WHEN MODELS KNOW WHEN THEY DO NOT KNOW CALIBRATION, CASCADING, AND CLEANING

Anonymous authors
Paper under double-blind review

ABSTRACT

When a model knows when it does not know, many possibilities emerge. The first question is how to enable a model to recognize that it does not know. A promising approach is to use confidence, computed from the model's internal signals, to reflect its ignorance. Prior work in specific domains has shown that calibration can provide reliable confidence estimates. In this work, we propose a simple, effective, and almost training-free framework that applies to both vision and language models, spanning model calibration, cascading, and data cleaning to better exploit a model's ability to recognize when it does not know. We build on two established empirical observations: higher confidence tends to correspond to higher accuracy within a single model, and models calibrated on a validation set can remain calibrated on a held-out in-distribution test set. These observations motivate the use of calibrated confidence as a routing and filtering signal. Building on this, we introduce two applications: (1) model cascading with calibrated advantage routing and (2) data cleaning based on a model ensemble. Using the routing signal derived from the comparability of calibrated confidences, we cascade large and small models to improve efficiency with almost no compromise in accuracy, and we further cascade two models of comparable scale to achieve performance beyond either model alone. Leveraging multiple experts and their calibrated confidences, we design a simple yet effective data-cleaning method that balances precision and detection rate to identify mislabeled samples in ImageNet and Massive Multitask Language Understanding (MMLU) datasets. Our results demonstrate that enabling models to recognize when they do not know is a practical step toward more efficient, reliable, and trustworthy AI.

1 INTRODUCTION

Modern deep neural networks have achieved remarkable success in both vision and language domains. However, models still frequently produce incorrect outputs [32] in regions of ignorance. Confidence, computed from intrinsic model signals, offers a promising path to estimate such ignorance. Prior work [9, 10, 17, 25, 28] has shown in specific domains that model calibration can yield reliable confidence estimates. In this work, we first introduce a unified framework for model calibration that applies to both vision and language models across diverse tasks.

We build on two empirical regularities established in prior work: (1) within a model, confidence functions defined from logits tend to be approximately monotonic with accuracy; and (2) confidence calibrated on a validation set can remain calibrated on a held-out in-distribution test set [8]. These properties motivate our use of calibrated confidence for cross-model comparison, routing, and data cleaning.

Building on calibrated confidence, we propose two applications that are broadly applicable to both vision and language models. The first application is model cascading. Prior work has introduced routers or gating modules to decide whether an input should be handled by a larger model, an external resource, or a human [3, 14, 24, 26, 29]. We propose an almost training-free cascading method: using calibrated confidence, we construct a reliable routing signal—confidence advantage—that determines whether to invoke a larger model. On tasks spanning image classification (ImageNet-1K [23]), code generation (Mostly Basic Python Problems (MBPP) [2], and BigCodeBench [33]), reasoning and mathematical tasks (ARC-Challenge [4] and GSM8K [5]), and knowledge-intensive

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

QA (MMLU) [13], we show that this method achieves an effective balance of accuracy and efficiency. We further extend cascading to large–large model pairs, where the combined cascade surpasses the performance of both standalone models. Notably, on ImageNet-1K, cascading two state-of-the-art models achieves accuracy beyond the previous state of the art.

The second application is data cleaning. Previous work (e.g., Confident Learning [18]) has demonstrated that confidence can signal mislabeled data: when a model is highly confident yet disagrees with the given label, the label is likely incorrect. We build upon this idea by using calibrated confidence as a reference signal. Since even well-calibrated individual models may still err, we introduce an ensemble approach for data cleaning, where calibrated confidence provides a tunable trade-off between precision and detection rate. This method is simple yet effective. We apply it to ImageNet-1K and MMLU test sets, and through large-scale manual verification of ImageNet labels, we show that it outperforms prior methods in both accuracy and detection power.

In summary, we present an almost training-free framework for calibration, cascading, and cleaning across vision and language tasks. Our contributions are threefold: (1) we characterize the condition under which calibrated confidence is comparable across models, introduce CrossECE, and derive an error bound relating confidence advantage to accuracy advantage; (2) we introduce calibrated confidence advantage as a routing signal for compute-efficient model cascading; and (3) we develop a label-issue detection method based on cross-model agreement and calibrated confidence, validated through large-scale human annotation. Together, these components provide a practical framework for confidence-based model coordination and data curation.

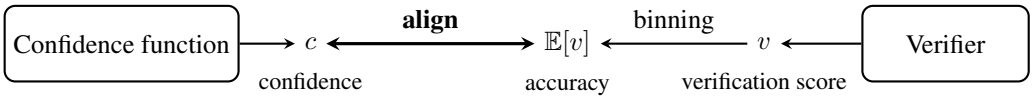
2 METHOD

This section introduces our formalization of model calibration, key empirical observations, and two downstream applications. We begin with a unified formulation applicable to both vision and language models, followed by two calibration methods used in our experiments: temperature scaling and Platt scaling [21]. We then present key observations—higher confidence corresponds to higher accuracy, and calibration learned on validation data generalizes to held-out test sets—that establish that calibrated confidence is both reliable and comparable across models. Building on these findings, we develop two applications: confidence-based model cascading to balance efficiency and accuracy or to combine strong models for improved performance, and agreement-driven data cleaning where confidence naturally balances precision and recall.

2.1 CALIBRATION OF VISION AND LANGUAGE MODELS

Formulation. Previous work has proposed calibration formulations tailored to specific cases, such as image classification models. Here we provide a unified formulation of model calibration that applies to both vision and language models, accommodates diverse downstream tasks, and supports different calibration methods.

In brief, calibration aligns the model's confidence c_M with the expected score $\mathbb{E}[v]$ from a verifier v, where the expectation is estimated by averaging within bins.



A calibration setup consists of the following components:

- **Model M takes input $X \in \mathcal{X}$ and generates internal states** $z = z_M(X, Y)$, used to sample a prediction $\hat{Y} \in \mathcal{Y}$ and to calculate confidence. For an image classification model, X is an input image, Y is a class, and z is a vector of logits over the label set $\mathcal{Y}$; for a language model, X is a prompt, Y is a possible answer sequence, and z comprises logits over the sequence and token spaces.
- **Verifier v assesses the correctness of an input–answer pair:** Specifically, a verifier v takes a pair (X, Y) as input and outputs a verification score $v(X, Y) \in [0, 1]$. For cases where correctness can be clearly determined, we consider a *binary verifier*, i.e., $v(X, Y) \in \{0, 1\}$ with $v(X, Y) = 0$ (invalid) and $v(X, Y) = 1$ (valid). For example, in classification tasks

(e.g., image classification and multiple-choice benchmarks such as MMLU) where each input X has a corresponding ground-truth label Y^*, the verifier is defined as $v(X, Y) = \mathbb{1}^{Y=Y^*}$. For code-generation benchmarks, the verifier checks whether the generated code executes correctly and produces the expected output. A verifier must extract task-relevant information from model outputs, typically in a simplified manner. For example, in MMLU, one may only check whether the first generated token matches the correct label. For code tasks, the verifier needs to extract executable code segments from the output and run them. On average, the binary verification score reduces to *accuracy*.

- **The confidence function $c(z) \in [0, 1]$ computes the confidence of an input–output pair from the model's internal states** $z = z_M(X, Y)$. For classification models, a common choice is $c = \mathrm{softmax}(z/T)$, a temperature-scaled softmax applied to logits, which yields likelihoods over all class labels. For language models, given an output sequence $Y_1Y_2 \dots Y_n$, the conditional joint likelihood $\ell(Y \mid X) = \prod_{i=1}^{n} \ell(Y_i \mid X, Y_1, \dots, Y_{i-1})$ computed via the softmax function can be used as confidence. Alternatively, one may let the language model self-report its confidence, which amounts to an additional sampling step from the model. As noted above, the verifier usually evaluates specific subsequences or restricted sub-token spaces. In such cases, the confidence function should also be constrained accordingly. For example, in MMLU only the first token (subsequence) and the four options $\{A, B, C, D\}$ (sub-token space) are relevant. Thus the confidence function should be restricted to this subset and renormalized to preserve its probabilistic interpretation. In what follows, we will also absorb z_M into c and use the shorthand $c_M(X, Y)$.
- **Binning $\{B_i\}_{i=1}^{N}$ is a coarse-graining of the data, over which expectations can be estimated.** Binning is essential with binary verifiers, since individual correctness is non-statistical. Accuracy must be considered as an average over equivalence classes (bins). Common binning methods include: **Confidence-based binning:** Partition the interval $[0, 1]$ into N equal intervals and assign each sample to a bin according to its confidence value. **Sample-based binning (i.e., histogram equalization):** Sort samples in ascending order of confidence and split them evenly into N bins $\{B_i\}_{i=1}^{N}$. This partitioning prevents samples from concentrating in a few confidence bins, thereby improving robustness. In this work, we adopt sample-based binning.

