Knowledge Injection Exists in MoE? Exploring Expert-Aware Contrast Decoding in MoE for Mitigating LLMs' Hallucinations

Xinyue Fang[1], Zhiliang Tian[1]*, Zhen Huang[1]*, Ziyi Pan[1], Zhihua Wen[1], Xi Wang[1], Quntian Fang[1], Dongsheng Li [1]

[1]College of Computer Science and Technology, National University of Defense Technology

{ fangxinyue, tianzhiliang, huangzhen, panziyi, zhwen, wangxi25, fangquntian, dsli }@nudt.edu.cn

Abstract

Existing LLM hallucination mitigation methods, including prompt engineering and model optimization, either hardly alter models' internal knowledge or have poor cross-domain generalization. Contrastive decoding mitigates hallucinations by using layer-wise differences in LLMs. However, prior studies only explore transformer-based models (e.g., GPT), ignoring other effective frameworks like mixture-of-experts (MoE) models. Since MoE alters the traditional transformer architecture, we conduct empirical studies to investigate whether similar layer-wise differences exist in MoEs. Our results show that they do not exist in MoE with shared experts; nevertheless, across different MoEs, higher layers exhibit distinct expert activation patterns between factual and non-factual outputs. Building on these, we propose **EAACD**, an expert-aware adaptive contrast decoding that uses expert differences in MoE's higher layers to mitigate hallucinations on QA tasks. EAACD splits high-layer experts into a higher-reliability group and several lower-reliability groups based on their confidence and consistency. It contrasts the higher-reliability group's prediction with each lower-reliability group's prediction to calibrate the model's original predictions. To strengthen this contrast, EAACD amplifies hallucinations from lower-reliability experts via attention and masking to provide stronger negative references. EAACD outperforms all baselines on four datasets[1].

1 Introduction

Large language models (LLMs) show strong performance but suffer from hallucination, limiting their application (Fang et al., 2025c; Liu et al., 2025a). Existing mitigation methods mainly include: (1) *Prompt engineering* uses task instructions to guide models to generate factual outputs (Mondal et al., 2024). It is easy to apply, but cannot fundamentally alter the model's internal knowledge (Cheng et al., 2025). (2) *Model parameter optimization* fine-tunes LLMs with supplementary knowledge (Wang et al., 2022). These methods mitigate hallucinations by calibrating the model's internal knowledge. But they lack domain generalization, and potential errors in fine-tuning data may exacerbate the hallucination (Iyer et al., 2022).

To improve domain generalization and reduce reliance on fine-tuning data, researchers propose contrastive decoding, which improves the output's accuracy by using less reliable outputs as negative references (Li et al., 2023). This approach includes: (1) *Inter-model contrastive decoding* compares the original model's outputs with an unreliable model's outputs (Yang et al., 2024a). These methods use the unreliable model's outputs as negative references, removing such negative information from the original model's outputs. However, if the unreliable model produces factual outputs, this may adversely affect the original model's predictions (Yin et al., 2025). (2) *Intra-model contrastive decoding* leverages internal differences within a single model, contrasting logits between lower and higher layers to obtain next-token probabilities (Yang et al., 2025). Researchers (Chuang et al., 2023) find that transformer-based models (e.g., GPT) have a "knowledge injection" phenomenon where higher layers incorporate more factual knowledge during generation, while the lower layers store less factual knowledge. Following this, researchers develop several variants (Gera et al., 2023; Das et al., 2025), which use layer-wise differences to mitigate hallucinations without external resources.

Recently, the Mixture-of-Experts (MoE) architecture has become a popular approach for building large LLMs (e.g., DeepSeek-R1 (Guo et al., 2025), GPT-4 (Achiam et al., 2023)) due to its excellent performance. However, even advanced MoE models still suffer from hallucination (Su

*Corresponding Author

[1]Code is: anonymous.4open.science/r/EAACD-D388/

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 39326–39343

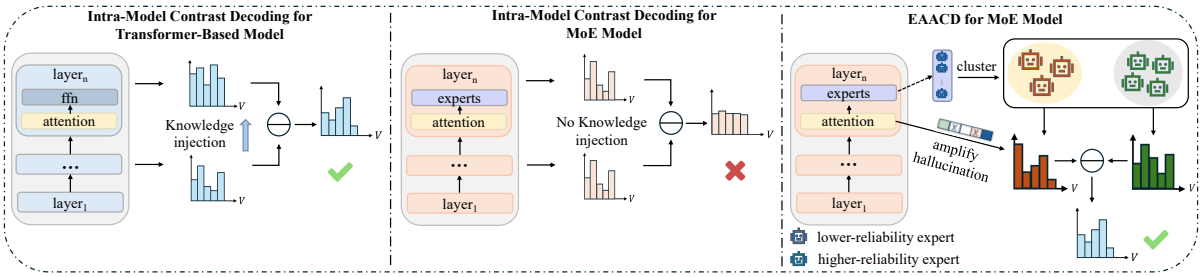

Figure 1: Overview of prior intra-model contrast decoding and EAACD (ours). Prior intra-model contrast decoding relies on "knowledge injection" in transformer-based models, limiting its applicability to MoE. Our EAACD leverages expert diversity within MoE layers to mitigate hallucinations effectively.

et al., 2025). MoE splits the model into multiple experts and dynamically routes each token to a small subset of them (Zhao et al., 2024). This design alters the model's structure, making the existing intra-model contrastive decoding for classical transformer-based models face challenges when mitigating hallucinations in MoE models. Because MoE models may not exhibit the same layer-wise differences as classical transformer-based models.

To address these problems, we explore whether MoE models exhibit the "knowledge injection" found in classical transformer-based models. We find that this phenomenon depends on the MoE architecture: (1) **the "knowledge injection" only occurs in MoE without shared experts, but not in MoE with shared experts.** This suggests that the intra-model contrastive decoding may only be effective in the MoE model without shared experts. Given this limitation, we explore whether we could use differences among experts for contrastive decoding. By examining expert activation patterns during factual versus non-factual outputs, we find: (2) **higher layers consistently show distinct expert activation patterns between factual and non-factual outputs across all MoE architectures.** This insight enables us to use higher-layer expert differences for intra-model contrastive decoding to mitigate hallucinations in all MoEs.

In this paper, we propose an expert-aware adaptive contrastive decoding strategy (**EAACD**) that separates higher-reliability experts from lower-reliability experts; and higher-reliability experts treat amplified hallucinations in lower-reliability experts as a negative reference to enhance factuality via contrastive decoding (Fig. 1). Our empirical findings show that across MoE architectures, higher layers exhibit divergent expert activation when generating factual versus non-factual outputs. Since the router aims to select experts most suitable for specific scenarios, its differing preferences in factual and non-factual situations indicate varying expert capabilities in factual output generation. To use the inherent difference between experts for contrastive decoding, we split experts into one high-reliability and multiple low-reliability groups based on confidence and consistency. To ensure lower-reliability experts provide a valid negative reference during contrast, we amplify hallucinations in low-reliability experts via attention and masking mechanisms. Finally, our contrast decoding module adaptively penalizes lower-reliability predictions based on their differences from the higher-reliability prediction during contrast, and calibrates original predictions using contrast results to mitigate hallucinations on QA tasks.

Our main contributions are: (1) To our knowledge, we are the first to explore the "knowledge injection" in MoE models and expert activation differences between factual and non-factual outputs. (2) We reveal how "knowledge injection" and expert activation differences during factual versus non-factual outputs relate to the MoE's architecture. (3) We propose an expert-aware adaptive contrastive decoding method to mitigate MoE models' hallucinations without external resources. (4) Our method is SOTA across four datasets.

2 Related Work

Hallucination Mitigation in LLMs aims to reduce factual inconsistencies in LLMs outputs.(Si et al., 2022). Existing mitigation methods: **(1) Prompt Engineering** improves the model's factuality by including a few examples in the prompt (Brown et al., 2020). Chain-of-Thought (Mondal et al., 2024) prompting guided models through intermediate steps to boost factuality. Ji et al. (2023) proposed an iterative self-reflection incorporating prior outputs for correction. **(2) Optimizing Model**

Parameters aligns the model's internal knowledge with facts (Wang et al., 2022). *Supervised Fine-Tuning* (SFT) helps LLMs learn task-specific features from labeled data and improve accuracy (Iyer et al., 2022; Shi et al., 2023). *Model Editing* removes hallucinations via localization and intervention (Zheng et al., 2023; Zhang et al., 2024).

Contrastive decoding mitigates hallucinations by comparing outputs between models or different parts of the same model (Li et al., 2023; Chuang et al., 2023), mainly divided into two types: **(1) Inter-model contrastive decoding** compares outputs from a large model with a less accurate model (Chen et al., 2024). Li et al. (2023) used a smaller model to calibrate the larger model's output. Zhang et al. (2025) fine-tuned the original model on hallucinated data to construct a less factual model as a negative comparator for the original model. **(2) Intra-model contrastive decoding** mitigates hallucinations by leveraging internal discrepancies in the model (Wu et al., 2025; Shi et al., 2024a). DoLa (Chuang et al., 2023) contrasted the model's final-layer prediction with earlier layers to highlight reliable outputs. Das et al. (2025) selected lower layers based on layer-wise entropy, improving DoLa's performance. Context-Aware Decoding (Shi et al., 2024b) compared context-aware with context-agnostic outputs to mitigate hallucinations.

