LLMs LEARN DIFFERENT FORMS OF METACOGNITION WHEN TRAINED TO PREDICT THEIR OWN ACCURACY

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models are trained to always produce an answer, regardless of whether they possess the relevant knowledge, which leads them to fabricate facts. Prior work has shown that LLMs' confidence estimates correspond poorly to their actual performance, and that fine-tuning can substantially improve them. However, what models actually learn during such training remains poorly understood. We investigate how LLMs acquire metacognitive monitoring, the ability to know what one knows, by training 10 open-weight LLMs to predict their own accuracy on factual multiple-choice questions before answering them. We find that trained confidence reflects two distinct signals. While on questions close to the training data, it tracks the model's true accuracy, in other domains, it instead tracks output consistency: the concentration of the model's answer distribution. Output consistency tracking emerges early in training and generalizes across datasets, whereas accuracy tracking develops later and remains local to the training distribution. These results suggest that calibration training may not teach models to generally detect errors they commit confidently, and they raise broader questions about the nature of metacognition in artificial systems.

1 INTRODUCTION

Large language models are increasingly integrated into work and daily life, with capabilities spanning coding, mathematical reasoning, creative writing, and information synthesis (Griot et al., 2025). However, they struggle to recognize the boundaries of their knowledge and often exhibit overconfidence (Sun et al., 2025). This is partly a consequence of training and evaluation practices that reward always producing an answer, which incentivizes hallucination when the model lacks the relevant information (Kalai et al., 2025). As users increasingly trust model outputs (Colombatto et al., 2025; Steyvers et al., 2025), the absence of a reliable signal indicating when a model is likely to be wrong poses substantial risks (Weidinger et al., 2021).

A promising way to address this limitation is to equip LLMs with metacognitive monitoring, the ability to assess their own knowledge states. Prior work has pursued this in several ways. Prompting-based methods elicit verbalized confidence estimates (Tian et al., 2023; Xiong et al., 2024). Sampling-based methods instead estimate uncertainty from the output consistency of a model's answers across samples (Kuhn et al., 2023; Manakul et al., 2023). Finally, fine-tuning and probing approaches train models, or lightweight heads on their internal representations, to predict their own correctness (Kadavath et al., 2022; Kapoor et al., 2024a). These generally yield better confidence estimates than prompting alone.

While prior work has achieved impressive calibration performance (Wang et al., 2025; Kapoor et al., 2024b), the research community has focused primarily on achieving high metrics on complex tasks, with little investigation into the underlying metacognitive processes. The mechanisms by which models assess their knowledge remain poorly understood—a critical gap given the importance of such skills in everyday user interactions. Understanding whether models truly assess their knowledge or merely track superficial signals has important implications for deployment: statistical shortcuts may fail under distribution shifts, in adversarial contexts, or when models need to reliably abstain from answering, even if they achieve high performance on standard benchmarks.

In this work, we investigate what LLMs learn when trained to estimate their own performance. To isolate metacognitive monitoring from other sources of error, such as faulty reasoning or planning,

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.

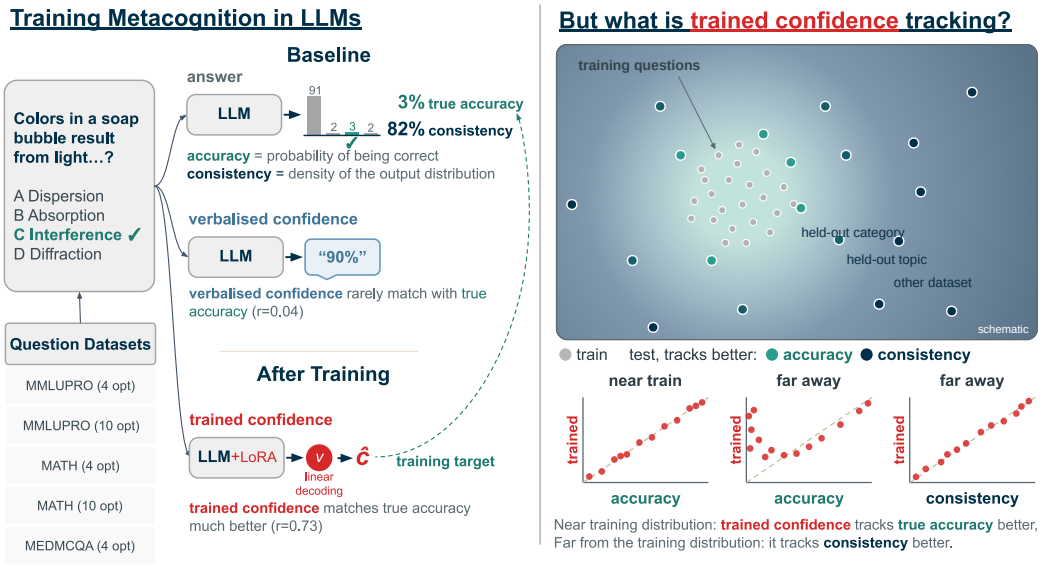

Figure 1: **Trained confidence tracks accuracy locally and consistency globally** Left side: experimental setup — models answer factual multiple-choice questions from five dataset variants. Accuracy is the probability assigned to the correct option, and output consistency is the normalized concentration of the answer distribution; the two can differ sharply, e.g., when a model reliably selects a wrong answer. Before training, verbalized confidence is essentially uncorrelated with true accuracy ($r = 0.04$). After fine-tuning with LoRA and a linear probe, trained confidence predicts accuracy much better ($r = 0.73$). Right side: schematics of the results — trained confidence tracks true accuracy on questions near the training distribution but output consistency on more distant ones, such as held-out topics or other datasets.

we focus on factual multiple-choice questions, which a model either knows the answer to or does not. Following Kadavath et al. (2022) we adopt a prospective setting in which the model estimates its accuracy on a question before answering it. This way, confidence reflects whether the model knows the answer rather than whether it can verify an answer it has already given. We fine-tune 10 open-weight LLMs, together with a linear probe on their hidden states, to predict their own accuracy on questions from science, medicine, and arithmetic datasets. Because train and test sets are split by topic, we can evaluate how learned confidence transfers to unseen domains. Beyond standard correlation metrics, we examine how trained confidence relates to other properties of the model's output distribution, how its behavior changes with semantic distance from the training data, and how it emerges over the course of training.

Contributions Consistent with prior work, we find that verbalized confidence is essentially uncorrelated with true accuracy before training, and that fine-tuning substantially improves it across all 10 models and 5 datasets. However, we found that trained confidence reflects two distinct signals: it tracks true accuracy on questions close to the training data, but output consistency, i.e., the concentration of the answer distribution, on more distant ones. This distinction matters because the two can differ sharply, for instance, when a model reliably selects a wrong answer. Finally, we find that output consistency tracking emerges early in training, whereas accuracy tracking develops later and remains confined to the training split and similar questions.

2 METHODS

As outlined in the introduction, we aim to isolate metacognitive monitoring from other processes. We therefore use multiple-choice questions that models answer directly, without chain-of-thought, a task requiring pure knowledge recall rather than classical reasoning tasks on which multiple metacognitive processes would need to be aggregated to guess the model's own performance. The

multiple-choice format also allows us to compute a model's accuracy and the concentration of its answer distribution exactly from the probabilities it assigns to each option. Confidence is elicited prospectively, before the model answers, so that it reflects whether the model knows the answer rather than whether it can verify an answer it has already produced.

2.1 DATASETS

To evaluate metacognitive performances on factual knowledge we used 3 distinct datasets: MMLU-PRO(Wang et al., 2024) which includes around 10k questions on various scientific topics, MEDMCQA (Pal et al., 2022), which contains 100k questions from multiple medical domains; and a custom MATH dataset (see Appendix A) which includes 100k calculation questions. These datasets include multiple choice questions, and most can be answered without reasoning by most modern AI models.

Let's introduce some notations: all these datasets $\mathcal{D}$ contain questions s^{q} with either 4 or 10 different options depending on the dataset $(s_j^{\mathrm{opt}})_{j\leq M_{\mathcal{D}}}$ (their concatenation is written $s^{\mathrm{opt}} = s_0^{\mathrm{opt}} s_1^{\mathrm{opt}} \dots s_{M_{\mathcal{D}}-1}^{\mathrm{opt}}$) and a single correct answer among these options s^{ans}. A dataset is written as a list of items $\mathcal{I}$: $\mathcal{D} = (\mathcal{I}_i)_{i\leq N_{\mathcal{D}}} = (s_i^{\mathrm{q}}, (s_{i,j}^{\mathrm{opt}})_{j\leq M_{\mathcal{D}}}, s_i^{\mathrm{ans}})_{i\leq N_{\mathcal{D}}}$.

We produced variations of these datasets by making the number of options vary: MMLU-PRO$_4$ with 4 options, MMLU-PRO$_{10}$ with 10 options (original), MEDMCQA$_4$ (original), MATH$_4$ (custom) and MATH$_{10}$ (custom). Additionally, we removed items longer than 500 tokens (using Mistral-7B tokenizer) for hardware requirements in order to run all experiments on a single H100 80GB (this resulted in around 5% of the original items removed in some datasets).

In all datasets, items are sorted into various categories. As such, each dataset variant is split into a train set and a test set such that (i) the accuracy of models remains similar on both the train and test sets; (ii) the train set includes around 90% of the questions (and the test set includes around 10%); (iii) for each category of question, (usually the topic of the question, such as Math, Physics, ...), all questions in this category are either in the train set or the test set (i.e. the split is a partitioning of the categories rather than individual questions).

More details about the dataset splitting method are in Appendix C. This ensures that the train and test sets are thematically distinct while retaining comparable metrics in accuracy to avoid statistical biases when comparing metacognitive performances between the train and test sets. The categories split are shown in Table 2 in Appendix C and accuracy statistics are reported in Table 3 in Appendix C. In practice, the categories composing MATH datasets are closer than MMLU-PRO and MEDMCQA, as they refer to subcategories of arithmetic computations, while the latter refer to completely different topics.

2.2 MODELS

We evaluated and finetuned 10 different LLMs: Qwen2.5-7B, Qwen2.5-7B-Instruct (Qwen et al., 2025), Qwen3.5-9B-Base, Qwen3.5-9B (Yang et al., 2025), Llama-2-7b-chat (Touvron et al., 2023), Llama-3.2-3B-Instruct (Grattafiori et al., 2024), phi-4 (Abdin et al., 2024), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), Ministral-3-3B-Instruct-2512 and Ministral-3-8B-Instruct-2512 (Liu et al., 2026). These were chosen for covering a wide range of recent development in LLMs, including base and instruct tuned models from various companies and training methods. All models used in this paper are quantized in 4-bits using unsloth (UnslothAI et al., 2025) and were chosen for their capability to fit on a single H100 80GB GPU. Thus this study does not include very large models for compute limitation reasons.

2.3 EVALUATION PROCEDURES

Accuracy evaluation Models are evaluated on each dataset split by using a question/answer prompt (see Appendix B for formatting details) and evaluating the probability to generate each option. The probability associated with the correct option is the accuracy of the model on the question.

$$Acc(\mathcal{M}_\theta, \mathcal{I}) = p_\theta(s^{\mathrm{ans}} | s^{\mathrm{q}} s^{\mathrm{opt}}) \tag{1}$$

with $\mathcal{M}_\theta$ the model, $\mathcal{I}$ the item composed of a question s^{q}, a list of options $(s_j^{\text{opt}})_{j \in M_\mathcal{D}}$ and $p_\theta(s^{\text{ans}}|s^{\text{q}}s^{\text{opt}})$ the generative probability function of model $\mathcal{M}_\theta$ for generating s^{ans} after prompt $s^{\text{q}}s^{\text{opt}}$ (concatenation of the question with the options).

The accuracy of a model on dataset split $\mathcal{D}$ is the average accuracy on the dataset split and is evaluated for both the train and the test splits:

$$Acc(\mathcal{M}_\theta, \mathcal{D}) = \frac{1}{N_\mathcal{D}} \sum_{i \leq N_\mathcal{D}} Acc(\mathcal{M}_\theta, \mathcal{I}_i) \tag{2}$$

Output consistency evaluation Aside from accuracy, the output consistency of the model on each question was also evaluated. Output consistency is a measurement of how often the LLM would give the same answer to the same question and is mathematically defined as

$$Con(\mathcal{M}_\theta, \mathcal{I}) = 1 - \frac{H(p_\theta(.|s^{\text{q}}s^{\text{opt}}))}{\log(M_\mathcal{D})} \tag{3}$$

with $M_\mathcal{D}$ the number of options for dataset $\mathcal{D}$ from which item $\mathcal{I}$ is extracted, H the entropy function and $p_\theta(.|s^{\text{q}}s^{\text{opt}})$ the probability distribution of sampling one of the possible options in $(s_i^{\text{opt}})_{i \leq M_\mathcal{D}}$.

In other words, the output consistency is the entropy of the output distribution (only considering the possible options) scaled in [0,1] such that an output consistency of 0 corresponds to a uniform distribution and an output consistency of 1 represents an output distribution associating probability 1 to one of the options. Output consistency evaluation is performed on both train and test splits.

Metacognitive evaluation Lastly, metacognitive performance is measured using two different methods. Before training, it is evaluated with a verbal statement from the LLM using an additional pre-prompt s^{pre} before the question to ask the LLM to give its accuracy estimate on the given question instead of answering it (see Appendix B for prompting details).

$$Met_{verb}(\mathcal{M}_\theta, \mathcal{I}) = \text{parse_number} \circ \text{greedy}(\mathcal{M}_\theta, s^{\text{pre}}s^{\text{q}}s^{\text{opt}}, 5) \tag{4}$$

with $\text{greedy}(\mathcal{M}_\theta, t, n)$ the function that autoregressively generates n tokens after context t using LLM $\mathcal{M}_\theta$ and parse_number the function that parses the first number in a text (transforms from string to integer/float).

