducing the computational cost associated with larger models. This process continues until the query reaches the largest model (M_k), at which point the response from M_k is used as the final answer. Figure 1 shows our strategy of escalating uncertain queries to larger models to improve efficiency.

2.2 Confidence Measurement

To determine $P(T)$, we measure the probability of the first token after prompting the model with a question and answer choices. Given a formatted prompt x, $P(T)$ is the probability that the predicted token belongs to the answer set. The final answer is determined based on the highest $P(T)$:

$$P(T) = p(c \mid x), \quad c \in \{A, B, C, D\}. \quad (1)$$

The value of $P(T)$ is directly obtained from the model's probability distribution over answer choices, corresponding to the predicted probability of the most confident option. For generation tasks, we use a different prompt that asks the model to answer the query in a predefined format *"Answer: "*, and measure the probability of the first token being *"Answer"*. To establish a threshold for $P(T)$, we conducted sensitivity analysis to identify suitable values. Based on the observations from the analysis, we set the threshold to 0.9 for all experiments in this work.

[1] https://github.com/NYCU-NLP-Lab/ConfDrivenInference

[2] Further discussions on confidence measurement methods, routing, and hybrid inference are provided in Appendix A.

To compute $P(IK)$, we trained a multi-layer perceptron using hidden states from the 24th transformer layer of an LLM as input (Mahaut et al., 2024), with supervision indicating whether the model correctly answered each query. The model is trained on 80% of the dataset, with 10% used for validation and 10% for testing across all datasets in this work, unless otherwise specified.

3 Experiments

3.1 Experimental Setup

We evaluate our strategy on MMLU, a multiple-choice benchmark assessing general knowledge, reasoning, and problem-solving across 57 subjects in humanities, sciences, and other domains.

For reproducibility, we conduct experiments using the open-source LLaMA instruction-tuned models (Dubey et al., 2024), and Qwen 3 series (Yang et al., 2025), which are available in sizes of (3B, 8B, 70B) and (4B, 8B, 32B) respectively. The specific open-source models used in our experiments are as follows: Llama-3.2-3B-Instruct, Meta-Llama-3.1-8B-Instruct, Llama-3.3-70B-Instruct, Qwen3-4B, Qwen3-8B, Qwen3-32B.[3] Additionally, we incorporate GPT-4o to explore its applicability in commercial API scenarios.[4]

Accuracy is measured as the proportion of questions for which the predicted answer matches the ground truth. In addition to accuracy, we assess the cost-effectiveness of different model combinations by measuring the end-to-end execution time, and calculating the computational cost of evaluating the models. To estimate the computational cost of the forward pass in LLMs, we adopt (Kaplan et al., 2020)'s approach. The number of add-multiply operations required for a forward pass in a Transformer-based model is given by:

$$C_{\text{forward}} \approx 2N + 2n_{\text{layer}} \times n_{\text{ctx}} \times d_{\text{model}} \,, \qquad (2)$$

where N is the number of model parameters, n_{ctx} is the number of context tokens, and d_{model} denotes the dimension of the residual stream. For simplification, we approximate N using:

$$N \approx 12 \times n_{\text{layer}} \times d_{\text{model}}^2 \,, \qquad (3)$$

Substituting into the original equation gives:

$$C_{\text{forward}} \approx 2N + 2n_{\text{ctx}} \times \sqrt{\frac{N n_{\text{layer}}}{12}} \,, \qquad (4)$$

LLM(s)	Acc.	PD (↓)	Reduced CC (↑)	Time (↓)
3B	63.78%	–	–	–
8B	70.21%	–	–	–
70B	83.57%	–	–	–
3B → 8B	69.44%	-0.77%	20.34%	-18.90%
3B → 70B	81.89%	-1.68%	32.47%	-26.51%
8B → 70B	83.22%	-0.35%	36.46%	-33.03%
3B → 8B → 70B	81.54%	-2.03%	44.48%	–

Table 1: Experimental results with LLaMA models on MMLU. Acc., Reduced CC, Time, and PD denote accuracy, reduced computational cost (measured in GFLOPs), end-to-end time, and performance drop. Each metric is calculated relative to those of M_k.

