BaseCal: Unsupervised Confidence Calibration via Base Model Signals

Anonymous ACL submission

Abstract

Reliable confidence is essential for trusting the outputs of LLMs, yet widely deployed post-trained LLMs (PoLLMs) typically compromise this trust with severe overconfidence. In contrast, we observe that their corresponding base LLMs often remain well-calibrated. This naturally motivates us to calibrate PoLLM confidence using the base LLM as a reference. This work proposes two ways to achieve this. A straightforward solution, **BaseCal-ReEval**, evaluates PoLLM's responses by feeding them into the base LLM to get average probabilities as confidence. While effective, this approach introduces additional inference overhead. To address this, we propose **BaseCal-Proj**, which trains a lightweight projection to map the final-layer hidden states of PoLLMs back to those of their base LLMs. These projected states are then processed by the base LLM's output layer to derive base-calibrated confidence for PoLLM's responses. Notably, BaseCal is an unsupervised, plug-and-play solution that operates without human labels or LLM modifications. Experiments across five datasets and three LLM families demonstrate the effectiveness of BaseCal, reducing Expected Calibration Error (ECE) by an average of 42.90% compared to the best unsupervised baselines.

1 Introduction

Hallucinations have become a critical challenge for large language models (LLMs). To mitigate this risk, a primary direction is to equip model responses with confidence scores that are well-calibrated with their actual accuracy (Guo et al., 2017; Geng et al., 2024). Such confidence enables abstention from low-confidence answers to mitigate hallucinations or otherwise alert users to potential errors. However, widely adopted post-trained language models (PoLLMs) have been found to exhibit significant overconfidence, often assigning high confidence even to incorrect responses (Achiam et al., 2023; Zhu et al., 2023).

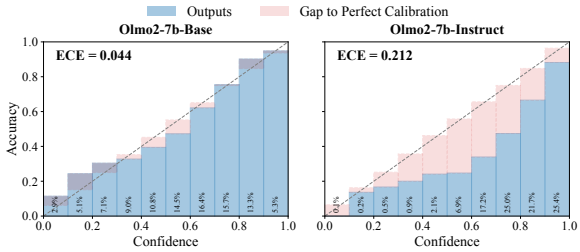


Figure 1: Calibration plots of PoLLM (right) and base LLM (left) on TriviaQA. The dashed line indicates perfect calibration; bars below it denote overconfidence.

Considerable efforts have been devoted to mitigating this overconfidence. One direction involves supervised strategies, such as calibration-oriented fine-tuning (Wang et al., 2025; Xiao et al., 2025) or temperature scaling using human annotation (Guo et al., 2017). However, these supervised methods rely heavily on human-labeled data, which is often difficult to obtain in real-world applications. In contrast, unsupervised approaches attempt to estimate confidence from PoLLM itself, such as aggregated token probabilities (Malinin and Gales, 2021), verbalized confidence (Tian et al., 2023; Xiong et al., 2024), and sampling-based consistency (Farquhar et al., 2024; Kuhn et al., 2023). While alleviating the reliance on human labels, these methods remain constrained by the quality of PoLLM signals, which often retain a certain degree of miscalibration (Tan et al., 2025; Simhi et al., 2025).

Inspired by findings that base LLMs are well-calibrated in multiple-choice tasks (Luo et al., 2025; Xiao et al., 2025), we investigate whether they can serve as an external reference to enhance PoLLM calibration in general QA tasks, i.e., free-form QA. Figure 1 (more results in §4.1) illustrates that base LLMs exhibit significantly superior confidence calibration compared to PoLLMs on the widely used QA dataset TriviaQA. Motivated by this observation, we seek to calibrate PoLLMs with their base counterparts.

A straightforward solution, **BaseCal-ReEval**, is to feed the PoLLM's response into its corresponding base LLM and utilize the base LLM's average token probabilities for this response as the confidence. While simple and effective, BaseCal-ReEval necessitates an additional forward pass on the base LLM, which significantly increases inference latency and resource consumption.

To mitigate this cost, we propose **BaseCal-Proj**, which approximates the base LLM's confidence via a lightweight projection. Specifically, we extract the final-layer states of PoLLM and base LLM for the same input questions and PoLLM-generated responses, and train a projection module to map the state of PoLLM to the target base state. During inference, the PoLLM's final-layer states are projected and passed through the base LLM's output layer to approximate the base probability distribution. We then calculate the confidence by averaging the probabilities assigned to the PoLLM-generated tokens under this distribution. Since BaseCal-Proj involves only a lightweight projection and an output layer, it incurs negligible overhead compared to BaseCal-ReEval and sampling-based methods (Kuhn et al., 2023; Farquhar et al., 2024).

Experiments across five datasets and three LLM families demonstrate the significant effectiveness of our methods. Compared to the best unsupervised baselines, BaseCal-ReEval reduces ECE by an average of 42.90%, and BaseCal-Proj achieves a 35.32% reduction but with significantly lower inference overhead. Further analysis shows that (i) these improvements persist across various model scales and post-training strategies; and (ii) BaseCal-Proj exhibits strong generalization capabilities on unseen questions. In conclusion, BaseCal serves as a plug-and-play framework to restore PoLLM calibration without parameter modification, making it easily adaptable to real-world applications.

2 Related Works

2.1 Overconfidence of LLMs

Emerging evidence indicates that post-training can induce systematic overconfidence, thus degrading model calibration (Xiao et al., 2025; Leng et al., 2025; Wang et al., 2025). This effect occurs in both instruction tuning and RLHF (Zhu et al., 2023; Leng et al., 2025), and strengthens with more tunable parameters (Chen et al., 2023). Prior work attributes this phenomenon to factors such as data overlap between fine-tuning and pretraining corpora (Wang et al., 2025), reward bias (Leng et al., 2025), preference collapse (Xiao et al., 2025), and catastrophic forgetting (He et al., 2023). To mitigate overconfidence, prior work has explored domain-specific fine-tuning (Xiao et al., 2025), confidence-aware reward (Leng et al., 2025), and feature-preserving adaptation (He et al., 2023).

2.2 Confidence Calibration for LLMs

Confidence calibration aims to align model confidence with the correctness of its predictions. On the one hand, **supervised methods** rely on human-labeled data to perform calibration, e.g., temperature scaling (Guo et al., 2017; Xie et al., 2024; Joy et al., 2023; Yu et al., 2022) optimizes a single temperature parameter to rescale probabilities. Other methods fine-tune models to provide calibrated confidence (Kapoor et al., 2024; Tao et al., 2024; Chen et al., 2023). Despite their effectiveness, supervised methods suffer from both reliance on human-labeled data (Shen et al., 2024) and limited generalization (Liu et al., 2025).

In contrast, **unsupervised methods** avoid reliance on human-annotated labels. *Prompt-based methods* directly query the model about the correctness of its own output, including P(true), which estimates confidence as the probability that the model judges its answer to be correct (Kadavath et al., 2022), and verbalized confidence, where confidence is explicitly expressed in natural language (Tian et al., 2023; Zhang et al., 2024; Lin et al., 2022; Xiong et al., 2024). *Sampling-based methods* estimate confidence from the uncertainty across response samples, either via semantic agreement (Manakul et al., 2023; Xiong et al., 2024; Lin et al., 2024; Raj et al., 2025) or through semantic entropy (Farquhar et al., 2024; Kuhn et al., 2023; Nikitin et al., 2024). However, these methods remain limited by their reliance on signals from the PoLLMs themselves, which preserve some extent of overconfidence (Tan et al., 2025; Simhi et al., 2025).

