

Routing Entropy: A Hidden Self-Verifier for Free in Mixture-of-Experts LLMs

Zhongyang Li

Department of Computer Science
The Ohio State University
li.17258@osu.edu

Ziyue Li

Department of Computer Science
University of Maryland, College Park
litzy619@umd.edu

Tianyi Zhou

Mohamed bin Zayed University of Artificial Intelligence
tianyi.zhou@mbzuai.ac.ae

Abstract

Modern LLMs are observed to be overconfident in their own outputs, making them poor self-verifiers. We found that for mixture-of-experts (MoE) LLMs, the Routing Entropy (RE) is a more reliable self-verifier than output confidence (OC). In other words, more confident expert choices in MoE usually lead to higher-quality outputs. We examined RE for self-verification on multiple recent MoE LLMs. While RE consistently outperforms OC, combining them leads to the best performance. Moreover, our RE-based analysis shows that better MoEs are usually more confident in expert choices. These novel insights further motivate an RE-regularizer for post-training of MoE routers. Our extensive experiments demonstrate the superiority of RE for self-verification across diverse reasoning and knowledge benchmarks. Furthermore, RE-regularization in post-training significantly enhances MoE’s internal uncertainty calibration, yielding a more reliable model without sacrificing its core generative performance.

1 Introduction

Large Language Models (LLMs) are known to produce fluent but unreliable outputs, often exhibiting hallucinations and reasoning errors (Perković et al., 2024). A practical strategy to mitigate such failures is to equip models with the ability to assess their own uncertainty or correctness. In most existing approaches, this is done using output-based signals, such as the maximum softmax probability of generated tokens (confidence) (Mahaut et al., 2024). However, these signals have become increasingly unreliable in modern LLMs due to overalignment: Supervised Fine-Tuning (SFT) and Reinforcement Learning from Human Feedback (RLHF) tend to encourage models to produce confident and definitive responses, even when they are incorrect or uncertain (Ouyang et al., 2022; Lin et al., 2022). As a result, output confidence is often poorly calibrated, failing to reflect internal uncertainty signal.

This limitation motivates us to look beyond the output layer and instead examine the model’s internal computation. In this work, we focus on Mixture-of-Experts (MoE) LLMs, where each token is dynamically routed to a subset of experts (Shazeer et al., 2017; Fedus et al., 2022). We find that the routing distribution over experts provides a simple yet useful signal: when the model is confident/correct, the routing is more concentrated, while uncertain/incorrect responses are often associated with more dispersed routing patterns. Based on this observation, we propose *Routing Entropy* (RE) to quantify the uncertainty of

the routing distribution and use it for self-verification of output correctness. Unlike output confidence, RE is derived entirely from internal states and requires no additional training.

To extract the self-verification signal, we analyze where this routing hesitation actually occurs. We observe that shallow transformer layers mostly handle basic syntax, while deep layers process complex semantics. Similarly, the model makes its most critical choices during the final tokens. By aggregating RE strictly from these deep layers and concluding tokens, we effectively filter out early noise and capture the internal uncertainty signal. Remarkably, this targeted internal metric consistently outperforms standard output confidence across diverse benchmarks without any extra training. We discover that internal RE and external output confidence are highly complementary. By fusing these two distinct signals with a simple XGBoost classifier (Chen & Guestrin, 2016), we achieve state-of-the-art error detection, as summarized in Figure 1(a). We further demonstrate that adding redundant features only degrades performance, proving that this two-feature combination is the optimal and minimal pair for reliable self-verification.

Finally, we move beyond self-verification to actively improving it and model performance. Standard SFT usually forces models to be overly confident, even when they make mistakes. To fix this risky behavior, we introduce an RE-Regularizer applied to the post-training of routers in MoE. During training, if the model answers correctly, we encourage the router to be decisive and maintain low entropy. However, if the model hallucinates or fails, we force the router to show hesitation and maintain high entropy. This simple training objective improves internal uncertainty calibration during router post-training. It substantially boosts the model’s ability to detect its own errors without sacrificing any generation accuracy. As shown in Figure 1(b) and 1(c), this approach consistently improves task performance and generalizes robustly across diverse MoE architectures.

Our main contributions can be summarized as:

- **Routing Behavior reflects Correctness:** We discover that the routing weights in MoE LLMs provide a natural, built-in self-verifier. The distribution over internal experts effectively bypasses the overconfidence trap on the outputs.
- **Routing Entropy is a Better Self-Verifier:** We demonstrate that the RE of later tokens in deep layers, especially when combined with output confidence, achieves the best self-verification performance across diverse tasks.
- **RE-Regularized Post-Training of Routers:** We propose a novel RE-Regularizer to post-train routers in MoE LLMs. This approach successfully aligns routing weights with the correctness of the generated content and boosts MoE performance across diverse benchmarks.

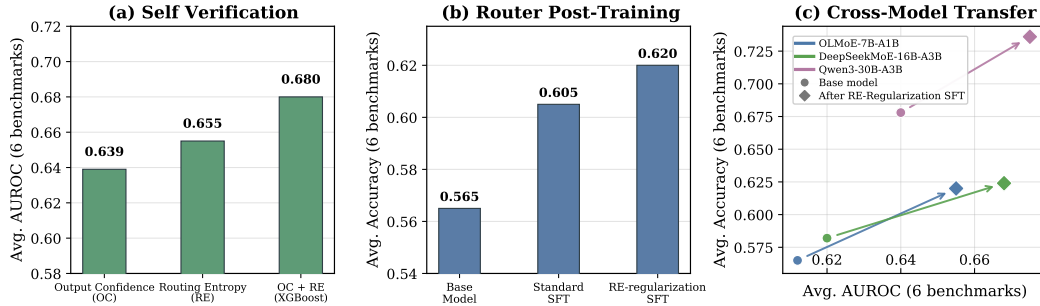


Figure 1: **Performance summary across three downstream analyses.** (a) **Self-Verification:** Routing Entropy (RE) as a single metric outperforms the Output Confidence (OC) baseline, while OC+RE fusion with XGBoost achieves the highest average AUROC. (b) **Router Post-Training:** RE-regularized SFT consistently improves average accuracy over both the base model and standard SFT. (c) **Cross-Model Transfer:** Across diverse MoE architectures, RE-regularized post-training yields simultaneous gains in both average AUROC and average accuracy, indicating robust cross-architecture generalization.