For large values of N (as model sizes are typically in the billions of parameters), the context-dependent term becomes significant only when n_{ctx} is comparable to N. Thus, the complexity of a forward pass is approximately:

$$C_{\text{forward}} \approx 2N \qquad (5)$$

To clarify the differences in computational cost (CC) across model configurations, we provide the specific number of queries handled by each model stage during evaluation. The compute cost is defined as:

$$\text{CC} = 2 \times N \times k \qquad (6)$$

where N is the number of model parameters and k is the number of queries evaluated by that model. The factor of 2 accounts for the two operations (multiply and accumulate) in the matrix multiplications. Appendix D presents the number of queries across different stages.

3.2 Results with Multi-Scale Open-Source Models

Table 1 presents the experimental results using LLaMA models. To provide a more comprehensive assessment beyond accuracy, we also report Macro-Precision, Macro-Recall, and Macro-F1 scores for all model configurations. These metrics offer a class-agnostic view of performance by treating each class equally, regardless of its frequency. The results are presented in Table 2.[5]

Several key observations can be made. First, as expected, utilizing the largest 70B model yields the best performance. However, this comes at a significantly higher computational cost compared to smaller models, highlighting the inherent trade-off between performance and efficiency. Second,

[3] Model details are presented in Appendix B.

[4] Prompts used in this work are provided in Appendix C.

[5] The end-to-end time for the 3B → 8B → 70B chain is omitted due to computation resource constraints.

LLM(s)	P	R	F1
3B	0.6444	0.6372	0.6368
8B	0.7055	0.7021	0.7017
70B	0.8386	0.8354	0.8357
3B → 8B	0.6974	0.6941	0.6940
3B → 70B	0.8198	0.8160	0.8166
8B → 70B	0.8350	0.8317	0.8322
3B → 8B → 70B	0.8162	0.8124	0.8130

Table 2: Macro-Precision (P), Macro-Recall (R), and Macro-F1 (F1) scores of LLaMA models on MMLU.

LLM(s)	Acc.	PD (↓)	Reduced CC (↑)	Time (↓)
4B	66.73%	–	–	–
8B	72.43%	–	–	–
32B	79.51%	–	–	–
4B → 8B	70.21%	-3.07%	17.80%	2.55%
4B → 32B	75.03%	-5.63%	52.70%	-40.21%
8B → 32B	80.00%	0.62%	33.18%	-23.76%
4B → 8B → 32B	74.69%	-6.06%	55.06%	–

Table 3: Experimental results with Qwen 3 models on MMLU.

by employing the proposed confidence-driven strategy, computational complexity is notably reduced with only a minor drop in performance. A similar reduction is also observed in the end-to-end execution time. This demonstrates the potential of leveraging LLM confidence for multi-scale model selection. Third, comparing the results of (8B, 3B → 8B) and (8B, 8B → 70B), we observe that applying the proposed strategy with a model one size smaller results in minimal performance degradation while achieving an approximate 20-36% reduction in computational cost. Notably, 8B → 70B achieves comparable accuracy to 70B (83.22% vs. 83.57%) with a 36% reduction in cost. We calculate McNemar's statistical significance test and observe no significant difference ($p = 0.4048$), supporting the efficiency of our approach.[6] However, extending the transfer chain (8B → 70B and 3B → 8B → 70B) leads to a more pronounced performance drop, but with increased cost savings. This is expected, as errors may accumulate and propagate under such a design. Additionally, in the 3B → 8B → 70B process, the smaller model (e.g., 3B) may overestimate its confidence, leading to premature task retention and suboptimal delegation, ultimately affecting the final reasoning outcome.

As shown in the Table 3, our proposed confidence-based model selection method also demonstrates promising results on the Qwen 3 series.[7] Specifically, the 8B → 32B cascade achieves an accuracy of 0.800, which is slightly higher than the full 32B model (0.795), while saving approximately 33.18% of the compute. This confirms the generalizability of our method beyond the LLaMA series and its effectiveness across different model families. We further extend our method to open-ended question answering, demonstrating promising results on PopQA with a 7% reduction in cost and less than a 2% drop in performance. The detailed results are provided in Appendix F. To sum up, these findings support the effectiveness of the confidence-driven strategy in balancing performance and computational cost.