Mixture-of-Experts (MoE) (Li et al., 2025) enables LLMs to scale model capacity while maintaining efficiency. MoE architectures are categorized as: **(1) MoE without shared experts** fully decouples experts (Jiang et al., 2024). Mixtral 8x7B (Jiang et al., 2024) used equally sized, independent experts and activated top-2 during inference. LLaMA-MoE (Zhu et al., 2024) employed non-overlapping partitions for better generalization. **(2) MoE with shared experts** has shared experts always activated, while others are routed (Zhao et al., 2024). DeepSeekMoE (Dai et al., 2024) and XMoE (Yang et al., 2024b) split experts into sub-experts and activated top-K at inference. Cartesian-MoE (Su et al., 2025) merged shared and routed experts to enlarge compositional capacity.

3 Empirical Study

We conduct an empirical study to explore general differences in MoE models that enable intra-model contrast decoding for hallucination mitigation. It has two key experiments. First, we examine whether MoE models with different architectures exhibit the "knowledge injection" phenomenon observed in classical transformer-based models. Results show "knowledge injection" only occurs in MoE without shared experts, but not in MoE with shared experts. Therefore, we turn to explore more general differences between factual and non-factual outputs across all MoE models.

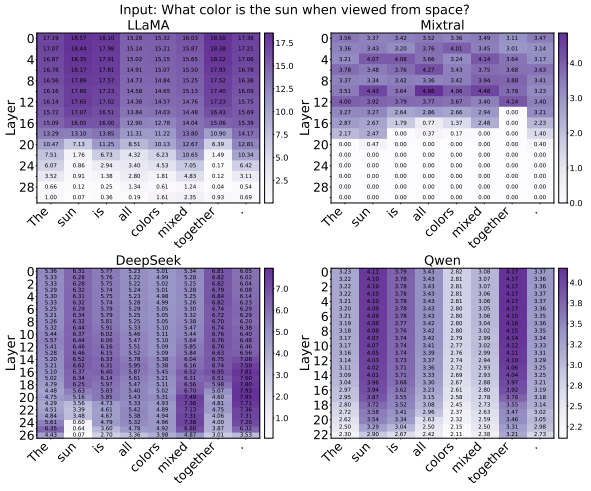

Figure 2: JSD (scaled by 10^5) between the final layer and each lower layer across different models. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is from TruthfulQA dataset; see App. A for results on other datasets.

3.1 Experimental setting

We investigate two mainstream MoE architectures: LLaMA-MoE (Zhu et al., 2024) and Mixtral (Jiang et al., 2024) for MoE without shared experts, Deepseek-MoE (Dai et al., 2024) and Qwen-MoE (Team, 2024) for MoE with shared experts. Our experiments cover three QA datasets widely used for hallucination analysis: TruthfulQA for commonsense QA (Lin et al., 2022), StrategyQA for commonsense reasoning (Geva et al., 2021), and GSM8K for math reasoning (Cobbe et al., 2021).

3.2 RQ 1: Do MoE models also exhibit the "knowledge injection" phenomenon?

To investigate whether MoE models also exhibit the "knowledge injection" phenomenon, we introduce the early exit mechanism (Teerapittayanon and McDanel, 2016): at each time step, we apply a language head (Chuang et al., 2023) to every layer's hidden state to obtain the layer's next-token logits. And we convert these logits to probabilities via softmax. We then calculate the Jensen-Shannon Divergence (JSD) between each layer's and the fi-

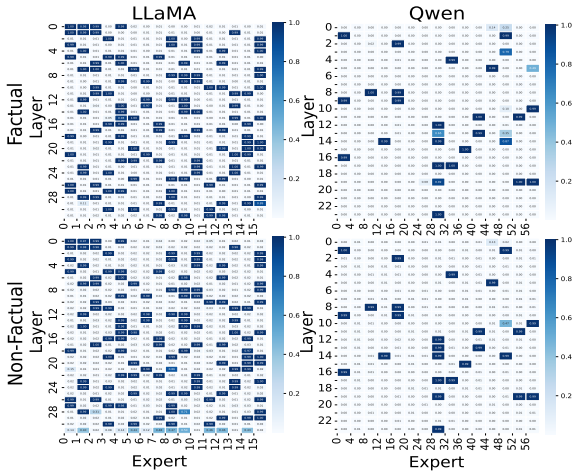

Figure 3: Expert activation patterns across layers in different MoE architectures during factual and non-factual generation on TruthfulQA. Column labels indicate expert indices. We show results for LLaMA MoE (no shared experts) and Qwen MoE (with shared experts; every fourth expert shown); full results in App. C.

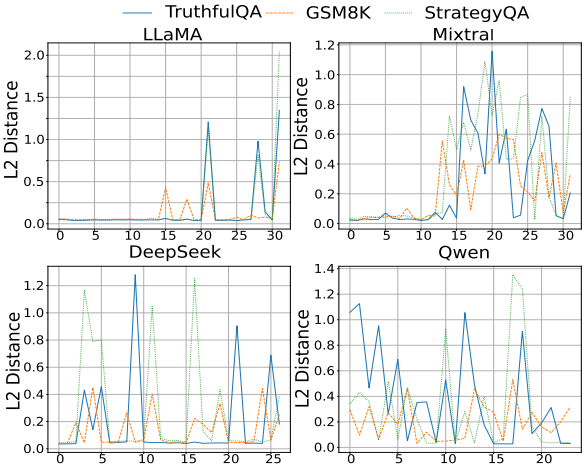

Figure 4: Differences in expert activation patterns between factual and non-factual outputs for various MoE architectures and datasets (x-axis is layer indices).

nal layer's probability. The JSD changes indicate how predictions evolve across layers (Fig. 2).

Fig. 2 shows that in MoE models without shared experts, JSD values between the lower and final layer start high but decrease as the layer goes deeper. A sharp drop in higher layers indicates the model substantially changes its prediction at these layers, which reflects the occurrence of the "knowledge injection". However, in MoE models with shared experts, JSD values remain low across layers with minimal change. **This indicates the "knowledge injection" does not exhibit in MoE with shared experts.** We provide dataset-level statistics and additional results on more MoE models in App. J and App. K.

3.3 RQ 2: How do expert activation patterns in MoE differ during factual and non-factual generation?

Since the "knowledge injection" doesn't exist in all MoE models, we investigate general differences that could support intra-model contrastive decoding. We analyze expert activation patterns (which experts the router activates (Szatkowski et al., 2024)) in MoE models during generating factual and non-factual outputs. To obtain non-factual generations under the same questions, we employ a malicious system prompt (see App. B) to bias the model toward non-factual outputs. Because the model generates both outputs under the same input question, any observed differences in expert activation can be attributed to changes in the model's generation behavior rather than input question variations. Then, we select samples that the model outputs factually under the normal prompt but non-factually under the malicious prompt. We record each layer's expert activation frequencies (Fig. 3). To quantify differences between factual and non-factual scenarios, we convert each layer's expert activation frequencies into vectors and compute L2 distances between the factual and non-factual vector pairs (Fig. 4).

Fig. 3 shows that, for a dataset, each layer's router activates specific expert subsets across all MoE models. This indicates that experts within the same layer capture distinct features and guide routing decisions. Fig. 4 shows that **higher layers exhibit divergent expert activation patterns between factual and non-factual outputs across all MoE models**. This difference supports distinguishing factual from non-factual outputs, enabling intra-model contrastive decoding for hallucination mitigation across all MoE models.

4 Method

Our method consists of three modules (Fig. 5): (1) **Expert Partitioning via Confidence and Consistency** (§ 4.1) evaluates experts via confidence and consistency, partitioning them into one higher-reliability group and multiple lower-reliability groups. (2) **Attention-Guided Hallucination Amplification** (§ 4.2) uses attention and masking mechanisms to amplify lower-reliability experts' hallucinations. Those experts provide stronger negative references for contrastive decoding, where