2 Related Works

2.1 Self-Verification and Uncertainty Estimation in LLMs

Accurately estimating the uncertainty of Large Language Models (LLMs) is vital for mitigating hallucinations (Ji et al., 2023). Traditional approaches primarily rely on output-based metrics, such as the maximum softmax probability or sequence likelihood (Jiang et al., 2020). However, modern LLMs often exhibit severe overconfidence, especially after alignment tuning (e.g., RLHF), rendering output confidence poorly calibrated (OpenAI, 2023). To address this, recent works extract internal hidden states to train auxiliary classifiers or linear probes for truthfulness prediction (Azaria & Mitchell, 2023; Zou et al., 2023). While effective, these methods require labeled datasets and additional training overhead. In contrast, our Routing Entropy (RE) offers an internal metric inherent to MoE architectures, bypassing the calibration issues of output logits. Recent studies have further explored router properties and optimization in MoE architectures, such as routing dynamics (Li et al., 2025b), routing control (Li et al., 2025a), and manifold alignment (Li et al., 2026).

2.2 Regularization Strategies for LLM Post-training

While SFT improves instruction-following, it often causes systemic overconfidence (OpenAI, 2023). To mitigate this, prior works typically apply regularizations directly to the output distribution, such as label smoothing or confidence penalties (Szegedy et al., 2016; Pereyra et al., 2017). However, in MoE architectures, the rich epistemic signals within the routing mechanism remain largely unexploited during training. Unlike traditional output-level penalties, we introduce a novel internal regularization. By actively penalizing or encouraging routing entropy based on ground-truth correctness, we explicitly align the router’s internal ambiguity with its actual generation accuracy.

3 Preliminaries

3.1 Mixture-of-Experts and Routing Mechanism

A MoE model decomposes the standard feed-forward layer into multiple expert sub-networks and a routing mechanism that dynamically assigns each input token to a subset of experts. Formally, given an input representation $\mathbf{h} \in \mathbb{R}^d$ at a given layer, the MoE layer computes:

$$\mathbf{y} = \sum_{i=1}^N g_i(\mathbf{h}) \cdot \text{Expert}_i(\mathbf{h}), \quad (1)$$

where N is the number of experts, $g_i(\mathbf{h})$ is the routing weight for expert i , and Expert_i denotes the i -th expert network (typically a two-layer MLP).

The routing mechanism is implemented by a gating network (router) that takes $\mathbf{h}$ as input and produces a distribution over experts. The router first computes logits via a linear projection:

$$\mathbf{z} = \mathbf{h}\mathbf{W}_g + \mathbf{b}_g, \quad (2)$$

where $\mathbf{W}_g \in \mathbb{R}^{d \times N}$ and $\mathbf{b}_g \in \mathbb{R}^N$ are learnable parameters. The routing weights are then obtained by applying the softmax function:

$$g_i(\mathbf{h}) = \frac{\exp(z_i)}{\sum_{j=1}^N \exp(z_j)}. \quad (3)$$

The routing distribution reflects the model’s internal uncertainty over expert selection. When the router assigns similar weights to multiple experts (*i.e.*, high entropy in the routing distribution), the routing decision is ambiguous; conversely, when one expert dominates (*i.e.*, low entropy), the routing is more decisive. This routing entropy has been shown to correlate with prediction uncertainty and can be leveraged for calibrated uncertainty estimation.

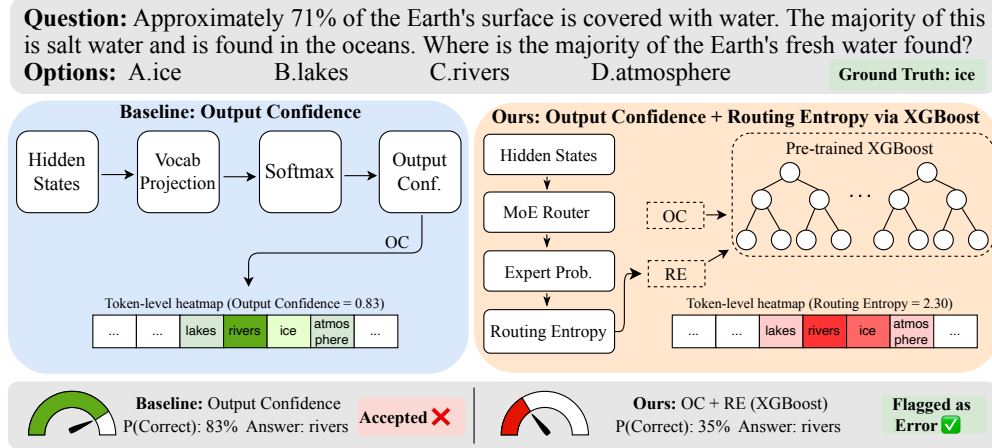


Figure 2: **A case study comparing output confidence (OC) with our proposed Routing Entropy (RE) for self verification.** When predicting the incorrect answer (“rivers”), the baseline method (left) fails to detect the error due to the model’s severe overconfidence at the output layer ($P(\text{Correct}) = 83\%$). In contrast, our approach (right) captures the model’s internal hesitation through the MoE routing distribution. The high routing entropy ($RE = 2.30$) during the generation of the hallucinated token reveals the model’s true uncertainty. By fusing the external output confidence (OC) and internal routing entropy (RE) via a simple XGBoost classifier, our method effectively recalibrates the probability of correctness (35%) and successfully flags the error.