LLM(s)	Acc.	Tokens	PD	Reduced Tokens
GPT-4o	86.43%	36,225	–	–
3B → GPT-4o	84.48%	30,671	-1.95%	15.33%
8B → GPT-4o	85.66%	28,941	-0.77%	20.11%
70B → GPT-4o	86.85%	14,505	0.42%	59.96%

Table 4: Experimental results with GPT-4o.

3.3 Results with Commercial API

To evaluate our strategy in a more realistic scenario, we tested it using a commercial API. Since OpenAI does not disclose the total parameter count for GPT-4o, we use the number of output tokens as a proxy for the compute cost metric. This metric aligns with mainstream API cost calculation methods.

Table 4 compares the results of using GPT-4o as the final model in our method versus directly prompting GPT-4o. While other approaches result in a slight performance drop for a 15–20% cost reduction, the 70B → GPT-4o approach slightly improves performance and achieves approximately 60% cost savings. Our strategy effectively balances performance and cost, particularly by leveraging an intermediate 70B model before GPT-4o, achieving significant token cost reduction while maintaining comparable performance. Even alternative approaches with minor performance loss provide meaningful savings, demonstrating the practicality of our method for optimizing API usage.[8]

4 Discussion

4.1 Ablation Analysis

In this section, we present an ablation study on $P(IK)$, focusing on performance drop since the transfer chain cost remains similar with or without $P(IK)$.[9] Table 5 presents the experimental

[6] Appendix E presents detailed significance test results.

[7] The end-to-end time for the 4B → 8B → 32B chain is omitted due to computation resource constraints.

[8] Appendix G provides an evaluation of cost across settings.

[9] Results of $P(IK)$ are reported in Appendix H.

Setting	LLM(s)	PD	Reduced CC
w/ $P(IK)$	8B → 70B	-0.35%	36.46%
	3B → 8B → 70B	-2.03%	44.48%
w/o $P(IK)$	8B → 70B	-1.26%	36.83%
	3B → 8B → 70B	-4.27%	47.17%

Table 5: Ablation studies of $P(IK)$.

results. First, the results indicate the usefulness of training a classifier to assess $P(IK)$. Specifically, the performance drop is significantly greater when using the transfer chain with the 70B model as the largest model without $P(IK)$. Second, even without $P(IK)$, which requires training data for assessment, the performance drop remains smaller than the computational cost saved. These results validate the effectiveness of the confidence-driven multi-scale model selection, even without training data. This finding also highlights the method's practicality in closed-source or API-based settings, where internal hidden states are not accessible and $P(IK)$ cannot be computed. In such cases, $P(T)$ can still be derived from log probabilities provided by the model. As shown in Table 5, relying solely on $P(T)$ still achieves competitive performance.[10]

4.2 Generalization to OOD Dataset

A key question when evaluating a $P(IK)$ classifier trained on MMLU is whether it can generalize to other datasets. Hence, we investigate if a classifier that assesses a model's NLU across different question types can be directly applied to new datasets without modification. Such a finding would suggest that a well-trained $P(IK)$ classifier captures a model's intrinsic ability for diverse NLU tasks, independent of the specific dataset characteristics.

To explore this hypothesis, we employ GPQA: A Graduate-Level Google-Proof Q&A Benchmark (Rein et al., 2023). GPQA represents an out-of-distribution (OOD) setting relative to MMLU, as it comprises highly challenging, graduate-level questions. By applying the same $P(IK)$ classification approach to GPQA, we aim to assess whether the classifier retains its effectiveness in measuring model performance in an OOD context.

Table 6 presents the experimental results. First, GPQA proves to be significantly more challenging than MMLU for all model scales. Second, the proposed framework results in a modest decrease in computational costs while maintaining competitive performance. However, the substantial difference in computational cost savings between MMLU and GPQA highlights the difficulty of extending the method to OOD applications. This may be due to the fact that GPQA's questions are too challenging for smaller models, such as 3B and 8B, making it difficult for them to effectively handle queries. We looked into the amount of queries that the $P(IK)$ classifier predicts positive in the 8B → 70B case. Out of 448 samples in the test set, only 21 samples were routed to 8B by the classifier, showing that the classifier is conservative while facing OOD queries rather than aggressively routing to the 8B model. However, among these 21 samples, 67% of them (14 samples) are "correct", meaning that the 8B model can indeed handle these queries, showing that the $P(IK)$ classifier remains useful in the OOD scenario. This preliminary exploration identifies this challenge for future research in advancing confidence-driven multi-scale model selection.