In general, confidence functions include learnable parameters. The process of optimizing these parameters on a validation set so that the confidence better reflects verifier scores is called **model calibration**. An *ideal calibration* means confidence perfectly reflects accuracy expectation, i.e.,

$$c_M(X, Y) = \mathbb{E}[v(X', Y') \mid (X', Y') \in B_i, \text{ where } (X, Y) \in B_i].$$

In practice, we approximate the expectation by the empirical average, and similarly average confidence within each bin, the ideal calibration condition becomes:

$$\frac{1}{|B_i|} \sum_{(X',Y') \in B_i} c_M(X', Y') = \frac{1}{|B_i|} \sum_{(X',Y') \in B_i} v(X', Y').$$

The left-hand side is the average confidence $\bar{c}_i$ of the bin B_i, and the right-hand side is the corresponding average accuracy $\bar{v}_i$. For an uncalibrated or imperfectly calibrated model, we measure the quality of its confidence estimates by the deviation between these two quantities. The expected calibration error (ECE) is defined as

$$\mathrm{ECE} = \sum_{i=1}^{N} \frac{|B_i|}{n} |\bar{c}_i - \bar{v}_i|,$$

where n denotes the total number of samples and $|B_i|$ denotes the number of samples in the i-th bin.

If the model adopts deterministic sampling such as greedy decoding (so that $\hat{Y}$ is uniquely determined by X) and we only care about the confidence of the final prediction, then the above dependence of confidence and accuracy on (X, Y) pairs can be simplified to depend solely on X. In what follows, we will also use the shorthand notation $c_M(X)$, which implicitly denotes $c_M(X, \hat{Y})$.

We employ two model calibration methods: *temperature scaling* (for vision tasks) and *Platt scaling* (for language tasks). Both methods involve a few calibration parameters trained on a validation set

that is held out from the training and test sets, and both are optimized using the negative log-likelihood (NLL) loss. A notable result is that, despite not directly optimizing ECE, this optimization objective tends to reduce the ECE of the calibrated model, as shown in Fig.1 and Tab.1.

Temperature Scaling. Temperature scaling introduces a single learned scalar parameter $T > 0$ applied to the logit vector z over classes to calculate confidence: $c = \text{softmax}(z/T)$. The temperature T is trained using the same objective as in model training, namely the NLL loss over the class labels. Temperature scaling can be used for calibrating multiclass classifiers.

Platt Scaling. Platt scaling [22] is a simple parametric calibration method tailored for binary classification. For a model output sequence $Y_1 \dots Y_n$, we obtain n logit vectors $z_1 \dots z_n$. Taking their values at the sampled tokens, $\hat{z}_i = z_i^{(Y_i)}$, we use their arithmetic mean $\bar{z} = \frac{1}{n}\sum_i \hat{z}_i$ as a logit estimate for the sequence. Platt scaling then applies a sigmoid function to compute confidence:

$$c = \sigma(a\bar{z} + b),$$

where $a, b \in \mathbb{R}$ are learned parameters and σ is the sigmoid function. These parameters are trained by fitting the confidence to the binary verifier's judgment (0 for invalid, 1 for valid) using the NLL loss. Since it is inherently designed for binary outcomes, Platt scaling is particularly suitable for program evaluation tasks, where the result can only be a pass or a failure.

We next present two key empirical observations that make calibrated confidences reliable:

- **Higher confidence implies higher accuracy.** For the confidence functions defined above, even without calibration, samples with higher confidence tend to have higher accuracy (see Fig.1). The calibration methods described above do not alter the confidence ranking of samples during training, so this approximate monotonic relationship between confidence and accuracy is essential for these methods to be ultimately calibrated.
- **Calibration trained on the validation set generalizes to the test set.** As shown in Fig.1, a confidence function calibrated on the validation set also remains calibrated on the test set. This indicates that confidence reflects the model's underlying estimation of the data distribution, rather than being specialized only to the training set. Hence, calibrated confidence can be expected to remain reliable on in-distribution new data.

The reliability of calibration is also reflected in the quantitative ECE (see Tab.1). Moreover, calibrated confidences across different models become comparable, which enables us to leverage confidence advantage between models to estimate their relative accuracy on specific problems.

2.2 Cascading with Calibrated Advantage Routing

Building on the comparability of confidences across models, we introduce the notion of *confidence advantage*. Consider two models M_1 and M_2, define the confidence advantage of M_1 compared to M_2 in the i-th bin B_i as $a_i^c = \bar{c}_i^1 - \bar{c}_i^2$, and the accuracy advantage as $a_i^v = \bar{v}_i^1 - \bar{v}_i^2$, where superscripts 1 and 2 denote the respective model indices.

A natural idea is to use confidence advantage as a routing signal for model cascading. Consider a large model M_l and a small model M_s. Both models are calibrated on the validation set, and histogram equalization is applied based on the confidence of the small model. Define the advantage function on the validation set as $a_i^c = \bar{c}_i^l - \bar{c}_i^s$. By the generalization property of the advantage function, for data x outside the validation set, if its confidence under the small model falls into the i-th bin, then its expected advantage under the large model will also be close to a_i^c. This calibrated advantage function reflects how much better M_1 performs than M_2 on bin B_i.

Based on this, we design the following cascading method: consider the bins $\{B_i\}_{i=1}^N$, sort them in ascending order of a_i^c, and select the top K bins, corresponding to the K bins where the large model shows the smallest advantage over the small model, where K is a hyperparameter balancing accuracy and efficiency. For a new input x, we first perform inference with the small model to obtain the output $\hat{Y}$ and confidence $c_s(x)$. If $c_s(x)$ falls into one of the selected K bins, this indicates that the large model has only a marginal advantage on x, and thus we adopt the small model's result. Otherwise, if $c_s(x)$ does not belong to these bins, we then invoke the large model for inference and use its output. The pseudocode of this algorithm is provided in Appendix Alg. 1.

While cascading between a large and a small model provides a balance between accuracy and efficiency, we can also cascade two large models to push the performance limit. Our experiments show that such large–large cascades can surpass the performance of either individual model.

2.3 Data Cleaning with Model Ensemble

Even when the verifier contains occasional errors (e.g., mislabeled data in the dataset), calibrated confidence still provides a useful signal about the accuracy of model outputs. This in turn allows us to use model confidence to correct erroneous labels. We demonstrate our approach on data cleaning for the ImageNet and MMLU datasets, which are known to contain mislabeled samples.

Basic Cleaning. The basic method is to treat cases where the model's top-1 prediction (the class with the highest confidence) disagrees with the given label as labeling errors. However, this ignores the model's own error probability, even if the model achieves high accuracy. Introducing multiple models as a mixture-of-experts can alleviate this issue. For example, consider two models M_1 and M_2. The most basic filtering rule is to check whether both M_1 and M_2 produce top-1 predictions inconsistent with the ground-truth label. If both disagree, the label is likely incorrect. This approach depends on the accuracy of individual models and the independence between experts: the more independent the experts, the better the performance.

Cleaning with Calibration. A more refined approach is to incorporate confidence, since models are highly accurate only on samples where they assign high calibrated confidence. A label is then flagged as erroneous when both models' top-1 predictions disagree with the ground-truth label **and** their corresponding confidences fall into the top-K confidence bins, where the hyperparameter K thus controls the trade-off between precision and recall in data cleaning.

Confident Learning. As a comparison, previous work has also used calibrated confidence for data cleaning, notably the Confident Learning (CL) [18] framework, in particular its Method 2. Let the given label of an input X be $Y_{\text{label}}(X)$. For each class Y, define its class-level average confidence as

$$\bar{c}(Y) = \frac{1}{|\{X' \mid Y_{\text{label}}(X') = Y\}|} \sum_{X': Y_{\text{label}}(X')=Y} c_M(X', Y),$$

where $c_M(X', Y)$ is the calibrated confidence of model M on class Y for input X'.

For a new sample (X, Y_{label}), consider the set of candidate classes

$$\mathcal{C}(X) = \{Y \mid c_M(X, Y) > \bar{c}(Y)\}.$$

Among these candidates, take the class $Y^* = \arg\max_{Y \in \mathcal{C}(X)} c_M(X, Y)$. If $Y^* \neq Y_{\text{label}}$, then the label is predicted to be incorrect; otherwise, it is retained as correct. Appendix Alg.2 presents pseudocode for the three methods described above.