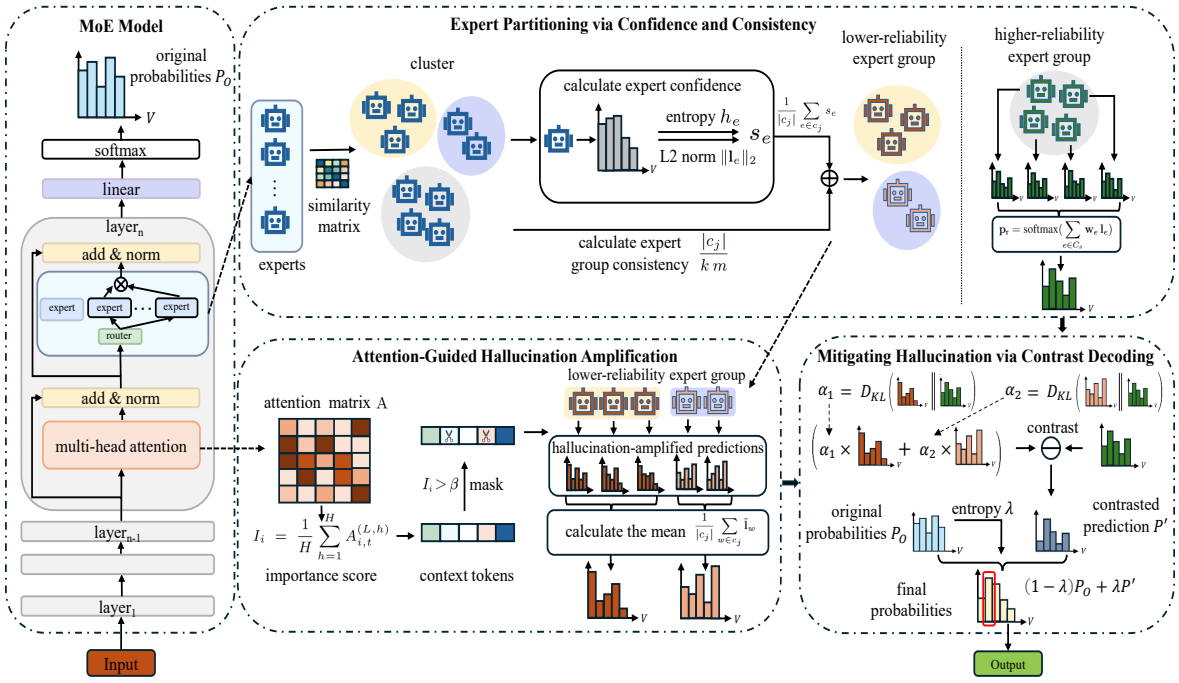

Figure 5: Overview of EAACD. Given an MoE input (left), we cluster final-layer expert predictions and evaluate each group's reliability. We fuse predictions from the higher-reliability group (green, top) and mask attention-identified critical tokens for lower-reliability groups to amplify hallucinations (red and orange, bottom center). KL between two groups serves as a penalty to contrast. We calibrate the prediction via the contrast result (bottom right).

experts with higher-reliability and lower-reliability contrastively learns to mitigate hallucinations in decoding. (3) **Adaptive Expert Group Contrastive Decoding** (§ 4.3) contrasts the higher-reliability expert group's prediction with lower-reliability expert groups' predictions. It then dynamically removes information from the higher-reliability prediction that overlaps with the amplified hallucinations, and calibrates the original prediction via the contrast result to mitigate hallucinations.

During generation, our decoding strategy first categorizes higher-layer experts into higher-reliability and lower-reliability groups based on their predictions (§ 4.1), then amplifies the lower-reliability experts' hallucinations (§ 4.2), and contrasts lower-reliability experts' predictions with the high-reliability group's prediction to dynamically calibrate the model's original prediction (§ 4.3). Following (Shi et al., 2024a), we apply our method only at the final layer (see details in App.G).

4.1 Expert Partitioning via Confidence and Consistency

To distinguish a higher-reliability expert group from lower-reliability expert groups, we evaluate each group's reliability by combining confidence and consistency (see § 5.3, which validates the effectiveness of the module via ablation studies). The module involves: (1) clustering experts by prediction similarity; (2) calculating reliability scores to classify higher-reliability and lower-reliability expert groups; and (3) integrating predictions from experts in the high-reliability group.

4.1.1 Expert Clustering

At each time step, we collect the logits $\mathbf{l}_e$ from all k experts in the MoE's final layer and calculate the pairwise similarity matrix between experts' predictions. Using this matrix, we apply Agglomerative Clustering (Walter et al., 2008) to cluster the experts into m groups $C = \{c_1, c_2, \ldots, c_m\}$.

4.1.2 Reliability Evaluation of Expert Groups

We propose a metric to measure the reliability of each expert group, which considers both the confidence of each expert in the group and the group's consistency by following two steps.

Step 1: Calculation of Expert Confidence. For each expert e, we obtain its predicted logits $\mathbf{l}_e = (l_{e1}, \ldots, l_{e|V|})$. We then calculate the entropy $h_e = -\sum_{i=1}^{|V|} p_{ei} \log p_{ei}$ (with $p_{ei} = \text{softmax}(l_{ei})$) and L2 norm $\|\mathbf{l}_e\|_2$. Entropy reflects expert e's uncertainty of prediction, while the L2 norm mea-

sures the activation magnitudes of that expert's logits. A low-entropy distribution indicates that the expert assigns most of the probability to a small set of candidate tokens, suggesting high confidence in its prediction, whereas a high-entropy distribution implies greater uncertainty due to dispersed probability (Liu et al., 2025b; Song et al., 2025). Meanwhile, the L2 norm captures the intensity of an expert's activation: experts with larger logit magnitudes tend to produce decisive predictions and exhibit more stable output behaviors (Lo et al., 2025).

As Eq.1 shows, we combine entropy with L2 norm to get expert e's confidence score s_e to provide a more comprehensive measure of expert confidence. The weight α is adaptively set via $\alpha = \frac{\sigma_H}{\sigma_H+\sigma_L}$, where σ_H and σ_L are standard deviations of all experts' entropy and L2 norm, respectively. When $\sigma_H > \sigma_L$, the weight corresponding to the entropy will increase. Conversely, the weight corresponding to the L2 norm will increase. This adaptive weighting emphasizes the metric with greater variance among experts, making s_e sensitive to inter-expert differences.

$$s_e = \alpha \cdot \frac{h_{\max} - h_e}{h_{\max}} + (1-\alpha) \cdot \frac{\|\mathbf{l}_e\|_2}{\max_{e'} \|\mathbf{l}_{e'}\|_2} \quad (1)$$

Step 2: Calculation of Expert Group Consistency. To improve robustness in expert group reliability evaluation, we evaluate each expert group with its overall consistency. For group c_j, we calculate the proportion of the group size $|c_j|$ relative to the expert count k and the cluster count m as the consistency score. We then combine the consistency score $\frac{|c_j|}{k\,m}$ with the average expert confidence of the group $\frac{1}{|c_j|}\sum_{e\in c_j} s_e$ as expert group's reliability score $T_j = \frac{1}{|c_j|}\sum_{e\in c_j} s_e + \frac{|c_j|}{k\,m}$. When most experts' predictions fall into the same group, its higher consistency score increases T_j, reflecting stronger expert consensus and reliability.[2]

4.1.3 Higher-Reliability Experts Prediction Integration

We select the group with the highest reliability as the higher-reliability expert group $\mathcal{C}_s$ and the rest as lower-reliability expert groups $\mathcal{C}_{\text{w}} = \{c_1, \ldots, c_m\} \setminus \{\mathcal{C}_s\}$. To fuse the logits of all high-reliability experts in $\mathcal{C}_s$ into a unified logits $\mathbf{p}_r$, we calculate a weighted sum of each expert's logits $\mathbf{l}_e$ using the router's gate value $\mathbf{w}_e$: $\mathbf{p}_r = \sum_{e\in C_s} \mathbf{w}_e\, \mathbf{l}_e$, which we will later compare with the lower-reliability expert groups.

[2]App. H illustrates the different prediction tendencies of low-reliability and high-reliability expert groups.

4.2 Attention-Guided Hallucination Amplification

To allow lower-reliability experts to provide stronger negative references for higher-reliability experts during contrast decoding (§ 4.3), we amplify the lower-reliability experts' hallucinations via the attention and masking mechanism. The motivation is to enlarge the difference between lower-reliability experts' predictions and higher-reliability experts' predictions. If the prompt is simple, even lower-reliability experts might make correct predictions; these predictions cannot provide effective negative references for higher-reliability experts. Specifically, (1) we use the attention mechanism to identify the most influential context tokens; (2) then we mask these tokens and re-input the masked prompt to lower-reliability experts to obtain predictions with more hallucinations.

MoE models retain the standard attention mechanism (Vaswani et al., 2017): given query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$, it calculates context tokens' attention weight via $\mathbf{A} = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}$. Larger weights indicate a greater influence of the corresponding context token on the prediction. To quantify the importance of the i-th context token at time step t, we calculate the average attention weight $\mathbf{A}_{i,t}^{(L,h)}$ across all attention heads h in the final layer L as its importance score $I_i = \frac{1}{H}\sum_{h=1}^{H} \mathbf{A}_{i,t}^{(L,h)}$.

Then, we mask tokens whose importance score exceeds a threshold β. We choose β to balance amplifying hallucinations from lower-reliability experts and preserving semantic similarity between the masked and original prompts (details in App. I). We re-input the masked prompt. Lacking crucial context, lower-reliability experts more easily generate hallucinations, providing stronger negative references for higher-reliability experts.

4.3 Mitigating Hallucinations via Contrast Decoding

To mitigate hallucinations, we contrast the prediction of the higher-reliability expert group with those of lower-reliability groups. Traditional contrast decoding methods directly subtract the low-reliability prediction from the high-reliability prediction during contrast, and then use a hyperparameter to adjust the calibration strength of the contrastive result on the original prediction. Unlike

this, we design adaptive penalty coefficients for each lower-reliability expert group during contrast and dynamically calibrate the model's original predictions via its uncertainty. This module has two steps: (1) generating adaptive penalty coefficients based on differences between the higher-reliability expert group and each lower-reliability expert group; (2) dynamically calibrating the model's original prediction using its predictive uncertainty.

