CONTENTS

A APPENDIX

A.1 TRAINING LOSS OF LLMS AND VLMS

To adapt a pretrained LLM or VLM to a specific domain or task, models are typically trained on a supervised dataset $\mathcal{D} = (\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^n$ of input-output pairs. Training is commonly performed using the standard teacher-forcing objective, which minimizes the negative log-likelihood of the target sequence:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \ln \pi_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i) - \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{|\boldsymbol{y}_i|} \ln \pi_\theta(y_{i,k}|\boldsymbol{x}_i, \boldsymbol{y}_{i,<k}).$$

This objective maximizes the likelihood that the model generates the correct output sequence conditioned on the input and the ground-truth prefix at each step. The parameters are updated using gradient descent or its variants:

$$\theta \leftarrow \theta - \eta\, \nabla_\theta\, \mathcal{L}_{\mathrm{SFT}}(\theta), \quad \text{with} \quad \theta_{t=0} = \theta_0,$$

where $\eta > 0$ is the learning rate. Teacher forcing stabilizes fine-tuning by supplying the true prefix $\boldsymbol{y}_{<k}$ during training, enabling the model to align its predictions closely with the target data distribution in the new domain.

A.2 PROOF OF THEOREM 1

In this section, we give the detailed proof of our Theorem 1, we start by proving the following theorem:

Theorem 2. *For a data $\boldsymbol{x}_v$ and its generation $\boldsymbol{y}_v$ that await valuation, at any time $t \geq 0$ of training using a training data $(\boldsymbol{x}_i, \boldsymbol{y}_i), i \in [n]$, the training data exhibits larger value to the valuation data as the following increases:*

$$\sum_{k=1}^{|\boldsymbol{y}_v|}\sum_{k'=1}^{|\boldsymbol{y}_i|} \alpha_{k,k'}(t) \cdot \left\langle \mathbf{h}_{\boldsymbol{x}_v, \boldsymbol{y}_{v,<k}}(t), \mathbf{h}_{\boldsymbol{x}_i, \boldsymbol{y}_{i,<k'}}(t) \right\rangle +$$
$$\sum_{k=1}^{|\boldsymbol{y}_v|} \left\langle \boldsymbol{w}_{\boldsymbol{y}_{v,k}}(t) - \sum_{z\in\mathcal{V}} \pi_{\theta(t)}(z|\boldsymbol{x}_v) \cdot \boldsymbol{w}_z(t), (\boldsymbol{w}_{\boldsymbol{y}_{i,k}} - \sum_{z\in\mathcal{V}} \pi_{\theta(t)}(z|\boldsymbol{x}_v) \cdot \boldsymbol{w}_z(t)) \right\rangle \tag{3}$$

Proof.

$$\begin{aligned}
\frac{d}{dt} \ln \pi_{\theta(t)}(\mathbf{y}_v|\mathbf{x}_v) &= \left\langle \nabla \ln \pi_{\theta(t)}(\mathbf{y}_v|\mathbf{x}_v), \frac{d}{dt}\theta(t) \right\rangle \\
&= \left\langle \nabla \ln \pi_{\theta(t)}(\mathbf{y}_v|\mathbf{x}_v), -\eta \nabla \mathcal{L}_D(\theta) \right\rangle \\
&= \left\langle \nabla \ln \pi_{\theta(t)}(\mathbf{y}_v|\mathbf{x}_v), \eta \sum_{i=1}^{n} \nabla \ln \pi_{\theta(t)}(y_i|\mathbf{x}_i) \right\rangle
\end{aligned}$$

As per the unconstrained features Assumption, the model's trainable parameters are

$$\theta = \left(\boldsymbol{W}, \mathbf{h}_{\boldsymbol{x}_v}, \left\{\mathbf{h}_{\boldsymbol{x}_v, \boldsymbol{y}_{v,<k}}\right\}_{k\in\{2,\ldots,|\boldsymbol{y}_v|\}}, \left\{\mathbf{h}_{\boldsymbol{x}_i, \boldsymbol{y}_{i,<k'}}\right\}_{i\in[n], k'\in\{1,\ldots,|\boldsymbol{y}_i|\}}\right).$$