LLM(s)	Acc.	PD	Reduced CC
3B	31.47%	–	–
8B	31.92%	–	–
70B	52.23%	–	–
8B → 70B	51.79%	-0.44%	3.93%
3B → 8B → 70B	51.56%	-0.67%	5.11%

Table 6: Experimental results of LLaMA models in OOD scenario.

5 Conclusions

We propose a confidence-driven multi-scale model selection strategy to optimize cost-effective inference. By combining confidence estimates of $P(T)$ (token probabilities) and $P(IK)$ (trained classifiers), our approach dynamically routes queries to the most appropriate model, significantly reducing computational costs while maintaining high accuracy. Our experiments evaluating LLaMA 3 and Qwen 3 models on MMLU show that the proposed strategy matches the performance of the largest model while reducing compute cost. When applied to GPT-4o, it significantly cuts token usage and improves cost efficiency. Moreover, we demonstrate that our approach remains effective in both in-distribution and out-of-distribution settings by leveraging classifiers trained with MMLU on GPQA, a significantly more challenging dataset. Future work should focus on refining confidence estimation for better generalization across diverse tasks, particularly generative tasks.

[10] Results for different $P(T)$ thresholds are in Appendix I.

Limitations

First, our approach relies on a classifier trained on a specific dataset to estimate $P(IK)$. While we observe some level of generalization to out-of-distribution datasets such as GPQA, performance may degrade in significantly different domains or tasks. Future work should explore methods to improve generalization, such as domain adaptation techniques or unsupervised confidence estimation. Second, our experiments primarily focus on NLU tasks within the MMLU and GPQA benchmarks. Our experiments with PopQA served as a preliminary attempt to generalize confidence-based routing beyond NLU tasks. The applicability of our method to other NLP tasks, such as generative language modeling, open-domain QA, or real-time conversational AI, remains an open challenge. Future work should investigate extending confidence-based model selection to broader NLP applications. Third, while our method reduces overall compute usage, it may introduce additional latency for queries that are escalated through multiple models before reaching the final response. Although we use compute cost as a proxy for efficiency, actual end-to-end latency can be affected by system implementation, hardware, and network conditions, especially in API-based settings. Future work could incorporate latency-aware routing or empirically evaluate real-world latency across deployment scenarios. Despite these limitations, our findings indicate that confidence-driven model selection is a promising direction for optimizing LLM deployment, particularly in scenarios where computational efficiency is a critical concern. Addressing these challenges will be key to further improving the reliability and applicability of the approach across diverse settings.

Acknowledgments

This research was partially supported by the National Science and Technology Council, Taiwan, under Grant NSTC 114-2221-E-A49-057-MY3, and by the project “Advancements in Industrial and Financial Large Language Models: Construction and Applications,” funded by the Research Center for Information Technology Innovation (CITI), Academia Sinica. Chung-Chi Chen’s work was supported by AIST policy-based budget project “R&D on Generative AI Foundation Models for the Physical Domain”.

References

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Jiuhai Chen and Jonas Mueller. 2024. Quantifying uncertainty in answers from any language model and enhancing their trustworthiness. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5186–5200, Bangkok, Thailand. Association for Computational Linguistics.

Justin Chih-Yao Chen, Sukwon Yun, Elias Stengel-Eskin, Tianlong Chen, and Mohit Bansal. 2025. Symbolic mixture-of-experts: Adaptive skill-based routing for scalable heterogeneous reasoning. *arXiv preprint arXiv:2503.05641*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *Preprint*, arXiv:2305.05176.

Yu-Neng Chuang, Leisheng Yu, Guanchu Wang, Lizhe Zhang, Zirui Liu, Xuanting Cai, Yang Sui, Vladimir Braverman, and Xia Hu. 2025. Confident or seek stronger: Exploring uncertainty-based on-device llm routing from benchmarking to generalization. *Preprint*, arXiv:2502.04428.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. In *The Twelfth International Conference on Learning Representations*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know. *Preprint*, arXiv:2207.05221.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *Preprint*, arXiv:2302.09664.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. *Preprint*, arXiv:2205.14334.