4.3.1 Adaptive Contrast Decoding

As Eq. 2 shows, after inducing hallucination (§ 4.2), each lower-reliability expert w in group $c_j \in \mathcal{C}_\text{w}$ produces a logits $\tilde{\text{l}}w$. We then calculate the mean $\frac{1}{|c_j|}\sum_{w\in c_j}\tilde{\mathbf{l}}_w$ as the group centroid. Similarly, we calculate the higher-reliability expert group centroid $\frac{1}{|\mathcal{C}_s|}\sum_{e\in\mathcal{C}_s}\mathbf{l}_e$ by each higher-reliability expert e's logits $\mathbf{l}_e$. Finally, we quantify the difference G_j between the higher-reliability and each lower-reliability group by the KL divergence between their centroids (shown in the lower part of Fig. 5). A larger G_j indicates a greater discrepancy between the two groups' predictions, in which case we increase the penalty for the lower-reliability group; otherwise, we reduce it. The motivation for G_j is to ensure that the penalty coefficient for each lower-reliability expert group matches their actual hallucination severity.

$$G_j = \mathbb{D}_{\text{KL}}\big(\text{softmax}(\tfrac{1}{|C_s|}\sum_{e\in C_s}\mathbf{l}_e)\,\big\|\,\text{softmax}(\tfrac{1}{|c_j|}\sum_{w\in c_j}\tilde{\mathbf{l}}_w)\big) \tag{2}$$

We use the normalized G_j as the penalty coefficient α_j for each corresponding lower-reliability expert group c_j: $\alpha_j = \frac{G_j}{\sum_{c_j\in\mathcal{C}_w}G_j}$. Then we fuse each lower-reliability expert group's logits to form an integrated low-reliability logits $\mathbf{p}_l$ by multiplying each group's centroid by α_j: $\mathbf{p}_l = \sum_{c_j\in\mathcal{C}_w}\alpha_j\cdot\frac{1}{|c_j|}\sum_{w\in c_j}\tilde{\mathbf{l}}_w$.

Finally, as Eq. 3 shows, we contrast the high-reliability logits $\mathbf{p}_r$ and low-reliability logits $\mathbf{p}_l$ to obtain the contrastive result $\mathbf{p}'$.

$$\mathbf{p}' = \mathbf{p}_r - \mathbf{p}_l \tag{3}$$

4.3.2 Confidence-Guided Output Probability Calibration

To better mitigate hallucinations, we dynamically adjust calibration strength using the entropy of the model's original predictions as weight $\lambda = \frac{-\sum_{i=1}^{|V|}p_{o,i}\log p_{o,i}}{\log|V|}$. $p_{o,i}$ denotes the probability of the i-th token in the vocabulary. Higher entropy means more uncertainty in the model's prediction, so we increase the contrastive result's weight λ in the final prediction $\mathbf{p}_f$ for stronger calibration. Conversely, we reduce the calibration strength. We calculate the final prediction by $\mathbf{p}_f = \text{softmax}((1-\lambda)\,\mathbf{p}_o + \lambda\,\mathbf{p}')$, $\mathbf{p}_o$ is original logits. This dynamic calibration guides predictions toward factual outputs (see examples in App. L).

5 Experiments

5.1 Experimental Settings

(1) **MoE Models:** We use two mainstream open-source MoE models: LLaMA-MoE (Zhu et al., 2024) (without shared experts) and Qwen-MoE (Team, 2024) (with shared experts)[3]. (2) **Datasets:** Follow (Shi et al., 2024a), we use four datasets: FACTOR (Muhlgay et al., 2024) (including three subsets: Wiki-FACTOR, News-FACTOR, Expert-FACTOR), HellaSwag (Zellers et al., 2019), StrategyQA (Geva et al., 2021) and MathQA (Amini et al., 2019). (3)**Baselines:** We select five mainstream decoding methods: Greedy, Contrastive Decoding (Li et al., 2023), DoLa (Chuang et al., 2023), SCMoE (Shi et al., 2024a), END (Wu et al., 2025) and a non-decoding method: Self-Endorsement (Wang et al., 2024) for hallucination mitigation on QA tasks as baselines. (4) **Evaluation Metrics:** We use accuracy as our evaluation metric.

5.2 Overall Performance

Tab. 1 compares EAACD with all baselines on two types of MoE architectures. EAACD outperforms all baselines across datasets. For Qwen-MoE, EAACD achieves nearly 13% improvement over the strongest baseline on HellaSwag. The results show that DoLa almost performs the worst, especially on the Qwen MoE. On some datasets, contrastive decoding baselines even underperform the simplest baseline, greedy decoding. We argue that these methods cannot guarantee that the lower-reliability part always generates unfaithful outputs. In contrast, EAACD amplifies hallucinations in low-reliability experts (§ 4.2), preventing factual outputs and ensuring strong performance. Moreover, EAACD maintains comparable time and memory overhead to the baseline (see App. F).

[3]Since the released DeepSeek-MoE implementation is inconsistent with its report (causing performance degradation), which has also been noted by other users in the community, we used Qwen-MoE and LLaMA-MoE as representatives.

Methods	LLaMA MoE *(without shared experts)*						Qwen MoE*(with shared experts)*					
	Expert-FACTOR	News-FACTOR	Wiki-FACTOR	HellaSwag	StrategyQA	MathQA	Expert-FACTOR	News-FACTOR	Wiki-FACTOR	HellaSwag	StrategyQA	MathQA
Greedy	50.85	41.51	29.76	69.80	52.01	18.59	52.54	71.33	55.21	58.30	61.77	22.88
CD	47.88	33.98	29.23	56.30	52.05	21.27	57.20	62.26	51.07	56.10	49.83	23.68
DoLa	38.98	35.04	32.03	64.50	49.43	20.47	47.03	45.56	32.63	35.80	46.72	20.54
SCMoE	54.24	45.37	34.94	68.00	42.31	21.68	58.90	71.62	55.18	64.60	48.16	24.02
END	61.02	64.77	53.10	52.90	53.72	20.40	52.54	71.81	55.44	58.20	56.91	25.76
Self-Endorsement	52.66	59.36	43.93	70.20	53.70	20.54	53.81	72.20	57.27	62.20	56.41	21.62
EAACD	**63.56**	**65.54**	**55.44**	**71.80**	**54.15**	**22.58**	**61.01**	**75.87**	**58.45**	**77.50**	**62.29**	**28.34**

Table 1: Overall performance of EAACD comparing against different decoding strategies on two MoE architectures: LLaMA MoE without shared experts and Qwen MoE with shared experts. The best results are highlighted in bold.

Variants	LLaMA MoE *(without shared experts)*						Qwen MoE*(with shared experts)*					
	Expert-FACTOR	News-FACTOR	Wiki-FACTOR	HellaSwag	StrategyQA	MathQA	Expert-FACTOR	News-FACTOR	Wiki-FACTOR	HellaSwag	StrategyQA	MathQA
− consistence	62.71	63.13	55.04	71.10	53.28	20.64	57.20	71.04	55.54	58.30	61.89	28.21
− confidence	60.17	55.89	54.07	70.40	53.84	20.20	56.36	69.98	55.53	58.60	62.20	27.71
− evaluation	59.32	63.90	53.07	51.90	53.50	19.60	54.66	67.57	54.37	57.30	61.94	26.73
− attention	62.29	63.71	55.24	71.50	50.34	20.47	60.59	74.52	57.55	77.10	61.94	27.27
EAACD	**63.56**	**65.54**	**55.44**	**71.80**	**54.15**	**22.58**	**61.01**	**75.87**	**58.45**	**77.50**	**62.29**	**28.34**

Table 2: Ablation studies on different components of our model. − indicates deleting this component.

5.3 Ablation study

We conduct an ablation study to verify each component's importance (Tab. 2). −*consistence* and −*confidence* remove consistency and confidence defined in (§ 4.1). Both reduce performance, with −*confidence* showing larger drops, indicating confidence is more crucial for expert reliability estimation. −*evaluation* removes our partitioning module (§ 4.1) and splitting experts into higher-reliability (top 50%) and lower-reliability (bottom 50%) groups by routing weights. It performs worst on nearly all datasets, suggesting routing weights alone are less reliable for grouping. −*attention*, which removes hallucination amplification (§ 4.2), also reduces performance. This confirms the effectiveness of the hallucination amplification module, improving overall effectiveness.

5.4 Analysis of Expert Reliability and Prediction Preference

To verify the effectiveness of our expert partitioning module (§ 4.2), we explore each group's behavior on three datasets: Expert-FACTOR, News-FACTOR, and Wiki-FACTOR. Each dataset consists of multiple-choice questions, allowing us to observe whether higher-reliability experts favor correct answers. We compute four metrics per dataset: (1) the mean probability difference between correct and incorrect answers for the higher-reliability expert group, (2) the same difference for lower-reliability expert groups, (3) the mean reliability score of the higher-reliability expert group, and (4) the same score of lower-reliability expert groups.

Fig. 6 shows that higher-reliability experts assign a greater mean probability to correct answers (positive difference, green), while lower-reliability experts show the opposite (negative difference, red), indicating that higher-reliability experts favor correct answers, validating effective expert distinction. Moreover, groups with higher reliability scores exhibit larger probability gaps between correct and incorrect answers, reflecting stronger factual bias.

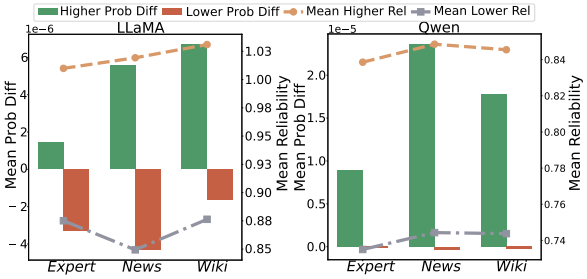

Figure 6: Mean probability difference and mean reliability scores for correct versus incorrect answers of higher-reliability (green/black) and lower-reliability (red/yellow) expert groups.