3.2 Output Confidence and Entropy as Self-Verifiers

In the context of Large Language Models (LLMs), a self-verifier aims to quantify the model’s uncertainty regarding its generated response. A reliable self-verifier should assign a high uncertainty score to erroneous outputs (e.g., hallucinations or incorrect reasoning) and a low uncertainty score to factually and logically correct answers.

Standard uncertainty estimation relies on the output softmax distribution over the vocabulary. For a generated token, let $p(v)$ be the probability of generating token v from the vocabulary $\mathcal{V}$. Two widely used output-level metrics are **Output Confidence (OC)** and Output Entropy (OE). Output Confidence is defined as the maximum softmax probability:

$$OC = \max_{v \in \mathcal{V}} p(v). \quad (4)$$

Similarly, Output Entropy measures the vocabulary-level ambiguity:

$$OE = - \sum_{v \in \mathcal{V}} p(v) \log p(v). \quad (5)$$

To evaluate an entire sequence, these token-level scores are typically averaged. However, modern LLMs are prone to overconfidence, meaning $p(v)$ often exhibits artificially high values even during incorrect reasoning steps. This significantly diminishes the discriminative power of output-level verifiers.

4 Application I: Routing Entropy for Robust Self-Verification

Standard self-verification relies on Output Confidence (OC), the maximum softmax probability at the vocabulary layer. However, as illustrated in Figure 2, OC often fails to reflect internal uncertainty signal. This motivates us to look beyond the output layer and leverage internal routing states. We formulate a self-verifier based strictly on the model’s internal

routing states. Let g_i denote the routing weight assigned to the i -th expert by the gating network. We define the token-level **Routing Entropy (RE)** as the entropy of this routing distribution:

$$RE = - \sum_{i=1}^N g_i \log g_i, \quad (6)$$

where N is the total number of experts.

A higher RE indicates that the router is uncertain about which experts to activate, reflecting internal ambiguity in feature processing. Conversely, a lower RE implies a decisive routing choice. To obtain the sequence-level uncertainty score, we simply aggregate this routing entropy across the generated tokens. By bypassing the vocabulary-level projection, Routing Entropy serves as a highly discriminative self-verifier that faithfully captures the model’s true epistemic uncertainty.

While Routing Entropy (RE) excels at capturing deep epistemic hesitation, standard Output Confidence (OC) still provides useful information regarding surface-level vocabulary probability. These two metrics represent distinct sources of uncertainty: OC is derived from the final vocabulary projection, whereas RE originates from the internal expert selection process. Since they are complementary, combining them can lead to a more robust self-verifier.

To effectively leverage this complementarity, we introduce a dual-track feature fusion strategy. Instead of relying on a heuristic linear combination, we employ a lightweight XGBoost classifier to capture the non-linear interactions between internal and external signals. For each generated response, we construct a feature vector $\mathbf{x} \in \mathbb{R}^2$:

$$\mathbf{x} = [OC, RE] \quad (7)$$

The XGBoost model, consisting of a collection of decision trees, maps this feature vector to a calibrated probability of correctness:

$$\hat{y} = \text{XGBoost}(\mathbf{x}) = \sigma \left(\sum_{k=1}^K f_k(\mathbf{x}) \right) \quad (8)$$

where f_k denotes an individual decision tree, K is the number of trees, and σ is the sigmoid function. This fusion mechanism allows the model to dynamically prioritize the internal routing signal when the output layer exhibits overconfidence. As illustrated in the following case study, this approach successfully intercepts factual errors that would be missed by relying on output confidence alone.

5 Application II: Learning Self-Aware Routing via Entropy Regularization

We further explore whether routers in MoE models can be explicitly trained to align their routing ambiguity with generative correctness. Our goal is to build internal uncertainty calibration directly into the model architecture during the alignment phase.

During standard SFT, the model is optimized purely for token prediction accuracy, which ignores the internal routing states. To bridge this gap, we propose an **Entropy-Regularized Post-Training** framework. We explicitly penalize high routing entropy on the ground-truth training data to encourage decisive expert selection. Consequently, when the model encounters unfamiliar queries during inference, it naturally struggles to find a confident routing path and exhibits higher routing entropy.

Given a training sample x_j , let $\mathbf{g}_j$ denote its corresponding routing weight distribution. We apply a regularization term $\mathcal{L}_{\text{reg}}(j)$ based on the entropy $\mathcal{H}$ of the routing weights:

$$\mathcal{L}_{\text{reg}}(j) \triangleq \mathcal{H}(\mathbf{g}_j) = - \sum_{i=1}^N g_{j,i} \log g_{j,i}, \quad (9)$$

where N is the total number of experts, and $g_{j,i}$ denotes the routing probability assigned to the i -th expert. By minimizing $\mathcal{L}_{\text{reg}}(j)$, we enforce the routing distribution to be sharp, reducing internal ambiguity when processing familiar concepts.

To ensure that these decisive routing patterns also lead to correct predictions, our proposed training objective applies this regularization alongside the standard cross-entropy loss $\mathcal{L}_{\text{CE}}$ defined on the outputs:

$$\mathcal{L}_{\text{task}}(j) = \mathcal{L}_{\text{CE}}(f(x_j, \mathbf{g}_j), y_j), \quad (10)$$

where $f(x_j, \mathbf{g}_j)$ is the model’s output prediction and y_j is the ground-truth label.