Matéo Mahaut, Laura Aina, Paula Czarnowska, Momchil Hardalov, Thomas Müller, and Lluis Marquez. 2024. Factual confidence of LLMs: on reliability and robustness of current estimators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4554–4570, Bangkok, Thailand. Association for Computational Linguistics.

Seungeun Oh, Jinhyuk Kim, Jihong Park, Seung-Woo Ko, Tony QS Quek, and Seong-Lyun Kim. 2024. Uncertainty-aware hybrid inference with on-device small and remote large language models. *arXiv preprint arXiv:2412.12687*.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof qa benchmark. *Preprint*, arXiv:2311.12022.

Jie Ren, Yao Zhao, Tu Vu, Peter J. Liu, and Balaji Lakshminarayanan. 2023. Self-evaluation improves selective generation in large language models. *Preprint*, arXiv:2312.09300.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2024. Large language model routing with benchmark datasets. In *First Conference on Language Modeling*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *Preprint*, arXiv:2306.13063.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Zesen Zhao, Shuowei Jin, and Z Morley Mao. 2024. Eagle: Efficient training-free router for multi-llm inference. *arXiv preprint arXiv:2409.15518*.

A Related Work

A.1 Confidence Measurement Methods

Kadavath et al. (2022) study the calibration of prediction made by LLMs, and proposed two facets of model confidence: $P(T)$ and $P(IK)$. $P(T)$, originally named as **P(True)**, represents the degree to which a model thinks the input statement is true. For example, given a claim or statement, the value of $P(T)$ can be evaluated by prompting the model: *"Is <statement> true? (A) True (B) False Answer: <answer>"*, and then measure the probability of the token "*(A)*" predicted by the model at position *<answer>*.

On the other hand, $P(IK)$ (stands for "I Know") describes how likely a model can give correct response to a query. The purpose of $P(IK)$ is to identify the problems that the model can or cannot answer, for the further goal of increasing the trustworthiness and reducing hallucination. The evaluation of $P(IK)$ can be done in multiple ways: Kadavath et al. (2022) trains a binary classification head on top of the language model by manually collecting training samples from various datasets. Mahaut et al. (2024) estimates $P(IK)$ with a similar setting to $P(T)$: prompting the model *"Do you know the answer to the following question: <question> (Yes/No/Maybe)? Answer: <answer>"* and measure the token probability of "*Yes*" predicted by the model at position <*answer*>.

When internal information of LLMs is not available, measuring the confidence of model responses becomes challenging. One straightforward way is to use ***verbalized confidence*** (Lin et al., 2022), which directly prompts the model to output confidence values for their response. However, empirical results (Xiong et al., 2024) have shown that LLMs tend to be overconfident while verbalizing their confidence level, leading to poor prediction performance. Some black-box methods focus on prompt design, e.g., ask the model to provide multiple answers and their confidence at the same time, also known as *top-k prompting* (Tian et al., 2023). Other methods investigate the distribution or semantic characteristics between multiple responses, such as ***self-consistency*** (Wang et al., 2023) and ***semantic entropy*** (Kuhn et al., 2023). In sum, current black-box methods relies heavily on multiple sampling to learn the nuanced relationship between responses. This would require significantly more computation at inference time compared to white-box methods, therefore they might not be the most efficient approach to estimate model confidence.

A.2 Routing and Hybrid Inference

Recent efforts in model routing and hybrid inference have introduced efficient strategies to balance LLM performance and cost. Training-free methods like Eagle (Zhao et al., 2024) and RouteLLM (Ong et al., 2024) dynamically assign queries based on benchmark performance or preference-based scoring, while hybrid systems such as U-HLM architectures (Oh et al., 2024) combine small and large models for faster, cost-efficient inference. Hybrid LLM (Ding et al., 2024) also addresses this trade-off by training a query router that predicts response quality gaps between small and large models, enabling routing decisions tailored to the expected performance of each query. Their approach estimates whether the small model can match the large model's response using BART scores, and tunes routing thresholds to control quality-cost trade-offs. SYMBOLIC-MOE (Chen et al., 2025) further explores symbolic, skill-based routing to match inputs with appropriate experts without gradient-based optimization, combining model profiling with task-specific keyword extraction.