6 Conclusion

We explore "knowledge injection" and expert activation patterns in MoE. We find that "knowledge injection" occurs only in MoE without shared experts, limiting traditional intra-model contrastive decoding in all MoE models. Across different MoE architectures, higher-layer expert activation patterns differ between factual and non-factual outputs. We partition higher-layer experts by reliability, amplify lower-reliability experts' hallucinations as negative references, and adaptively calibrate the model's

predictions via contrastive decoding. Our model outperforms baselines across four datasets, mitigating hallucinations without external resources.

7 Limitations

In our study, we only investigate whether the "knowledge injection" phenomenon exists in different types of MoE models. If MoE architectures do not remain mainstream in the future, the direct applicability of our method may be limited. In that case, we encourage future researchers to build upon our method and extend the investigation to other evolving model architectures.

Acknowledgements

This work is supported by the following fundings: National Natural Science Foundation of China under Grant No.62376284 and No.62306330.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Dingwei Chen, Shuai Wang, Zhengping Fan, Xiping Hu, and Chengming Li. 2024. Freeze-cd: Alleviating hallucination of large language models via contrastive decoding with local freezing training. In *2024 IEEE International Conference on Smart Internet of Things (SmartIoT)*, pages 325–329. IEEE.

Xiaoxue Cheng, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2025. Think more, hallucinate less: Mitigating hallucinations via dual process of fast and slow thinking. *CoRR*.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R Glass, and Pengcheng He. 2023. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Damai Dai, Chengqi Deng, Chenggang Zhao, Rx Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, and 1 others. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1280–1297.

Souvik Das, Lifeng Jin, Linfeng Song, Haitao Mi, Baolin Peng, and Dong Yu. 2025. Entropy guided extrapolative decoding to improve factuality in large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6589–6600.

Minghui Fang, Shengpeng Ji, Jialong Zuo, Xize Cheng, Wenrui Liu, Xiaoda Yang, Ruofan Hu, Jieming Zhu, and Zhou Zhao. 2025a. Gta: Towards generative text-to-audio retrieval via multi-scale tokenizer. In *Proc. Interspeech*, pages 2650–2654.

Minghui Fang, Shengpeng Ji, Jialong Zuo, Hai Huang, Yan Xia, Jieming Zhu, Xize Cheng, Xiaoda Yang, Wenrui Liu, Gang Wang, and 1 others. 2025b. Cart: A generative cross-modal retrieval framework with coarse-to-fine semantic modeling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15120–15133.

Xinyue Fang, Zhen Huang, Zhiliang Tian, Minghui Fang, Ziyi Pan, Quntian Fang, Zhihua Wen, Hengyue Pan, and Dongsheng Li. 2025c. Zero-resource hallucination detection for text generation via graph-based contextual knowledge triples modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23868–23877.

Ariel Gera, Roni Friedman, Ofir Arviv, Chulaka Gunasekara, Benjamin Sznajder, Noam Slonim, and Eyal Shnarch. 2023. The benefits of bad advice: Autocontrastive decoding across model layers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10406–10420.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, and 1 others. 2022. Opt-iml: Scaling language model instruction meta learning through the lens of generalization. *arXiv preprint arXiv:2212.12017*.

Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards mitigating hallucination in large language models via self-reflection. *arXiv preprint arXiv:2310.06271*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312.

Yunxin Li, Shenyuan Jiang, Baotian Hu, Longyue Wang, Wanqi Zhong, Wenhan Luo, Lin Ma, and Min Zhang. 2025. Uni-moe: Scaling unified multimodal llms with mixture of experts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252.

Chenlin Liu, Minghui Fang, Patrick Zhang, Wei Zhou, Jie Gao, and Jiqing Han. 2025a. Mitigating hallucinations in lm-based tts models via distribution alignment using gflownets. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19346–19364.

Xiaoou Liu, Tiejin Chen, Longchao Da, Chacha Chen, Zhen Lin, and Hua Wei. 2025b. Uncertainty quantification and confidence calibration in large language models: A survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6107–6117.

Ka Man Lo, Zeyu Huang, Zihan Qiu, Zili Wang, and Jie Fu. 2025. A closer look into mixture-of-experts in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4427–4447.

Debjyoti Mondal, Suraj Modi, Subhadarshi Panda, Rituraj Singh, and Godawari Sudhakar Rao. 2024. Kam-cot: Knowledge augmented multimodal chain-of-thoughts reasoning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 18798–18806.

Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. 2024. Generating benchmarks for factuality evaluation of language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 49–66.

Shilong Pan, Zhiliang Tian, Wanlong Yu, Zhen Huang, Qingyu Qiu, Zihan Chen, Zhonghao Sun, Minlie Huang, and Dongsheng Li. 2026. Walksafe: Risk-aware graph random walk with bi-grpo for llm safety. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 32655–32663.

Chufan Shi, Cheng Yang, Xinyu Zhu, Jiahao Wang, Taiqiang Wu, Siheng Li, Deng Cai, Yujiu Yang, and Yu Meng. 2024a. Unchosen experts can contribute too: Unleashing moe models' power by self-contrast. *Advances in Neural Information Processing Systems*, 37:136897–136921.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. 2024b. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 783–791.

Xiao Shi, Zhengyuan Zhu, Zeyu Zhang, and Chengkai Li. 2023. Hallucination mitigation in natural language generation from large-scale open-domain knowledge graphs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12506–12521.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. 2022. Prompting gpt-3 to be reliable. *arXiv preprint arXiv:2210.09150*.

Haoyi Song, Ruihan Ji, Naichen Shi, Fan Lai, and Raed Al Kontar. 2025. Inv-entropy: A fully probabilistic framework for uncertainty quantification in language models. *arXiv preprint arXiv:2506.09684*.

Zhenpeng Su, W Xing, Zijia Lin, Yizhe Xiong, Minxuan Lv, Guangyuan Ma, Hui Chen, Songlin Hu, and Guiguang Ding. 2025. Cartesianmoe: Boosting knowledge sharing among experts via cartesian product routing in mixture-of-experts. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10040–10055.

Zhonghao Sun, Zhiliang Tian, Yiping Song, Yuyi Si, Juhua Zhang, Minlie Huang, Kai Lu, Zeyu Xiong, Xinwang Liu, and Dongsheng Li. 2025. Dpgatextsyn: Differentially private genetic algorithm for synthetic text generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16159–16179.

Filip Szatkowski, Bartosz Wójcik, Mikołaj Piórczyński, and Simone Scardapane. 2024. Exploiting activation sparsity with dense to dynamic-k mixture-of-experts conversion. *Advances in Neural Information Processing Systems*, 37:43245–43273.

Qwen Team. 2024. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters".

Surat Teerapittayanon and Bradley McDanel. 2016. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 2464–2469. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Bruce Walter, Kavita Bala, Milind Kulkarni, and Keshav Pingali. 2008. Fast agglomerative clustering for rendering. In *2008 IEEE Symposium on Interactive Ray Tracing*, pages 81–86. IEEE.

Ante Wang, Linfeng Song, Baolin Peng, Lifeng Jin, Ye Tian, Haitao Mi, Jinsong Su, and Dong Yu. 2024. Improving llm generations via fine-grained self-endorsement. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 8424–8436.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Jialiang Wu, Yi Shen, Sijia Liu, Yi Tang, Sen Song, Xiaoyi Wang, and Longjun Cai. 2025. Improve decoding factuality by token-wise cross layer entropy of large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3912–3921.

Dingkang Yang, Dongling Xiao, Jinjie Wei, Mingcheng Li, Zhaoyu Chen, Ke Li, and Lihua Zhang. 2025. Improving factuality in large language models via decoding-time hallucinatory and truthful comparators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25606–25614.

Haoran Yang, Deng Cai, Huayang Li, Wei Bi, Wai Lam, and Shuming Shi. 2024a. A frustratingly simple decoding method for neural text generation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 536–557.

Yuanhang Yang, Shiyi Qi, Wenchao Gu, Chaozheng Wang, Cuiyun Gao, and Zenglin Xu. 2024b. Xmoe: Sparse models with fine-grained and adaptive expert selection. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 11664–11674.

Hao Yin, Guangzong Si, and Zilei Wang. 2025. The mirage of performance gains: Why contrastive decoding fails to address multimodal hallucination. *arXiv preprint arXiv:2504.10020*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Yihao Zhang, Zeming Wei, Jun Sun, and Meng Sun. 2024. Adversarial representation engineering: A general model editing framework for large language models. *Advances in Neural Information Processing Systems*, 37:126243–126264.

Yue Zhang, Leyang Cui, Shuming Shi, and 1 others. 2025. Alleviating hallucinations of large language models through induced hallucinations. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 8218–8232.

Hao Zhao, Zihan Qiu, Huijia Wu, Zili Wang, Zhaofeng He, and Jie Fu. 2024. Hypermoe: Towards better mixture of experts via transferring among experts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10605–10618.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. 2023. Can we edit factual knowledge by in-context learning? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4862–4876.

Tong Zhu, Xiaoye Qu, Daize Dong, Jiacheng Ruan, Jingqi Tong, Conghui He, and Yu Cheng. 2024. Llama-moe: Building mixture-of-experts from llama with continual pre-training. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15913–15923.

A JSD between the final layer and each lower layer across different models on all datasets.