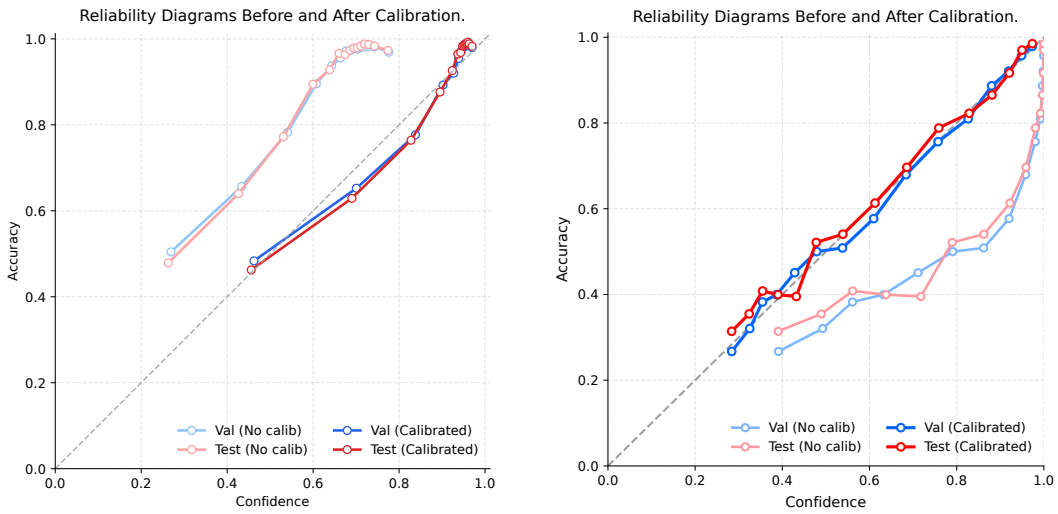
3 Experiments

In this section, we first show that calibration is reliable: calibrated confidence is monotonic in accuracy and generalizes from the calibration split to a held-out test split. Building on this, we evaluate cascades in both vision and language. In small–large cascades, our approach assigns to the small model the subset it can solve, achieving a favorable accuracy–small model ratio trade-off; in large–large pairs, our approach exploits complementary strengths to further improve accuracy. Finally, we use cross-model agreement to drive data cleaning and show that higher confidence consistently yields higher cleaning precision. We split the validation set in half, using one part for calibration and the other for testing, and report all results on the test split.

3.1 Calibration Can Be Reliable

We first validate the two key conclusions from Section 2.1—that higher confidence correlates with higher accuracy, and that calibration generalizes to the held-out test set. For consistency, we also adopt equal-count binning to partition predictions, reporting bin-wise accuracy and confidence for both validation and test sets.

Calibration on ImageNet Classification We first present the results on ImageNet using a single state-of-the-art model, `EVA02-Large-448`[6]. As shown in Figure 1a, the accuracy consistently



(a) ImageNet-1K (vision domain)

(b) MMLU (language domain)

Figure 1: **Reliability Diagrams Before and After Calibration.** Each panel plots accuracy vs. confidence across 15 equal-count bins. Light curves represent uncalibrated predictions; dark curves represent calibrated predictions. After calibration, confidence becomes more monotonic with accuracy and aligns more closely with the diagonal, indicating improved reliability and generalization from the calibration split to the held-out test split. Left: ImageNet model (`EVA02-Large-448`). Right: MMLU results from a `LLama-8B-Instruct`.

Table 1: **ECE Before and After Calibration on Validation and Test Sets.** Results are reported for both the vision domain (ImageNet) and the language domain (MMLU). Calibration substantially reduces ECE on the validation set, and the improvement generalizes to the held-out test set.

Dataset	Validation ECE (↓)		Test ECE (↓)	
	Before	After	Before	After
ImageNet	0.2617	**0.0271**	0.2624	**0.0296**
MMLU	0.1902	**0.0913**	0.1830	**0.0901**

increases with confidence across all bins, confirming the positive correlation between confidence and accuracy. Moreover, calibration performed on the validation set transfers well to the held-out test set, as evidenced by the close alignment between the ECE 1 on validation and test, demonstrating strong generalization of the calibration.

Calibration on Language Model Generation We perform the same evaluation procedure on large language models. Specifically, we use calibrated confidence scores from `LLaMA-8B-Instruct`[7] and validate them on the `MMLU` benchmark. As shown in Figure 1b, validation and test accuracies remain closely aligned across confidence bins, confirming that the calibrated confidence scores generalize well even in the language modeling setting.

3.2 Cascading Small and Large Models

In this section, we use the calibrated confidence advantage as a routing signal for combining small and large models. Experiments show that the raw confidence advantage already serves as an effective indicator for the portion of data the small model can reliably handle. Calibration further stabilizes this signal, leading to more consistent routing decisions and achieving a better accuracy–small-model ratio trade-off on several datasets.

We apply our proposed method to both image and language models. For evaluation, we consider the following baselines:

1. **Random Cascading (RC)** — Data instances are randomly assigned to the small model, and we report the *theoretical* accuracy computed as the average accuracy of the small model rather than running an explicit experiment.

2. **Confidence-Based Cascading (UC)** — Instances with the highest confidence advantage from the small model are selected without applying any post-hoc calibration. This corresponds to the first (non-parametric) stage of our method alone.
3. **Confidence-Based Cascading with Calibration Method (CC)** — Instances are selected based on the confidence advantage after applying a post-hoc temperature-scaling calibrator (hereafter referred to as ***t-scaling***). This corresponds to the full version of our proposed method. We further show that the performance of our approach is not sensitive to the specific choice of calibration method, and that similar cascading behavior can be achieved with alternative calibrators; detailed results are provided in Appendix A.3.
4. **Matrix Factorization Routing (MF)** — We additionally compare with the matrix–factorization-based router implemented in `RouteLLM` [19], a widely used state-of-the-art baseline for preference-driven routing. This method learns a low-dimensional scoring function to predict the relative quality between a small and a large model from pairwise preference data, and routes an input to the large model when the predicted win probability exceeds a learned threshold. Implementation and experimental settings of this baseline are detailed in Appendix D.

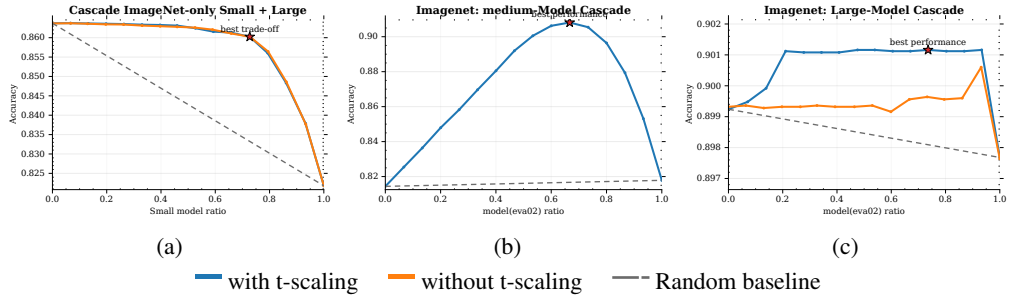


Figure 2: **Cascade Results on ImageNet-1K.** Each panel shows cascade accuracy vs. the routing ratio p. Panel (a) uses a small–large model pair, while panels (b) and (c) combine models of the same size, with panel (b) using a medium–medium pair and panel (c) using a large–large pair.

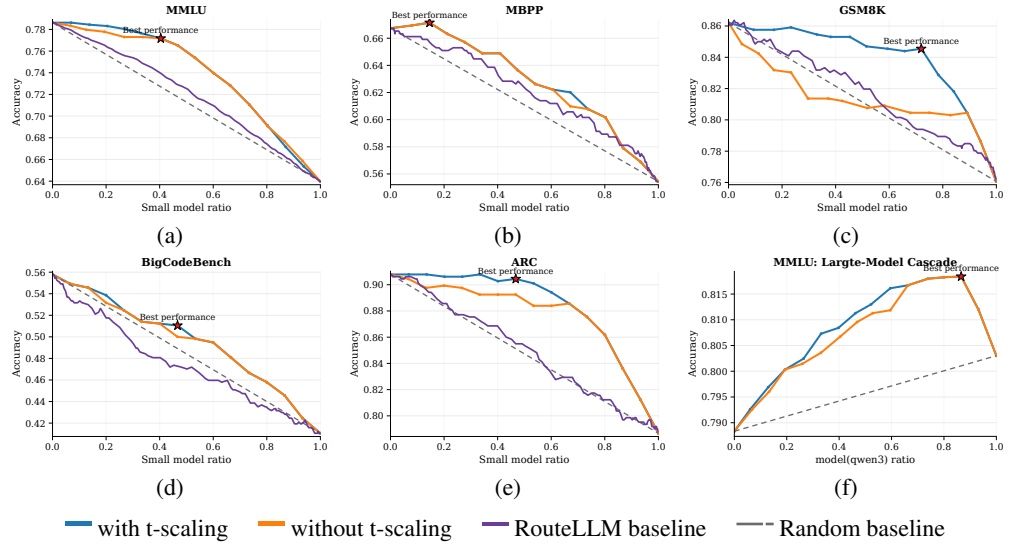


Figure 3: **Cascade Results on Language Models.** Panels (a)–(e) show small–large model cascades on five datasets. Panel (f) shows a large–large cascade, where p denotes the fraction of samples routed to the first large model.