After training, the procedure involves reading from the trained probe $\mathbf{v}$ at the end of the transformer stack (see Section 2.4). The output of the probe is passed through a sigmoid function to scale it between 0 and 1.

$$Met_{prob}(\mathcal{M}_\theta, \mathcal{I}) = \sigma(\mathbf{v} \cdot \mathbf{t}_\theta^{end}(s^{\text{pre}}s^{\text{q}}s^{\text{opt}})) \tag{5}$$

with $\mathbf{t}_\theta^{end}(s^{\text{pre}}s^{\text{q}}s^{\text{opt}})$ the function that returns the output at the end of the transformer stack of LLM $\mathcal{M}_\theta$ after input $s^{\text{pre}}s^{\text{q}}s^{\text{opt}}$, σ the sigmoid function and $\mathbf{v}$ a vector learnt during training. Another training setup puts the probe at the middle of the transformer stack instead of the end. In that case it is $\mathbf{t}_\theta^{mid}(s^{\text{pre}}s^{\text{q}}s^{\text{opt}})$ that is used, namely the latent activation at the middle of the transformer stack.

2.4 Training Procedures

The training setup consists in adding LoRA adapters in the original quantized LLM used to compute the accuracy, output consistency and verbalised metacognition (QLoRA (Dettmers et al., 2023)) and adding a linear regression (called probe) from the output activations to a single scalar at the end of the transformer stack. This setup is inspired from Kadavath et al. (2022) who shown that adding a probe yields better results than training token generation and later from Gaven et al. (2025) that implemented it using QLoRA.

Models were trained on the train split of each dataset variant 5 times with an MSE loss to predict their true accuracy:

$$Loss(\phi) = \mathbb{E}_{\mathcal{I} \in \mathcal{D}} \left[(Met_{prob}(\mathcal{M}_\phi, \mathcal{I}) - Acc(\mathcal{M}_\theta, \mathcal{I}))^2 \right]^{\frac{1}{2}} \tag{6}$$

This loss on ϕ was minimized on the train set for a single epoch for variants of both MedMCQA and MATH datasets (100k items in the train split is 100k steps) and for 8 epochs for MMLU-PRO

(10k items in train split $\times 8 = 80$k steps) using QLoRA to approach the same quantity of training steps than MEDMCQA and MATH. Hyperparameters are detailed in Appendix D.

Training metrics and training curves are reported in Appendix G. In this experiment, we train metacognition alone: the additional LoRA adaptors can only be used to predict metacognition and make the network unable to answer questions anymore.

2.5 ANALYSES

2.5.1 CORRELATION METRICS

We analyzed the data using Pearson correlations between verbalized confidence, true accuracy, and output consistency. The choice of metrics was motivated by the fact that our data are continuous and that we aim to quantify how much of a linear relationship exists between two variables. More precisely, we used differences in Pearson correlations to quantify how much a trained confidence linearly reflects true accuracy or output consistency:

$$\begin{aligned}\Delta r(\mathcal{M}_\theta, \mathcal{D}) =& r\left[(Acc(\mathcal{M}_\theta, \mathcal{I}))_{\mathcal{I}\in\mathcal{D}}, (Met_{prob}(\mathcal{M}_\theta, \mathcal{I}))_{\mathcal{I}\in\mathcal{D}}\right] \\ &- r\left[(Con(\mathcal{M}_\theta, \mathcal{I}))_{\mathcal{I}\in\mathcal{D}}, (Met_{prob}(\mathcal{M}_\theta, \mathcal{I}))_{\mathcal{I}\in\mathcal{D}}\right]\end{aligned} \tag{7}$$

for some (trained) model $\mathcal{M}_\theta$ tested on dataset $\mathcal{D} = (\mathcal{I}_i)_{i\leq N_\mathcal{D}}$.

2.5.2 ERROR METRICS

While correlation can be very useful for studying how well a LLM is predicting its accuracy on a dataset, it cannot measure whether the answer to an individual question reflects true accuracy or rather output consistency. For this specific situation, we used the differences in absolute errors for some (trained) model $\mathcal{M}_\theta$ tested on question $\mathcal{I}$:

$$\Delta e(\mathcal{M}_\theta, \mathcal{I}) = \|Acc(\mathcal{M}_\theta, \mathcal{I}) - Met_{prob}(\mathcal{M}_\theta, \mathcal{I})\| - \|Con(\mathcal{M}_\theta, \mathcal{I}) - Met_{prob}(\mathcal{M}_\theta, \mathcal{I})\| \tag{8}$$

To recap, the correlation is used to compute the performance on a set of questions (usually a dataset), while the error is used to compute the performance on a single question. Indeed, in this study, we investigate whether trained confidence matches a linear or U-shape pattern with true accuracy or consistency. Therefore correlation is a better suited tool to study this dependency at the dataset level than error (Brier score) so we only keep it for individual questions and more specific experiment.

2.5.3 EMBEDDINGS

We embed each question using the Qwen3-Embedding-0.6B model (Zhang et al., 2025), which maps text to a 1024-dimensional vector such that semantically similar texts have nearby representations. We use these embeddings to quantify how similar questions are, both within and across datasets, and compute them on the training and test splits of each dataset. Each item $\mathcal{I}$ is embedded together with its options ($s^{\mathrm{q}}s^{\mathrm{opt}}$), giving an embedding $e_\mathcal{I}$; the embeddings of the same question in the 4- and 10-option variants of a dataset are therefore similar but not identical. The distance between two items is their cosine distance, consistent with the similarity measure used to train the embedding model, and we compute the distance from a test item $\mathcal{I}$ to a training dataset $\mathcal{D}$ as the average distance to its $k = 100$ nearest training items where $\mathcal{N}_k(\mathcal{I}, \mathcal{D}) \subset \mathcal{D}$ is the set of the k training items closest to $\mathcal{I}$:

$$d_k(\mathcal{I}, \mathcal{D}) = \frac{1}{k} \sum_{\mathcal{I}'\in\mathcal{N}_k(\mathcal{I},\mathcal{D})} \left(1 - \frac{e_\mathcal{I} \cdot e_{\mathcal{I}'}}{\|e_\mathcal{I}\| \, \|e_{\mathcal{I}'}\|}\right), \tag{9}$$

3 RESULTS

We first compare confidence estimates before and after training (Section 3.1). We then analyze which signals trained confidence tracks and how this depends on the similarity between test questions and the training data (Sections 3.2–3.4). Finally, we also examine how these signals emerge over the course of training (Appendix E).

3.1 FINE-TUNING SUBSTANTIALLY IMPROVES CONFIDENCE ESTIMATION

As a baseline, we elicit verbalized confidence estimates from untrained models. These estimates are essentially uncorrelated with true accuracy on the test splits ($r = 0.04 \pm 0.01$, mean $\pm$ s.e.m. across models and datasets). After fine-tuning, confidence estimates correlate strongly with accuracy on test splits ($r = 0.73 \pm 0.03$). This improvement holds for every model and dataset (Appendix H Figure 11 for the details). Note that the two conditions differ both in training and in elicitation format (verbal report vs. linear probe): since an untrained probe produces arbitrary outputs, verbalized confidence is the natural baseline before any form of training. This step simply builds the models we are going to investigate in the rest of the paper.

3.2 TRAINED CONFIDENCE TRACKS TRUE ACCURACY OR OUTPUT CONSISTENCY DEPENDING ON THE DATASET

To examine the learned relationship at the level of individual questions, we plot trained confidence against true accuracy. Figure 2 (top left) shows this relationship for Qwen3.5-9B trained on the training split of MEDMCQA$_4$ and evaluated on its test split, using data from the first training run. The relationship follows a U-shape: confidence closely matches accuracy on questions the model answers correctly with high probability, slightly underestimates accuracy on questions of intermediate accuracy, and strongly overestimates it on many questions with accuracy close to zero, i.e., questions on which the model reliably selects an incorrect answer. Moreover, the model never predicts an accuracy below chance level (25% in this example), although many questions fall below this threshold because the model consistently selects the same incorrect answer.

In contrast, plotting trained confidence against output consistency for the same model and test split reveals an almost linear relationship (Figure 2, bottom left), with a higher correlation than with true accuracy ($r = 0.94$ vs. $r = 0.81$). This suggests that the model may use its output consistency as a proxy for accuracy. Indeed, a model that performs well on a task achieves high accuracy by consistently selecting the correct answer. Consequently, the more competent the model, the better its output consistency on a question predicts whether it will answer correctly.

This pattern reverses on MATH$_4$ (Figure 2, right column of the left panel), where Qwen3.5-9B is trained on the training split and evaluated on the test split of MATH$_4$, again using data from the first run on this training set. Here, confidence tracks true accuracy almost perfectly ($r = 0.99$), whereas its relationship with output consistency is non-linear ($r = 0.84$) and follows a transposed U-shape.

These observations generally hold across models. On MMLU-PRO$_4$, MMLU-PRO$_{10}$, and MEDMCQA$_4$, most models show a U-shaped relationship between trained confidence and true accuracy (more details about this in Appendix H), and a nearly linear relationship between trained confidence and output consistency. To quantify this difference, Figure 2 (right panel) reports Δr, the difference between the correlation of trained confidence with true accuracy and trained confidence correlation with output consistency, computed on the test split of each dataset. On these three datasets (blue), Δr is negative ($\Delta r = -0.12 \pm 0.01$, mean $\pm$ s.e.m. across models), indicating that trained confidence is more linearly related to output consistency than to true accuracy. Conversely, on the MATH datasets (red), trained confidence is almost linearly related to true accuracy, while its relationship with output consistency follows a U-shape ($\Delta r = 0.27 \pm 0.06$).

3.3 OUTPUT CONSISTENCY TRACKING TRANSFERS ACROSS DATASETS

We hypothesize that this difference reflects the distance between training and test questions. Although all splits are category-disjoint, the categories of the MATH datasets are much closer to each other than those of MMLU-PRO and MEDMCQA: the MATH test category (easy$*$) lies between training categories (easy$-$, easy$+$, medium$*$ and hard$*$), whereas the test categories of the other datasets (e.g., philosophy or ophthalmology) are distinct topics. To test this hypothesis, we evaluate each model in a 5x5 design: each model (trained on train split of dataset X) is tested on all datasets test split (Figure 3). This cross evaluation further increases the distance between training and test questions. If our hypothesis holds, models should track output consistency on all datasets other than their training dataset, with the exception of the two MATH and MMLU-PRO variants, which share the same questions and categories.

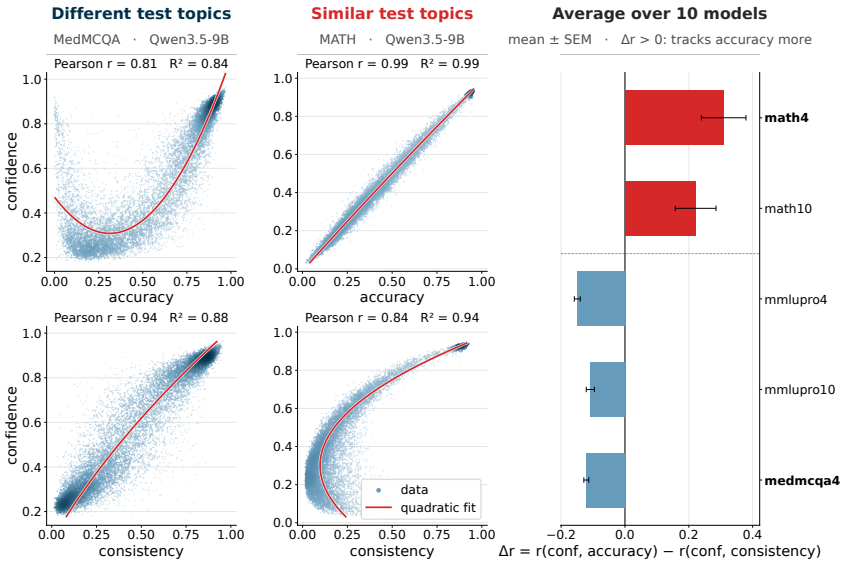

Figure 2: **Relationship between predicted accuracy, true accuracy and output consistency.** Left panel shows the trained confidence of Qwen3.5-9B against true accuracy (top) and output consistency (bottom) on the test split of MEDMCQA$_4$ and MATH$_4$. Red lines are quadratic fits (R^2 shown above each plot). Right panel represents Δr on each test split; positive values (red) indicate that confidence tracks true accuracy more than output consistency, negative values (blue) the reverse. Error bars: s.e.m. across the 10 models (each averaged over 5 runs).

Consistent with this prediction, models trained on MATH datasets linearly track true accuracy when evaluated on either MATH variant ($\Delta r = 0.23 \pm 0.07$), but switch to tracking output consistency when evaluated on MMLU-PRO or MEDMCQA ($\Delta r = -0.14 \pm 0.02$). Conversely, models trained on MMLU-PRO or MEDMCQA keep tracking output consistency across all evaluation datasets ($\Delta r = -0.13 \pm 0.02$). Accuracy tracking on MATH may therefore not be a property of training on the math questions themselves, but arises only when models are trained on nearby questions akin to the model being familiar with its performance on math but not on more general scientific topics.

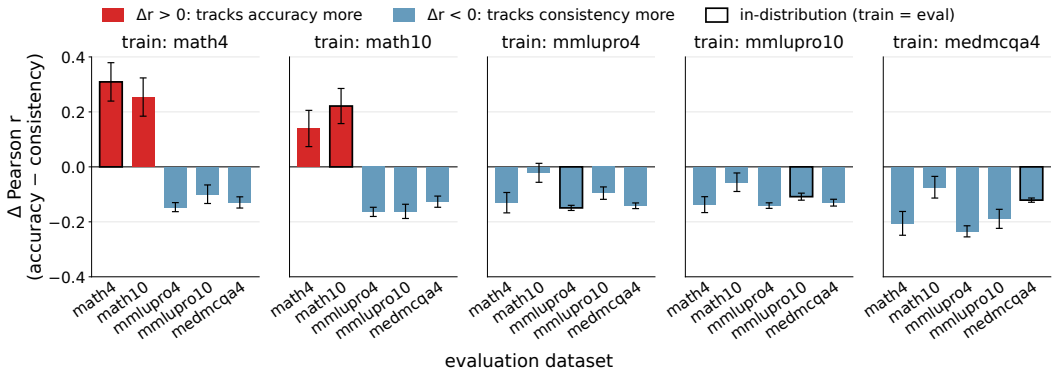

Figure 3: **Cross-dataset evaluation of trained confidence.** Each panel shows models trained on one dataset and evaluated on the test split of all five datasets. Bars represent Δr; positive values (red) indicate that confidence tracks true accuracy more than output consistency, negative values (blue) the reverse. In-distribution evaluations (tested on the test split from the training dataset - thematically distinct but closer with respect to the other datasets) are highlighted. Error bars: s.e.m. across the 10 models (each averaged over 5 runs).