With a regularization coefficient $\lambda \geq 0$, the final objective on sample x_j is:

$$\mathcal{L}_{\text{RE}}(j) = \mathcal{L}_{\text{task}}(j) + \lambda \cdot \mathcal{L}_{\text{reg}}(j). \quad (11)$$

During training, we only update the router parameters while keeping all expert parameters frozen. The gradient update is performed via backpropagation on $\mathcal{L}_{\text{RE}}$ with respect to the router parameters:

$$\theta_{\text{router}}^{(t+1)} = \theta_{\text{router}}^{(t)} - \eta \nabla_{\theta_{\text{router}}} \mathcal{L}_{\text{RE}}, \quad (12)$$

where $\theta_{\text{router}}^{(t)}$ represents the parameters of the routers at training step t , and η is the learning rate.

6 Experiments

6.1 Experimental Setup

Models. We evaluate our approach on three recent open-weight MoE large language models: OLMoE-7B-A1B (Muennighoff et al., 2024), DeepSeekMoE-16B-A3B (Dai et al., 2024), and Qwen3-30B-A3B (Yang et al., 2025). Specifically, OLMoE features a 16-layer transformer with 64 experts per layer (activating 8 per token), totaling 6.9B parameters with 1.3B active parameters per token. DeepSeekMoE uses a 28-layer transformer with 2 shared and 64 routed experts per layer (activating all shared plus 6 routed experts), totaling 16.4B parameters with 2.8B active per forward pass. Qwen3-30B-A3B employs a 48-layer transformer with 128 experts per layer (activating 8 per token), totaling 30.5B parameters with 3.3B active per token. Evaluating across these diverse MoE designs ensures that the effectiveness of our Routing Entropy metric is architecture-agnostic.

Baselines. We compare our approach against standard output-level metrics: Output Confidence (OC) and Output Entropy (OE). To evaluate against internal state methods, we include INSIDE (Chen et al., 2024), a strong recent baseline that computes the covariance matrix of dense hidden states for hallucination detection.

Training Set. To train the XGBoost fusion classifier and to perform the entropy-regularized post-training, we construct a diverse training set comprising 49,000 samples. As illustrated in Figure 3, the dataset spans five major task categories: General Knowledge (BIG-Bench and SuperGLUE), Commonsense Reasoning (CommonsenseQA and SocialQA), Science QA (OpenBookQA and SciQ), Reading Comprehension (MultiRC), and Coreference Resolution (KnowRef). This broad composition guarantees comprehensive coverage of different semantic domains.

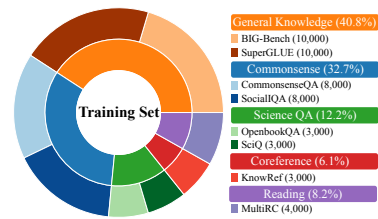


Figure 3: Training set statistics.

Benchmarks. To comprehensively evaluate the discriminative power and generation accuracy of RE, we test it across six diverse evaluation benchmarks: MMLU, HellaSwag (Hella), PIQA, ARC-Challenge (ARC-C), ARC-Easy (ARC-E) and WinoGrande (Wino). To ensure the robustness of our self-verifier, we strictly enforced a partition between training set and all evaluation benchmarks. Full details about the training set and evaluation benchmarks are provided in Appendix A.1.

6.2 Performance on Self Verification

We evaluate the performance of our approach across six benchmarks. The results are measured by the Area Under the Receiver Operating Characteristic Curve (AUROC) and Precision-Recall curve.

AUROC Analysis. Table 1 shows that Output Confidence (OC) is a strong baseline among output-level metrics. However, our proposed Routing Entropy (RE) consistently outperforms OC across all six datasets. As a single metric, RE achieves an average AUROC of 0.655, compared to 0.639 for OC. This proves that internal routing hesitation is a more reliable indicator of model errors than external output probabilities. We also test combining internal and external metrics. While a simple linear combination of OC and RE improves performance, using an XGBoost classifier to fuse these two features gives the best results. As shown in Table 1, the XGBoost combination of OC and RE achieves the highest average AUROC of 0.680.

Precision-Recall Analysis. Figure 4 further illustrates this advantage using Precision-Recall (PR) curves on the ARC-E and ARC-C datasets. The curve for RE stays clearly above the OC baseline. Moreover, the combined approach (OC + RE) achieves the highest Average Precision (AP). This demonstrates that fusing the internal routing entropy with the external output confidence is a highly effective way for self-verification. These consistent performance trends generalize seamlessly across diverse architectures, with detailed results for DeepSeekMoE and Qwen3 provided in Appendix A.2.

Table 1: **Main Results on Error Detection (AUROC) on OLMoE-7B-A1B.** Comparison of discriminative power across common reasoning and knowledge benchmarks. Output Confidence serves as a strong baseline. Our Routing Entropy (RE) achieves the highest single-metric performance, outperforming both the standard output-level metrics and the competitive hidden states baseline (INSIDE). The overall best results are in **bold**.