Cascading Results on ImageNet In the ImageNet experiments, we evaluate a single group of ImageNet-only models, consisting of a small model and a large model trained exclusively on ImageNet. As shown in Figure 2, we plot the accuracy curves of UC, CC, and a random-pick

baseline as the proportion of small-model predictions increases. Because the small model on ImageNet is already well calibrated, UC and CC yield very similar performance. Nevertheless, both approaches consistently outperform random selection under the same proportion of small-model usage, demonstrating the effectiveness and robustness of the method on image classification models.

Cascading Results on Language Models In the language model experiments, we select five diverse and widely adopted datasets: `MBPP`, `BigCodeBench`, `MMLU`, `GSM8K`, and `ARC-Challenge`. We use `LLaMA-70B` as the large model and `LLaMA-8B` as the small model. As shown in Figure 3, CC consistently outperforms matrix factorization routing and the random-pick baseline across all datasets. Moreover, CC consistently outperforms UC across the entire accuracy–small-model-ratio trade-off curve across all evaluated datasets, providing strong evidence that calibrated confidence leads to more effective routing in language modeling.

3.3 Cascading Same-Size Models

In this section, we extend our approach to pairs of same-size models. We further show that using calibrated confidence as the routing signal enables two models with comparable overall capacity to complement each other by specializing on the data where each performs best, thereby improving overall accuracy. As illustrated in Figures 2b and 2c, when we cascade two medium-sized models (DeiT-B [27] and TinyViT [30]), each achieving around 82% accuracy individually, As illustrated in Figures 2b and 2c, when we cascade two medium-sized models—each achieving around 82% accuracy individually—the combined system reaches an astonishing peak accuracy of up to 90%. Moreover, cascading two state-of-the-art ImageNet models in this manner consistently achieves higher accuracy than either model alone.

Similarly, on the `MMLU` dataset, pairing `Qwen-32B` with `LLaMA-70B` yields further performance gains, as shown in Figure 3(f), demonstrating that calibrated confidence not only guides efficient routing but also facilitates effective collaboration between high-capacity models.

Table 2: Cascade accuracy (%) of two EVA-02-Large models on the original and Cleaned half ImageNet validation sets. Model 1: `eva02_large_patch14_448.mim_m38m_ft_in22k_in1k`; Model 2: `eva02_large_patch14_448.mim_in22k_ft_in22k_in1k`.

Model Combination	Original ImageNet	Cleaned ImageNet
EVA02-L (M38M) + EVA02-L (IN22k)	90.116	96.266

3.4 Data Cleaning

In this section, we demonstrate the effectiveness of leveraging model prediction agreement for data cleaning on both image and language models. We further show that cleaning precision improves as model confidence increases, highlighting a strong positive correlation between confidence and the reliability of the cleaned data.

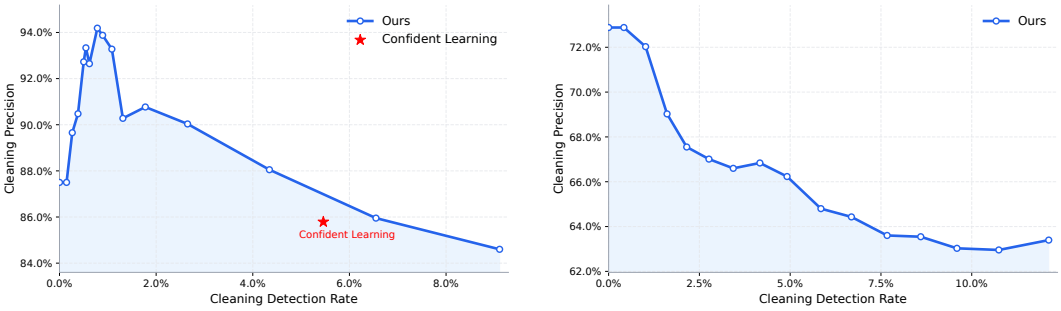
ImageNet Cleaning We use two high-performing ImageNet models (ranked 3rd and 4th[1] as of the time of writing) as complementary experts to flag potential label issues. This avoids using the same models for both cleaning and evaluation, ensuring that the subsequent accuracy measurement remains independent. We compare our method against `CL`, a widely used label-cleaning approach. To assess quality, we randomly sampled 1,000 flagged images for manual verification by trained annotators, which serves as ground truth for estimating precision and detection rate. As shown in Table 3, our method recovers most mislabeled samples found by `CL` and additionally discovers a much larger set of label errors with high verified precision, demonstrating its stronger ability to improve dataset integrity. We categorize errors into *Wrong Label*, *Multi-Object*, *Similar Class*, *Subclass*, and *Superclass*, with definitions and examples in Appendix C.

Table 3: Comparison between our cleaning method and `CL` on ImageNet. We report the number of samples flagged by each method, the overlap between them, and the accuracy of samples uniquely flagged by each approach.

[1] `https://github.com/huggingface/pytorch-image-models/blob/main/results/results-imagenet.csv`

Method	Overlap	Flagged Samples	Unique Accuracy (%)
Ours	1283	994	0.665
Confident Learning		82	0.537

Furthermore, to validate the conclusion of Section 2.3 — that the accuracy of identified mislabeled samples decreases as confidence decreases — we present the results in Figure 4a. As the detection rate increases (by progressively lowering the confidence threshold), the cleaning precision consistently declines, corroborating our claim. Moreover, across the entire detection range, our method achieves higher cleaning precision than `CL` at the same detection rate, demonstrating its superior precision.



(a) ImageNet-1K (b) MMLU

Figure 4: **Cleaning Precision vs. Detection Rate.** The horizontal axis (**Cleaning Detection Rate**) is defined as the ratio of flagged samples to the total dataset size. The vertical axis (**Cleaning Precision**) measures the fraction of truly mislabeled samples among the flagged ones, estimated using human verification. Samples are sorted by calibrated confidence in descending order and grouped into 15 equal-count bins; progressively lowering the confidence threshold increases detection rate but typically decreases precision. The red star in panel (a) denotes the operating point of `CL`.

Finally, after completing the data-cleaning process, we apply our cascade method to combine the two state-of-the-art ImageNet models (ranked first and second). As shown in Table 2, the combined model achieves an impressive accuracy of 96.266%, substantially surpassing the performance of either individual model. This result suggests that, when combined with proper data cleaning, the current state-of-the-art models already approach the upper bound of ImageNet performance, indicating that ImageNet has become a nearly saturated benchmark.

Language Data Cleaning. Analogous to the vision setting, we use `Qwen-32B` [31] and `LLaMA-70B` as expert models to identify potentially mislabeled samples in language benchmarks including `MMLU`, `ARC`, and `MBPP`. Owing to the high cost of large-scale manual verification, we use responses generated by `GPT-4o` [1] in thinking mode as pseudo-labels for estimating precision and detection rate.

Applying the same confidence-based filtering procedure as in the vision experiments, we partition flagged samples into 15 equal-count bins and observe a consistent monotonic trend across all datasets: precision decreases as bin confidence decreases. These results suggest that calibrated confidence provides a robust signal for balancing precision and detection rate in language-model settings.

4 Limitations and Conclusion

Limitations. Our framework assumes usable confidence and correctness signals, which are most natural in classification and multiple-choice tasks. Extending it to free-form generation remains challenging because both confidence estimation and automatic verification can be unreliable. Recent confidence estimators and LLM-based evaluators offer promising directions [11, 15, 20].

Conclusion. We propose a simple, multimodal, and almost training-free framework that leverages calibrated confidence for model cascading and data cleaning. Our results show that calibrated confidence serves as a practical signal for identifying when a model does not know, enabling favorable accuracy–efficiency trade-offs and improved data quality. Overall, our work highlights the potential of transforming model self-awareness into actionable signals for building more efficient and reliable AI systems.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, presentation, reference formatting, and LaTeX preparation. The authors reviewed and revised all AI-assisted material and take full responsibility for the final content, claims, and artifacts in this work.

REFERENCES

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

[2] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[3] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 527–536. PMLR, 2017.

[4] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

[6] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 149:105171, 2024.

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

[8] Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. Predicting with confidence on unseen distributions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1134–1144, 2021.

[9] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 1321–1330. PMLR, 2017.