3.4 DISTANCE TO THE TRAINING DATA DETERMINES WHICH SIGNAL DOMINATES

The previous results suggest that trained confidence reflects two signals: true accuracy, which is linearly tracked only for questions close to the training data, and output consistency, which is tracked in datasets far from the training distribution. Nonetheless, this hypothesis has only been tested at the level of the full dataset, namely whether predicted confidence on the full test set reflects generally more true accuracy or output consistency. However, a test set could include some questions that are predicted to be closer to true accuracy and others that better reflect output consistency.

To examine this relationship even closer at the level of individual questions within each dataset, we plot one point per training dataset (5 datasets) per test question (dataset test splits = 38581 test questions) showing Δe, the difference between the true accuracy and output consistency prediction errors, computed after averaging accuracy, output consistency, and confidence across all models trained on this training set (10 base LLMs x 5 runs). This experiment makes it possible to explore this relationship beyond pure linear or U-shape tracking at the dataset scale and is therefore complementary to the correlation studies conducted so far.

Figure 4 shows that Δe increases steadily with distance from the training data for all models and training datasets. Test questions that are similar to at least one training question yield confidence estimates that track true accuracy, whereas questions dissimilar to all training questions yield estimates that track output consistency, regardless of the training dataset.

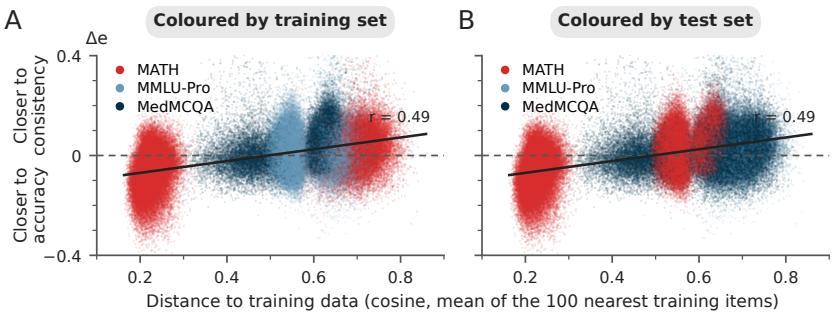

Figure 4: **Distance to the training data and the signal tracked by trained confidence.** Each point is one test question for one training dataset; its y-value is Δe computed from accuracy, consistency and confidence averaged over 10 models $\times$ 5 training runs. Negative values mean confidence is closer to true accuracy, positive values closer to output consistency. Points are colored by training dataset (A) or test dataset (B). The x-axis is the mean cosine distance to the 100 nearest training questions. Black line: linear fit ($r = 0.49$); dashed: $\Delta e = 0$; 0.13% points beyond ± 0.4 not shown.

4 DISCUSSION

We investigated what LLMs learn when trained to predict their own accuracy. Although models are explicitly trained to predict true accuracy, their confidence reflects two distinct signals. On questions close to the training data, confidence tracks true accuracy, whereas on more distant questions it tracks output consistency better. We also show that output consistency tracking emerges early in training (see Appendix E) and transfers across datasets, while accuracy tracking develops later and remains confined to questions similar to those seen during training.

Another way to understand these processes is to see output consistency tracking as a form of first-person metacognition: the model reads out a property of its own internal state, namely how concentrated its beliefs over the answer options are. Because this signal is available for any question, it generalizes across domains and is learned quickly. True accuracy tracking resembles a form of third-person metacognition: by observing its own performance on training questions, the model learns how well it performs on specific topics, much as an external observer could learn to predict the model's behavior from examples. This form of accuracy tracking can be seen as implicit contamination. The model does not generalize whether it knows the answer to a new question, akin to knowing someone's movie preferences is not necessarily informative about their food tastes, but the

model rather learns its own accuracy on questions it has seen and on closely related ones. This is consistent with our results on MATH datasets, whose test split is category-disjoint from the training split yet highly similar to it. The scope of third-person metacognition is therefore bounded both by the model's capacity and by its exposure to its own behavior, and does not easily generalize to other types of questions (MMLU-PRO and MEDMCQA). Additionally, Appendix F further shows that this dissociation persists when the probe is placed at the middle of the network rather than at its end. Consistency tracking is preserved, whereas true accuracy tracking largely disappears. First, this shows that consistency tracking does not stem from a leak from the network's output, where consistency is directly encoded in the logits — which are computed from the output of the network. Second, it suggests that output consistency tracking may be the simplest and lightweight form of metacognition of the two.

This picture qualifies the concern raised in the introduction that trained confidence may rely on shortcuts that fail under distribution shift. In our experiments, the signal that fails to transfer is accuracy tracking, whereas output consistency tracking, though less precise, is the one that generalizes.

Why would output consistency be the signal that generalizes? A model has no direct access to the truth, only to its own beliefs. LLMs are trained on large corpora that contain inaccurate information, so a model may hold a false belief as firmly as a true one, and both can produce the same internal signal. From the model's perspective, output consistency with its own beliefs is then the best available estimate of correctness, and errors made with high output consistency may not be detectable from the inside. More generally, no cognitive system is omniscient, and it is unclear whether a finite system can fully represent the limits of its own knowledge: this is similar to a map that would have to represent the world at the same time as a complete drawing of itself - the information capacity may prevent it.[1] Perfect metacognition may therefore be out of reach, and the more relevant question is what the upper bound on self-knowledge is. We speculate that combining a general first-person signal with third-person knowledge of one's performance on familiar domains may approach this bound.

These results raise the question of what metacognitive monitoring actually is, not only in LLMs but also in humans. Caziot & Mamassian (2021) showed that human perceptual confidence reflects self-consistency more closely than true accuracy, suggesting that consistency may be the true basis of metacognition. Our results are consistent with this view. Individuals do not have access to the truth, only to their perception of it, so the best estimate of whether one is correct is whether one's answer is consistent with one's own beliefs. Confidence should therefore reflect self-consistency rather than true accuracy. Our models illustrate this in an artificial system: although they are explicitly trained to predict their true accuracy, the optimal behavior found by the optimizer is a signal that generalizes beyond familiar questions: output consistency. Nonetheless, despite these similarities the comparison has a limitation: these two studies do diverge in the exact definition of consistency: in our study output consistency is the peakiness of the output distribution, while in Caziot & Mamassian (2021), self-consistency is the probability to answer the same answer again. Kumaran (2026) has shown that verbal confidence in LLMs tends to match better with commitment in a commit vs. opt-out task, where commitment could be comparable to self-consistency in Caziot & Mamassian (2021).

Finally, these results raise the question of how a model arbitrates between the two signals for a given question. Our results suggest that distance to the training data is associated with this balance — a form of implicit contamination vs. generalization — but the underlying mechanisms should be studied further. Answering this question would require understanding how a model assesses whether the best way to predict its performance is through its internal consistency or its knowledge of its observed behavior: a form of meta-metacognition.

Our study has several limitations. We focus on factual multiple-choice questions answered without chain-of-thought, and on models between 3B and 9B parameters quantized to 4 bits showing only a small spectrum of LLM confidence interactions. It only includes a couple of datasets and in the training section confidence is compared before and after training using different elicitation formats (verbal report vs. linear probe). Still this does not affect most findings in the paper. Finally, distance to the training data is measured with a single embedding model. Extending these analyses to open-ended generation, reasoning tasks, and larger models is an important direction for future work.

[1] Informally, consider a system with n bits of capacity, m of which encode knowledge. A complete description of its knowledge limits must be stored in the remaining $n - m$ bits, yet must cover both the m bits of knowledge and the $n - m$ bits of the description itself, i.e., information about all n bits.

AI USE STATEMENT

In this work, we have used AI to help implement methods (Claude Code to improve code performance) but we did not use AI tools for any of these tasks: Generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. We used AI for these tasks: help generate some of the figures (code and help draft/adjust figure 1 half by hand and half by Claude Opus), format a couple of references in bibtex, edit and generate parts of code, edit a research paper to improve readability and help format some of the appendices and tables from the code and raw results. We have reviewed all AI-assisted work: we have read and updated most of the generated readability and writing suggestions multiple times by all authors, and we checked and adjusted by hand most of the code proposed by AI. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendices reference most of the technical details used during this research project. Code and data will be made available after the anonymity period.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. Phi-4 technical report, 2024. URL https://arxiv.org/abs/2412.08905.

Baptiste Caziot and Pascal Mamassian. Perceptual confidence judgments reflect self-consistency. *Journal of Vision*, 21(12):8–8, 2021. doi: 10.1167/jov.21.12.8. URL https://doi.org/10.1167/jov.21.12.8.

Clara Colombatto, Jonathan Birch, and Stephen Fleming. The influence of mental state attributions on trust in large language models. *Communications Psychology*, 3, 05 2025. doi: 10.1038/s44271-025-00262-1.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023. URL https://arxiv.org/abs/2305.14314.

Loris Gaven, Thomas Carta, Clément Romac, Cédric Colas, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Magellan: Metacognitive predictions of learning progress guide autotelic llm agents in large goal spaces, 2025. URL https://arxiv.org/abs/2502.07709.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu,

Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent,

Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Maxime Griot, Coralie Hemptinne, Jean Vanderdonckt, and Demet Yuksel. Large language models lack essential metacognition for reliable medical reasoning. *Nature Communications*, 16, 01 2025. doi: 10.1038/s41467-024-55628-6.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

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, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know, 2022. URL https://arxiv.org/abs/2207.05221.

Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why language models hallucinate, 2025. URL https://arxiv.org/abs/2509.04664.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Katherine Collins, Arka Pal, Umang Bhatt, Adrian Weller, Samuel Dooley, Micah Goldblum, and Andrew Gordon Wilson. Large language models must be taught to know what they don't know, 2024a. URL https://arxiv.org/abs/2406.08391.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Arka Pal, Samuel Dooley, Micah Goldblum, and Andrew Wilson. Calibration-tuning: Teaching large language models to know what they don't know. In Raúl Vázquez, Hande Celikkanat, Dennis Ulmer, Jörg Tiedemann, Swabha Swayamdipta, Wilker Aziz, Barbara Plank, Joris Baan, and Marie-Catherine de Marneffe (eds.), *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, pp. 1–14, St Julians, Malta, March 2024b. Association for Computational Linguistics. URL https://aclanthology.org/2024.uncertainlp-1.1/.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation, 2023. URL https://arxiv.org/abs/2302.09664.

Dharshan Kumaran. Reported confidence in llms tracks commitment more than correctness, 2026. URL https://arxiv.org/abs/2606.29490.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, Baptiste Rozière, Baudouin De Monicault, Clémence Lanfranchi, Corentin Barreau, Cyprien Courtot, Daniele Grattarola, Darius Dabert, Diego de las Casas, Elliot Chane-Sane, Faruk Ahmed, Gabrielle Berrada, Gaëtan Ecrepont, Gauthier Guinet, Georgii Novikov, Guillaume Kunsch, Guillaume Lample, Guillaume Martin, Gunshi Gupta, Jan Ludziejewski, Jason Rute, Joachim Studnia, Jonas Amar, Joséphine Delas, Josselin Somerville Roberts, Karmesh Yadav, Khyathi Chandu, Kush Jain, Laurence Aitchison, Laurent Fainsin, Léonard Blier, Lingxiao Zhao, Louis Martin, Lucile Saulnier, Luyu Gao, Maarten Buyl, Margaret Jennings, Marie Pellat, Mark Prins, Mathieu Poirée, Mathilde Guillaumin, Matthieu Dinot, Matthieu Futeral, Maxime Darrin, Maximilian Augustin, Mia Chiquier, Michel Schimpf, Nathan Grinsztajn, Neha Gupta, Nikhil Raghuraman, Olivier Bousquet, Olivier Duchenne, Patricia Wang, Patrick von Platen, Paul Jacob, Paul Wambergue, Paula Kurylowicz, Pavankumar Reddy Muddireddy, Philomène Chagniot, Pierre Stock, Pravesh Agrawal, Quentin Torroba, Romain Sauvestre, Roman Soletskyi, Rupert Menneer, Sagar Vaze, Samuel Barry, Sanchit Gandhi, Siddhant Waghjale, Siddharth Gandhi, Soham Ghosh, Srijan Mishra, Sumukh Aithal, Szymon Antoniak, Teven Le Scao, Théo Cachet, Theo Simon Sorg, Thibaut Lavril, Thiziri Nait Saada, Thomas Chabal, Thomas Foubert, Thomas Robert, Thomas Wang, Tim Lawson, Tom Bewley, Tom Bewley, Tom Edwards, Umar Jamil, Umberto Tomasini, Valeriia Nemychnikova, Van Phung, Vincent Maladière, Virgile Richard, Wassim Bouaziz, Wen-Ding Li, William Marshall, Xinghui Li, Xinyu Yang, Yassine El Ouahidi, Yihan Wang, Yunhao Tang, and Zaccharie Ramzi. Ministral 3, 2026. URL https://arxiv.org/abs/2601.08584.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models, 2023. URL https://arxiv.org/abs/2303.08896.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa : A large-scale multi-subject multi-choice dataset for medical domain question answering, 2022. URL https://arxiv.org/abs/2203.14371.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Mark Steyvers, Heliodoro Tejeda, Aakriti Kumar, Catarina Belem, Sheer Karny, Xinyue Hu, Lukas W. Mayer, and Padhraic Smyth. What large language models know and what people think they know. *Nature Machine Intelligence*, 7(2):221–231, 2025. doi: 10.1038/s42256-024-00976-7.