Closely related to our study, Luo et al. (2025) optimizes a temperature parameter to rescale PoLLM's output probabilities to match those of base LLMs. In contrast to their probability-level adjustments, we recover calibration from the model's hidden states, thereby leveraging richer internal information. More importantly, while their method is restricted to multiple-choice formats, our approach is more general and natively supports free-form generation, which is the predominant interaction paradigm for current LLMs

3 Preliminaries

Confidence calibration aims to align model confidence with the actual correctness probability of LLMs' responses. Let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^*)\}, i = 1, \ldots, N$ denote a dataset of N samples, where $\boldsymbol{x}_i$ is an input prompt and $\boldsymbol{y}_i^*$ is the corresponding ground-truth answer. Given $\boldsymbol{x}_i$, the model $\mathcal{M}$ generates a response $\boldsymbol{y}_i = (y_1, \ldots, y_T)$, where T is the length of the response. The response correctness $z_i \in \{0, 1\}$ is commonly evaluated by comparing the response with the ground-truth $\boldsymbol{y}_i^*$. A model is perfectly calibrated if

$$\mathbb{P}(z = 1 \mid \mathrm{c}(\boldsymbol{x}, \boldsymbol{y}) = q) = q. \quad (1)$$

which means that among all responses of which the model predicts a confidence of q (e.g., 70%), the proportion of correct answers should be q.

To quantify the degree of miscalibration, we employ the Expected Calibration Error (**ECE**) (Guo et al., 2017), which partitions the samples into M equal-width confidence bins $\{B_1, \ldots, B_M\}$, then calculates the absolute gap between average accuracy $\mathrm{acc}(B_m)$ and average confidence $\mathrm{conf}(B_m)$:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right|. \quad (2)$$

We primarily focus on free-form QA as it aligns better with the primary usage paradigm of current LLMs. In this task, the aggregated token probability (Malinin and Gales, 2021) is commonly treated as the **Vanilla** confidence, as it represents the most direct utilization of the model's signal, without requiring specific prompt design (Tian et al., 2023) or multiple sampling (Farquhar et al., 2024). We employ average aggregation following Orgad et al. (2025) to mitigate the effect of sequence length.

$$\mathrm{c}(\boldsymbol{x}_i, \boldsymbol{y}_i) = \frac{1}{T} \sum_{t=1}^{T} p(y_t \mid \mathbf{x_i}, \boldsymbol{y}_{<t}). \quad (3)$$

4 Calibration with Base LLMs

4.1 Motivation

Current unsupervised calibration methods are limited by the signals from PoLLMs, e.g., verbalized confidences are often very high (Leng et al., 2025), and consistency can fail under self-consistent errors (Tan et al., 2025). This motivates us to identify a more reliable external reference beyond PoLLMs. Inspired by findings that base LLMs are well-calibrated in multiple-choice tasks (Luo et al., 2025; Xiao et al., 2025; Achiam et al., 2023), we investigate whether this superior calibration persists in more challenging free-form QA tasks.

The experiments are conducted on the widely used QA dataset TriviaQA, using Qwen2.5 (Yang et al., 2024), Llama3.1 (Grattafiori et al., 2024), and the fully open-source Olmo2 (Walsh et al., 2025), which provide checkpoints at different post-training stages. This enables us to analyze the effects of various post-training strategies. As shown in Figure 2, PoLLMs are consistently more miscalibrated than their base counterparts across all three model families, indicating that the better calibration of base LLMs persists in free-form QA. Moreover, results on Olmo2 indicate that diverse post-training methods all impair confidence calibration, revealing a shared limitation of existing post-training approaches. Leveraging these insights, our core idea is to calibrate PoLLMs with their base counterparts, thereby eliminating the reliance on miscalibrated signals from the PoLLMs.

4.2 BaseCal-ReEval

As a direct instantiation of our idea, BaseCal-ReEval evaluates the PoLLM's generation by passing the identical input sequence through the base LLM to extract the corresponding target token probabilities as the confidence score (Figure 3b).

Let $\mathcal{M}_\mathrm{p}$ and $\mathcal{M}_\mathrm{b}$ denote the PoLLM and its corresponding base LLM, respectively. Given an input prompt $\boldsymbol{x}$, let $\boldsymbol{y}^\mathrm{p} = (y_1^\mathrm{p}, \ldots, y_T^\mathrm{p})$ denote the response sequence generated by $\mathcal{M}_\mathrm{p}$. To quantify the base LLM's confidence in $\boldsymbol{y}^\mathrm{p}$, we employ $\mathcal{M}_\mathrm{b}$ to score the sequence. Specifically, for each position t, we compute the probability that $\mathcal{M}_\mathrm{b}$ assigns to the token y_t^p produced by $\mathcal{M}_\mathrm{p}$, conditioned on the prompt and the preceding tokens $\boldsymbol{y}_{<t}^\mathrm{p}$. The sequence-level confidence $c_\mathrm{b}(\boldsymbol{x}, \boldsymbol{y})$ is then defined as the average probability of the generated tokens under the base LLM:

$$\mathrm{c}_\mathrm{b}(\boldsymbol{x}, \boldsymbol{y}^\mathrm{p}) = \frac{1}{T} \sum_{t=1}^{T} P_{\mathcal{M}_\mathrm{b}}(y_t^\mathrm{p} \mid \boldsymbol{x}, \boldsymbol{y}_{<t}^\mathrm{p}). \quad (4)$$

4.3 BaseCal-Proj

Despite its effectiveness, BaseCal-ReEval necessitates a full forward pass of the base LLM, incurring additional computational and memory overhead during inference. To mitigate this, we propose

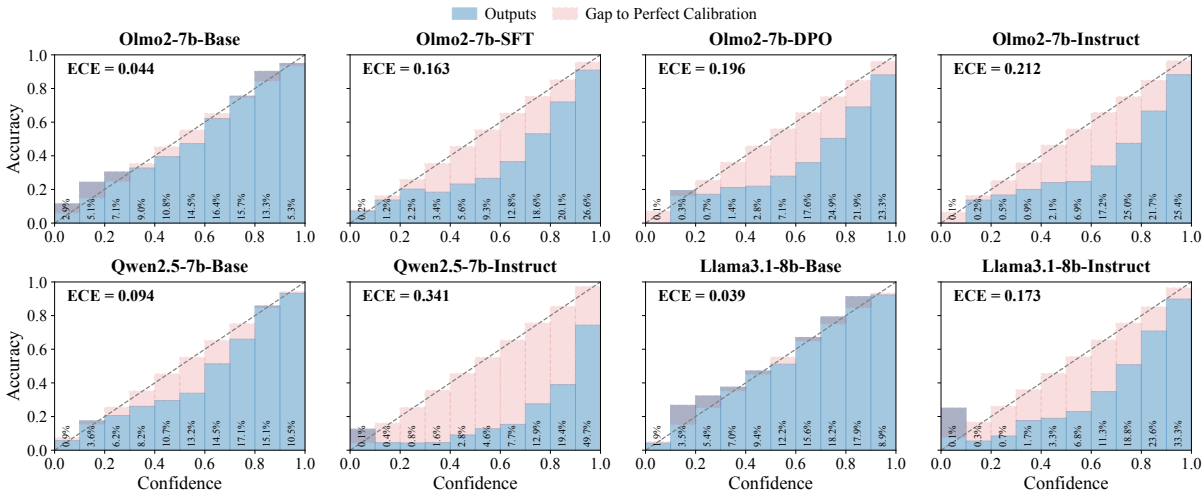
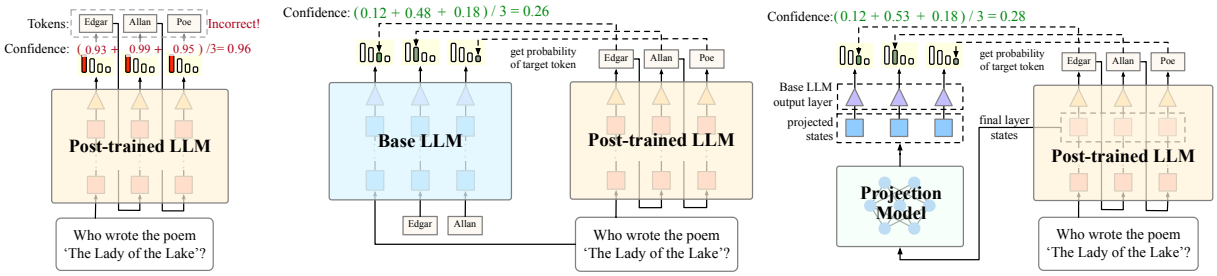


Figure 2: Calibration plots on TriviaQA across three model families. The top row presents Olmo2 checkpoints after different post-training stages. The dashed line is perfect calibration, while bars below denote overconfidence.