[10] Chirag Gupta and Aaditya Ramdas. Distribution-free calibration guarantees for histogram binning without sample splitting. *ArXiv*, abs/2105.04656, 2021. URL `https://api.semanticscholar.org/CorpusID:234357974`.

[11] Jinyi Han, Tingyun Li, Shisong Chen, Jie Shi, Xinyi Wang, Guanglei Yue, Jiaqing Liang, Xin Lin, Liqian Wen, Zulong Chen, and Yanghua Xiao. Mind the generation process: Fine-grained confidence estimation during llm generation, 2025. URL `https://arxiv.org/abs/2508.12040`.

[12] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[13] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[14] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations (ICLR)*, 2021.

[15] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL `https://aclanthology.org/2023.emnlp-main.153/`.

[16] Francesco Maria Molfese, Luca Moroni, Luca Gioffrè, Alessandro Scirè, Simone Conia, and Roberto Navigli. Right answer, wrong score: Uncovering the inconsistencies of llm evaluation in multiple-choice question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

[17] Jeremy Nixon, Michael Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. *arXiv preprint arXiv:1904.01685*, 2019.

[18] Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. Confident learning: Estimating uncertainty in dataset labels, 2022. URL https://arxiv.org/abs/1911.00068.

[19] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

[20] Tejaswini Pedapati, Amit Dhurandhar, Soumya Ghosh, Soham Dan, and Prasanna Sattigeri. Large language model confidence estimation via black-box access. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=WrWYChkyRI.

[21] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. 1999. URL https://api.semanticscholar.org/CorpusID:56563878.

[22] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pp. 61–74. MIT Press, 1999.

[23] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[24] Noam Shazeer, Azalia Mirhoseini, Piotr Maziarz, Andy Davis, Quoc V. Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017.

[25] Claudio Spiess, David Gros, Kunal Suresh Pai, Michael Pradel, Md Rafiqul Islam Rabin, Amin Alipour, Susmit Jha, Prem Devanbu, and Toufique Ahmed. Calibration and correctness of language models for code. *arXiv preprint arXiv:2402.02047*, 2024.

[26] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

[28] Deng-Bao Wang, Lei Feng, and Min-Ling Zhang. Rethinking calibration of deep neural networks: Do not be afraid of overconfidence. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=NJS8kp15zzH.

[29] Xiaofang Wang, Dan Kondratyuk, Eric Christiansen, Kris M Kitani, Yair Alon, and Elad Eban. Wisdom of committees: An overlooked approach to faster and more accurate models. *arXiv preprint arXiv:2012.01988*, 2020.

[30] Kan Wu, Jinnian Zhang, Houwen Peng, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. Tinyvit: Fast pretraining distillation for small vision transformers. In *European conference on computer vision*, pp. 68–85. Springer, 2022.

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

[32] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, pp. 1–46, 2025.

[33] Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, et al. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. *arXiv preprint arXiv:2406.15877*, 2024.

Appendix

A Additional Analyses of Model Cascading

A.1 Average Performance Gap Recovered (APGR)

We summarize routing curves using the Average Performance Gap Recovered (APGR). Let $q \in [0, 1]$ denote the fraction of examples delegated from the weaker to the stronger model, and let $P(q)$ denote the resulting performance. For $P_{\text{strong}} > P_{\text{weak}}$, define

$$\widetilde{P}(q) = \frac{P(q) - P_{\text{weak}}}{P_{\text{strong}} - P_{\text{weak}}}, \qquad \text{APGR} = \int_0^1 \widetilde{P}(q)\, dq. \tag{1}$$

Random delegation has expected performance $(1 - q)P_{\text{weak}} + qP_{\text{strong}}$, giving a theoretical APGR of 0.5. This reference is not a measured baseline or a lower bound. APGR can exceed one when routing exploits complementary errors and exceeds the stronger model's accuracy. It is undefined when the standalone accuracies are equal and can be sensitive to small accuracy gaps. The delegation fraction is not wall-clock time: a sequential cascade also incurs the first model's inference cost. We distinguish APGR from the explicit cost proxy in Eq. 6.

A.2 Out-of-Distribution (OOD) Evaluation

To evaluate the robustness of our cascading method under out-of-distribution (OOD) conditions, we select MMLU-Adversarial [16] as the OOD dataset for language models. For vision models, we choose three representative corruption types from ImageNet-C — Gaussian Noise, Motion Blur, and Fog — as OOD benchmarks, and evaluate the performance of the proposed cascading method on these datasets.

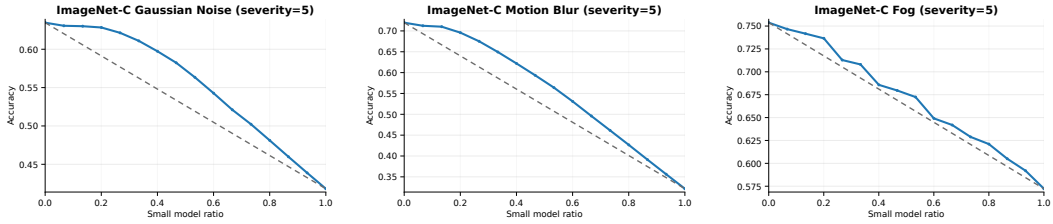


Figure 5: OOD evaluation on ImageNet-C [12] for the three evaluated corruption types at severity level 5. Results for severity levels 1 and 3 are provided in the appendix.

Figures 5 and 6 show results under specific distribution shifts with calibration fitted on the source domain. On ImageNet-C at severity 5, APGR is 0.6364 for Gaussian noise, 0.5999 for motion blur, and 0.5409 for fog. The fog result is close to the theoretical random reference of 0.500 and indicates substantial degradation. In the source-transfer language experiment, APGR decreases from 0.7013 on MMLU to 0.6301 on MMLU-Adversarial. These measurements support useful routing in some evaluated shifts, not general robustness to arbitrary OOD inputs.

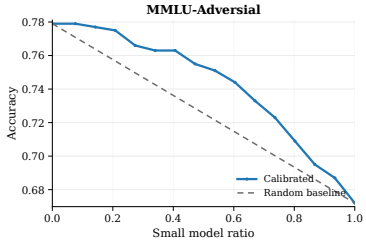


Figure 6: OOD Evaluation on MMLU-Adversarial

A.3 Effect of Calibration Methods on Model Cascading

We evaluate several calibration methods. As shown in Table 4, the cascading performance remains largely consistent across different calibration method choices, indicating that CC method is not sensitive to the specific calibration technique used.

Table 4: APGR of model cascading with different Post-hoc calibration methods.

APGR / Dataset	MMLU	MBPP	GSM8K
Histogram Binning	0.7006	0.6609	0.7489
Temperature Scaling	0.7013	0.6772	0.7593
Platt Scaling	0.7013	0.6897	0.7496

A.4 Sensitivity to the Number of Bins

Additional experiments are conducted to analyze how the number of bins affects cascading performance. As shown in Table 5, the method remains stable across a wide range of bin choices.

Table 5: Effect of the number of bins on model cascading (APGR).

Dataset	#Bins = 5	#Bins = 15	#Bins = 50	#Bins = 100
MMLU	0.6917	0.7013	0.7044	0.7035
GSM8K	0.7378	0.7496	0.7547	0.7551

A.5 Additional OOD Results on ImageNet-C (Severity Levels 1 and 3)

Table 6 reports the source-calibrated system across three corruption severities. Performance generally decreases with severity; for fog at severity 5, APGR approaches the theoretical random reference of 0.5. Target-domain recalibration is a separate protocol, reported in Appendix H.3.

Table 6: APGR of Model Cascading on OOD Image Datasets.

Severity	Gaussian Noise	Motion Blur	Fog
Original ImageNet	0.7984	–	–
1	0.8158	0.8014	0.7675
3	0.7588	0.7267	0.7220
5	0.6364	0.5999	0.5409

B Annotation and Verification Procedure

B.1 ImageNet Cleaning Verification Guidelines

Guidelines for Label-Cleaning Validation (Full Instructions)

Columns in the Excel Sheet

- Image Index
- Image
- Model 1 Prediction
- Model 2 Prediction
- Ground-Truth (GT) Label
- Your Answer

Important Note. All provided samples are cases where the predictions from Model 1 and/or Model 2 do *not* match the ground-truth (GT) label. However, the predictions from Model 1 and Model 2 may be identical to each other.

Your Task. Determine the correctness of the GT label by referencing the *Image*, the *Model 1 & 2 Predictions*, and the *GT Label*. Based on your evaluation, enter an integer **1–4** in the "Your Answer" cell for that row.