Fengfei Sun, Ningke Li, Kailong Wang, and Lorenz Goette. Large language models are overconfident and amplify human bias, 2025. URL https://arxiv.org/abs/2505.02151.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D. Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback, 2023. URL https://arxiv.org/abs/2305.14975.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi,

Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

UnslothAI, Daniel Han-Chen, and Michael Han-Chen. Unsloth. https://github.com/unslothai/unsloth, 2025.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

Ziming Wang, Zeyu Shi, Haoyi Zhou, Shiqi Gao, Qingyun Sun, and Jianxin Li. Towards objective fine-tuning: How llms' prior knowledge causes potential poor calibration?, 2025. URL https://arxiv.org/abs/2505.20903.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Ethical and social risks of harm from language models, 2021. URL https://arxiv.org/abs/2112.04359.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms, 2024. URL https://arxiv.org/abs/2306.13063.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. URL https://arxiv.org/abs/2506.05176.

APPENDIX

Data and experimental setup

A Custom MATH dataset: question generation, difficulty levels, categories, and distractor design.

B Question formatting: answer and confidence prompts, how option probabilities are read out, and how verbalized confidence is parsed.

C Dataset splitting: category-level train/test split procedure, resulting categories, and per-model accuracy on each split.

D Hyperparameters: training settings shared across all models and datasets.

Additional results

E Emergence during training: how output consistency tracking and accuracy tracking develop across training checkpoints.

F Middle probe: replication of all main analyses with the probe at the middle of the network, ruling out a logit-leak explanation.

Supplementary figures

G Training curves: train and test loss curves during training for both end and middle probes.

H Individual model plots: per-model confidence performance before and after training, and cross-dataset correlations and Δr for both probes.

A CUSTOM MATH DATASET

The MATH datasets consist of procedurally generated arithmetic questions of the form "What is $a \circ b$?", with $\circ \in \{+, -, \times\}$. Answering them does not require any world knowledge, and difficulty can be controlled precisely through the magnitude of the operands. This gives a set of question categories that are thematically more homogeneous, in contrast to MMLU-PRO and MEDMCQA while remaining challenging and distinct questions.

Operands and difficulty levels. We generate 120 000 questions across three difficulty levels, defined by the number of digits of the smaller operand b. The larger operand a is written first. Pairs (a, b) are sampled uniformly without replacement from the ranges below, so no question appears twice within a level:

- **easy** (30% of questions): $b \in [1, 9]$, $a \in [10, 99\,999]$;
- **medium** (40%): $b \in [10, 99]$, $a \in [100, 99\,999]$;
- **hard** (30%): $b \in [100, 999]$, $a \in [1\,000, 99\,999]$.

Within each level, the three operators are assigned in rotation, so each operator covers exactly one third of the level. Because $a > b$ in every range, results of subtractions are always strictly positive. Each combination of level and operator defines a category (e.g., medium$*$ for medium multiplications), which yields the 9 categories listed in Table 1. Following the procedure of Appendix C, the category easy$*$ is held out as the test split and the 8 remaining categories form the training split. This gives 108 000 training and 12 000 test questions.

Level	Operators	b range	a range	#questions per operator
easy	$+, -, \times$	$[1, 9]$	$[10, 99\,999]$	12 000
medium	$+, -, \times$	$[10, 99]$	$[100, 99\,999]$	16 000
hard	$+, -, \times$	$[100, 999]$	$[1\,000, 99\,999]$	12 000

Table 1: Categories of the MATH datasets. Each level is split evenly between the three operators, yielding 9 categories (e.g., easy$+$, easy$-$, easy$*$).

Distractor generation. Distractors are designed so that questions cannot be solved by shallow heuristics such as checking the magnitude of the result or its last digit. Each distractor is obtained by altering some digits of the correct result r, whose decimal representation has L digits:

1. **Candidate positions.** If $L > 2$, the first and last digits of r are kept fixed and only the inner positions $\{1, \ldots, L-2\}$ may be altered. If $L \leq 2$, all positions may be altered.
2. **Number of altered digits.** We draw $n = \min(1 + k,\ 4,\ |\text{candidates}|)$ with $k \sim \text{Poisson}\left(\frac{L-1}{2}\right)$. Longer results therefore tend to have more altered digits, up to at most 4.
3. **Shared positions.** A set of n distinct positions is sampled uniformly among the candidates. The same positions are used for all distractors of a question.
4. **Distractor values.** For each distractor, the digits at these positions are replaced by uniformly random digits. We reject combinations that equal the correct result's digits at these positions and combinations already used by another distractor. This guarantees that all options are distinct.

As a result, all options of a question share the same leading digit, the same last digit and, in general, the same number of digits. They differ only at the altered positions. For example, the medium$*$ question "What is $48213 * 87$?" ($r = 4\,194\,531$) could receive the altered positions $\{2, 4\}$ and the options $\{4\,194\,531,\ 4\,134\,231,\ 4\,194\,831,\ 4\,164\,531\}$. Note that a single distractor may keep some of the correct digits, since only the full combination at the altered positions must differ.

The correct answer is inserted among the $M_{\mathcal{D}} - 1$ distractors, and the options are shuffled uniformly, so the position of the correct answer is uniformly distributed.

4- and 10-option variants. MATH_4 and MATH_{10} are generated with the same random seed. Operand pairs are sampled before any distractor is drawn, so the two variants contain exactly the same questions and category split. They differ only in their number of options (3 and 9 distractors, respectively), and thus in their distractors.

B QUESTION FORMATTING

All models are evaluated zero-shot, with no few-shot examples, no system prompt and no chat template. Instruction-tuned models therefore receive the same raw text as base models. In every prompt, the question q is introduced by `"Question :"`, so that $s^{\mathrm{q}} =$ `"Question :"` $\oplus\, q$. It is followed by the options, one per line. Options are labelled with capital letters in their dataset order: (A) to (D) for 4-option variants and (A) to (J) for 10-option variants. The options block is $s^{\mathrm{opt}} = s_0^{\mathrm{opt}} \cdots s_{M_{\mathcal{D}}-1}^{\mathrm{opt}}$, where $s_j^{\mathrm{opt}} =$ `"\n("` $\oplus\, L_j \oplus$ `") "` $\oplus\, o_j$ contains the letter L_j and the text o_j of option j. The prompt then ends with a cue line, which differs between answering and confidence elicitation. For readability, the cue line is left implicit in the notation of the main text.

Answer prompt. To evaluate true accuracy and output consistency, the options are followed by the cue `"\nAnswer : Option ("`, so that the next token the model produces is the letter of its chosen option. For example, a MATH_4 item is formatted as:

```
Question :What is 48213 * 87?
(A) 4134231
(B) 4194531
(C) 4194831
(D) 4164531
Answer : Option (
```

Reading out the answer distribution. We do not sample answers. We rather run a single forward pass on the prompt and read the next-token distribution $p_\theta(\cdot \mid s^{\mathrm{q}} s^{\mathrm{opt}})$ at the last position. Prompts are batched with left padding, so that the last position always corresponds to the final prompt token. Tokenizers may encode an option letter in several ways depending on the surrounding whitespace and punctuation. As such, the probability of option j is the sum of the probabilities of all tokens in the vocabulary that could correspond to letter L_j:

$$p_\theta(s_j^{\mathrm{opt}} \mid s^{\mathrm{q}} s^{\mathrm{opt}}) = \sum_{t \in V(L_j)} p_\theta(t \mid s^{\mathrm{q}} s^{\mathrm{opt}}), \tag{10}$$

where $V(L_j)$ contains the tokens among `"`L_j`"`, `"`L_j `"`, `"`L_j`)"`, `"`L_j`) "`, `" `L_j`"`, `" `L_j `"`, `" `L_j`)"` and `" `L_j`) "` that exist in the model's vocabulary and could represent the same L_j answer. The accuracy of the model on an item (Section 2.3) is the probability assigned in this way to the letter of the correct option. It is thus a soft measure of correctness: the expected accuracy of an answer sampled from the model, rather than a binary score.

Confidence prompt. To elicit verbalized confidence, the question is preceded by a pre-prompt s^{pre} that asks the model to estimate its probability of answering correctly and specifies the expected answer format:

```
You are going to be asked to guess your likelihood
to be right on a given question.  Give your answer
with the format 'Accuracy prediction:  X%' but replace
X with your percentage chance to be right on this
question.\n
```

The options are then followed by the cue `"\nAccuracy prediction:"` instead of the answer cue. The model is never asked to answer the question in this prompt. The same MATH_4 item thus becomes:

```
You are going to be asked to guess your likelihood to be right on a
given question. Give your answer with the format 'Accuracy prediction:
X%' but replace X with your percentage chance to be right on this
question.
Question :What is 48213 * 87?
(A) 4134231
(B) 4194531
```

```
(C) 4194831
(D) 4164531
Accuracy prediction:
```

(Line breaks inside the instruction are added here for display only.) We generate exactly 5 tokens with greedy decoding. The completion is then searched for numbers immediately followed by a percent sign (regular expression `(\d+(?:\.\d+)?)%`). If exactly one such number is found, it is taken as the model's confidence, rescaled from percent to $[0, 1]$. Otherwise, the response is considered invalid and has no confidence estimate. Verbalized confidence is computed on the test split of each dataset, on the same items as those used to evaluate the trained models.

C DATASET SPLITTING

Each dataset variant is split into a training set and a test set by assigning entire categories to one side or the other, never individual questions. We pool all items of the original dataset test split before partitioning. The split should satisfy two goals: (i) the training set should contain about 90% of the items, and (ii) every model should have a similar accuracy on both splits. The second goal ensures that differences in metacognitive performance between train and test cannot be explained by a difference in task difficulty. The splits should form a partition of the categories and not mix items from similar categories across splits which could create a form of implicit form of contamination from training to test set (except for MATH which has implicitly similar categories). In addition, these constraint have to be respected for all models at once at best for results across models to be comparable.

This appendix will describe how this procedure is done through the optimization of an objective function described below.

Objective. Let $\mathcal{C}$ be the set of categories of a dataset, N_c the number of items in category c, and $N = \sum_{c \in \mathcal{C}} N_c$ the total number of items. A split is defined by the set of test categories $\mathcal{T} \subset \mathcal{C}$. The remaining categories $\mathcal{C} \setminus \mathcal{T}$ form the training set. The fraction of training items is

$$f_{\text{train}}(\mathcal{T}) = 1 - \frac{1}{N} \sum_{c \in \mathcal{T}} N_c. \tag{11}$$

For a model $\mathcal{M}_\theta$ and a set of categories $\mathcal{S}$, we write $Acc(\mathcal{M}_\theta, \mathcal{S})$ for the average accuracy (Section 2.3) over all items in these categories, so that categories contribute in proportion to their size. We select the split that minimizes

$$\mathcal{L}(\mathcal{T}) = |f_{\text{train}}(\mathcal{T}) - 0.9| + \lambda \frac{1}{|\mathcal{M}|} \sum_{\mathcal{M}_\theta \in \mathcal{M}} |Acc(\mathcal{M}_\theta, \mathcal{C} \setminus \mathcal{T}) - Acc(\mathcal{M}_\theta, \mathcal{T})|, \tag{12}$$

where $\mathcal{M}$ is the set of 10 models studied in this paper and $\lambda = 1$ (arbitrary choice).

Search procedure. Since the number of possible partitions grows exponentially with the number of categories, we minimize $\mathcal{L}$ by random search over 10 000 candidate splits (random seed 42) and keep the split with the lowest loss. To concentrate the search around the target 90/10 ratio, each candidate is generated as follows:

1. Draw a target test fraction $\tau \sim \mathcal{U}(0.05, 0.15)$.
2. Shuffle the categories uniformly at random.
3. Add categories to $\mathcal{T}$ in the shuffled order until their cumulative share of items reaches τ.

Candidates in which either split is empty are discarded.

This procedure makes it possible to rather efficiently test various splits which could satisfy the aforementioned objective.

Resulting splits. Table 2 lists the selected categories, and Table 3 reports the resulting number of items and the accuracy of each model on both splits. For MATH_4 and MATH_{10}, the search was run independently on each variant and returned the same split: the category easy$*$ is held out, which contains exactly 10% of the items. For MMLU-PRO, the two searches returned different splits: {computer science, other} for MMLU-PRO_4 and {computer science, history, philosophy} for MMLU-PRO_{10}. To make the two variants differ only in their number of options, we applied the MMLU-PRO_{10} split to MMLU-PRO_4 as well, which increases the loss on MMLU-PRO_4 only slightly, from 0.0361 to 0.0368 and moved the difference of accuracy between train and test by around 1%.

After optimization, the training fraction is 89.5% and 89.6% for MMLU-PRO_4 and MMLU-PRO_{10}, and 90.0% for the other variants. The per-model absolute train–test accuracy gap, averaged over models (second term of $\mathcal{L}$), is 3.2, 2.9, 0.7, 4.4 and 3.3 points on MMLU-PRO_4, MMLU-PRO_{10}, MEDMCQA_4, MATH_4 and MATH_{10}, respectively. The largest individual gaps occur for the two

Dataset	Split	Categories
MMLU-PRO	train (11)	biology, business, chemistry, economics, engineering, health, law, math, other, physics, psychology
	test (3)	computer science, history, philosophy
MEDMCQA	train (18)	anaesthesia, anatomy, biochemistry, dental, ENT, forensic medicine, gynaecology & obstetrics, medicine, microbiology, orthopaedics, pathology, pediatrics, pharmacology, psychiatry, radiology, social & preventive medicine, surgery, unknown
	test (3)	ophthalmology, physiology, skin
MATH	train (8)	easy−, easy+, medium−, medium∗, medium+, hard−, hard∗, hard+
	test (1)	easy∗

Table 2: Category-level train/test split of each dataset, selected by the procedure of Appendix C. The 4- and 10-option variants of MMLU-PRO and MATH share the same split. Counts in parentheses are the number of categories per split.