Our work complements these by proposing a confidence-driven selection mechanism that integrates both token-level certainty ($P(T)$) and internal knowledge estimation ($P(IK)$) to decide when to escalate queries. Unlike prior routing models that often rely on offline training (Shnitzer et al., 2024), symbolic rules, or task-specific heuristics, our approach enables dynamic, instance-level decision-making with minimal architectural overhead, and demonstrates robustness under both in-distribution and OOD conditions.

B Models

The open-source models were obtained from HuggingFace with the following identifiers:

- `meta-llama/Llama-3.2-3B-Instruct` (used for the 3B model)
- `meta-llama/Meta-Llama-3.1-8B-Instruct` (used for the 8B model)
- `meta-llama/Llama-3.3-70B-Instruct` (used for the 70B model)
- `Qwen/Qwen3-4B` (used for the 4B model)
- `Qwen/Qwen3-8B` (used for the 8B model)
- `Qwen/Qwen3-32B` (used for the 32B model)

All of the experiments were conducted on 4 NVIDIA RTX A6000 GPUs, each with 48GBs of VRAM. For generation tasks, we used a deterministic decoding setup with `do_sample=False`, and applied nucleus sampling with `top_p=0.9` and `temperature=1.0`. These parameters were held consistent across all model configurations unless otherwise noted. All results are based on a single run per model configuration without repeated sampling.

C Input Format

The prompts used in this work are presented in Table 7.

D Query Distribution Across Stages

The number of queries processed at each stage is summarized in Table 8. For example, in the 3B $\rightarrow$ 8B $\rightarrow$ 70B configuration, the compute cost is $2 \times (487 \times 3 \times 10^9 + 198 \times 8 \times 10^9 + 745 \times 70 \times 10^9) = 1.1 \times 10^5$ GFLOPs. In contrast, in the 8B

Prompt used for multiple choice questions. The following are multiple choice questions about {subject}. Please determine the correct answer in the format "Answer: <answer>", where <answer>is A, B, C or D. Question: {question} Choices: {choices} Answer:
Prompt used for open-ended questions. (system) The following are questions about entity relationship. Please determine the correct answer in the format "Answer: <answer>". (user) Question: {question} (assistant) Answer:

Table 7: Prompt templates used in this work.

LLM(s)	3B	8B	70B
3B → 8B	487	943	–
3B → 70B	487	–	943
8B → 70B	–	594	836
3B → 8B → 70B	487	198	745

Table 8: Number of queries processed by each model stage under different configurations.

Comparison	Chi-squared	p-value
3B → 8B vs 8B	4.00	0.0455
8B → 70B vs. 70B*	N/A	**0.4048**
3B → 8B → 70B vs. 70B	19.22	0.00001
3B → 70B vs. 70B	20.48	0.00001
3B → GPT-4o vs. GPT-4o	15.84	0.00007
8B → GPT-4o vs. GPT-4o	4.00	0.0455
70B → GPT-4o vs. GPT-4o	**0.22**	**0.6395**

Table 9: Significance test results. The exact p-value for 8B → 70B vs 70B is calculated using a binomial test instead of the chi-squared approximation.

→ 70B configuration, the cost is $2 \times (594 \times 8 \times 10^9 + 836 \times 70 \times 10^9) = 1.27 \times 10^5$ GFLOPs. This illustrates that even though 3B→8B→70B involves more model stages, the total cost can be lower due to early-stage filtering, which reduces the number of queries reaching the larger models.

E Statistical Significance Analysis

To assess whether the observed differences in performance are statistically significant, we conducted McNemar's test on key model comparisons. The results are summarized in Table 9.

The results indicate that in the cases of 8B → 70B vs. 70B and 70B → GPT-4o vs. GPT-4o, the p-values exceed 0.05. This suggests that there is no statistically significant difference in performance between these configurations. However, our method achieves comparable performance while significantly reducing computational costs. These results support our claim that the proposed method offers an effective balance between performance and resource usage.