Classification Categories

1. **(GT wrong):** The image does not contain the object described by the GT label.
2. **(GT and model correct):** The object in the GT label *and* the object in at least one of the model predictions (Model 1 or 2) are present in the image.
3. **(Only GT correct):** The image *only* contains the object described by the GT label (and does not contain the objects from either of the model predictions).
4. **Unable to determine.**

Supplementary Instructions
Tools. You may use LLMs (e.g., Gemini, ChatGPT) and search engines to assist your judgment. When using a search engine, adding the keyword "ImageNet" can refine your search. This website can also provide other images from a specific category, e.g., `https://salient-imagenet.cs.umd.edu/explore/`. However, you should not accept an LLM's answer directly. For challenging cases—such as distinguishing between two similar biological species—you must use these tools to learn about the key features and then make your own informed decision.
Focus. Your primary goal is to assess the correctness of the GT label. However, it is essential to consider the model predictions, as comparing the different labels is often critical for making an accurate judgment.
Clarification for Category 1 (GT wrong). If the image clearly does not contain the object specified by the GT label, you should select Category 1 immediately. You do not need to evaluate whether the model predictions are present in the image.
Examples for Category 2 (GT and model correct).

- **Multiple Objects:** The image contains several objects. The model prediction(s) correctly identify one object, while the GT label correctly identifies a different one.
- **Hierarchical/Aspectual Labels:** The model prediction(s) and the GT label describe the image at different levels of specificity or from different perspectives, but both are technically correct (e.g., "off-road vehicle" vs. "four-wheeler"; "laptop" vs. "notebook").

Handling Lack of Information.

- If you can clearly determine that the image contains the object from a model prediction, but you lack sufficient information to judge the object in the GT label, choose **Category 1 (GT wrong)**.
- If you can clearly determine that the image contains the object from the GT label, but you lack sufficient information to judge the object(s) in the model prediction(s), choose **Category 3 (Only GT correct)**.
- Choose **Category 4 (Unable to determine)** only when there is insufficient information to determine the presence of *both* the GT label's object and the model predictions' objects.

Handling Identical Label Names (Rare Case). In the very rare situation where the GT label and a model prediction are the same name but refer to different actual categories, please select **Category 1 (GT wrong)**.

B.2 MMLU Cleaning Verification Guidelines

To verify the correctness of samples in the Confmat_{00} set (where both models fail), we reconstruct the original MMLU questions and prompt a GPT model as an independent judge. Each question is presented with its four multiple-choice options, and the model is instructed to answer with *exactly one capital letter (A, B, C, or D)* without explanation. We then compare the model's response with the ground-truth label to estimate the true accuracy of samples in this error region. Temperature is set to 0.0 to minimize randomness.

C Representative Examples of ImageNet Label Errors

(1) Wrong Label: the annotated label does not correspond to any object in the image (e.g., a digital watch labeled as "stopwatch").



Label: stopwatch
True: digital watch



Label: projectile
True: warplane



Label: water jug
True: water bottle

(2) Multi-Object: images containing multiple objects where only one was labeled, leading to incomplete annotations (e.g., "phone" without lamp, "screen" without desk, or "shovel" without the tent).



Label: phone
True: phone + lamp



Label: screen
True: screen + desk



Label: shovel
True: shovel + mountain tent

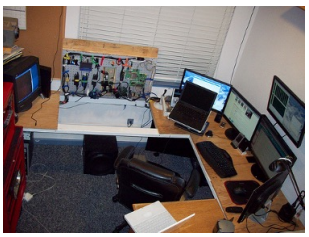
(3) Similar Class: cases where the annotated label names a visually similar class.



Label: desktop
Similar: notebook

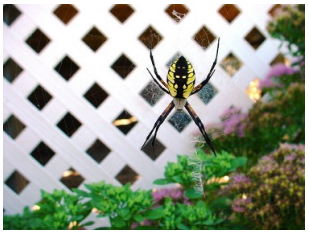


Label: maillot
Similar: maillot



Label: screen
Similar: monitor

Super/Sub Class: errors where the annotated label is at a different granularity level from the ground truth. This includes cases where the label is too broad (super-class) or too specific (sub-class) compared to the true category, as well as sibling-class mismatches within the same hierarchy.



Label: garden spider
Sub: black and gold garden spider



Label: mushroom
Sub: agaric



Label: water jug
True: water bottle

D Implementation Details of the Matrix Factorization Router

In this section, we describe the implementation and experimental settings for the Matrix Factorization (MF) routing baseline adopted in our comparisons.

Training settings. We implement the MF router in three ways:

1. **Pretrained:** We directly use the pretrained router checkpoint provided by the authors of RouteLLM.
2. **Fine-tuned:** We fine-tune the pretrained router on the target task dataset using the same loss and optimization settings as in the original work.
3. **From scratch:** We train the router from scratch on the target task dataset using randomly initialized parameters.

In Figure 3, for each dataset we select the implementation that achieves the highest APGR as the baseline for comparison with other methods.

Table 7: APGR of different MF router implementations across datasets.

Implementation	MMLU	GSM8K	MBPP	ARC	Bigcodebench
Pretrained	0.5418	0.5672	0.5500	0.5102	0.4320
Fine-tuned	0.5497	0.5718	0.5479	0.5114	0.4310
From scratch	0.5267	0.6234	0.5909	0.4933	0.4185

E Algorithm Pseudocode

F Conditional Calibration and Cross-Model Comparability

Ordinary within-model calibration does not imply that two models are calibrated on the same subsets of inputs. This distinction matters for cascading: the routing bins are determined by the first model, whereas the second model's confidence and accuracy are evaluated on those same bins. We therefore examine calibration conditional on the routing partition rather than infer comparability from the two models' marginal ECE values.

Algorithm 1: Model Cascading with Calibrated Advantage Routing

Input: Large model M_l, small model M_s; validation set $\mathcal{V}$; number of bins N; selection budget K ($1 \le K \le N$).
Output: A router that decides whether to use M_s or M_l for a new input x.
Phase I: Calibration and bin statistics on $\mathcal{V}$.
Calibrate M_l and M_s on a held-out validation set $\mathcal{V}$ to obtain calibrated confidence functions $c_l(\cdot)$ and $c_s(\cdot)$.
foreach $(X, Y) \in \mathcal{V}$ **do**
  Compute the small-model confidence $c_s(X)$ (or $c_s(X, \hat{Y})$ if label-specific).
Apply histogram equalization on $\{c_s(X) : (X, Y) \in \mathcal{V}\}$ to form N bins $\{b_i\}_{i=1}^N$ of (approximately) equal size.
for $i \leftarrow 1$ **to** N **do**
  Compute $\bar{c}_i^s := \frac{1}{|b_i|} \sum_{(X,Y)\in b_i} c_s(X)$.
  Compute $\bar{c}_i^l := \frac{1}{|b_i|} \sum_{(X,Y)\in b_i} c_l(X)$.
  Compute the confidence advantage $a_i^c := \bar{c}_i^l - \bar{c}_i^s$.
Sort bins $\{b_1, \ldots, b_N\}$ in ascending order of a_i^c and let $\mathcal{S} \subset \{b_1, \ldots, b_N\}$ be the first K bins (smallest advantages).
Phase II: Routing on a new input x.
Obtain the small-model prediction $\hat{Y}_s(x)$ and confidence $c_s(x)$.
Find the bin index i such that $c_s(x) \in b_i$.
if $b_i \in \mathcal{S}$ **then**
  return use M_s: output $\hat{Y}_s(x)$.
else
  Run the large model to obtain $\hat{Y}_l(x)$; **return** use M_l: output $\hat{Y}_l(x)$.

F.1 A BIN-WISE IDENTITY FOR THE ROUTING SIGNAL

Let B_i be a bin induced by the first model's confidence, let $w_i = |B_i|/n$, and let $v_m(x) \in [0, 1]$ be the verification score of model $m \in \{S, L\}$. Define, on the *same* bins,

$$\bar{c}_{m,i} = \frac{1}{|B_i|} \sum_{x\in B_i} c_m(x), \qquad \bar{v}_{m,i} = \frac{1}{|B_i|} \sum_{x\in B_i} v_m(x), \qquad r_{m,i} = \bar{v}_{m,i} - \bar{c}_{m,i}. \tag{2}$$

The empirical accuracy advantage $a_i^v = \bar{v}_{L,i} - \bar{v}_{S,i}$ and confidence advantage $a_i^c = \bar{c}_{L,i} - \bar{c}_{S,i}$ satisfy the exact identity

$$a_i^v - a_i^c = r_{L,i} - r_{S,i}. \tag{3}$$

Define the cross-model calibration error by

$$\mathrm{CrossECE}(L \mid S) = \sum_i w_i |r_{L,i}|, \qquad \mathrm{ECE}_S = \sum_i w_i |r_{S,i}|. \tag{4}$$

The triangle inequality immediately gives

$$\sum_i w_i |a_i^v - a_i^c| \le \mathrm{CrossECE}(L \mid S) + \mathrm{ECE}_S. \tag{5}$$

Scope of the result. If both residuals vanish in every nonempty routing bin, the confidence advantage equals the accuracy advantage in each bin. More generally, the bound controls their *average* discrepancy; it does not certify every bin or guarantee a correct ordering of bins with nearly equal advantages. For example, a pairwise ordering is preserved when the true advantage gap exceeds the sum of the two corresponding advantage-estimation errors. These are empirical identities on a fixed evaluation partition, not a guarantee that a partition learned on calibration data transfers to a shifted test distribution. Nor do they establish global optimality over arbitrary input-dependent routers or a guarantee for label cleaning. Temperature and Platt scaling minimize a calibration fitting objective, not ECE directly, so their ECE improvements remain empirical.