Model	MMLU-PRO$_4$		MMLU-PRO$_{10}$		MEDMCQA$_4$		MATH$_4$		MATH$_{10}$	
	train	test	train	test	train	test	train	test	train	test
#items ($N_{\mathcal{D}}$)	10 685 / 1 256		10 337 / 1 202		108 642 / 12 123		108 000 / 12 000		108 000 / 12 000	
Llama-2-7b-chat	35.5	33.1	18.3	17.3	37.8	36.0	31.8	32.2	16.3	19.2
Llama-3.2-3B-Instruct	42.1	41.8	25.6	25.5	66.9	65.6	34.1	28.6	13.7	12.8
Mistral-7B-Instruct-v0.3	46.1	48.1	28.3	29.0	52.1	50.6	31.2	26.9	12.8	13.5
Ministral-3-3B-Instruct	46.3	47.2	31.1	31.9	49.7	49.8	37.1	32.5	15.8	16.1
Ministral-3-8B-Instruct	55.1	58.2	37.6	41.7	58.8	58.9	81.1	78.5	46.8	38.3
phi-4	61.1	67.9	46.6	54.1	68.7	68.6	52.7	52.9	25.5	28.4
Qwen2.5-7B	44.5	47.7	26.6	29.3	46.1	45.3	31.9	34.8	14.6	16.4
Qwen2.5-7B-Instruct	55.4	59.8	37.8	41.3	59.2	58.2	52.3	50.1	29.7	29.8
Qwen3.5-9B-Base	50.4	54.6	33.6	37.3	54.1	54.0	54.1	43.9	30.0	22.6
Qwen3.5-9B	52.2	56.5	33.9	38.6	57.0	56.6	55.9	44.9	29.1	21.8
Mean	48.9	51.5	31.9	34.6	55.0	54.3	46.2	42.5	23.4	21.9
$\lvert\Delta\rvert$ (train, test)	2.6		2.7		0.7		3.7		1.5	

Table 3: Accuracy (%) of the evaluated models on the train and test splits of each dataset variant. **Mean** averages over the 10 models, and $|\Delta|$ is the absolute difference between the train and test means. This is a looser measure than the per-model gap minimized by the splitting objective (Appendix C). #items is given as train / test. MMLU-PRO$_{10}$ contains fewer items than MMLU-PRO$_4$ because of the 500-token length limit: with 10 options, some questions exceed the limit that they satisfy with 4 options.

Qwen3.5-9B models on the MATH datasets (7 to 11 points). With only nine categories, MATH admits few splits close to the 90/10 ratio (because of the repartition of question in each difficulty), which limits how well accuracy can be balanced for every model.

D HYPERPARAMETERS

Hyperparameters used to train models are shown in Table 4.

Hyperparameter	**Value**
Optimizer	Paged AdamW 8bits
Learning rate	5×10^{-5}
LR schedule	Linear
Warmup ratio	0.1
Weight decay	10^{-2}
Batch size	16
Train sampling strategy	by length

Table 4: Hyperparameters used for calibration training. The same values are used for all models and datasets.

E OUTPUT CONSISTENCY TRACKING EMERGES EARLY IN TRAINING

We next investigate how the two signals emerge during training. Figure 5 shows, for each dataset, Δr and the correlation between trained confidence and output consistency over training, on both the training split and the test split of the training dataset. On all datasets, the correlation with output consistency peaks early in training and then remains stable. On MMLU-PRO and MEDMCQA, confidence initially tracks output consistency more than accuracy on both splits ($\Delta r < 0$). As training proceeds, Δr increases on the training split until confidence tracks true accuracy almost perfectly, whereas it remains negative on the test split. Models thus first learn to track output consistency, and only later learn the accuracy of individual training questions, a form of accuracy tracking that does not extend to the test split. On MATH, in contrast, Δr is negative only very briefly at the start of training, then quickly becomes positive on both splits, earlier than on the other datasets. We attribute this difference to the high similarity between the math questions, accuracy tracking may be learnt fast enough due to their low variance for accuracy tracking to dominate in only a few steps of training. Nevertheless, models trained on MATH still track output consistency on the other datasets (Figure 3 in Main text).

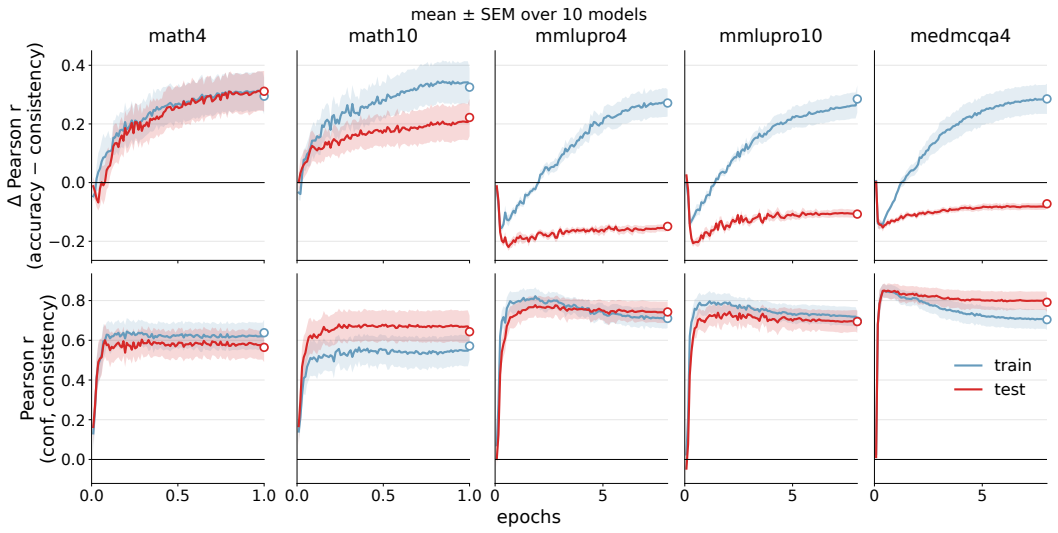

Figure 5: **Development of output consistency tracking during training.** Each column shows one training dataset. The top row represents Δr over training; positive values indicate that confidence tracks true accuracy more than output consistency, negative values the reverse. Bottom row represents Pearson correlation between trained confidence and output consistency. Curves are computed on the train and test splits of the training dataset. Shaded areas: s.e.m. across the 10 models. For this specific experiment models were trained on MedMCQA for 8 epochs instead of 1 to investigate the overfitting regime.

F MIDDLE PROBE

The confidence matching output consistency results could be linked to the fact that consistency is inherently encoded in the logits, and therefore we could find traces of it in the output of the transformer block which could contaminate the confidence probe trained at the end of the network. To verify whether our observations of output consistency tracking are due to this form of contamination we retrained all networks from scratch to replicate all the previous experiments this time putting the probe at the middle of the network instead of the end. By doubling the rank of LoRA adapters (from 8 to 16) we could conserve the same number of training parameters and kept the same hyperparameters for all experiments.

First, Figure 12 in Appendix H shows that LLMs all learnt to predict an accuracy much higher than the baseline verbalised accuracy. Nonetheless, this performance is still on average lower than with the probe at the end of the network ($r_{mid} = 0.55 \pm 0.04$ against $r_{end} = 0.73 \pm 0.03$ — s.e.m. computed on models variance).

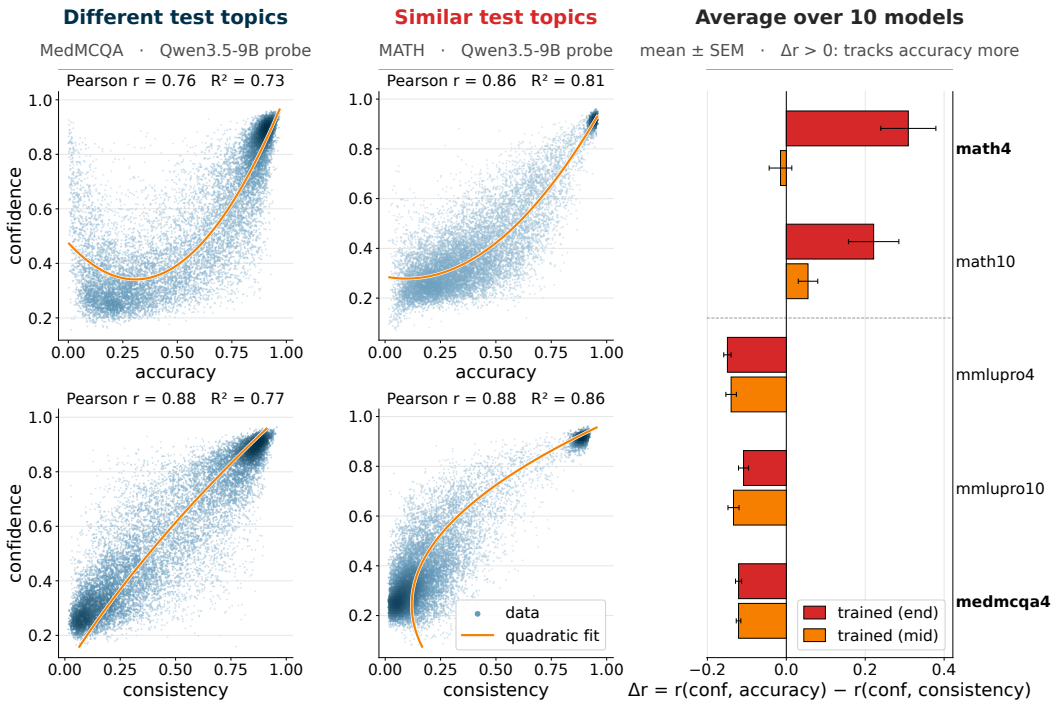

Figure 6: **Relationship between predicted accuracy, true accuracy and output consistency with middle probe.** Left panel shows the trained confidence of Qwen3.5-9B against true accuracy (top) and output consistency (bottom) on the test split of MEDMCQA$_4$ and MATH$_4$ with the probe located at the middle of the network. Orange lines are quadratic fits (R^2 shown above each plot). Right panel represents Δr on each test split for both middle and end probes; positive values (red) indicate that confidence tracks true accuracy more than output consistency, negative values (blue) the reverse. Error bars: s.e.m. across the 10 models (each averaged over 5 runs).

Then, looking at the true accuracy vs. trained confidence (mid probe) plots for Qwen3.5-9B (Figure 6 left column) we still noticed a very similar U-shape pattern on MEDMCQA$_4$ with a better linear relationship between trained confidence and output consistency ($r = 0.88$) than with true accuracy ($r = 0.76$). Therefore the output consistency tracking pattern remains. However, the other pattern, true accuracy matching greatly diminished ($r = 0.86$ for true accuracy tracking and $r = 0.88$ for consistency tracking) which is the opposite of what the leak theory would have predicted. Overall on MATH datasets models do not track true accuracy much better than output consistency ($\Delta r = 0.02 \pm 0.02$) but still predict a confidence closer to output consistency ($\Delta r = -0.13 \pm 0.01$ s.e.m. across models). Additionally, the output consistency prediction over

true accuracy prediction with the middle probe is quantitatively comparable with the end probe ($\Delta r = -0.13 \pm 0.01$ for middle and $\Delta r = -0.12 \pm 0.01$ for end probe).

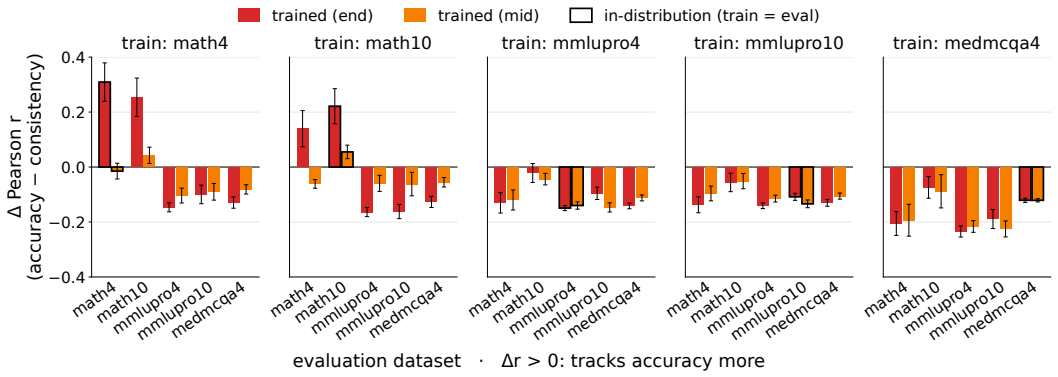

Figure 7: **Cross-dataset evaluation of trained confidence with the middle probe.** Each panel shows models trained on one dataset and evaluated on the test split of all five datasets. Bars represent Δr; positive values (red) indicate that confidence tracks true accuracy more than output consistency, negative values (blue) the reverse. In-distribution evaluations (tested on the test split from the training dataset - thematically distinct but closer with respect to the other datasets) are highlighted. Error bars: s.e.m. across the 10 models (each averaged over 5 runs).

When testing models trained on a specific dataset on all the others, consistency tracking results remain consistent: true accuracy prediction on MATH datasets falls by a large margin ($\Delta r = 0.01 \pm 0.02$ s.e.m. across models) but they keep generalising output consistency on the other datasets ($\Delta r = -0.07 \pm 0.02$). On the other hand, the models trained on variants of MMLU-PRO and MEDMCQA show a similar Δr than with the end probe ($\Delta r = -0.13 \pm 0.02$ for middle probe vs. $\Delta r = -0.12 \pm 0.02$ for end probe — see Figure 7).