Module	Metric	MMLU	Hella	ARC-C	ARC-E	PIQA	Wino	Avg.
Output	Confidence (OC)	0.641	0.654	0.612	0.718	0.545	0.664	0.639
	Entropy (OE)	0.638	0.641	0.609	0.720	0.531	0.650	0.632
Routing	Confidence (RC)	0.574	0.589	0.541	0.605	0.522	0.561	0.565
	Entropy (RE)	0.671	0.668	0.639	0.742	0.537	0.675	0.655
Hidden States	INSIDE	0.645	0.660	0.625	0.730	0.540	0.670	0.645
Combined	OC + RE (Linear)	0.678	0.672	0.648	0.746	0.551	0.685	0.663
	OC + RE (XGBoost)	0.692	0.687	0.674	0.765	0.569	0.694	0.680

Table 2: **Statistical Significance Analysis of AUROC Improvement (Δ) over OC Baseline.**

Model / Method	Avg. AUROC	Δ AUROC	95% CI of Δ	p -value
OC (Baseline)	0.639	–	–	–
RE (Standalone)	0.655	+0.016	[+0.012, +0.020]	< 0.001**
OC + RE (XGBoost)	0.680	+0.041	[+0.035, +0.047]	< 0.001**

6.3 Performance on RE-Regularized Post-Training

We evaluate the effectiveness of our RE-Regularized Supervised Fine-Tuning (SFT) against standard SFT. Table 3 shows the results for both task accuracy and error detection across different amounts of training data. Detailed results on DeepSeekMoE and Qwen3 are available in Appendix A.3.

Improved Accuracy and internal uncertainty calibration. Our RE-regularized approach consistently outperforms standard SFT at every data scale. When using 100% of the labeled data, our method achieves the highest average accuracy (0.620) and a much stronger error

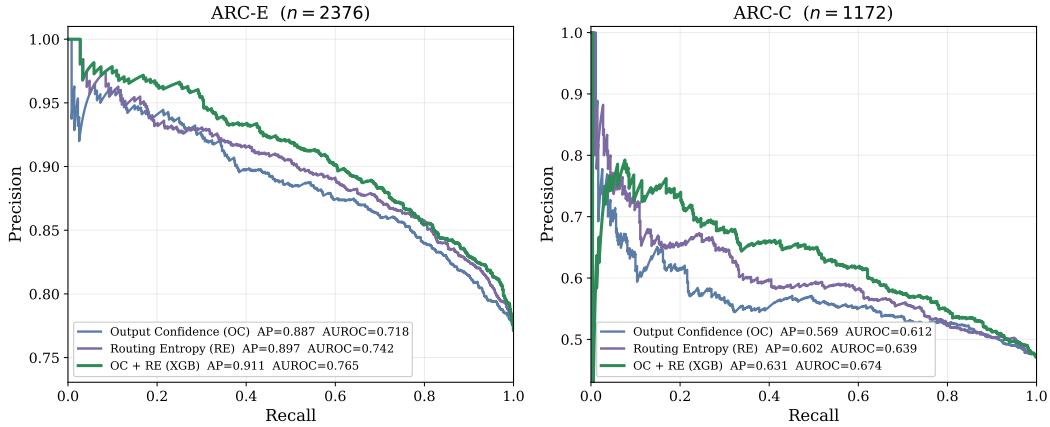


Figure 4: **Precision-Recall (PR) curves evaluating error detection performance on the ARC-E and ARC-C datasets (OLMoE-7B-A1B).** We compare the baseline Output Confidence (OC), our proposed internal Routing Entropy (RE), and their combination via an XGBoost classifier (OC + RE). Relying solely on the internal RE consistently outperforms the standard external OC baseline in both Average Precision (AP) and AUROC. By fusing these complementary signals, our combined approach (OC + RE) achieves the best overall performance across both the easier (ARC-E) and more challenging (ARC-C) tasks, demonstrating the effectiveness of utilizing routing hesitation for reliable self-verification.

detection AUROC (0.644). In contrast, standard SFT only reaches 0.605 accuracy and 0.625 AUROC. This proves that aligning routing entropy with correctness inherently calibrates the model, making its standard output confidence more reliable.

High Data Efficiency. Table 3 also highlights the strong data efficiency of our method. Remarkably, our approach trained on just 75% of the data (0.612 accuracy, 0.638 AUROC) easily surpasses the performance of standard SFT trained on the full 100% dataset (0.605 accuracy, 0.625 AUROC). This shows that internal routing regularization is a highly efficient way to train better MoE models.

Table 3: **Effectiveness of RE-Regularized Post-Training on OLMoE-7B-A1B across Data Scales.** We evaluate the impact of incorporating Routing Entropy regularization during SFT under varying proportions of ground-truth (GT) labels. Notably, our RE-regularized approach demonstrates exceptional data efficiency, surpassing the fully-trained (100% GT) standard SFT baseline using only a fraction of the training data.

Training Setup	Task Accuracy $\uparrow$			Self-Verification (AUROC) $\uparrow$		
	MMLU	ARC-C	Avg. (All 6)	MMLU	ARC-C	Avg. (All 6)
Base Model	0.580	0.530	0.565	0.615	0.590	0.612
<i>Standard SFT</i>						
25% Labeled	0.595	0.548	0.582	0.622	0.595	0.618
50% Labeled	0.603	0.561	0.591	0.626	0.598	0.621
75% Labeled	0.609	0.574	0.598	0.629	0.602	0.624
100% Labeled	0.615	0.585	0.605	0.632	0.605	0.625
<i>RE-Regularized SFT (Ours)</i>						
25% Labeled	0.598	0.554	0.587	0.636	0.609	0.622
50% Labeled	0.610	0.572	0.599	0.645	0.615	0.631
75% Labeled	0.621	0.589	0.612	0.652	0.621	0.638
100% Labeled	0.628	0.602	0.620	0.658	0.628	0.644

Table 4: **Ablation Study on Spatiotemporal Aggregation (OLMoE-7B-A1B)**. We analyze AUROC of Routing Entropy when aggregated over different transformer depths (Layers 1-16) and generation steps (Tokens). Isolating the signals from the deeper layers (9-16) and the concluding reasoning tokens (Last 5) yields the most faithful self verification.