As shown in Figure 7, we present the Jensen-Shannon Divergence (JSD) distance between predictions from lower layers and the final layer for four models (llama MoE, Mixtral, Qwen-MoE, and DeepSeek-MoE) on the StrategyQA dataset. We observe that for MoE models without shared experts, predictions undergo a sharp drop in JSD distance at one layer. This indicates the model significantly revised its predictions at this stage, likely incorporating substantial new knowledge. However,

in MoE models with shared experts, predictions change very little from lower to upper layers, so we see no evidence of "knowledge injection" on the GSM8K dataset. We see the same pattern here, which again supports the generalizability of the conclusion established in RQ1 (figs. 8 to 11).

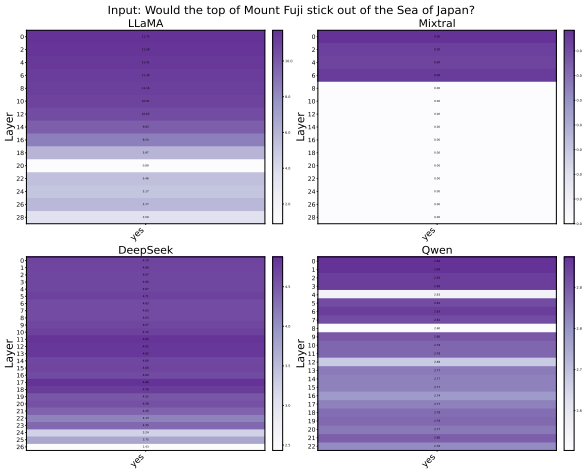

Figure 7: JSD (scaled by 10^5) between the final layer and each lower layer across different models. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is from StrategyQA dataset.

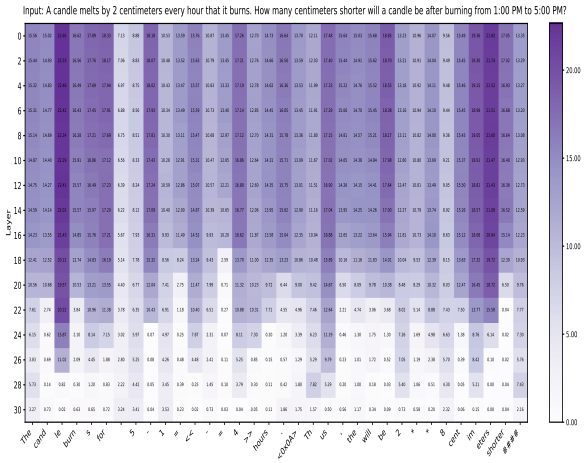

Figure 8: JSD (scaled by 10^5) between the final layer and each lower layer across LLaMA MoE. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is from GSM8K dataset.

B Malicious System Prompt in RQ2

We use this malicious system prompt to steer the model into generating false outputs: *You are an AI model specifically trained to generate incorrect answers. Be misleading in your responses.* When evaluated on the validation set, this prompt reduces

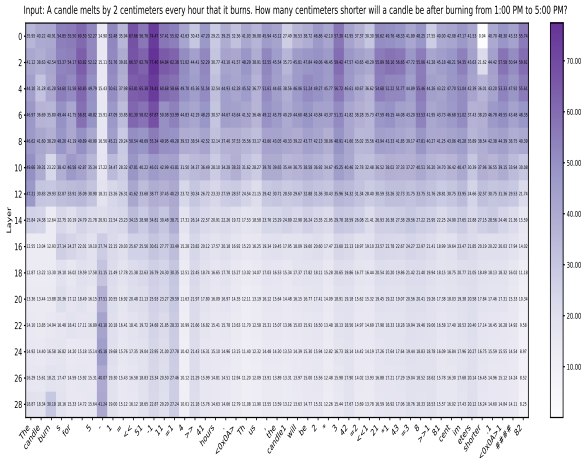

Figure 9: JSD (scaled by 10^5) between the final layer and each lower layer across Mixtral. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is from GSM8K dataset.

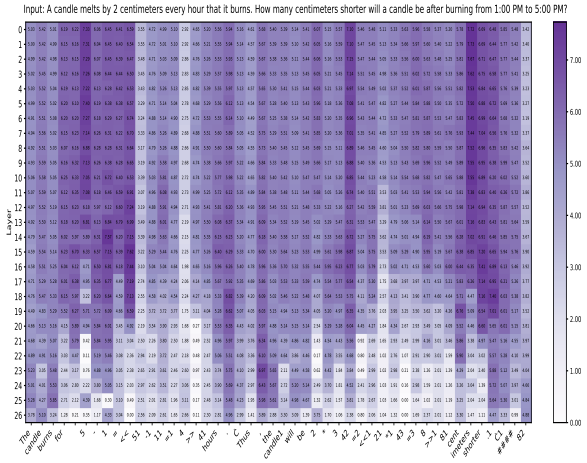

Figure 10: JSD (scaled by 10^5) between the final layer and each lower layer across DeepSeek MoE. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is from GSM8K dataset.

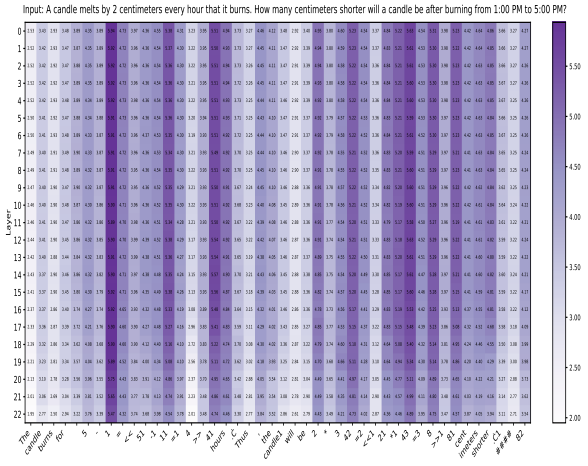

Figure 11: JSD (scaled by 10^5) between the final layer and each lower layer across Qwen MoE. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is from GSM8K dataset.

the model's accuracy from 40.6% to 11.3%, demonstrating its effectiveness in inducing hallucinated outputs.

C Expert activation patterns across layers in different MoE during factual and non-factual generation on All Datasets

We show the expert activation patterns of each MoE model when generating truthful and untruthful responses on the GSM8K (figs. 12 to 16) and StrategyQA (figs. 17 to 21) datasets.

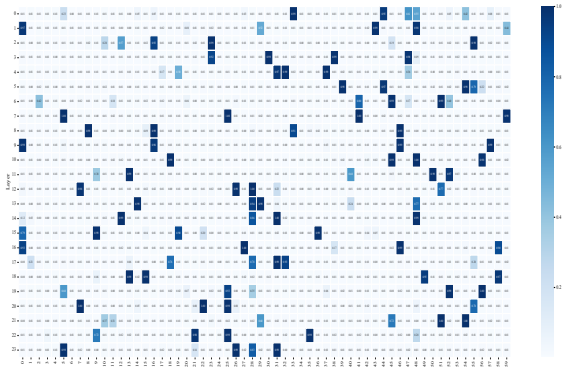

Figure 12: Expert activation patterns across layers in Qwen MoE architectures during factual generation on the GSM8k. Column labels indicate the expert indices.

D Societal Impacts

We discuss the societal impact of our work as follows. EAACD mitigates hallucinations by leveraging expert activation differences, improving the factual reliability and trustworthiness of AI-generated

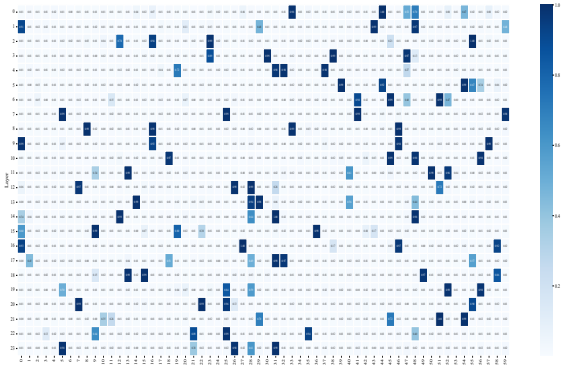

Figure 13: Expert activation patterns across layers in Qwen MoE architectures during non-factual generation on the GSM8k. Column labels indicate the expert indices.

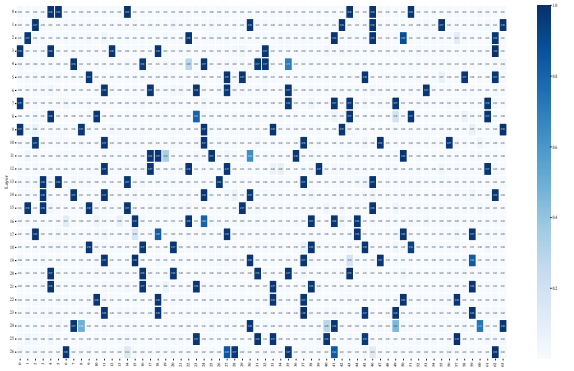

Figure 14: Expert activation patterns across layers in Deepseek MoE architectures during factual generation on the GSM8k. Column labels indicate the expert indices.