Algorithm 2: Data Cleaning with Calibrated Confidence

Input: Noisy dataset $\mathcal{D}$, calibrated models $M_1, M_2, \ldots, M_m$, number of bins N, number of selected confidence bins K ($1 \le K \le N$).
Output: A set of potentially erroneous labels for review.

Method 1: Top-1 Disagreement.

foreach $(X, Y_{label}) \in \mathcal{D}$ **do**
  foreach *model* $M_j,\ j = 1, \ldots, m$ **do**
    Compute $Y_j^*(X) = \arg\max_Y c_{M_j}(X, Y)$.
  if $Y_j^*(X) \neq Y_{label}$ *for all* j **then**
    Flag Y_{label} as potentially erroneous.

Method 2: High-Confidence Disagreement.

On the validation set, for each model M_j, apply histogram equalization to its confidences $\{c_{M_j}(X)\}$ to form N bins $\{b_i^j\}_{i=1}^N$. Let $\mathcal{B}^j_{\text{top-}K}$ denote the K bins (for model M_j) with the highest mean calibrated confidence.

foreach $(X, Y_{label}) \in \mathcal{D}$ **do**
  foreach *model* $M_j,\ j = 1, \ldots, m$ **do**
    Compute

$$Y_j^*(X) = \arg\max_Y c_{M_j}(X, Y), \quad c_{M_j}(X) = \max_Y c_{M_j}(X, Y).$$

  if $Y_j^*(X) \neq Y_{label}$ ***and*** $c_{M_j}(X) \in \mathcal{B}^j_{top\text{-}K}$ *for all* j **then**
    Flag Y_{label} as potentially erroneous.

Method 3: CLEANLAB-inspired Averaging.

For each class Y, compute

$$\bar{c}(Y) = \frac{1}{|\{X' : Y_{\text{label}}(X') = Y\}|} \sum_{X' : Y_{\text{label}}(X') = Y} c_M(X', Y).$$

foreach $(X, Y_{label}) \in \mathcal{D}$ **do**
  Define $\mathcal{C}(X) = \{Y : c_M(X, Y) > \bar{c}(Y)\}$.
  Let $Y^*(X) = \arg\max_{Y \in \mathcal{C}(X)} c_M(X, Y)$.
  if $Y^*(X) \neq Y_{label}$ **then**
    Flag Y_{label} as potentially erroneous.

Table 8: Calibration on routing-relevant slices. CrossECE evaluates Model L on bins induced by Model S. ECE-L uses Model L's own confidence bins.

Dataset	ECE-S	ECE-L	CrossECE ($L \mid S$)
ImageNet-1K	0.0380	0.0303	0.0303
ImageNet-C, Fog 5	0.3144	0.0337	0.0348
MMLU	0.0104	0.0223	0.0188
MMLU-Adversarial	0.0445	0.0344	0.0170
MBPP	0.0713	0.0451	0.0272

F.2 Cross-model calibration diagnostics

The vision pair is CoaT-S / EVA02-L-448; the language pair is LLaMA-3-8B / LLaMA-3-70B. The language diagnostics use Platt scaling. CrossECE is between 0.0170 and 0.0348 in these evaluations, supporting the use of the large model's confidence on the tested small-model slices. However, low CrossECE alone is insufficient: for Fog at severity 5, ECE-S is 0.3144, and the bound in Eq. 5 is 0.3492. This explains why the diagnostic cannot support an unqualified claim of reliable routing under severe shift. Per-bin residuals and bin counts are needed to assess individual routing decisions; aggregate ECE values do not supply that evidence.

G Additional Cascading Evaluations

G.1 ImageNet-Sketch: isolating the effect of calibration

We compare routing based on confidence before post-hoc calibration (UC) with calibrated-confidence routing (CC), using the same splits and routing protocol within each comparison. The theoretical random-routing reference has APGR 0.500; it is not an empirical lower bound, and a poor router can perform worse. The model pairs are CoaT-S / EVA02-L-448, TinyViT-11M / DeiT-B, and EVA02-

Table 9: Model accuracies and best observed cascade accuracy on ImageNet-Sketch. All accuracies are percentages. Best observed accuracy summarizes the evaluated curve and is not a guarantee for a deployment operating point.

Configuration	Model A	Model B	Best cascade
Small–large	35.95	67.20	67.20
Medium–medium	31.74	32.55	33.27
Large–large	67.20	70.67	71.25

L-448-IN22K / EVA02-L-448-M38M, respectively. In the small–large setting, the stronger model achieves 67.20%, providing an evaluation outside the near-saturated ImageNet regime. CC improves

Table 10: Calibration ablation on ImageNet-Sketch. ECE refers to Model A; random routing has theoretical APGR 0.500 in every row.

Configuration	ECE before	ECE after	UC APGR	CC APGR
Small–large	0.1987	0.0781	0.560	0.649
Medium–medium	0.1343	0.0445	1.109	1.259
Large–large	0.1318	0.0830	0.651	0.743

APGR over UC by 0.089, 0.150, and 0.092 in these three comparisons. APGR can exceed one when a cascade outperforms the stronger standalone model over part of the curve. Its normalization is sensitive to the standalone accuracy gap, especially for nearly matched models; APGR values should therefore be interpreted within a model pair rather than as directly comparable absolute gains across pairs.

G.2 Full-compute ensemble references

The plain-ensemble and maximum-confidence selection baselines evaluate both models on every input and use their uncalibrated outputs. CC first evaluates Model A and invokes Model B only on a fraction q of inputs. For per-input costs C_A and C_B, its relative cost is

$$R(q) = \frac{C_A + qC_B}{C_A + C_B}. \tag{6}$$

The tabulated costs use equal model costs for the two EVA02 variants and parameter-count-proportional costs (70B and 32B) for the language pair. They are inference-cost proxies, not measured latency or energy savings; sequence length, hardware, and batching can change the realized costs. The primary result is comparable observed accuracy at lower estimated cost: CC uses 53.35% and 79.11% of full two-model computation in the two settings. The accuracy differences relative to the plain ensemble are only 0.06 and 0.10 percentage points. Without uncertainty estimates or repeated-run evidence, these small differences do not establish statistically significant accuracy superiority. These comparisons isolate the efficiency benefit relative to full-compute ensembles; they do not imply that CC always uses less computation than either standalone model.

G.3 Additional language benchmarks and model pairs

G.4 Extension to visual question answering

We additionally apply CC to publicly available ViLT-VQA and BLIP-VQA models. Calibration and routing are determined on VQA-v2 and transferred to AdVQA without using AdVQA labels. The reported APGR is 3.324 on VQA-v2 and 0.612 on AdVQA, against the theoretical random reference

Table 11: Accuracy and computation relative to running both models. Call rates and relative costs are percentages. Model-B-only rows do not also run Model A.

Method	Accuracy (%)	B call rate	Relative cost
ImageNet-1K: EVA02-IN22K / EVA02-M38M			
Model A only	89.75	0.00	50.00
Model B only	89.92	100.00	50.00
Plain ensemble	90.05	100.00	100.00
Maximum-confidence selection	90.03	100.00	100.00
CC cascade	90.11	6.70	53.35
MMLU: LLaMA-3-70B / Qwen3-32B			
Model A only	78.03	0.00	68.63
Model B only	81.07	100.00	31.37
Plain ensemble	81.41	100.00	100.00
Maximum-confidence selection	81.24	100.00	100.00
CC cascade	81.51	33.40	79.11

Table 12: Cascading in language settings where the stronger model's accuracy is below 70%. The pair is LLaMA-3-8B / LLaMA-3-70B. Random APGR is the theoretical reference.

Dataset	Small acc. (%)	Large acc. (%)	Random	CC APGR
BigCodeBench	38.77	55.09	0.500	0.570
MBPP	55.65	67.76	0.500	0.686

Table 13: APGR for same-family and different-family language-model pairs in the model-pair ablation. These values characterize each pair and do not isolate model diversity from changes in model size, accuracy, or pretraining.