Unfolding the gradients with respect to these parameters yields:

$$\frac{d}{dt}\ln \pi_{\theta(t)}(\boldsymbol{y}_v|\boldsymbol{x}_v) = \left\langle \nabla_{\boldsymbol{W}} \ln \pi_{\theta(t)}(\boldsymbol{y}_v|\boldsymbol{x}_v), \sum_{i}^{n} \nabla_{\boldsymbol{W}} \ln \pi_{\theta(t)}(\boldsymbol{y}_i|\boldsymbol{x}_i) \right\rangle$$
$$+ \sum_{k=1}^{|\boldsymbol{y}_v|} \underbrace{\left\langle \nabla_{\mathbf{h}_{\boldsymbol{x}_v,\boldsymbol{y}_{v,<k}}} \ln \pi_{\theta(t)}(\boldsymbol{y}_{v,k}|\boldsymbol{x}_v, \boldsymbol{y}_{v,<k}), \sum_{i'=1}^{n_k} \nabla_{\mathbf{h}_{\boldsymbol{x}_v,\boldsymbol{y}_{v,<k}}} \ln \pi_{\theta(t)}(\boldsymbol{y}_{i',k}|\boldsymbol{y}_{v,<k}) \right\rangle}_{\text{(II) Training data have the same } (\boldsymbol{x}_v, \boldsymbol{y}_{v,<k})}. \tag{4}$$

where n_k is the number of training data whose input and prediction before token k are the same as valuation data $(\boldsymbol{x}_v, \boldsymbol{y}_{v,<k})$. Since we have

$$\nabla_{\mathbf{W}} \ln \pi_{\theta(t)}(z|\mathbf{x}) = \left(\mathbf{e}_z - \sum_{z' \in \mathcal{V}} \pi_{\theta(t)}(z'|\mathbf{x}) \cdot \mathbf{e}_{z'} \right) \mathbf{h}_{\mathbf{x}}^{\top}(t),$$
$$\nabla_{\mathbf{h}_{\mathbf{x}}} \ln \pi_{\theta(t)}(z|\mathbf{x}) = \mathbf{W}_z(t) - \sum_{z' \in \mathcal{V}} \pi_{\theta(t)}(z'|\mathbf{x}) \cdot \mathbf{W}_{z'}(t).$$

Putting this back in (4) together with a few algebra steps, yields

$$\frac{d}{dt}\ln \pi_{\theta(t)}(\boldsymbol{y}_v|\boldsymbol{x}_v) = \text{(I)} + \text{(II)} \tag{5}$$

where:

$$\text{(I)} = \sum_{k=1}^{|\boldsymbol{y}_v|} \sum_{i=1}^{n} \sum_{k'=1}^{|\boldsymbol{y}_i|} \alpha_{k,k'}(t) \cdot \left\langle \mathbf{h}_{\boldsymbol{x}_v,\boldsymbol{y}_{v,<k}}(t), \mathbf{h}_{\boldsymbol{x}_i,\boldsymbol{y}_{i,<k'}}(t) \right\rangle \tag{6}$$

$$\text{(II)} = \sum_{k=1}^{|\boldsymbol{y}_v|} \left\langle \boldsymbol{w}_{\boldsymbol{y}_{v,k}}(t) - \sum_{z \in \mathcal{V}} \pi_{\theta(t)}(z|\boldsymbol{x}_v) \cdot \boldsymbol{w}_z(t), \sum_{i'=1}^{n_k} (\boldsymbol{w}_{\boldsymbol{y}_{i',k}} - \sum_{z \in \mathcal{V}} \pi_{\theta(t)}(z|\boldsymbol{x}_v) \cdot \boldsymbol{w}_z(t)) \right\rangle \tag{7}$$

where $\alpha_{k,k'}(t) = \left\langle \mathbf{e}_{\boldsymbol{y}_{v,k}} - \pi_{\theta(t)}(\cdot|\boldsymbol{x}, \boldsymbol{y}_{v,<k}), \mathbf{e}_{\boldsymbol{y}_{i,k'}} - \pi_{\theta(t)}(\cdot|\boldsymbol{x}, \boldsymbol{y}_{i,<k'}) \right\rangle$. By taking the i-th sample, we can obtain Theorem 2. □

We observe the following:

(1) When the training input $\boldsymbol{x}_i$ differs from the valuation input $\boldsymbol{x}_v$, its influence on the valuation target arises solely through Term (I), which captures the contribution of the token embeddings and all network parameters except the token unembedding layer.

(2) The effect of the token unembeddings is concentrated in cases where the training and valuation data share the same input $\boldsymbol{x}$ and exhibit overlapping output predictions $\boldsymbol{y}$.
To eliminate this dependence on token unembeddings, we impose the following assumption:

Assumption 2 (Distinct Input). *The training dataset satisfies that no training input $\boldsymbol{x}_i$ is identical to the valuation input $\boldsymbol{x}_v$.*

Under the Assumption 2, the contribution from token unembeddings (Term (II)) vanishes, so that the influence of the training data on the valuation data arises entirely through the shared representation features captured in Term (I). This assumption is mild, as training inputs typically differ from valuation inputs in practice — especially in vision-language datasets, where the input images are almost always distinct. Extending this result to cases where training examples share the same input but differ in their outputs $\boldsymbol{y}$ is straightforward: the output prefix $\boldsymbol{y}_{<k}$ can be incorporated into the input $\boldsymbol{x}$, treating each unique pair $(\boldsymbol{x}, \boldsymbol{y}_{<k})$ as a distinct input, where $k-1$ indicates the point at which the outputs begin to differ. Combining Theorem 2 and Assumption 2 then yields Theorem 1.