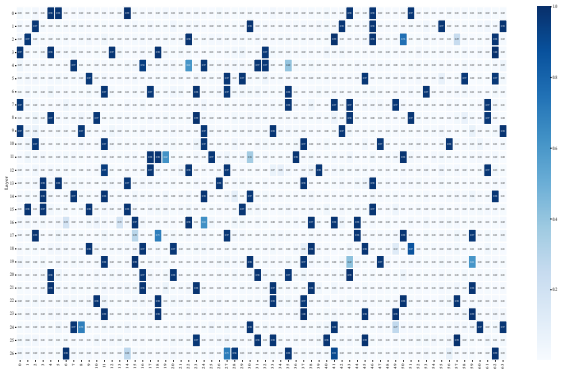

Figure 15: Expert activation patterns across layers in Deepseek MoE architectures during non-factual generation on the GSM8k. Column labels indicate the expert indices.

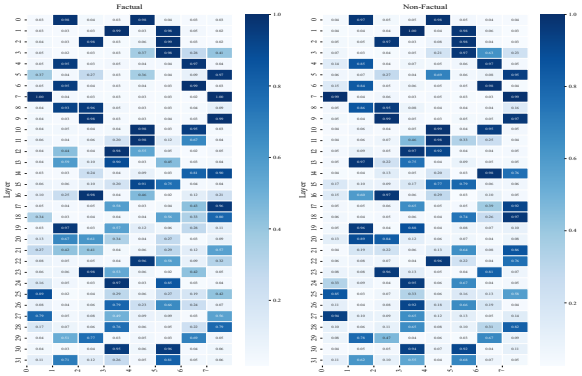

Figure 16: Expert activation patterns across layers in Mixtral architectures during factual and non-factual generation on the GSM8k. Column labels indicate the expert indices.

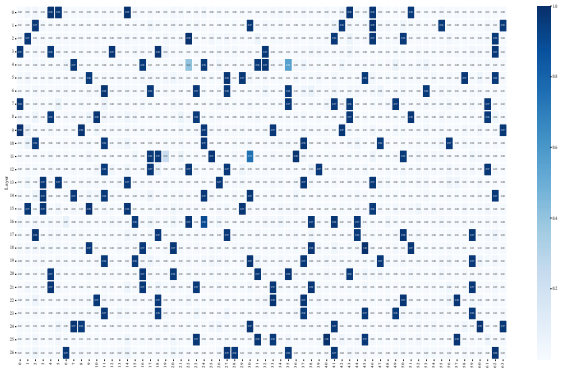

Figure 19: Expert activation patterns across layers in Deepseek MoE architectures during factual generation on the StrategyQA. Column labels indicate the expert indices.

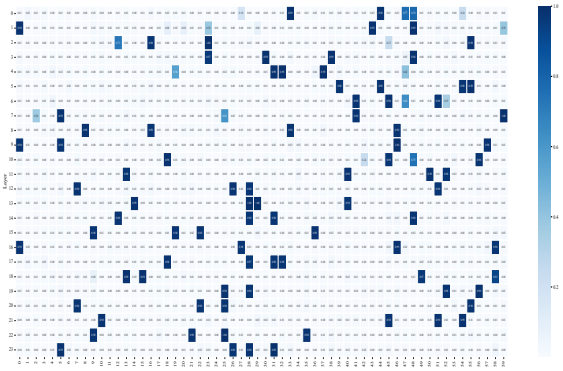

Figure 17: Expert activation patterns across layers in Qwen MoE architectures during factual generation on the StrategyQA. Column labels indicate the expert indices.

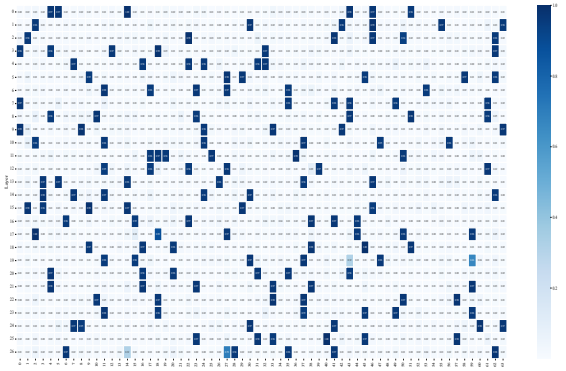

Figure 20: Expert activation patterns across layers in Deepseek MoE architectures during non-factual generation on the StrategyQA. Column labels indicate the expert indices.

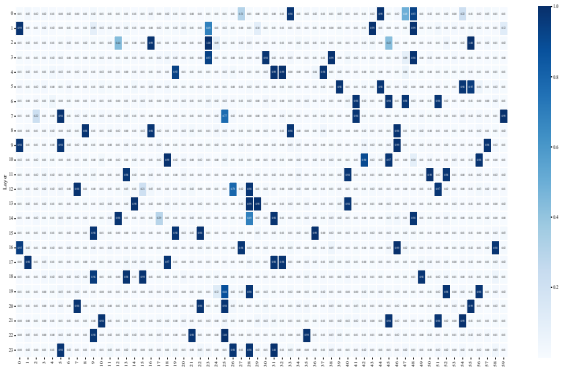

Figure 18: Expert activation patterns across layers in Qwen MoE architectures during non-factual generation on the StrategyQA. Column labels indicate the expert indices.

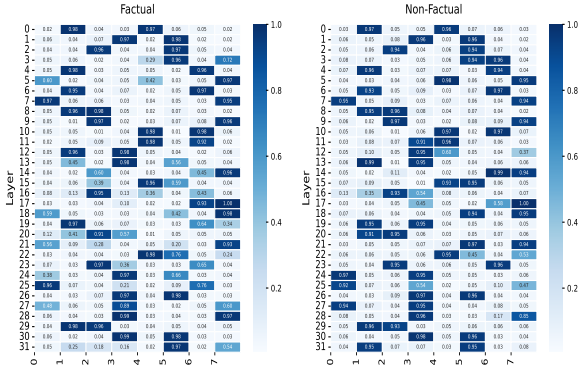

Figure 21: Expert activation patterns across layers in Mixtral architectures during factual and non-factual generation on the StrategyQA. Column labels indicate the expert indices.

text. This enhancement benefits applications in knowledge-sensitive domains such as healthcare and law. Importantly, EAACD achieves these improvements without relying on any external resources, making it particularly valuable for scenarios where access to knowledge bases or auxiliary models is limited. By enabling MoE models to generate more accurate and reliable outputs even under resource-constrained conditions, EAACD broadens the practical applicability of LLMs and promotes the responsible deployment of AI in the real world (Fang et al., 2025b,a).

Additionally, the hallucination amplification component in our method may introduce potential risks in real-world deployment if not properly controlled (Pan et al., 2026). To mitigate these risks, for practical applications, we plan to incorporate two protective measures. First, we will integrate an external knowledge base for post-hoc fact verification, enabling the system to identify and discard outputs that contain hallucinated content. Second, we will restrict the hallucination amplification module to offline evaluation or strictly sandboxed environments, ensuring that amplified hallucinations are used solely for internal contrastive signals and never exposed to end users (Sun et al., 2025). These safeguards aim to preserve the safety and reliability of the system while retaining the effectiveness of the proposed approach.

E Significance Test of Experimental Results

To assess whether our method significantly improves performance over the baseline on four datasets, we compare our results to the baselines using a paired t-test. We find that for all baselines, the significance tests between our method and theirs showed statistically significant differences (as shown in Tab. 3, all $P < 0.05$), demonstrating the effectiveness of our approach.

	Greedy	CD	DoLa	SCMoE	END	Self-Endorsement
LLaMA	0.0458	0.0277	0.0261	0.0207	0.0489	0.0407
Qwen	0.0497	0.0115	0.0069	0.0244	0.0397	0.0191

Table 3: Significance test results comparing different baselines with EAACD across all datasets.

F Latency and GPU Memory Overhead

To compare the latency and memory overhead, we record the cost for each model when generating 100 tokens from the same prompt under different decoding strategies. Specifically, we calculate the time difference before and after the model generates these 100 tokens as latency. For memory overhead, we compute the difference between the peak GPU memory used during forward passes and the GPU memory used before the first forward pass. For fairness in comparison, we compare our method with SCMoE: a contrastive decoding strategy specifically designed for MoE. To our knowledge, it is also the only contrastive decoding strategy tailored for MoE models to date. We use GPU A100 for our experiments. All results are in Tab.4.

		SCMoE	**EAACD**
LLaMA	Latency (s)	19.47	21.33
	Latency Ratio	1.0x	1.10x
	Memory Overhead (MB)	4859.07	5529.95
	Memory Ratio	1.0x	1.14x
Qwen	Latency (s)	60.40	76.06
	Latency Ratio	1.0x	1.26x
	Memory Overhead (MB)	4337.01	4707.45
	Memory Ratio	1.0x	1.09x

Table 4: Latency and GPU memory overhead of SCMoE and EAACD on LLaMA-MoE and Qwen-MoE. Ratios indicate the multiplicative factors of EAACD relative to SCMoE.

G The Reasons for Applying Contrastive Decoding at the Final Layer

We choose to apply contrastive decoding at the final layer because it provides a favorable trade-off between decoding reliability and inference efficiency. As shown in Sec. 3.3, higher-layer experts exhibit reliability differences across various MoE architectures. If we apply contrastive decoding across all higher layers, the intermediate predictions have not yet converged, and the later layers will modify them again. This makes early contrastive decoding unstable and increases the latency overhead. In contrast, the final layer produces stable predictions. Partitioning experts and performing contrastive decoding at this layer offers more reliable results with much lower overhead.