Model pair	MMLU	GSM8K	MBPP	ARC
Qwen-4B / LLaMA-70B	0.7610	0.9697	0.9561	0.8896
LLaMA-8B / LLaMA-70B	0.7013	0.7496	0.6897	0.7808

of 0.500. The value above one reflects the normalization discussed in Appendix G.1; it is not an accuracy or a multiplicative speedup. These results provide an additional application and a specific transfer experiment. They do not establish robustness to arbitrary distribution shifts or a controlled advantage over VQA systems using different underlying models.

H Calibration Ablations and Their Limits

H.1 Order-preserving confidence concentration

Strictly increasing transformations compress ImageNet-Sketch confidence scores into either $[0.0011, 0.0097]$ or $[0.9900, 0.9996]$, preserving their ordering. The reported calibrators are fitted on a disjoint calibration split. Equal-count histogram binning depends on rank partitions and estimates bin-level correctness from labels; it must not be confused with merely grouping examples into bins, which by itself does not calibrate a score. Histogram binning preserves its reported result

Table 14: Calibration and routing under confidence concentration. TS: temperature scaling; HB: histogram binning. ECE entries show before/after values. Random routing has theoretical APGR 0.500.

Regime	Method	ECE-S	ECE-L	APGR
Original	TS	0.1987 / 0.0779	0.1318 / 0.0831	0.649
Original	HB	0.1987 / 0.0237	0.1318 / 0.0235	0.644
Near 0	TS	0.3550 / 0.1115	0.6678 / 0.1469	0.627
Near 0	HB	0.3550 / 0.0237	0.6678 / 0.0235	0.644
Near 1	TS	0.6344 / 0.1785	0.3215 / 0.1315	0.625
Near 1	HB	0.6344 / 0.0237	0.3215 / 0.0235	0.644

under these rank-preserving transformations. This is evidence about concentration with distinguishable scores, not about information lost through exact ties or numerical saturation. If all scores are

identical, confidence alone cannot distinguish which examples should be routed. Global scaling also cannot in general eliminate heterogeneous calibration residuals in every bin.

H.2 Token-aware scores for program-level routing

Let p_t denote the probability assigned to generated token t. This independent MBPP ablation compares the sequence geometric mean $\exp(T^{-1}\sum_t \log p_t)$ with $\min_t p_t$ and the geometric mean of the lowest-confidence 5% of tokens. Each score is calibrated against the outcome of executing the *complete program*. The geometric-mean baseline in this ablation is distinct from the mean-logit scorer used in the main language protocol. The minimum-token score lowers ECE but substantially reduces

Table 15: Token-aware MBPP ablation. Lower ECE does not necessarily imply better routing.

Confidence scorer	ECE-8B	ECE-70B	APGR
Sequence geometric mean	0.1023	0.1143	0.644
Minimum token probability	0.0668	0.0531	0.545
Bottom-5% geometric mean	0.0888	0.0862	0.640

APGR; the bottom-5% score yields APGR close to the sequence geometric mean. This illustrates the distinction between marginal calibration and the information needed for routing. These experiments do not implement token-level model switching: token probabilities depend on tokenization and the generated prefix, while execution feedback supplies a program-level target. A low-probability token need not be an incorrect token.

H.3 Source calibration versus target recalibration

The original ImageNet-C transfer experiment fits calibration on ImageNet-1K and evaluates the fixed system under corruptions. A separate experiment recalibrates on the shifted data at severity 5. These protocols answer different questions: the latter uses target-domain calibration information and must not be reported as zero-shot transfer. Recalibration improves the reported routing result

Table 16: Source-calibrated versus target-recalibrated ImageNet-C results at severity 5. ECE refers to the small model.

	Source calibration		Target recalibration	
Corruption	ECE	APGR	ECE	APGR
Gaussian noise	0.0584	0.6364	0.0537	0.6364
Motion blur	0.1602	0.5999	0.0589	0.6122
Fog	0.3140	0.5409	0.0285	0.6888

for fog and motion blur but not Gaussian noise. The fog result is therefore evidence for the value of target recalibration, not evidence that source-only calibration withstands severe fog. In a separate source-calibration experiment on ImageNet-R, ECE is 0.7711 for EVA-IN22K and 0.6282 for CoaT-S, indicating a substantial failure of calibration transfer. These observations limit conclusions to the evaluated distributions and protocols.

I Additional Data-Cleaning Evaluation

I.1 Comparing models on an identical cleaned subset

Two cleaning operating points must be distinguished. The confidence-filtered protocol selects the 11 highest-confidence bins out of 15 before removing flagged examples; its reported cascade accuracy is 96.266%. A separate all-bin protocol removes 2,278 jointly flagged examples from a 25,000-example test split and retains 22,722 examples. Table 17 evaluates both standalone models and the cascade on this *same* retained subset. Removing flagged examples changes both the evaluation population and its difficulty. Consequently, cleaned-subset accuracy cannot be compared directly with accuracy on the original ImageNet benchmark as a state-of-the-art result. A flag identifies a *potential* label issue and does not establish that every removed example was mislabeled. Candidate-pool counts across all bins also must not be interpreted as the number removed by the confidence-filtered protocol.

Table 17: Accuracy (%) under matched evaluation sets. The all-bin cleaned subset is different from the 11-bin confidence-filtered operating point.

Method	Original (25,000)	Cleaned (22,722)
EVA02-L-448-IN22K	89.752	97.434
EVA02-L-448-M38M	89.916	97.474
CC cascade	90.108	97.804

I.2 Human-annotated language evaluation

To complement the GPT-based proxy evaluation, we evaluate label-issue detection on MMLU-Redux-2.0, a dataset of 5,700 manually re-annotated questions spanning all 57 MMLU subjects. The dataset and annotation study are available at `https://huggingface.co/datasets/edinburgh-dawg/mmlu-redux-2.0` and `https://arxiv.org/abs/2406.04127`. In this evaluation, 370 questions (6.49%) are treated as containing errors or requiring further review. This positive category includes review-worthy issues and should not be described solely as definitively incorrect answer labels. The results exhibit the expected coverage–precision trade-off using human annotations rather than agreement with a GPT judge. They support the cleaning method under this evaluation standard; they do not retrospectively establish the accuracy of every GPT judgment in the original experiment.

Table 18: Label-issue detection on human-annotated MMLU-Redux-2.0. Detection rate is the fraction of all questions flagged, not recall over the erroneous questions.

Detection rate (%)	Precision (%)
1.23	72.86
1.63	67.31
2.46	67.14
3.46	61.47
5.07	57.34

I.3 Model diversity and additional models

Model diversity can provide complementary errors, but different architectures or training seeds do not establish statistical independence. Shared training data, task biases, and ambiguous examples can induce correlated errors. Thus, unanimous disagreement with a dataset label should be interpreted as a candidate for review, not proof that the label is wrong.

An exploratory three-model cascade uses CoaT-S and two EVA02 variants (IN22K and M38M). It reports 89.1% accuracy while CoaT-S handles more than 60% of examples, compared with approximately 89.7% for EVA-IN22K alone. The precise cost of choosing between the two large models must be included before translating this call reduction into a compute or latency claim. In a separate cleaning experiment, three models achieve 86.4% precision at a 7.9% detection rate, compared with an interpolated two-model precision of 85.3% at the same rate. This is a modest observed difference rather than evidence of a statistically significant gain from adding a model.

J Task-Specific Confidence and Reporting Conventions

Confidence protocols. For image classification, we temperature-scale class logits and use the predicted-class probability, with classification correctness as the target. For MMLU and ARC, confidence is the predicted answer's probability renormalized over valid options; downstream routing and CrossECE use Platt scaling. For MBPP, BigCodeBench, and GSM8K, the main protocol averages generated-token logits, applies Platt scaling, and verifies the complete answer or program. This mean removes dependence on the number of summed terms but does not guarantee invariance to length, tokenization, or answer format. Appendix H.2 reports separate probability-based token aggregations. Results from different calibrators and scorers are not interchangeable.

Binning and reporting. Equal-count binning partitions scores by rank; histogram calibration additionally fits bin-level correctness using labeled calibration data. Neither histogram nor parametric

calibration is label-free. The framework keeps backbone models fixed and does not train a separate neural router, but it still fits calibrators, estimates bin statistics, and selects operating points on labeled calibration data before held-out evaluation. Cleaning flags potentially erroneous labels: detection rate is the dataset fraction flagged, and precision is the positive fraction under the stated annotation standard. Confidence filtering ranks bins by mean calibrated confidence, not test accuracy. The models form an ensemble rather than a jointly trained mixture-of-experts architecture, and the implementation does not require statistical independence.