F Results with Open-ended QA

One critical question for our framework is: *"how can we generalize the method to generative tasks where multiple-choice $P(T)$ cannot be computed?"* While some researchers have attempted to extend confidence estimation to generative tasks by either sampling multiple responses and transforming the original query into multiple-choice format (Ren et al., 2023), or by simply using average token probabilities over the generated output (Chuang et al., 2025), the performance is modest. In this paper, we propose a new confidence estimation technique called ***first-token confidence***. We prompt the model to respond in a specific format and then measure the probability of the first token being *"Answer"*. The rationale behind this approach is that we observed models often deviate from the expected format when they are uncertain or unable to answer confidently.

To test our method on generative tasks, we perform evaluation using PopQA, a large-scale open-domain question answering dataset consisting of 14k entity-centric QA pairs. A main problem we encountered while evaluating open-ended queries was the ineffectiveness of traditional keyword-based matching. Previous work (Kamalloo et al., 2023) emphasizes that exact match or simple overlap metrics understate LLM performance and miss hallucinations. To mitigate this, we leveraged the grounding API to search for relevant results online and summarize whether the model's response can be considered correct. Table 10 shows the evaluation results with and without grounding. Our results are in line with (Kamalloo et al., 2023), where simple lexical matching fails when models responded with a plausible answer. This is especially the case when evaluating datasets like PopQA, whose questions are somewhat ambiguous and can have multiple correct answers.

For fair evaluation, we argue that a model responding *"I don't know"* or asking for clarification should not be considered as incorrect. We propose using ***hallucination rate*** to reflect model's faithfulness, which evaluates how many responses are factually incorrect, verified by grounding API.

Table 11 presents the accuracy, hallucination rate

	Accuracy	Δ
3B (w/o grounding)	0.1916	-
3B (w/ grounding)	0.2669	+0.0753
8B (w/o grounding)	0.2520	-
8B (w/ grounding)	0.3514	+0.0994
70B (w/o grounding)	0.4767	-
70B (w/ grounding)	0.6608	+0.1841

Table 10: Accuracy and delta for different model sizes with and without grounding.

LLM(s)	Acc.	Hallucination Rate	Cost (USD)	PD	Reduced Cost
3B	0.2689	0.4169	0.0030	-	-
8B	0.3387	0.4679	0.0028	-	-
70B	0.6585	0.3208	0.0257	-	-
3B → 8B	0.3317	0.4923	0.0027	-2.07%	2.64%
3B → 70B	0.6131	0.3547	0.0226	-6.89%	12.29%
8B → 70B	**0.6459**	**0.3422**	**0.0239**	**-1.91%**	**7.11%**
3B → 8B → 70B	0.6075	0.3603	0.0219	-7.74%	14.78%

Table 11: Experimental results with LLaMA models on PopQA.

Model	Input (per 1M tokens)	Output (per 1M tokens)
3B	\$0.08	\$0.16
8B	\$0.10	\$0.20
70B	\$0.60	\$1.20
GPT-4o	\$2.50	\$10.00

Table 12: Token pricing for different models.

LLM(s)	Total Cost	Reduced Cost
3B	\$0.017	-
8B	\$0.021	-
70B	\$0.129	-
3B→8B	\$0.020	4.76%
3B→70B	\$0.096	25.58%
8B→70B	\$0.089	31.01%
3B→8B→70B	\$0.080	37.98%
GPT-4o	\$0.928	-
3B→GPT-4o	\$0.707	23.81%
8B→GPT-4o	\$0.650	29.96%
70B→GPT-4o	\$0.357	61.53%

Table 13: Estimated cost for each setting.

and cost of different LLaMA models on the PopQA dataset.[11] As expected, larger models generally demonstrate higher accuracy and lower hallucination rates. The 70B model achieves the best single-model performance, with an accuracy of 0.6585 and a hallucination rate of 0.3208, outperforming both the 3B and 8B variants by a significant margin.