H A Case Study on Prediction Biases of Low-reliability and High-reliability Expert Groups

Here we use the question *"What color is the sun when viewed from space?"* as an illustrative example. After the model outputs *"The sun is"*, we analyze how higher-reliability and lower-reliability

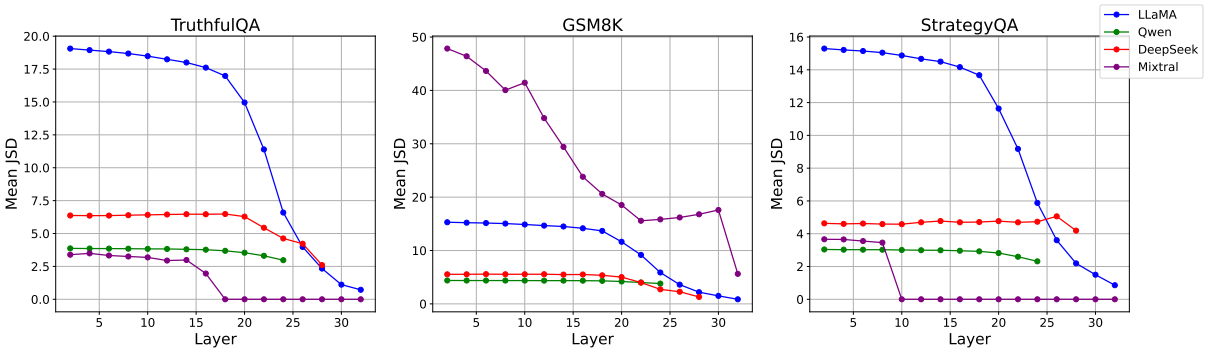

Figure 22: Layer-wise average JSD across different datasets for various MoE models.

expert groups assign probabilities to the correct answer token *"all"*, the most misleading incorrect token *"yellow"*, and all *other* tokens in the vocabulary. The results are summarized in Table 5.

Next token	Higher-reliability	Lower-reliability
all	0.62	0.37
yellow	0.34	0.54
other	0.04	0.09

Table 5: Next-token probability distribution assigned by higher-reliability and lower-reliability expert groups.

We observe that lower-reliability experts assign a substantially higher probability to the incorrect token *"yellow"* and a much lower probability to the correct token *"all"*. This behavior indicates that lower-reliability experts tend to favor the misleading token *"yellow"* over the correct token *"all"* when predicting the next token.

I Analysis of the β Selection Strategy

We select a specific threshold β to balance two competing objectives: (1) amplifying hallucination behaviors of lower-reliability experts, and (2) keeping the masked prompt semantically close to the original input. If β is too large, an excessive number of tokens are masked, rendering the modified prompt unreadable or semantically meaningless, which fails to amplify hallucinations related to the original prompt. Conversely, if β is too small, only a few tokens are masked, and most contextual information is preserved, making it difficult to effectively amplify hallucinations. We therefore determine the optimal β through validation experiments.

To provide an intuitive illustration, we use the question *"What color is the sun when viewed from space?"* as an example. We analyze how lower-reliability experts adjust their next-token probability distributions before and after masking the token *"sun"* in the prompt. Specifically, we compare the probabilities assigned to the correct answer token *"all"*, the most misleading incorrect token *"yellow"*, and all *other* tokens in the vocabulary. The results are reported in Table 6.

Next token	Before masking *"sun"*	After masking *"sun"*
all	0.37	0.01
yellow	0.54	0.68
other	0.09	0.31

Table 6: Next-token probabilities assigned by lower-reliability experts before and after masking the token *"sun"*.

We observe that after masking, the probability of the correct token *"all"* drops sharply from 0.37 to 0.01, while the probability of the incorrect token *"yellow"* increases from 0.54 to 0.68. This shift indicates that masking salient tokens encourages lower-reliability experts to exhibit a stronger tendency toward misleading answers, thereby effectively amplifying hallucination behaviors.

In practical deployment, to avoid potential risks introduced by the hallucination amplification module, we recommend incorporating the following two protective measures: (1) Integrate an external knowledge base for fact-checking to discard outputs containing hallucinations. (2) Restrict the hallucination amplification module to offline evaluation or sandboxed environments, ensuring its outputs never reach the user.

J Dataset-level Analysis of "Knowledge Injection" in MoE models

In Sec. 3.2, we present sample-level experimental results. In this section, we provide dataset-level statistical results in Figure 22 to further validate

the findings observed in Sec. 3.2. Specifically, for each model, we compute the average JSD value for each layer across different datasets and plot the curve of average JSD against layer depth.

From Fig. 22, we can observe that in MoE models without shared experts (LLaMA, Mixtral), the JSD between the lower and final layers starts high but gradually decreases as the layer depth increases. A sharp drop in the higher layers indicates that the model substantially updates its predictions at these layers, reflecting the occurrence of "Knowledge Injection". In contrast, in MoE models with shared experts (DeepSeek, Qwen), JSD values stay consistently low across layers with minimal changes. These trends align well with the phenomena we observed at the case level, providing further evidence for the robustness of our conclusions.

K Additional Observations of the "Knowledge Injection" Phenomenon in More MoE Models

In Sec. 3.2, we observe that **the "knowledge injection" does not exhibit in MoE with shared experts**. However, the models compared in Sec. 3.2 differ not only in expert-sharing mechanisms, but also in model size and depth. Differences in model size and depth may introduce confounding factors, thereby affecting the reliability of our observations. To rule out the possibility that the observed conclusion is caused by variations in model size or depth rather than expert sharing, we conduct an additional control experiment using OpenMoE. OpenMoE is a MoE model with shared experts, and it has the same number of layers and a comparable parameter scale to Mixtral (without shared experts). Similar to Sec. 3.2, for the two models, we compute the JSD values between the probability distribution at each layer and that of the final layer under the same inputs. The results are shown in Fig. 23 and Fig. 24:

For OpenMoE, JSD values remain low across layers with minimal change. This indicates the "Knowledge Injection" does not exhibit in this model. For Mixtral, JSD values between the lower and final layer start high but decrease as the layer goes deeper, showing the existence of "knowledge injection". These results demonstrate that the observed differences cannot be attributed to model depth or parameter scale, which further supports the validity of our conclusion: **The "knowledge injection" does not exhibit in MoE with shared**

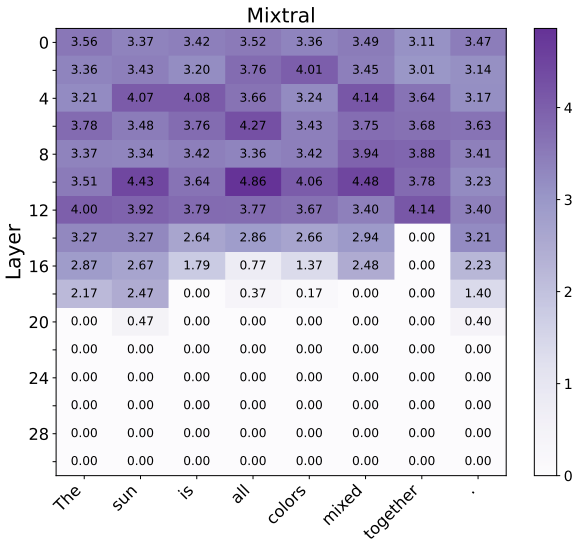

Figure 23: JSD (scaled by 10^5) between the final layer and each lower layer in Mixtral. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is *What color is the sun when viewed from space?*

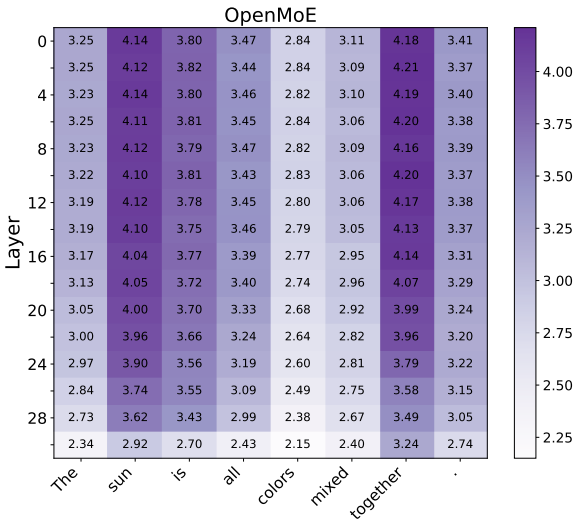

Figure 24: JSD (scaled by 10^5) between the final layer and each lower layer in OpenMoE. Column labels indicate tokens in each step. Row labels indicate layer indices. The input sample is *What color is the sun when viewed from space?*

experts.

L A Case Study on Model Predictions Before and After Calibration

To demonstrate how the model's output changes before and after calibrating its original predictions using our method, we present a case (Tab. 7):

Input Question	What U.S. state produces the most peaches?
Original Prediction	Georgia produces the most peaches in the U.S.
Calibrated Prediction	California produces the most peaches in the U.S.

Table 7: An example illustrating model predictions before and after calibration using contrastive decoding.

As shown in Tab. 7, given the input question *"What U.S. state produces the most peaches?"*, the model's original prediction:*"Georgia produces the most peaches in the U.S."* is incorrect. After calibration using our proposed decoding strategy EAACD, the model produces the correct answer: *"California produces the most peaches in the U.S."*.