Layer Aggregation	Token Aggregation	MMLU	ARC-C	Avg. (All 6 Tasks)
All Layers (1 to 16)	All Tokens	0.624	0.602	0.615
All Layers (1 to 16)	Last 5 Tokens	0.645	0.618	0.632
Shallow Layers (1 to 8)	Last 5 Tokens	0.562	0.531	0.554
Deep Layers (9 to 16)	Last 5 Tokens (Ours)	0.671	0.639	0.655

6.4 Ablation Study

Deep Layers and Final Tokens are More Informative. Table 4 compares different spatiotemporal aggregation methods. We find that routing signals from shallow layers (1-8) are much weaker (0.554 Avg. AUROC) than those from deep layers. Furthermore, focusing on the last 5 tokens of the generation provides a clearer uncertainty signal than averaging over all tokens. Our final strategy, using deep layers and the last 5 tokens, achieves the best average performance of 0.655. This suggests that the model’s true internal hesitation occurs during complex semantic processing in later stages.

Table 5: **Ablation Study on Feature Complementarity (OLMoE-7B-A1B)**. We train an XGBoost classifier using different subsets of the four uncertainty features. The combination of Output Confidence and Routing Entropy (OC + RE) forms the optimal minimal subset.

Features Used				Self-Verification (AUROC)			
OC	OE	RC	RE	MMLU	ARC-C	PIQA	Avg. (All 6 Tasks)
✓	✓			0.642	0.614	0.542	0.641
		✓	✓	0.673	0.642	0.539	0.657
✓		✓		0.648	0.618	0.540	0.646
	✓		✓	0.685	0.662	0.550	0.673
✓			✓	0.692	0.674	0.569	0.680
✓	✓		✓	0.688	0.665	0.568	0.674
✓	✓	✓	✓	0.683	0.655	0.571	0.670

Table 6: **Sensitivity Analysis on Token Window Size k (OLMoE-7B-A1B)**.

Token Window Size (k)	MMLU	ARC-C	Avg. (All 6 Tasks)
$k = 1$	0.648	0.615	0.636
$k = 3$	0.665	0.632	0.651
$k = 5$ (Ours)	0.671	0.639	0.655
$k = 10$	0.667	0.635	0.652
$k = \text{all}$	0.654	0.622	0.644

Output Confidence and Routing Entropy are the Optimal Pair. Table 5 explores different feature sets for the XGBoost classifier. We find that OC and RE form the most effective pair, reaching 0.680 average AUROC. Adding more features, such as Output Entropy (OE) or Router Confidence (RC), does not improve the results and even causes slight performance degradation. This indicates that OC and RE already capture the most relevant and complementary information for error detection, while additional features mainly introduce redundancy and noise.

Sensitivity to Regularization Strength. For the entropy-regularized post-training (Application II), the hyperparameter λ controls the trade-off between the standard generative loss and our internal routing penalty. We observe that a moderate regularization strength

Table 7: **Ablation Study on Regularization Strength λ (OLMoE-7B-A1B).** We evaluate how the weight of the Routing Entropy penalty affects both generative performance and self-verification. A moderate value ($\lambda = 0.05$) provides the best trade-off.

Regularization Coefficient (λ)	Avg. Task Accuracy $\uparrow$	Avg. Error Detection (AUROC) $\uparrow$
$\lambda = 0.00$ (Standard SFT)	0.605	0.625
$\lambda = 0.01$	0.611	0.634
$\lambda = 0.05$ (Ours)	0.620	0.644
$\lambda = 0.20$	0.614	0.649
$\lambda = 1.00$	0.552	0.610

($\lambda = 0.05$) achieves the optimal balance, yielding the highest task accuracy (0.620) and strong error detection (0.644). If λ is too small, the router fails to learn internal uncertainty calibration. Conversely, an excessively large λ (e.g., 1.0) forces the router to be artificially sharp even on unfamiliar tokens, which corrupts the expert representations and severely degrades generation accuracy (0.552).

7 Conclusion

In this paper, we show that the internal Routing Entropy (RE) of MoE LLMs is a highly effective self-verifier. Our experiments demonstrate that RE consistently outperforms traditional output confidence in detecting model errors. We find that the most reliable uncertainty signals come from the deeper layers and concluding tokens of the generation. Furthermore, incorporating RE-regularization during post-training significantly improves the model’s internal uncertainty calibration and task accuracy. These findings suggest that the internal decision-making process of MoEs holds valuable, untapped information for building more reliable and trustworthy large language models.

References

- Amos Azaria and Tom Mitchell. The internal state of an llm knows when it’s lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, 2023.
- Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.
- Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. Inside: Llm’s internal states retain the power of hallucination detection. *arXiv preprint arXiv:2402.03744*, 2024.
- Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.
- Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huajian Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.
- Ali Emami, Paul Trichelair, Adam Trischler, Kaheer Suleman, Hannes Schulz, and Jackie Chi Kit Cheung. The knowref coreference corpus: Removing gender and number cues for difficult pronominal anaphora resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.
- William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.
- Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8: 423–438, 2020.
- Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.
- Zhongyang Li, Ziyue Li, and Tianyi Zhou. C3po: Critical-layer, core-expert, collaborative pathway optimization for test-time expert re-mixing, 2025a. URL <https://arxiv.org/abs/2504.07964>.
- Zhongyang Li, Ziyue Li, and Tianyi Zhou. R2-t2: Re-routing in test-time for multimodal mixture-of-experts, 2025b. URL <https://arxiv.org/abs/2502.20395>.
- Zhongyang Li, Ziyue Li, and Tianyi Zhou. Routing manifold alignment improves generalization of mixture-of-experts llms. In *International Conference on Learning Representations*, volume 2026, pp. 21576–21595, 2026.

- Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 3214–3252, 2022.
- Matéo Mahaut, Laura Aina, Paula Czarowska, Momchil Hardalov, Thomas Müller, and Lluís Màrquez. 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)*, pp. 4554–4570, 2024.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.
- Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. Olmoe: Open mixture-of-experts language models. *arXiv preprint arXiv:2409.02060*, 2024.
- OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.
- Gabrijela Perković, Antun Drobniak, and Ivica Botički. Hallucinations in llms: Understanding and addressing challenges. In *2024 47th MIPRO ICT and electronics convention (MIPRO)*, pp. 2084–2088. IEEE, 2024.
- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.
- Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.
- Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarsz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.
- Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.
- Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.
- Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text (W-NUT)*, 2017.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mantyh, Dario Amodei, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A Appendix

A.1 Details of Benchmarks and Training Set

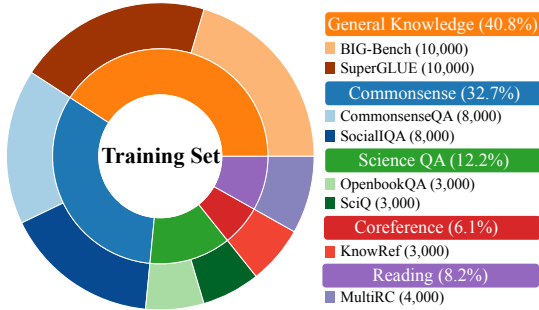


Figure 5: Distribution of Task Categories in the Training Set.

Evaluation Benchmarks	Sample Size
MMLU	14,042
HellaSwag	10,042
PIQA	1,838
WinoGrande	1,267
ARC-Challenge	1,172
ARC-Easy	2,376

Figure 6: Statistics of the core evaluation benchmarks. These datasets are strictly held out from the router post-training process.

Our training corpus consists of 49,000 curated instances across five core functional domains. This selection ensures that the MoE router encounters a diverse range of cognitive dimensions during the entropy-regularization phase:

- **BIG-Bench** (Srivastava et al., 2022): A massive collaborative suite encompassing tasks from logical induction to linguistic analysis, aimed at quantifying emergent behaviors in large-scale models.
- **SuperGLUE** (Wang et al., 2019): An advanced collection of NLU challenges, including causal reasoning and disambiguation, designed to be significantly more difficult than the original GLUE tasks.
- **CommonsenseQA** (Talmor et al., 2019): A dataset requiring prior world knowledge to solve multiple-choice questions where the correct answer is often implicit.
- **SocialQA** (Sap et al., 2019): A benchmark specifically focused on social intelligence, testing a model’s ability to reason about human interactions and emotional responses.
- **OpenBookQA** (Mihaylov et al., 2018): Modeled after open-book exams, this task requires combining provided facts with external common sense to reach a conclusion.
- **SciQ** (Welbl et al., 2017): A science-heavy dataset featuring questions from biology, chemistry, and physics, emphasizing factual recall in STEM fields.
- **MultiRC** (Khashabi et al., 2018): A complex reading comprehension task where each question may have multiple correct answers, necessitating evidence integration across several sentences.
- **KnowRef** (Emami et al., 2019): A pronoun resolution task that uses contextual difficulty to prevent models from relying on simple pattern matching or gender bias.

We assess the performance and self-verification capabilities of our models on six prominent benchmarks that span general knowledge and logical reasoning:

- **MMLU** (Hendrycks et al., 2021): A high-stakes multitask benchmark covering 57 distinct subjects. It serves as a primary measure for a model’s breadth of academic and professional knowledge.
- **HellaSwag** (Zellers et al., 2019): A challenging task for models to predict the most logical completion of a scenario. It is designed to be easy for humans (~95%) but difficult for uncalibrated models.
- **PIQA** (Bisk et al., 2020): This benchmark focuses on physical interaction and common sense, asking how everyday objects are used in various contexts.
- **ARC-Challenge** (Clark et al., 2018): Contains the difficult subset of grade-school science questions that require advanced reasoning beyond simple keyword matching.
- **ARC-Easy** (Clark et al., 2018): A standard science QA dataset focusing on fundamental factual knowledge and elementary reasoning.
- **WinoGrande** (Sakaguchi et al., 2020): An adversarial version of the Winograd Schema Challenge, requiring sophisticated pronoun resolution that cannot be solved through simple statistical associations.

A.2 Detailed Error Detection Results on DeepSeekMoE-16B-A3B and Qwen3-30B-A3B

To comprehensively demonstrate that the effectiveness of Routing Entropy (RE) is not limited to a specific architecture, we extend our self-verification analysis (Application I) to two additional state-of-the-art MoE models: DeepSeekMoE-16B-A3B (Dai et al., 2024) and Qwen3-30B-A3B (Yang et al., 2025).

DeepSeekMoE introduces a fine-grained expert segmentation and shared expert mechanism, while Qwen3 employs a massive routing space with 128 experts per layer. Despite these structural differences, the internal hesitation captured by the routing distribution remains highly indicative of the generation correctness. All evaluations in this section are performed on the standard SFT versions of these models to align with our baselines.

Table 8 and Table 9 present the detailed error detection performance (AUROC). We observe the exact same trend as in our main experiments: our proposed Routing Entropy (RE) consistently outperforms both the standard Output Confidence (OC) and the dense hidden-state baseline (INSIDE) across all datasets.

Specifically, on DeepSeekMoE, RE achieves an average AUROC of 0.646 compared to the OC baseline of 0.624. On Qwen3, RE yields 0.669 compared to the 0.648 OC baseline. Furthermore, fusing the internal RE with the external OC via the XGBoost classifier yields the highest self-verification performance on both models (0.670 for DeepSeekMoE and 0.692 for Qwen3). This confirms that RE provides a robust, architecture-agnostic uncertainty signal that generalizes seamlessly across diverse MoE scaling paradigms.