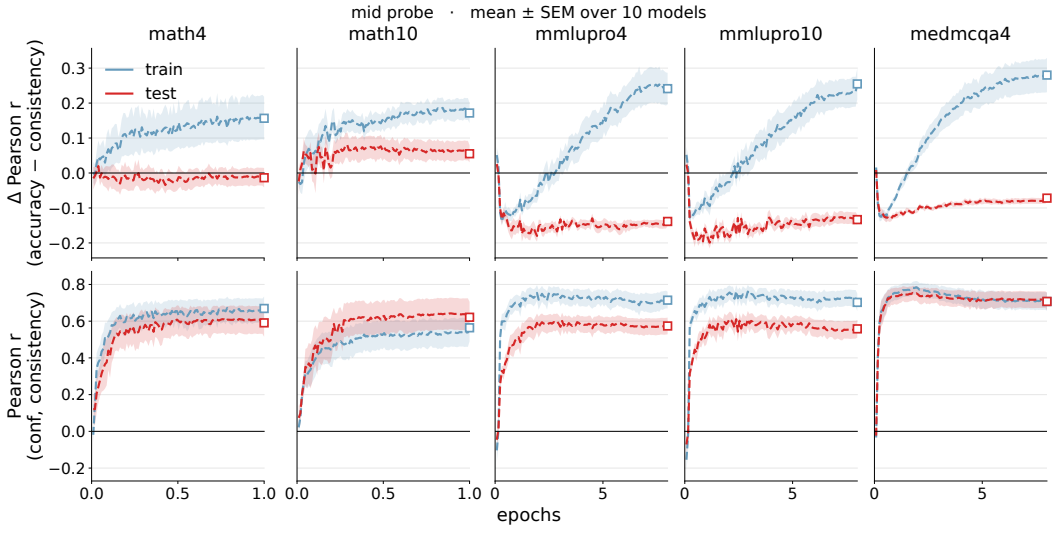

Figure 8: **Development of output consistency tracking during training with the middle probe.** Each column shows one training dataset. The top row represents Δr over training; positive values indicate that confidence tracks true accuracy more than output consistency, negative values the reverse. Bottom row represents Pearson correlation between trained confidence and output consistency. Curves are computed on the train and test splits of the training dataset. Shaded areas: s.e.m. across the 10 models.

The acquisition of these capabilities remain very similar than with the end probe: models learn output consistency tracking very early in training in MMLU-PRO and MEDMCQA dataset vari-

ants but shift faster on MATH datasets yet do not predict true accuracy much better than output consistency (see Figure 8).

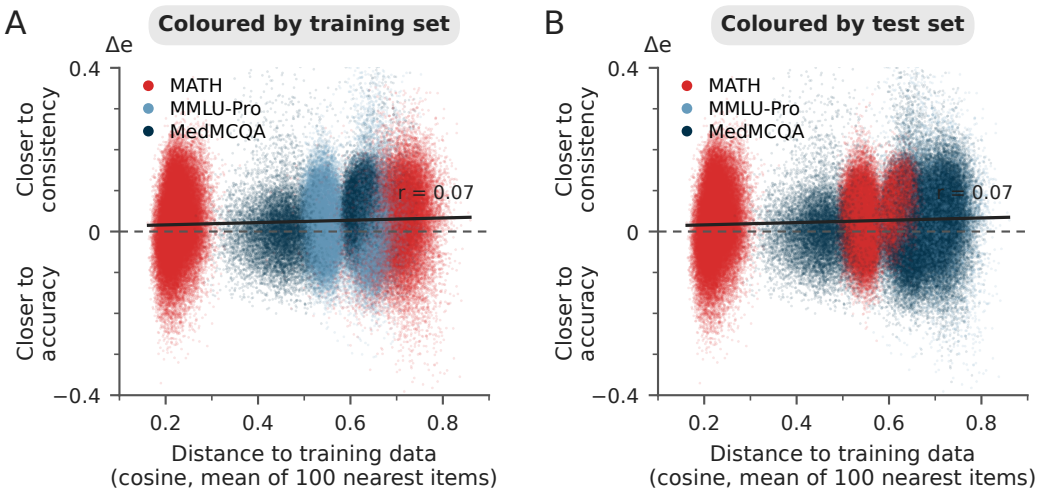

Figure 9: **Distance to the training data and the signal tracked by trained confidence with the mid probe.** Each point is one test question for one training dataset; its y-value is Δe computed from accuracy, consistency and confidence averaged over 10 models $\times$ 5 training runs. Negative values mean confidence is closer to true accuracy, positive values closer to output consistency. Points are colored by training dataset (A) or test dataset (B). The x-axis is the mean cosine distance to the 100 nearest training questions. Black line: linear fit ($r = 0.07$); dashed: $\Delta e = 0$; 0.10% of points beyond ± 0.4 not shown.

Lastly, because of this decrease in true accuracy on the MATH variants, in Figure 9 distance is no longer a good indicator of Δe ($r = 0.07$ for mid probe against $r = 0.49$ for end probe), the relative prediction error between accuracy over consistency ($\Delta e > 0$ means the model predicts consistency better than accuracy and vice versa). Indeed, with the middle probe, confidence seems to generally always be predictive of output consistency above true accuracy.

These results highlight the fact that this output consistency prediction does not arise from a leaking effect from the end of the transformer network but is far more ingrained in the LLM itself. Moreover, when using the middle probe, this time, it is the true accuracy tracking behavior that disappeared from our experiments suggesting this other form of metacognition could emerge from the latter parts of the network. This paper does not investigate in details where both of these metacognitive signals emerge from but suggest output consistency information arises early in training while this second form of metacognition may requires specific information only available in the later parts of the network.

G TRAINING CURVES

This section presents the training loss curves in Figure 10.

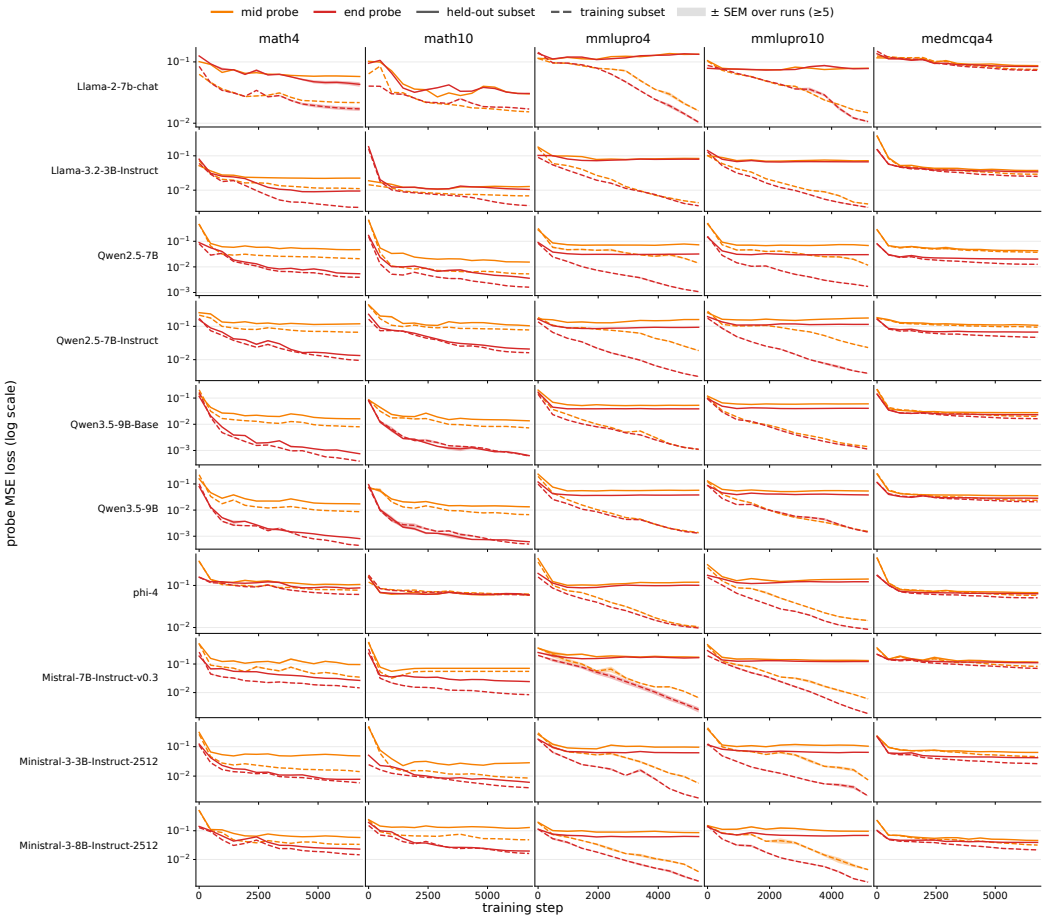

Figure 10: **Loss during training for both end and mid probes, train and test sets.**

H INDIVIDUAL MODEL PLOTS

H.1 TRAINING PLOTS

This section contains figures showing individual models behaviors. Figure 11 shows correlations between verbalised confidence and trained confidence (end probe) with true accuracy. Figure 12 also include the middle probe alongside the others. Figures 13 14 15 16 show respectively the scatter plot for trained confidence vs. accuracy or consistency for both end and mid probes. Figure 17 represents all the correlations between true accuracy and predicted accuracy (left column is end probe, right is mid probe) for models trained on the train set of a datasets and tested on all the other datasets' test set. Figure 18 shows the same type of representations but for correlation between output consistency and predicted accuracy and finally Figure 19 represents Δr for each individual model in the same settings.

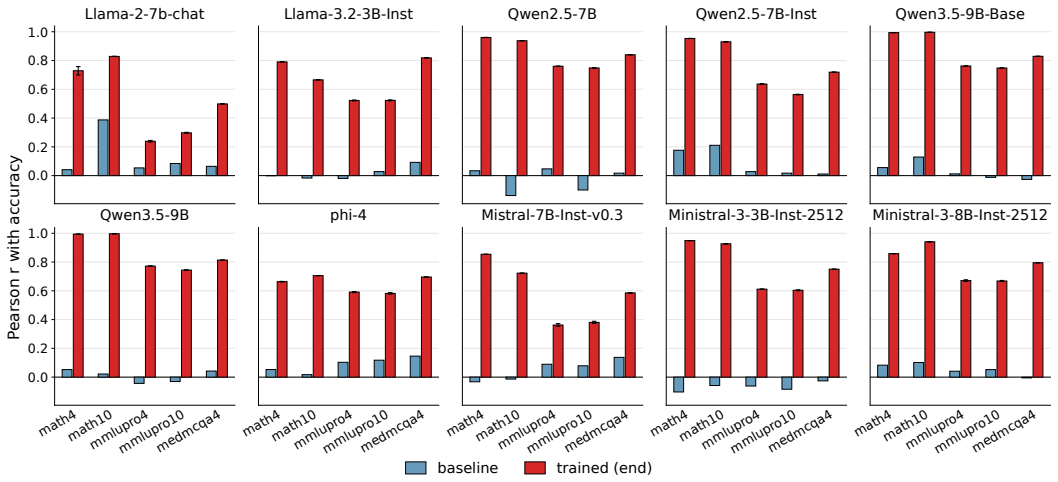

Figure 11: **Confidence performances before and after training on test splits for each dataset.** Bars represent the pearson r between accuracy and verbalised predicted accuracy (blue) or trained predicted accuracy (red) on each of the 5 datasets and the 10 models. Each bar is the average of 5 runs on the same dataset x model and errorbars represent the stderr between these trainings.

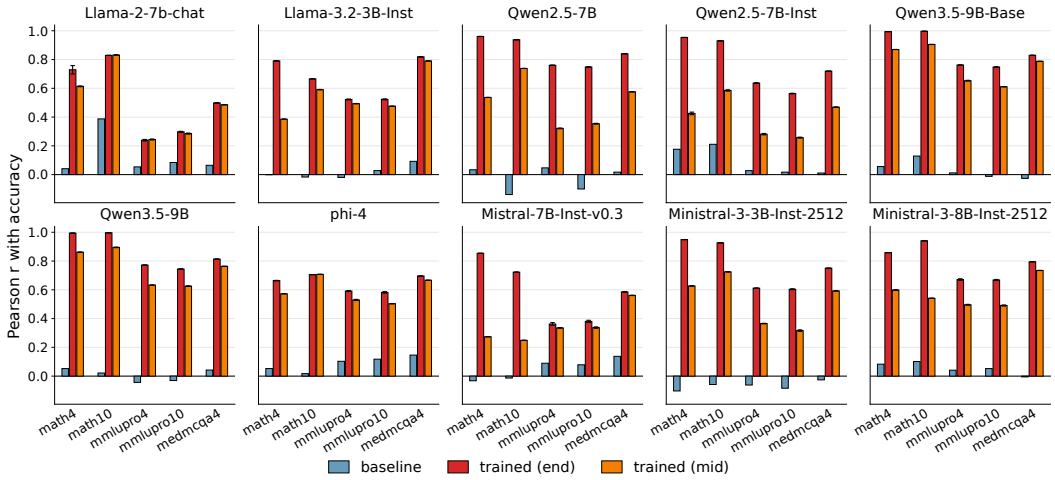

Figure 12: **Confidence performances before and after training on test splits for each dataset on both probes.** Bars represent the pearson r between accuracy and verbalised predicted accuracy (blue), trained predicted accuracy on the end probe (red) or trained predicted accuracy on the mid probe (orange) on each of the 5 datasets and the 10 models. Each bar is the average of 5 runs on the same dataset x model and errorbars represent the stderr between these trainings.