(a) Vanilla (b) BaseCal-ReEval (c) BaseCal-Proj

Figure 3: Frameworks of three methods. **(a) Vanilla:** use the PoLLM's own aggregated probabilities as confidence. **(b) BaseCal-ReEval:** evaluate the PoLLM-generated response with the corresponding base LLM. **(c) BaseCal-Proj:** learn a lightweight projection that maps PoLLM final-layer hidden states to the base-model space, and get confidence using the base LLM's output layer.

BaseCal-Proj, which learns a lightweight projection to map the PoLLM's final-layer hidden states into the base LLM's representation space. These projected states are then fed into the base LLM's output layer to derive the final probability, thereby bypassing the transformer blocks while preserving the base LLM's calibration benefit.

Prepare Hidden State Pairs. To learn the mapping from PoLLM hidden states to those of the base LLM, we extract the final-layer hidden states from both models, conditioned on *identical contexts*. Concretely, for the t-th token y_t^{p}, we extract the final-layer hidden state from the PoLLM, conditioned on the prompt and the prefix response:

$$\boldsymbol{h}_{t-1}^{\mathrm{p}} = \mathcal{M}_{\mathrm{p}}^{(L)}\big(\boldsymbol{x}; \boldsymbol{y}_{<t}^{\mathrm{p}}\big)\,, \tag{5}$$

where $\mathcal{M}^{(L)}(\cdot)$ denotes the mapping from the input sequence to the L-th (final) layer hidden state. Similarly, we feed the same concatenated sequence $(\boldsymbol{x}, \boldsymbol{y}_{<t}^{\mathrm{p}})$ into the base LLM to obtain the final-layer hidden state:

$$\boldsymbol{h}_{t-1}^{\mathrm{b}} = \mathcal{M}_{\mathrm{b}}^{(L)}\big(\boldsymbol{x}; \boldsymbol{y}_{<t}^{\mathrm{p}}\big)\,. \tag{6}$$

Repeating this procedure for all positions $t = 1, \ldots, T$, we get a sequence of hidden-state pairs $\{(\boldsymbol{h}_0^{\mathrm{p}}, \boldsymbol{h}_0^{\mathrm{b}}), \ldots, (\boldsymbol{h}_{T-1}^{\mathrm{p}}, \boldsymbol{h}_{T-1}^{\mathrm{b}})\}$ for each question.

Projection Training. Our goal is to minimize the discrepancy between the projected PoLLM hidden states and corresponding base-model representations. This objective requires two components: a projection model ϕ_θ and a loss function $\mathcal{L}$. For our primary implementation, we adopt a one-layer linear projection for ϕ_θ and Mean Squared Error (MSE) for $\mathcal{L}$ as a simple yet effective instantiation:

$$\mathcal{L}(\theta) = \frac{1}{T}\sum_{t=1}^{T} \big\|\phi_\theta\big(\boldsymbol{h}_{t-1}^{\mathrm{p}}\big) - \boldsymbol{h}_{t-1}^{\mathrm{b}}\big\|_2^2\,, \tag{7}$$
$$\phi_\theta\big(\boldsymbol{h}_{t-1}^{\mathrm{p}}\big) = \boldsymbol{W}\boldsymbol{h}_{t-1}^{\mathrm{p}} + \boldsymbol{b},$$

where $\boldsymbol{W} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{b} \in \mathbb{R}^{d}$ are learnable parameters. During training, we freeze both $\mathcal{M}_{\mathrm{p}}$ and $\mathcal{M}_{\mathrm{b}}$, optimizing only the projection parameters θ. Furthermore, we explore non-linear architectures and alternative loss functions in Section 5.4 and Appendix A.4.1, respectively.

Inference. Figure 3c shows the pipeline of BaseCal-Proj at inference time. We extract the final-layer hidden states of the post-trained LLM $\mathcal{M}_{\mathrm{p}}$, and transform them via the learned projection ϕ_θ. These projected states are then fed into the base LLM output layer $\boldsymbol{W}_{\mathrm{b}}^{\mathrm{o}}$ to get the probabilities of *target tokens* $[y_1^{\mathrm{p}}, \ldots, y_T^{\mathrm{p}}]$, which are then aggregated to the sequence-level confidence:

$$\tilde{\mathrm{c}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{T}\sum_{t=1}^{T} \mathrm{softmax}(\boldsymbol{W}_{\mathrm{b}}^{\mathrm{o}}\, \phi_\theta\big(\boldsymbol{h}_{t-1}^{\mathrm{p}}\big))[y_t^{\mathrm{p}}]. \quad (8)$$

Notably, the projected states are used only for confidence estimation. The PoLLM still performs generation with its original hidden states, thereby preserving the generation quality of PoLLM. Moreover, compared to standard inference, BaseCal-Proj introduces only a lightweight projection model and the base-model output layer, resulting in nearly negligible inference-time overhead.

5 Experiments

In this section, we conduct experiments to answer the following research questions[1]:

- **RQ1.** How do the proposed BaseCal-ReEval and BaseCal-Proj perform compared to the baselines in the confidence calibration task?
- **RQ2.** How does BaseCal-Proj generalize to unseen questions?
- **RQ3.** How do different projection models impact BaseCal-Proj?
- **RQ4.** How do the proposed methods perform across different PoLLMs, i.e., PoLLMs with different sizes and post-training strategies?
- **RQ5.** How does BaseCal-Proj benefit selective classification?

5.1 Experimental Setup

LLMs. We conduct experiments across a diverse set of model families, including Qwen2.5 (Yang et al., 2024), Llama3.1 (Grattafiori et al., 2024), and Olmo2 (Walsh et al., 2025) series. For each post-trained model, we use its corresponding pre-trained counterpart to perform calibration.

[1]Code will be released after the review process.

Datasets. Our evaluation covers an extensive range of free-form question answering benchmarks, including TriviaQA (Joshi et al., 2017), Natural Questions (NQ; Lee et al., 2019), SQuAD (Rajpurkar et al., 2016), and WebQuestions (WebQ; Berant et al., 2013). The correctness of responses is evaluated using LLM-as-a-judge, following Tian et al. (2023); Orgad et al. (2025). We further validate the effectiveness of LLM-as-a-judge via human verification. Appendix A.2 shows more details. To compare with DACA, we also include the widely-adopted multiple-choice benchmark MMLU (Hendrycks et al., 2021), which spans 57 subjects from STEM to humanities. Appendix A.1 shows detailed statistics of these datasets.

Metrics. Following previous works (Tian et al., 2023; Kadavath et al., 2022), we evaluate calibration using ECE (Guo et al., 2017) as introduced in Section 3 and the Brier Score (BS; BRIER, 1950), which measures the mean squared error between confidence and correctness labels.

Baselines. Since our approach does not rely on labeled data, we primarily compare it against *unsupervised* baselines.

(i) **Vanilla.** This approach directly employs the model's native probabilities as confidence. For free-form QA, we follow prior work (Orgad et al., 2025; Mahaut et al., 2024; Malinin and Gales, 2021) and aggregate the token-level probabilities of the generated sequences to obtain a single confidence score.

(ii) **P(True).** This method prompts the model to self-assess its own output and takes the predicted probability of answering "True" as the confidence score (Kadavath et al., 2022).

(iii) **Verbalization.** This method prompts PoLLMs to express confidence in natural language. We use the prompt from Tian et al. (2023).