When employing model cascades, we observe a trade-off between accuracy and hallucination. For example, the 3B → 8B → 70B cascade achieves an accuracy of 0.6075, which is slightly lower than using the 70B model alone. The two-stage (8B → 70B) strategy achieves the highest accuracy (0.6459) and the lowest hallucination rate (0.3422) among all dynamic settings, while saving over 7% in cost compared to using the 70B model exclusively. Notably, the performance drop (PD) is minimal at only 1.91%, demonstrating that dynamic model selection can yield promising results at a fraction of the cost.

G Cost Estimation

To promote transparency and enable fair comparison, we provide a detailed cost analysis of all model configurations used in our experiments. While our initial assumption was that self-hosting LLMs would generally offer lower cost than API-based solutions, we conducted a systematic evaluation using publicly available token pricing information.

Specifically, we obtained pricing for input and output tokens from SambaNova's cloud-hosted LLaMA models (3B, 8B, and 70B),[12] and compared them against OpenAI's GPT-4o API pricing.[13] We summarize the input and output token pricing of different models used in this work in Table 12. All prices reported were recorded in April 2025.

For each setting, we calculated the total cost based on the number of input and output tokens consumed during inference. Table 13 shows the per-token pricing as well as the estimated total cost across different model paths. As expected, direct use of GPT-4o incurs the highest cost (\$0.928 per run), while cascade-based configurations such as 70B→GPT-4o (\$0.357) or 8B→GPT-4o (\$0.650) offer a substantial reduction. Among open-source settings, 3B→8B→70B achieves strong performance while keeping the cost at only \$0.08, highlighting the method's efficiency in balancing accuracy and resource usage. This analysis supports the utility of our approach in both open-source and API-based scenarios.

H Evaluation of $P(IK)$

Before evaluating the impact of our framework, we first examine the performance of the $P(IK)$ classifier itself. As shown in Table 14, the trained classifiers achieve decent performance, indicating that the $P(IK)$ classifiers can effectively estimate model's self-knowledge, providing a reliable signal

[11] Table 12 presents the token pricing of the models.

[12] https://cloud.sambanova.ai/plans/pricing

[13] https://openai.com/api/pricing/

Model	Accuracy	F1	AUROC
3B	69.16%	62.76%	75.17%
8B	76.50%	69.21%	81.78%

Table 14: Performance of $P(IK)$ classifier of LLaMA 3B and 8B models.

8B → 70B			
Setting	**Accuracy**	**PD**	**Reduced CC**
$P(T) \geq 95\%$	0.8350	-0.08%	33.25%
$P(T) \geq 90\%$	0.8322	-0.42%	36.46%
$P(T) \geq 70\%$	0.8287	-0.84%	38.56%
$P(T) \geq 50\%$	0.8266	-1.09%	39.33%
8B → 70B (w/o $P(IK)$)			
$P(T) \geq 95\%$	0.8350	-0.08%	29.69%
$P(T) \geq 90\%$	0.8231	-1.51%	38.15%
$P(T) \geq 70\%$	0.7923	-5.19%	56.19%
$P(T) \geq 50\%$	0.7455	-10.79%	74.24%

Table 15: Model Performance Comparison for LLaMA 8B → 70B.

for guiding model selection. With this understanding, we can now analyze how incorporating $P(IK)$ affects the overall framework performance.

I Sensitivity Analysis of $P(T)$ threshold

We conducted a sensitivity analysis by varying the threshold of P(T), as shown in Table. The threshold of $P(T)$ controls how confidently the smaller model must be before retaining the answer rather than routing it to the larger model.

In the 8B → 70B setting, we observe that as the $P(T)$ threshold decreases from 95% to 50%, the routing rate to the larger model increases (i.e., reduced compute cost improves), but this comes at the expense of a gradual decline in overall accuracy and performance drop (PD). Importantly, with our full system (i.e., with both $P(T)$ and $P(IK)$), the accuracy remains relatively stable, dropping only 0.8% even when the threshold is lowered to 50%, while reducing compute cost by $\tilde{3}$9%. However, when we disable $P(IK)$ and rely on $P(T)$ alone, the performance becomes more sensitive to the threshold. For instance, at a 50% threshold, the accuracy drops significantly, despite higher compute savings. These results confirm that $P(IK)$ complements $P(T)$ by stabilizing performance even when we allow more aggressive routing to the smaller model. It also suggests that thresholds around 90% strike a good balance between accuracy and computation reduction.