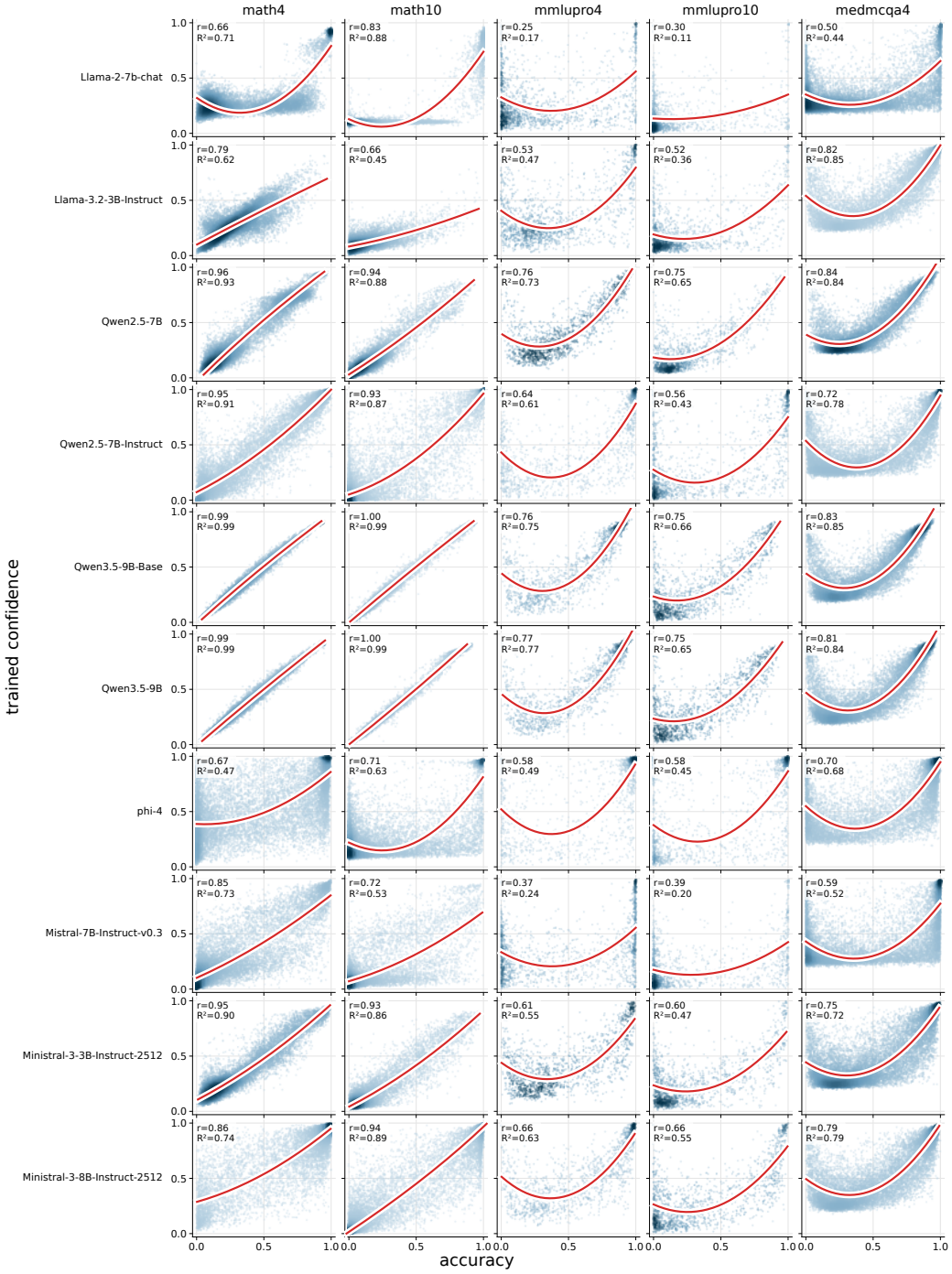

Figure 13: **Trained confidence vs. accuracy for all models and datasets with the end probe.** X axis is the datasets, Y axis the model. Line is a 2nd order polynomial fitted on the data, R^2 its fit coefficient and r is the Pearson correlation coefficient.

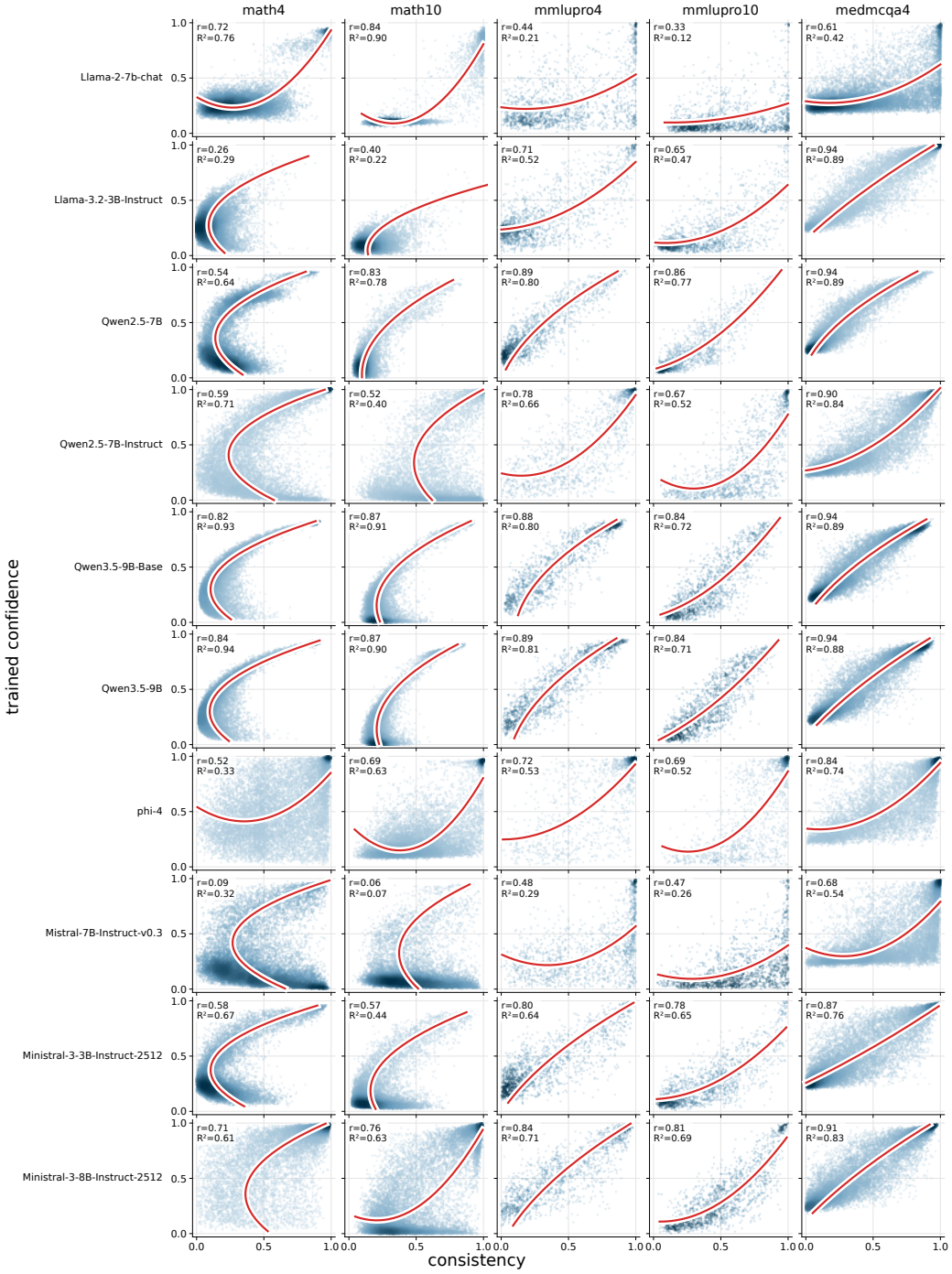

Figure 14: **Trained confidence vs. consistency for all models and datasets with the end probe.** X axis is the datasets, Y axis the model. Line is a 2nd order polynomial fitted on the data, R^2 its fit coefficient and r is the Pearson correlation coefficient.

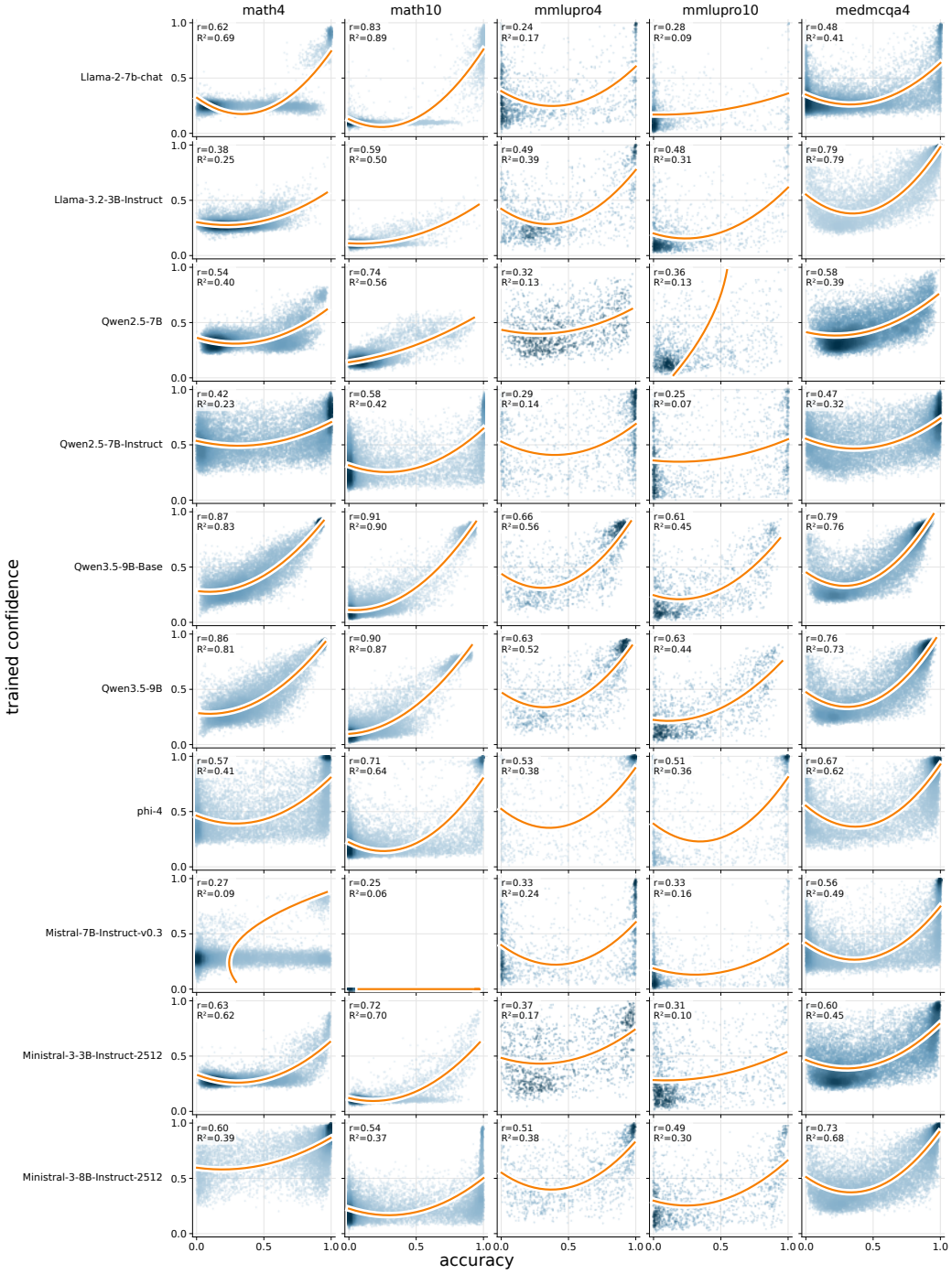

Figure 15: **Trained confidence vs. accuracy for all models and datasets with the mid probe.** X axis is the datasets, Y axis the model. Line is a 2nd order polynomial fitted on the data, R^2 its fit coefficient and r is the Pearson correlation coefficient.

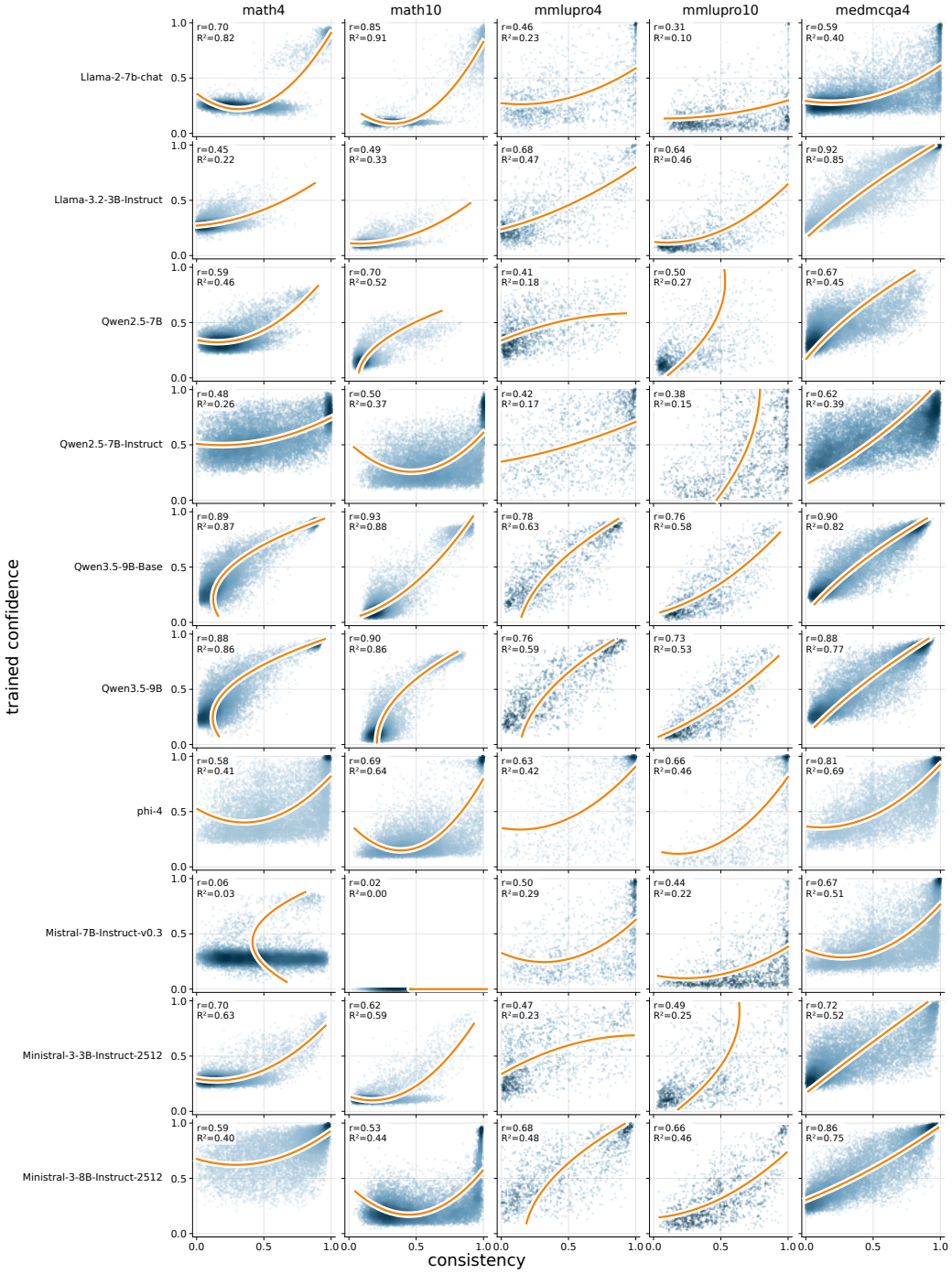

Figure 16: **Trained confidence vs. consistency for all models and datasets with the mid probe.** X axis is the datasets, Y axis the model. Line is a 2nd order polynomial fitted on the data, R^2 its fit coefficient and r is the Pearson correlation coefficient.