(iv) **Semantic Entropy (SE).** SE (Farquhar et al., 2024; Kuhn et al., 2023) samples multiple responses and computes the entropy over semantic clusters derived from these samples. Following Li et al. (2025), we normalize the SE values to the range $[0, 1]$.

(v) **DACA.** DACA (Luo et al., 2025), which is designed specifically for the multiple-choice format, uses examples where the PoLLM and base LLM produce the same choice, and then learns a temperature to align the PoLLM's probabilities with those of the pre-trained model.

Model	Method	Unsupervised	TriviaQA		NQ		WebQ		SQuAD		MMLU	
			ECE (↓)	BS (↓)	ECE (↓)	BS (↓)	ECE (↓)	BS (↓)	ECE (↓)	BS (↓)	ECE (↓)	BS (↓)
Llama3.1-8B -Instruct	Temp. Scaling	✗	0.0226	0.1702	0.0460	0.1743	0.0930	0.2446	0.0911	0.1818	0.0307	0.2021
	Vanilla	✓	0.1725	0.2090	0.4532	0.3882	0.3832	0.3800	0.5255	0.4546	0.1071	0.2152
	P(true)	✓	0.2476	0.2506	0.4439	0.3972	0.4581	0.4496	0.5532	0.5094	0.2971	0.3079
	Verbalization	✓	0.1769	0.2046	0.2689	0.2507	0.3565	0.3456	0.3603	0.3208	0.2011	0.2633
	Semantic Entropy	✓	0.2443	0.2302	0.4927	0.4371	0.3635	0.3446	0.4645	0.4003	0.4085	0.3991
	DACA	✓	-	-	-	-	-	-	-	-	0.0473	0.1804
	BaseCal-Proj	✓	0.0387	0.1850	0.2488	0.2358	0.2091	0.2764	0.3134	0.2816	**0.0336**	0.1500
	BaseCal-ReEval	✓	**0.0309**	**0.1724**	**0.2462**	**0.2349**	**0.1873**	**0.2737**	**0.2959**	**0.2755**	0.0375	**0.1473**
Qwen2.5-7B -Instruct	Temp. Scaling	✗	0.0895	0.1850	0.1304	0.1938	0.0738	0.2341	0.0966	0.2007	0.2261	0.2354
	Vanilla	✓	0.3406	0.3118	0.5562	0.4839	0.4486	0.4328	0.6012	0.5550	0.2569	0.2607
	P(true)	✓	0.2113	0.2134	0.3723	0.3708	0.4838	0.4829	0.5666	0.5650	0.3204	0.3256
	Verbalization	✓	0.2889	0.3031	0.4718	0.4623	0.3977	0.3957	0.4800	0.4646	0.1972	0.2546
	Semantic Entropy	✓	0.3583	0.3191	0.5192	0.4502	0.2497	0.2853	0.4041	0.3821	0.2858	0.2856
	DACA	✓	-	-	-	-	-	-	-	-	0.0703	0.1618
	BaseCal-Proj	✓	0.1393	0.1923	0.3382	0.2880	**0.1792**	**0.2696**	0.4085	0.3569	0.0889	0.1662
	BaseCal-ReEval	✓	**0.1120**	**0.1789**	**0.3161**	**0.2714**	0.1983	0.2801	**0.3215**	**0.2980**	**0.0393**	**0.1465**
Olmo2-7B -Instruct	Temp. Scaling	✗	0.0286	0.1939	0.0742	0.1826	0.0674	0.2191	0.0587	0.1769	0.1707	0.2275
	Vanilla	✓	0.2121	0.2465	0.4404	0.3739	0.3638	0.3514	0.4459	0.3733	0.2465	0.2762
	P(true)	✓	0.2055	0.2393	0.3120	0.3166	0.4071	0.4043	0.4822	0.4582	0.1940	0.2549
	Verbalization	✓	0.2054	0.2721	**0.2030**	0.2478	0.2537	0.3101	0.2956	0.3040	0.1533	0.2747
	Semantic Entropy	✓	0.1910	0.2197	0.4178	0.3646	0.2469	0.2769	0.3906	0.3558	0.3132	0.3607
	DACA	✓	-	-	-	-	-	-	-	-	0.0555	0.1898
	BaseCal-Proj	✓	0.0314	0.1966	0.2712	0.2511	0.1967	0.2652	0.2393	0.2357	0.0525	0.1768
	BaseCal-ReEval	✓	**0.0269**	**0.1947**	0.2304	**0.2312**	**0.1587**	**0.2483**	**0.2131**	**0.2216**	**0.0470**	**0.1717**

Table 1: Calibration results across four datasets and three PoLLMs. **Bold** indicates the best unsupervised calibration performance, while underlining denotes the second best. Since DACA is specifically designed for multiple-choice tasks, it is marked with "-" on free-form QA datasets.

Beyond *unsupervised* baselines, we include **Temperature Scaling** (TS; Guo et al., 2017) as a *supervised* reference baseline, which learns a temperature parameter by minimizing the negative log-likelihood with respect to answer correctness. More details are shown in Appendix A.3.

Implementation Details. The training of BaseCal-Proj leverages training set questions to derive hidden state pairs without relying on human-annotated labels. We mitigate overfitting via an early stopping mechanism, monitored by the loss on the validation set. Statistics for the training and validation sets are detailed in Appendix A.1.

5.2 Overall Performance Comparison (RQ1)

To answer **RQ1**, we compare the performance of our methods with the above baselines. Table 1 shows the results on five datasets across three PoLLMs. We analyze the results from various perspectives and obtain the following observations:

(i) Comparison with Unsupervised Methods. As shown in Table 1, BaseCal-ReEval and BaseCal-Proj reduce the average ECE by 55.65% and 49.78% compared to the Vanilla PoLLMs, respectively. Compared to other unsupervised methods, our methods achieve the best calibration performance in 29 out of 30 experimental settings (5 datasets $\times$ 3 PoLLMs $\times$ 2 metrics). While Verbalization works well in Olmo2-7B-Instruct on NQ, its performance varies greatly and drops substantially on other models like Qwen2.5-7B. This instability likely arises from their strong reliance on the instruction-following ability. In contrast, BaseCal-ReEval consistently ranks among the top two across all settings, significantly outperforming Verbalization (average ECE 0.1641 vs. 0.2874).

Notably, on multiple-choice datasets, DACA exhibits performance that significantly outperforms other baselines. However, compared to DACA, BaseCal-ReEval achieves even superior results (e.g., 0.0393 vs. 0.0703 ECE on Qwen2.5). This underlines the efficacy of directly leveraging base LLM confidence, as opposed to rescaling probabilities to approximate the base distribution. Meanwhile, BaseCal-Proj also remains superior or comparable to DACA, likely benefiting from the richer information within the hidden states compared to probability-level scaling by DACA.

(ii) Comparison with Supervised Methods. Despite being unsupervised, BaseCal achieves performance comparable to supervised TS on TriviaQA and MMLU. In the remaining settings, while BaseCal underperforms TS baseline, it maintains the best performance among unsupervised methods

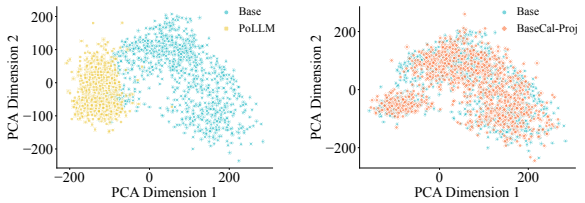


Figure 4: Visualization of hidden states for PoLLM vs. Base LLM (left) and BaseCal-Proj vs. Base LLM (right), for 2500 randomly sampled examples on TriviaQA.