Table 8: **Error Detection (AUROC) on DeepSeekMoE-16B-A3B.** The internal Routing Entropy (RE) consistently surpasses the standard Output Confidence (OC) baseline, confirming that our findings generalize to architectures with shared and fine-grained experts.

Module	Metric	MMLU	Hella	ARC-C	ARC-E	PIQA	Wino	Avg.
Output	Confidence (OC)	0.625	0.630	0.610	0.700	0.535	0.644	0.624
Hidden States	INSIDE	0.635	0.645	0.615	0.710	0.545	0.660	0.635
Routing	Entropy (RE)	0.652	0.655	0.630	0.725	0.550	0.664	0.646
Combined	OC + RE (XGBoost)	0.675	0.680	0.655	0.750	0.570	0.690	0.670

Table 9: **Error Detection (AUROC) on Qwen3-30B-A3B**. The performance trend holds on this larger-scale MoE model with 128 experts per layer. The XGBoost combination of OC and RE consistently achieves the highest discriminative power.

Module	Metric	MMLU	Hella	ARC-C	ARC-E	PIQA	Wino	Avg.
Output	Confidence (OC)	0.655	0.660	0.630	0.720	0.555	0.668	0.648
Hidden States	INSIDE	0.662	0.675	0.640	0.730	0.565	0.676	0.658
Routing	Entropy (RE)	0.678	0.682	0.655	0.740	0.570	0.689	0.669
Combined	OC + RE (XGBoost)	0.705	0.700	0.680	0.765	0.590	0.712	0.692

A.3 Detailed Post-Training Results on DeepSeekMoE-16B-A3B and Qwen3-30B-A3B

To provide further granularity to the generalization results, we present the detailed breakdown of task accuracy and self-verification performance for DeepSeekMoE-16B-A3B and Qwen3-30B-A3B.

Table 10 compares the standard Supervised Fine-Tuning (SFT) against our proposed RE-Regularized SFT. Consistent with our findings on OLMoE, standard SFT improves the generation accuracy but provides disproportionately small gains in error detection (AUROC) due to the overconfidence penalty inherently induced by standard cross-entropy loss.

In contrast, applying our Routing Entropy Regularization during the post-training phase yields simultaneous improvements in both dimensions across both architectures. For DeepSeekMoE, RE-Regularization pushes the average AUROC from 0.624 to 0.668, while also securing a slight gain in task accuracy. For the larger Qwen3 model, the regularizer proves equally effective, elevating the average AUROC from 0.648 to 0.675 and task accuracy from 0.725 to 0.736. These detailed metrics confirm that incorporating internal routing hesitation into the training objective serves as a universally effective calibrator for MoE architectures, regardless of their expert scale or routing topology.

Table 10: **Detailed Impact of RE-Regularized Post-Training on DeepSeekMoE-16B-A3B and Qwen3-30B-A3B**. We breakdown the generative accuracy and self-verification (AUROC) on representative benchmarks (MMLU and ARC-C) alongside the 6-task average. All fine-tuned models utilize 100% of the labeled training data.

Base Model	Training Setup	Task Accuracy $\uparrow$			Self-Verification (AUROC) $\uparrow$		
		MMLU	ARC-C	Avg.	MMLU	ARC-C	Avg.
DeepSeekMoE	Base Model	0.602	0.548	0.582	0.621	0.595	0.620
	Standard SFT	0.630	0.605	0.618	0.625	0.610	0.624
	RE-Regularized SFT	0.638	0.612	0.624	0.675	0.652	0.668
Qwen3-30B	Base Model	0.695	0.662	0.678	0.645	0.622	0.640
	Standard SFT	0.740	0.710	0.725	0.655	0.630	0.648
	RE-Regularized SFT	0.752	0.722	0.736	0.686	0.660	0.675

A.4 Data Decontamination and XGBoost Training Protocols

To guarantee zero data leakage between the 49k training corpus and the 6 evaluation benchmarks, we perform exact n-gram overlap checks. Table 11 reports the 13-gram overlap percentage and exact string match counts across all test sets.

Furthermore, Table 12 lists the hyperparameter search space and selected optimal configuration for the XGBoost classifier, tuned strictly via 5-fold cross-validation on the training set.

Table 11: **Data Decontamination Statistics.** Overlap analysis between the 49k training set and evaluation benchmarks.

Evaluation Benchmark	Test Sample Size	13-gram Overlap (%)	Exact Match (Count)
MMLU	14,042	0.04%	0
HellaSwag	10,042	0.01%	0
PIQA	1,838	0.00%	0
ARC-Challenge	1,172	0.08%	0
ARC-Easy	2,376	0.04%	0
WinoGrande	1,267	0.00%	0
Overall	30,737	0.03%	0

Table 12: **XGBoost Hyperparameter Search Space and Optimal Configuration.**

Hyperparameter	Search Space	Selected	Description
max_depth	{2, 3, 4, 5, 6}	3	Maximum depth of a tree.
n_estimators	{50, 100, 200, 300}	100	Number of boosting rounds.
learning_rate	{0.01, 0.05, 0.1, 0.2}	0.05	Step size shrinkage.
subsample	{0.6, 0.8, 1.0}	0.8	Subsample ratio of instances.
min_child_weight	{1, 3, 5}	5	Minimum sum of instance weight.
colsample_bytree	{1.0}	1.0	Subsample ratio of columns.
Optimization Metric	Validation AUROC (via 5-fold Cross-Validation)		