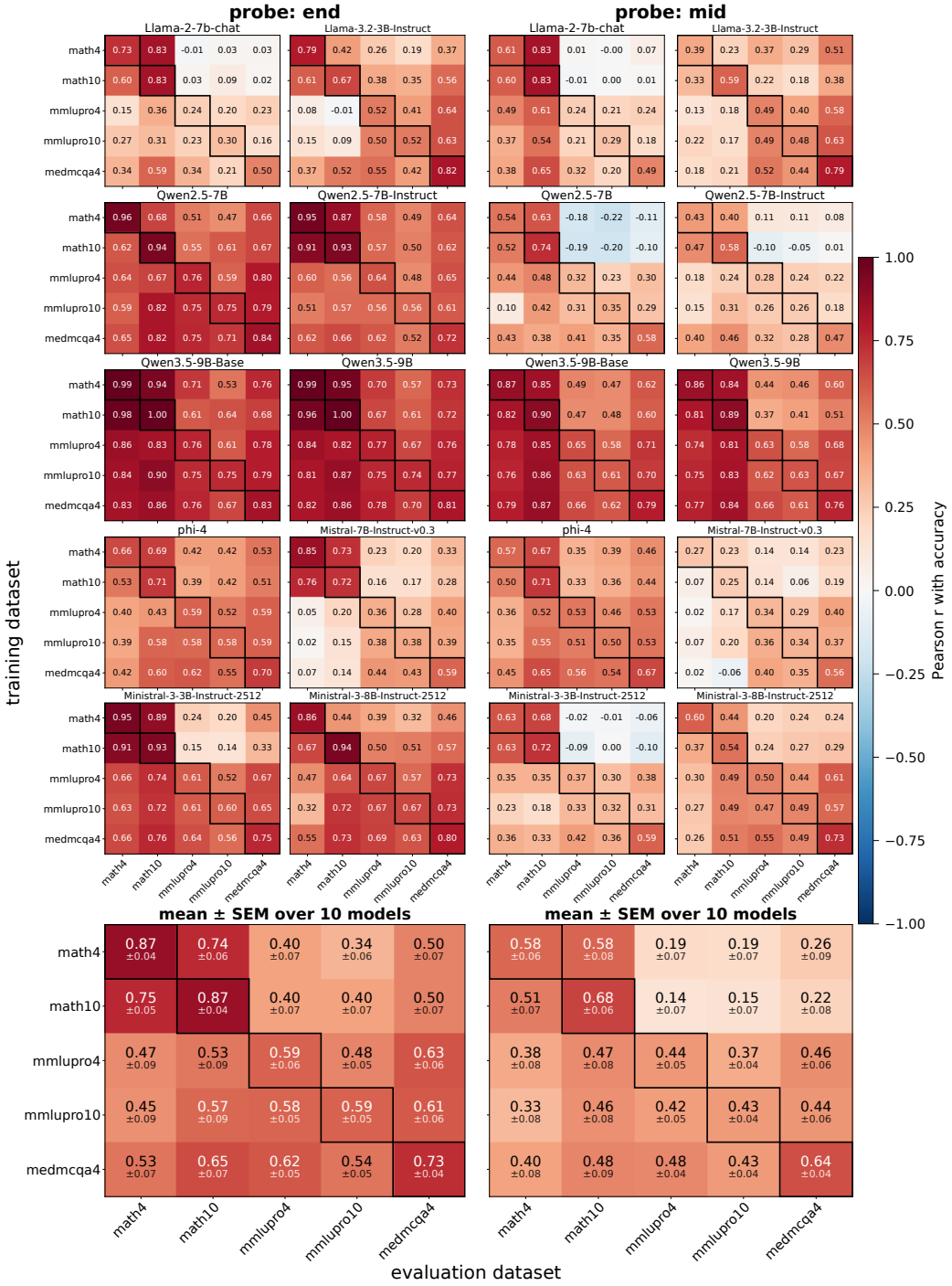

Figure 17: **Cross dataset true accuracy correlation with trained confidence across both probes.** Left column represents end probe results, right column, mid probe results and bottom plots show average + s.e.m. on all models together.

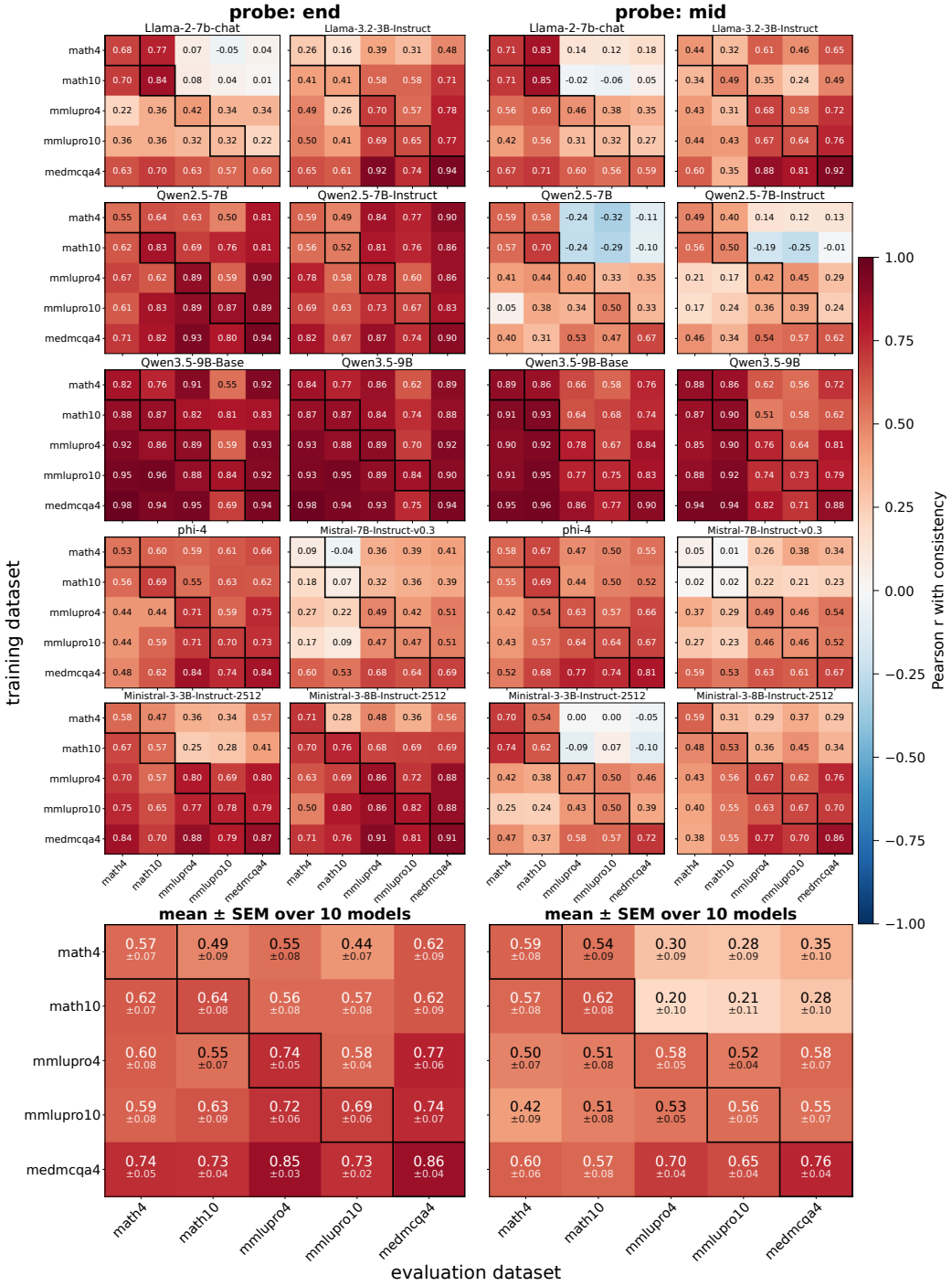

Figure 18: **Cross dataset output consistency correlation with trained confidence across both probes.** Left column represents end probe results, right column, mid probe results and bottom plots show average + s.e.m. on all models together.

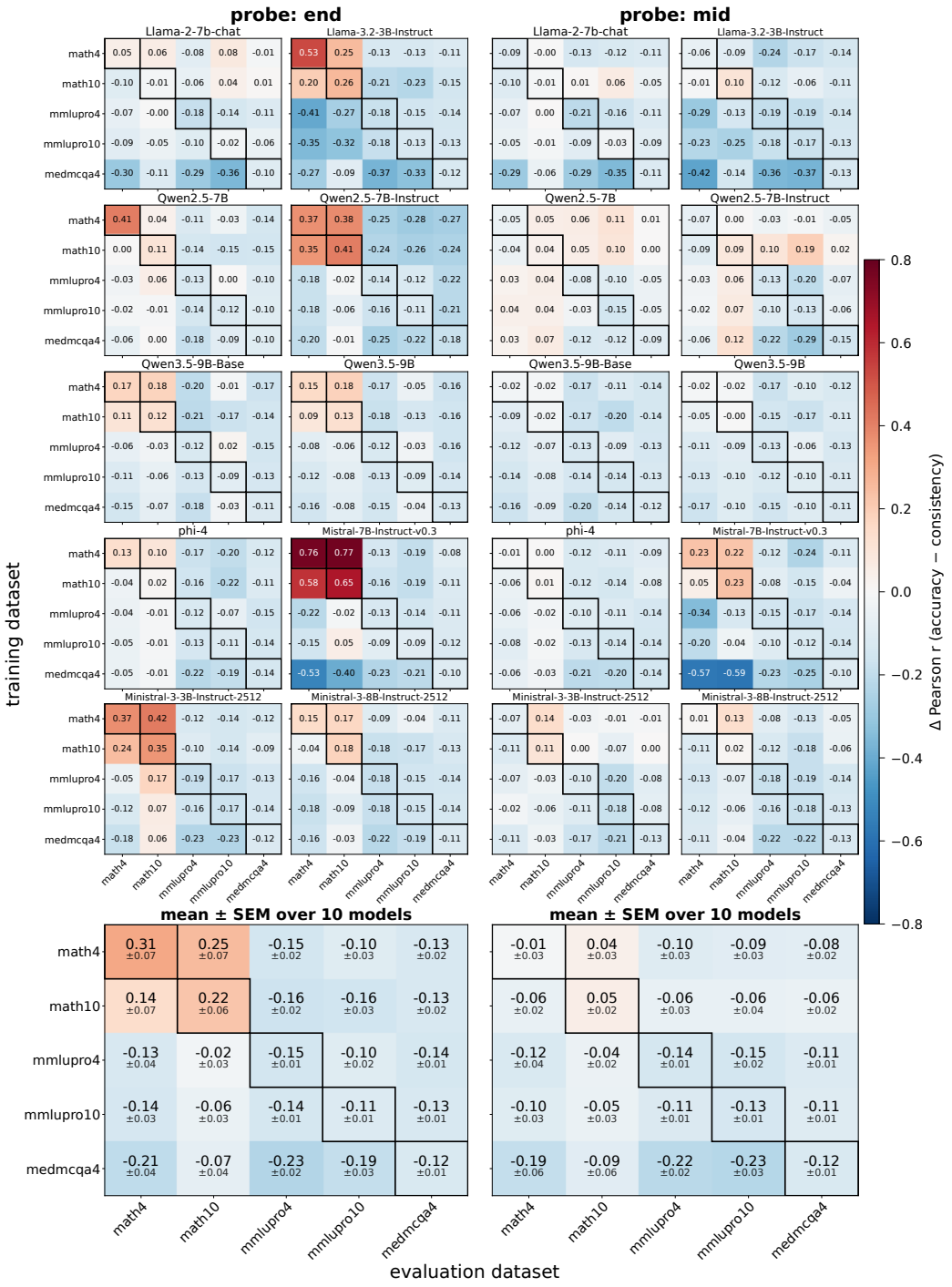

Figure 19: **Cross dataset Δr across both probes.** Left column represents end probe results, right column, mid probe results and bottom plots show average + s.e.m. on all models together.