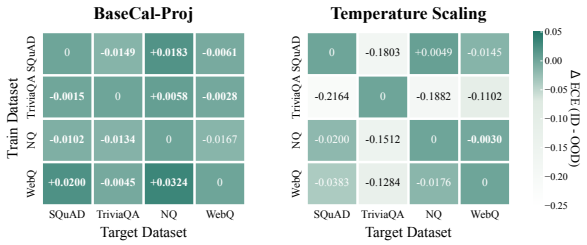


Figure 5: Comparison of ΔECE for BaseCal-Proj and Temperature Scaling for Llama3.1-8b. Darker colors indicate a better out-of-domain performance. The results for other LLMs are shown in Appendix A.4.2.

and significantly narrows the gap with supervised approaches. These results highlight BaseCal as a compelling, label-efficient alternative for scenarios where ground-truth data is unavailable.

(iii) Comparison between BaseCal-Proj and BaseCal-ReEval. As shown in Table 1, BaseCal-Proj achieves an average ECE of 0.1859 across all settings, closely approximating the performance of BaseCal-ReEval (0.1641), which directly uses base LLMs to get confidence. This means that BaseCal-Proj successfully recovers the calibration of base LLMs without the cost of an extra forward pass on them. To further understand the mechanism behind this, we visualize the hidden states of PoLLM, BaseCal-Proj, and base LLM in Figure 4. As illustrated, while the hidden states of the PoLLM are distinct from those of the base LLM, the projected states closely align with those of the base LLM. This visualization confirms that the projection layer effectively maps the post-trained hidden states back to the base LLM, thereby explaining their comparable calibration performance.

5.3 Generalization of BaseCal-Proj (RQ2)

Although BaseCal-Proj does not rely on human labels, it still requires questions for training, raising the question of whether it can generalize to unseen questions. To investigate this, we conduct a cross-dataset generalization evaluation across four QA datasets: SQuAD, NQ, TriviaQA, and WebQ. We compare the in-domain (ID) and out-of-domain (OOD) performance by calculating the difference in their ECE values, defined as $\Delta\text{ECE} = \text{ECE}_{\text{ID}} - \text{ECE}_{\text{OOD}}$. A higher ΔECE indicates superior generalization capability, where $\Delta\text{ECE} > 0$ signifies that the model's OOD calibration outperforms its ID performance. We compare the generalization of BaseCal-Proj against the supervised method, temperature scaling. Notably, BaseCal-Proj is trained solely using the questions from the training dataset, while temperature scaling utilizes both the questions and human-labeled ground truth.

Figure 5 presents the ΔECE heatmap for BaseCal-Proj and Temperature Scaling. Darker colors in the heatmap indicate better OOD calibration performance. Overall, **BaseCal-Proj achieves an average ΔECE of +0.0005, indicating that its OOD calibration performance is practically on par with its in-domain results**. In contrast, Temperature Scaling suffers from a significant performance degradation with an average ΔECE of -0.0886. This failure stems from its reliance on post-hoc rescaling to fit the specific correctness label distribution of the training data, rendering it ineffective when accuracy levels shift across datasets (e.g., between TriviaQA and NQ). In contrast, BaseCal-Proj is label-agnostic and focuses on recovering the intrinsic calibration information embedded within the model's hidden states. This prevents overfitting to specific datasets, allowing BaseCal-Proj to maintain consistent performance in unseen questions.

Model	Projection	TriviaQA		NQ		MMLU	
		ECE (↓)	BS (↓)	ECE (↓)	BS (↓)	ECE (↓)	BS (↓)
Qwen2.5-7B -Instruct	Non-Linear	0.0504	0.1968	0.2813	0.2573	0.0790	0.1633
	Linear	0.1393	0.1923	0.3382	0.2880	0.0889	0.1662
Llama3.1-8B -Instruct	Non-Linear	0.1526	0.2209	0.2052	0.2251	0.0166	0.1487
	Linear	0.0387	0.1850	0.2488	0.2358	0.0336	0.1500
Olmo2-7B -Instruct	Non-Linear	0.0783	0.2133	0.1933	0.2228	0.0508	0.1742
	Linear	0.0314	0.1966	0.2712	0.2511	0.0525	0.1768

Table 2: Performance comparison of BaseCal-Proj with different projection model architectures.

5.4 Effect of Projection Architecture (RQ3)

Previous experiments adopted a simple linear projection. Although its effectiveness has been empirically validated, an open question remains as to whether more expressive projection models can further improve calibration performance. We additionally evaluate a more complex projection model, a three-layer multilayer perceptron with ReLU nonlinearities. As shown in Table 2, the model with

Method	Olmo2-7b-SFT	Olmo2-7b-DPO	Olmo2-7b-RLVR
Vanilla	0.1628	0.1958	0.2121
Verbalization	0.3004	0.2274	0.2054
Semantic Entropy	0.2583	0.1966	0.1910
BaseCal-Proj	**0.0582**	**0.0269**	**0.0314**

Table 3: ECE of BaseCal-Proj and baselines across different post-training stages on TriviaQA.

Scale	ECE (↓)			BS (↓)		
	Vanilla	BC-Proj	BC-ReEval	Vanilla	BC-Proj	BC-ReEval
7B	0.3406	0.1393	**0.1120**	0.3118	0.1923	**0.1789**
14B	0.2687	0.0778	**0.0663**	0.2652	0.1715	**0.1674**
32B	0.2662	0.0854	**0.0542**	0.2514	0.1593	**0.1623**
72B	0.2089	0.0502	**0.0440**	0.2078	0.1433	**0.1396**

Table 4: Calibration performance across different LLM scales of Qwen2.5 instruct series on TriviaQA. **BC** denotes our BaseCal methods.

non-linearities achieves a slightly lower average ECE than the linear projection (0.1231 vs. 0.1381). These results suggest that a simple linear projection is already sufficient to effectively recover the calibration, and that increasing the model complexity yields only slight gains.

5.5 Impact of Different PoLLMs (RQ4)

To answer **RQ4**, we investigate the performance of our proposed method across PoLLMs with varying post-training strategies and scales.

First, we evaluate BaseCal-Proj with PoLLMs after different post-training strategies using Olmo-2 checkpoints, including SFT, DPO, and RLVR. As shown in Table 3, **BaseCal-Proj consistently achieves the best performance across all post-training strategies**. Notably, verbalization exhibits inconsistent performance across different strategies, even underperforming vanilla for DPO and SFT, due to its heavy reliance on the model's instruction-following capability. In contrast, BaseCal-Proj effectively recovers well-calibrated confidence agnostic to specific post-training strategies.

Subsequently, the influence of model size is analyzed using Qwen2.5-7b/14b/32b/72b-instruct. Table 4 shows that **BaseCal-Proj consistently yields substantial improvements across all scales**. Furthermore, BaseCal-Proj achieves superior performance on larger models, due to the stronger calibration capabilities of larger base LLMs, a trend also observed in previous work (Zhu et al., 2023).

5.6 Application (RQ5)

Selective classification (Geifman and El-Yaniv, 2017) enables a model to abstain from making predictions when confidence is low, thereby enhancing reliability by trading off coverage for higher accuracy. This mechanism is particularly critical for LLMs in decision-making tasks, where unreliable outputs can lead to severe consequences. In this section, we evaluate the effectiveness of our proposed BaseCal-Proj in the context of selective classification. Figure 6 compares the accuracy of the vanilla baseline and BaseCal-Proj by varying the confidence threshold from 0.5 to 0.95, where prediction below the threshold is rejected. Experimental results demonstrate that BaseCal-Proj consistently surpasses the vanilla method in accuracy across all confidence thresholds, highlighting its superior capability in identifying reliable predictions.

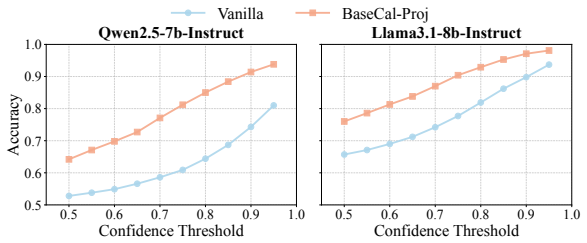


Figure 6: Selective classification accuracy on TriviaQA. Accuracy is computed on examples with confidence scores above thresholds ranging from 0.5 to 0.95.

6 Conclusion

To address the overconfidence in PoLLMs, this work first reveals a critical phenomenon where base LLMs retain superior calibration compared to their post-trained counterparts, even on general QA tasks. Motivated by this, we propose BaseCal, an unsupervised, plug-and-play framework that calibrates PoLLMs with their base counterparts without modifying PoLLM's parameters or compromising generation quality. Experiments demonstrate that BaseCal substantially mitigates overconfidence across diverse datasets, LLM size, and post-training strategies.

Beyond performance improvements, our findings provide a key insight into the effect of post-training on confidence calibration. The effectiveness of a simple linear projection suggests that calibration information is not entirely lost during post-training; instead, it remains recoverable via a simple linear transformation on the internal states. By restoring calibration without expensive retraining or human supervision, our method offers a practical solution for deploying reliable LLMs.

7 Limitations

Our primary focus is to provide a practical and efficient calibration framework based on the observation that base LLMs are relatively well-calibrated. While a theoretical analysis of the underlying mechanisms driving this phenomenon is an important direction, it falls outside the scope of this work and is left as a promising avenue for future research. Furthermore, we mainly evaluate standard LLMs. It remains to be explored whether our findings generalize to other specialized architectures, such as Mixture-of-Experts (MoE) or reasoning LLMs. Additionally, BaseCal-Proj requires access to the base LLM's output layer, which may limit its applicability in scenarios involving closed-source API-based models or environments where internal model parameters are inaccessible.

8 Ethics Statement

Data All data used in this study are publicly available and do not raise any privacy concerns.

AI Writing Assistance We used ChatGPT solely to refine and polish the textual expressions. It was not used to generate new ideas or influence the method design.

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*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

GLENN W. BRIER. 1950. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1 – 3.

Yangyi Chen, Lifan Yuan, Ganqu Cui, Zhiyuan Liu, and Heng Ji. 2023. A close look into the calibration of pre-trained language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1343–1367, Toronto, Canada. Association for Computational Linguistics.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Yonatan Geifman and Ran El-Yaniv. 2017. Selective classification for deep neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4885–4894, Red Hook, NY, USA. Curran Associates Inc.

Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A survey of confidence estimation and calibration in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6577–6595, Mexico City, Mexico. Association for Computational Linguistics.

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

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Guande He, Jianfei Chen, and Jun Zhu. 2023. Preserving pre-trained features helps calibrate fine-tuned language models. In *The Eleventh International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Tom Joy, Francesco Pinto, Ser-Nam Lim, Philip HS Torr, and Puneet K Dokania. 2023. Sample-dependent adaptive temperature scaling for improved calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14919–14926.

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

Sanyam Kapoor, Nate Gruver, Manley Roberts, Arka Pal, Samuel Dooley, Micah Goldblum, and Andrew Wilson. 2024. Calibration-tuning: Teaching large language models to know what they don't know. In *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, pages 1–14, St Julians, Malta. Association for Computational Linguistics.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Jixuan Leng, Chengsong Huang, Banghua Zhu, and Jiaxin Huang. 2025. Taming overconfidence in LLMs: Reward calibration in RLHF. In *The Thirteenth International Conference on Learning Representations*.

Rui Li, Jing Long, Muge Qi, Heming Xia, Lei Sha, Peiyi Wang, and Zhifang Sui. 2025. Towards harmonized uncertainty estimation for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22938–22953, Vienna, Austria. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2024. Generating with confidence: Uncertainty quantification for black-box large language models. *Transactions on Machine Learning Research*.

Hongfu Liu, Hengguan Huang, Xiangming Gu, Hao Wang, and Ye Wang. 2025. On calibration of LLM-based guard models for reliable content moderation. In *The Thirteenth International Conference on Learning Representations*.

Beier Luo, Shuoyuan Wang, Sharon Li, and Hongxin Wei. 2025. Your pre-trained LLM is secretly an unsupervised confidence calibrator. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

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

Andrey Malinin and Mark Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. 2024. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. *Advances in Neural Information Processing Systems*, 37:8901–8929.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. 2025. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *The Thirteenth International Conference on Learning Representations*.

Harsh Raj, Vipul Gupta, Domenic Rosati, and Subhabrata Majumdar. 2025. Improving consistency in large language models through chain of guidance. *Transactions on Machine Learning Research*.

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

Maohao Shen, Subhro Das, Kristjan Greenewald, Prasanna Sattigeri, Gregory W. Wornell, and Soumya Ghosh. 2024. Thermometer: Towards universal calibration for large language models. In *Forty-first International Conference on Machine Learning*.

Adi Simhi, Itay Itzhak, Fazl Barez, Gabriel Stanovsky, and Yonatan Belinkov. 2025. Trust me, I'm wrong: LLMs hallucinate with certainty despite knowing the answer. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 14665–14688, Suzhou, China. Association for Computational Linguistics.

Hexiang Tan, Fei Sun, Sha Liu, Du Su, Qi Cao, Xin Chen, Jingang Wang, Xunliang Cai, Yuanzhuo Wang, Huawei Shen, and Xueqi Cheng. 2025. Too consistent to detect: A study of self-consistent errors in LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 4755–4765, Suzhou, China. Association for Computational Linguistics.

Shuchang Tao, Liuyi Yao, Hanxing Ding, Yuexiang Xie, Qi Cao, Fei Sun, Jinyang Gao, Huawei Shen, and Bolin Ding. 2024. When to trust LLMs: Aligning confidence with response quality. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5984–5996, Bangkok, Thailand. Association for Computational Linguistics.

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

Evan Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, and 23 others. 2025. 2 OLMo 2 furious (COLM's version). In *Second Conference on Language Modeling*.

Ziming Wang, Zeyu Shi, Haoyi Zhou, Shiqi Gao, Qingyun Sun, and Jianxin Li. 2025. Towards objective fine-tuning: How LLMs' prior knowledge causes potential poor calibration? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14830–14853, Vienna, Austria. Association for Computational Linguistics.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. *Preprint*, arXiv:2411.04368.

Yuxi Xia, Pedro Henrique Luz De Araujo, Klim Zaporojets, and Benjamin Roth. 2025. Influences on LLM calibration: A study of response agreement, loss functions, and prompt styles. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3740–3761, Vienna, Austria. Association for Computational Linguistics.

Jiancong Xiao, Bojian Hou, Zhanliang Wang, Ruochen Jin, Qi Long, Weijie J Su, and Li Shen. 2025. Restoring calibration for aligned large language models: A calibration-aware fine-tuning approach. In *Forty-second International Conference on Machine Learning*.

Johnathan Xie, Annie S Chen, Yoonho Lee, Eric Mitchell, and Chelsea Finn. 2024. Calibrating language models with adaptive temperature scaling. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*.

An Yang, Baosong Yang, Beichen Zhang, and et al. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Yaodong Yu, Stephen Bates, Yi Ma, and Michael Jordan. 2022. Robust calibration with multi-domain temperature scaling. *Advances in Neural Information Processing Systems*, 35:27510–27523.

Mozhi Zhang, Mianqiu Huang, Rundong Shi, Linsen Guo, Chong Peng, Peng Yan, Yaqian Zhou, and Xipeng Qiu. 2024. Calibrating the confidence of large language models by eliciting fidelity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2959–2979, Miami, Florida, USA. Association for Computational Linguistics.

Chiwei Zhu, Benfeng Xu, Quan Wang, Yongdong Zhang, and Zhendong Mao. 2023. On the calibration of large language models and alignment. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9778–9795, Singapore. Association for Computational Linguistics.

A Appendix

A.1 Further Information about Datasets

Dataset	Train	Used Train	Dev	Used Dev	Test
TriviaQA	70,098	10000	17,524	2000	11,313
SQuAD	70,079	10000	17,520	2000	10,570
NQ	79,168	10000	8,757	2000	3,610
WebQ	3,022	3022	756	756	2,032
MMLU	98,843	10000	10,000	2000	14,042

Table 5: Statistics of the datasets. "Used Train" and "Used Dev" show the number of data used to train or validate our BaseCal-Proj method.

A.1.1 Statistics

Table 5 shows the detailed statistics of the datasets. To construct the training data for BaseCal-Proj, we randomly sample 10,000 questions from the original training sets, as this scale suffices for our lightweight linear projection model. To mitigate the risk of overfitting, we further sample 2,000 questions from the dev set for validation and employ an early stopping strategy based on the validation loss. For the WebQ dataset, which contains fewer instances, we utilize all 3,022 available questions for training and 756 for validation. Crucially, during both the training and validation phases, our method relies solely on the input questions without accessing any human-labeled ground truth.

A.1.2 Generation Prompts

In this section, we present the prompts used for response generation across the different datasets. For the QA datasets, including TriviaQA, SQuAD, WebQ, and NQ, we employ the following prompt template:

```
Answer the question briefly. {5 examples}
Question: {query}
Answer:
```

For the MMLU multiple-choice dataset, we utilize the following prompt to elicit answers and extract the option labels (A, B, C, or D):

```
Answer the following multiple-choice question. Reply with only A, B, C, or D. {5 examples}
Question: {question}
A. {choices[0]}  B. {choices[1]}
C. {choices[2]}  D. {choices[3]}
Answer:
```

A.2 Details about LLM-as-a-Judge

To assess the correctness of model outputs in QA tasks, we employ an LLM-as-a-judge to determine whether the generated responses are semantically equivalent to the ground truth, following Tian et al. (2023); Wei et al. (2024). For reproducibility, we utilize the powerful open-source Qwen2.5-14B-Instruct model as our judge. Inspired by the evaluation protocol from Wei et al. (2024), we use the specific prompt detailed in Section A.5 to verify response correctness. To validate the quality of the LLM judgment, we conducted a manual audit on 100 randomly sampled instances. Comparative verification against human ground truth reveals a disagreement rate of only 1% with human annotations, confirming the reliability of our automated evaluation process.

A.3 Further Information about Baselines

This section provides further details for the calibration baselines.

(1) Vanilla. Several studies have employed the aggregated token probabilities as the vanilla confidence (Orgad et al., 2025; Mahaut et al., 2024; Malinin and Gales, 2021). Following implementation in Orgad et al. (2025), we average the probabilities of all tokens in a response as the confidence.

(2) Verbalization. This method leverages the instruction-following capability of models to express confidence in natural language. Following the implementation of Tian et al. (2023), we employ the following prompt:

```
Provide the probability that your guess is correct. Give ONLY the probability, no other words or explanation. For example:
Probability: <the probability between 0.0 and 1.0 that your guess is correct, without any extra commentary whatsoever; just the probability!>
The question is:question
The best guess is: response
Probability:
```

The resulting output is then parsed using regular expressions to extract a floating-point value between 0.0 and 1.0, which serves as the confidence score. Notably, this method requires an additional forward pass on the PoLLMs after the initial response is generated, thereby increasing the overall computational overhead.

(3) P(True). This method follows the prompting strategy introduced by Kadavath et al. (2022), where the LLM is directly queried to assess the correctness of its own output. P(True) also requires an additional forward pass on the PoLLMs after the initial response is generated. Specifically, we construct the following prompt:

```
Question: {question}
Possible answer: {response}
Is the possible answer:
A. True  B. False
The possible answer is:
```

The model's confidence is then quantified as the probability it assigns to the token "A".

(4) Semantic Entropy. As proposed by Kuhn et al. (2023); Farquhar et al. (2024), semantic entropy estimates confidence by sampling multiple responses for the same question and computing their semantic consistency. Following the implementation details recommended by Kuhn et al. (2023), we set the sampling parameters as follows: temperature 0.5, number of samples 10, top_p = 1.0, and top_k = -1. Since this metric was originally designed for hallucination detection, its raw scores do not fall within the $[0, 1]$ interval. Therefore, following Li et al. (2025), we normalize the semantic entropy values to the range $[0, 1]$ to facilitate the calculation of ECE.

(5) DACA. DACA (Luo et al., 2025) is an unsupervised approach that optimizes a temperature parameter to scale the probability of PoLLM, thereby approximating the probability distribution of the base LLM. This probability-level scaling necessitates that both the base LLM and PoLLM produce

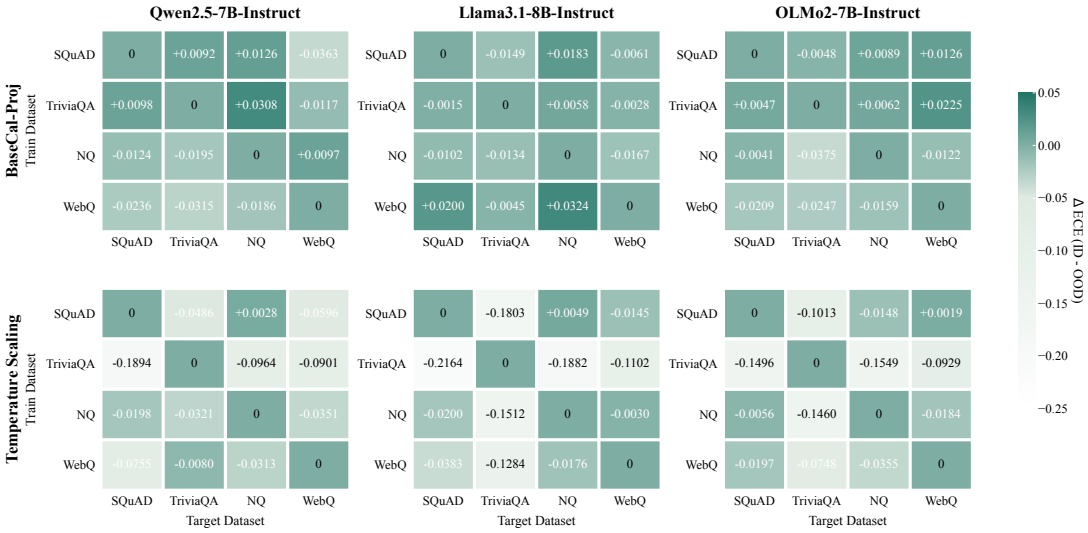


Figure 7: Comparison of ΔECE for BaseCal-Proj and Temperature Scaling. Darker colors indicate a better out-of-domain performance.

Model	Projection	TriviaQA		NQ		MMLU	
		ECE (↓)	BS (↓)	ECE (↓)	BS (↓)	ECE (↓)	BS (↓)
Qwen2.5-7B -Instruct	MSE	0.1393	0.1923	0.3382	0.2880	0.0889	0.1662
	MAE	0.1260	0.1898	0.3276	0.2806	0.0882	0.1665
	Cosine	0.5011	0.5009	0.2543	0.2548	0.0396	0.1592
Llama3.1-8B -Instruct	MSE	0.0387	0.1850	0.2488	0.2358	0.0336	0.1500
	MAE	0.0447	0.1854	0.2421	0.2339	0.0332	0.1500
	Cosine	0.6125	0.6057	0.2483	0.2597	0.0221	0.1489
Olmo2-7B -Instruct	MSE	0.0314	0.1966	0.2712	0.2511	0.0525	0.1768
	MAE	0.0297	0.1969	0.2676	0.2493	0.0587	0.1768
	Cosine	0.5651	0.5636	0.2929	0.2935	0.0384	0.1718

Table 6: Performance comparison of BaseCal-Proj trained with different loss functions.

the **exact same top-1 token** for a given input, a constraint theoretically proved and empirically analyzed in Section 3 (pp. 4-5) of Luo et al. (2025). This limitation restricts DACA's applicability primarily to multiple-choice tasks, as identical token generation is very rare for base LLM and PoLLM in free-form QA. Accordingly, we compare our method with DACA on the MMLU benchmark in the main experiments. We utilize the official implementation (Luo et al., 2025) and optimize DACA using the same training set employed for our proposed method.

(6) Temperature Scaling. We directly adopt the implementation of temperature scaling from Xia et al. (2025). This is a supervised method that optimizes an optimal temperature parameter to rescale probabilities, thereby minimizing the Expected Calibration Error (ECE). The training process requires questions and human-labeled answers to determine response correctness. In our implementation, we utilize the entire training dataset to optimize the temperature.

A.4 Further Information about BaseCal-Proj

A.4.1 Effect of Loss Function

The default BaseCal-Proj optimization utilizes the Mean Squared Error (MSE) to minimize the Euclidean distance between the projected PoLLM states and the target states:

$$\mathcal{L}_{\text{MSE}}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\left\|\phi_\theta\left(\boldsymbol{h}^{\text{p}}_{t-1}\right) - \boldsymbol{h}^{\text{b}}_{t-1}\right\|_2^2 .$$

To assess the effect of different loss functions, we explore two alternative objectives. First, we consider the Mean Absolute Error (MAE), which imposes an L_1 penalty:

$$\mathcal{L}_{\text{MAE}}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\left\|\phi_\theta\left(\boldsymbol{h}^{\text{p}}_{t-1}\right) - \boldsymbol{h}^{\text{b}}_{t-1}\right\|_1 .$$

Second, we examine the Cosine Loss, which prioritizes angular alignment over magnitude:

$$\mathcal{L}_{\text{Cos}}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\left(1 - \frac{\phi_\theta\left(\boldsymbol{h}^{\text{p}}_{t-1}\right)^\top \boldsymbol{h}^{\text{b}}_{t-1}}{\left\|\phi_\theta\left(\boldsymbol{h}^{\text{p}}_{t-1}\right)\right\|_2 \left\|\boldsymbol{h}^{\text{b}}_{t-1}\right\|_2}\right) .$$

To ensure a fair comparison, we maintain a consistent architecture using a single linear layer for all projection models. Table 6 shows the calibration performance of BaseCal-Proj trained with MSE, MAE, and Cosine Loss. The experimental results reveal that $\mathcal{L}_{\text{Cos}}$ exhibits significant instability across different benchmarks. While it achieves competitive calibration on MMLU, its performance degrades sharply on TriviaQA. This

discrepancy suggests that solely optimizing the angular alignment of hidden states is insufficient for confidence calibration. In contrast, both MSE and MAE demonstrate consistently superior performance across all tested models and datasets.

A.4.2 More Results about Generalization

Figure 7 illustrates the generalization gap, ΔECE, for the Qwen, Llama, and Olmo models. It can be observed that across all model-dataset combinations, the ΔECE for our method is close to 0, demonstrating that BaseCal-Proj achieves OOD performance comparable to its ID performance. In contrast, temperature scaling generally exhibits significantly lower ΔECE values, reaching as low as -0.2164. This indicates a significant degradation in the performance of temperature scaling when generalizing across different datasets.

A.5 Prompt for LLM-as-a-Judge

Prompt for LLM-as-a-Judge

Your job is to look at a question, some gold targets, and a predicted answer, and then assign a grade of either ["CORRECT", "INCORRECT", "NOT_ATTEMPTED"].
First, I will give examples of each grade, and then you will grade a new example.

The following are examples of CORRECT predicted answers.

Question: What are the names of Barack Obama's children?
Gold target: ["Malia Obama and Sasha Obama", "malia and sasha"]
Predicted answer 1: sasha and malia obama
Predicted answer 2: most people would say Malia and Sasha, but I'm not sure and would have to double check
Predicted answer 3: Barack Obama has two daughters. Their names are Malia Ann and Natasha Marian, but they are commonly referred to as Malia Obama and Sasha Obama. Malia was born on July 4, 1998, and Sasha was born on June 10, 2001.

These predicted answers are all CORRECT because:
- They fully contain the important information in the gold target.
- They do not contain any information that contradicts the gold target.
- Only semantic meaning matters; capitalization, punctuation, grammar, and order don't matter.
- Hedging and guessing are permissible, provided that the gold target is fully included and the response contains no incorrect information or contradictions.

The following are examples of INCORRECT predicted answers.

Question: What are the names of Barack Obama's children?
Gold target: ["Malia and Sasha"]
Predicted answer 1: Malia.
Predicted answer 2: Malia, Sasha, and Susan.
Predicted answer 3: Barack Obama does not have any children.
Predicted answer 4: I think it's either Malia and Sasha. Or it could be Malia and Jackie. Or it could be Joey and Malia.
Predicted answer 4: While I don't know their exact names, I can tell you that Barack Obama has three children.
Predicted answer 5: It's possible you may mean Betsy and Olivia. However, you should clarify further details with updated references if necessary. Is that the correct answer?
Predicted answer 6: It may be the case that Obama's child is named James. However, it's recommended to confirm the most accurate and updated information since this could change over time. This model may not always reflect the most current information.

These predicted answers are all INCORRECT because:
- A factual statement in the answer contradicts the gold target. Incorrect statements that have some hedging (e.g., "it is possible that", "although i'm not sure, i think") are also considered incorrect.

The following are examples of NOT_ATTEMPTED predicted answers.

Question: What are the names of Barack Obama's children?
Gold target: ["Malia and Sasha"]
Predicted answer 1: I don't know.
Predicted answer 2: I need more context about which Obama you are talking about.
Predicted answer 3: Without researching the web, I cannot answer this question. However, I can tell you that Barack Obama has two children.
Predicted answer 4: Barack Obama has two children. I know that one of them is Malia, but I'm not sure about the other one.

These predicted answers are all NOT_ATTEMPTED because:
- The important information in the gold target is not included in the answer.
- No statements in the answer contradict the gold target.

Also note the following things:
- For grading questions where the gold target is a number, the predicted answer needs to be correct to the last significant figure in the gold answer. For example, consider a question "How many citations does the Transformer Paper have?" with gold target "120k".
  - Predicted answers "120k", "124k", and 115k" are all CORRECT.

- Predicted answers "100k" and "113k" are INCORRECT.
  - Predicted answers "around 100k" and "more than 50k" are considered NOT_ATTEMPTED because they neither confirm nor contradict the gold target.
- The gold target may contain more information than the question. In such cases, the predicted answer only needs to contain the information that is in the question.
  - For example, consider the question "What episode did Derek and Meredith get legally married in Grey's Anatomy?" with gold target "Season 7, Episode 20: White Wedding". Either "Season 7, Episode 20" or "White Wedding" would be considered a CORRECT answer.
- Do not punish predicted answers if they omit information that would be clearly inferred from the question.
  - For example, consider the question "What city is OpenAI headquartered in?" and the gold target "San Francisco, California". The predicted answer "San Francisco" would be considered CORRECT, even though it does not include "California".
  - Consider the question "What award did A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity win at NAACL '24?", the gold target is "Outstanding Paper Award". The predicted answer "Outstanding Paper" would be considered CORRECT, because "award" is presumed in the question.
  - For the question "What is the height of Jason Wei in meters?", the gold target is "1.73 m". The predicted answer "1.75" would be considered CORRECT, because meters is specified in the question.
  - For the question "What is the name of Barack Obama's wife?", the gold target is "Michelle Obama". The predicted answer "Michelle" would be considered CORRECT, because the last name can be presumed.
- Do not punish for typos in people's name if it's clearly the same name.
  - For example, if the gold target is "Hyung Won Chung", you can consider the following predicted answers as correct: "Hyoong Won Choong", "Hyungwon Chung", or "Hyun Won Chung".

Here is a new example. Simply reply with either CORRECT, INCORRECT, NOT_ATTEMPTED. Don't apologize or correct yourself if there was a mistake; we are just trying to grade the answer.

Question: {question}
Gold target: {target}
Predicted answer: {predicted_answer}

Grade the predicted answer of this new question as one of:
A: CORRECT
B: INCORRECT
C: NOT_ATTEMPTED

Just return the letters "A", "B", or "C", with no text around it.