A.3 Additional Details of Influential and Mislabeled data detection

Training setting for baselines. While `For-Value` requires only a single forward pass, the influence function-based baselines `Hessian-free` and `DataInf` require fine-tuning the models to convergence. For text generation tasks, we follow the training setup in Kwon et al. (2024), except to llama-2-13B, we use float16 weights instead of 8-bit quantization. For image-to-text generation tasks, we apply LoRA to every query and value matrix within the model's attention layers. To fine-tune VLMs, we use a learning rate of 2×10^{-4}, LoRA hyperparameters $r = 8$ and $\alpha = 32$, float16 model weights, a batch size of 32, and train for 20 epochs.

Efficiency details. For larger 32B and 72B models in Fig. 4, we employ 4 A100 GPUs for inference and a single A100 for value computation. Baseline methods requiring training are fine-tuned on up to 8 GPUs, with the 32B model quantized to 8-bit to enable valuation on a single A100. Due to their long runtime, we restrict baselines to the sentence transformation task and, for 14B/32B models, sample 10% of valuation data—scaling time by a factor of 10 to estimate totals. Despite these adjustments, `For-Value` achieves substantially lower runtime without quantization and with fewer GPUs.

A.4 Additional Results

Complexity Analysis. Tab. 6 compares the training, computational, and memory costs of different methods. Traditional approaches such as IF, `Hessian-free`, `HyperINF`, and `DataInf` rely on gradient traces or Hessian computations, resulting in high costs that scale poorly with model size. In contrast, `Emb` and `For-Value` are training-free and algorithm-agnostic, which significantly reduces overhead. Although `HyperINF` is the strongest baseline in terms of accuracy, its cubic complexity makes it impractical for large LLMs—requiring about 6 hours for a Qwen-32B model (Fig. 4b). Although `Emb` achieves the best runtime efficiency, its performance lags behind other methods, as demonstrated in Tab. 1 and Tab. 2. Our method, `For-Value`, maintains strong performance while remaining highly efficient. Since $|\hat{\mathcal{V}}|$ is typically small (often under 2k), `For-Value` achieves much lower computational and memory costs than baselines.

Method	Training Free	Algorithm Agnostic	Training Complexity	Computational Complexity	Memory Complexity
Original IF	✗	-	$O(nEd_{in}dL)$	$O(nd_{in}^2 d^2 L + d_{in}^3 d^3 L)$	$O(D^2 L + nDL)$
`Hessian-free`	✗	✗	$O(nEd_{in}dL)$	$O(nd_{in}dL)$	$O(nd_{in}dL)$
`DataInf`	✗	✗	$O(nEd_{in}dL)$	$O(nd_{in}dL)$	$O(nd_{in}dL)$
`HyperINF`	✗	✗	$O(nEd_{in}dL)$	$O(nd^3 L)$	$O(nd^2 L)$
`Emb`	✓	✓	0	$O(nd)$	$O(nd)$
`For-Value` (ours)	✓	✓	0	$O(nd\|\hat{\mathcal{V}}\|)$	$O(nd\|\hat{\mathcal{V}}\|)$

Table 6: Comparison on complexity of the Influence Function (IF), `Hessian-free`, `DataInf`, `Emb`, and `For-Value`. Complexities are given assuming a multilayer perceptron (MLP) with L layers, each containing $d_{in} \times d$ neurons where d_{in} is input dimension and d is the output embedding dimension, trained for E epochs on n training samples. The parameter count is identical across layers ($D \in \mathbb{N}$), and the in-batch volcabulary size is $|\hat{\mathcal{V}}|$. Overall, `For-Value` achieves higher computational and memory efficiency than baseline methods.

Discussion on Parallel Computing: While previous studies focus on using a single GPU for fair comparison, we would like to highlight that `For-Value` can further improve efficiency through parallel computing with a large batch size, as it only requires forward calculations. In contrast, baseline methods require computing the gradient for each individual data sample, which restricts them to a batch size of one and makes scaling up challenging.

Qualitative Demonstration. Beyond quantitative results, we present qualitative examples identified by `For-Value`. Fig. 5 shows a target valuation sample alongside its most and least influential training samples as ranked by `For-Value`. Specifically, `For-Value` successfully identifies highly relevant training points — for example, selecting samples from the same reverse order of words task for sentence transformation, or matching the same subject or artistic style in image-to-text tasks. In contrast, the least influential samples are clearly less relevant and often differ entirely in task or content from the target valuation data.

A.5 Additional Details of Select Data for Finetuning

Mathematics: GSM8K As the baseline methods require LoRA, we begin with a one-epoch warmup training on Llama3-8B Meta (2024) using the whole training set to avoid utilizing gradients from randomly initialized LoRA modules (with a rank of $r = 32$). Next, we calculate influence scores

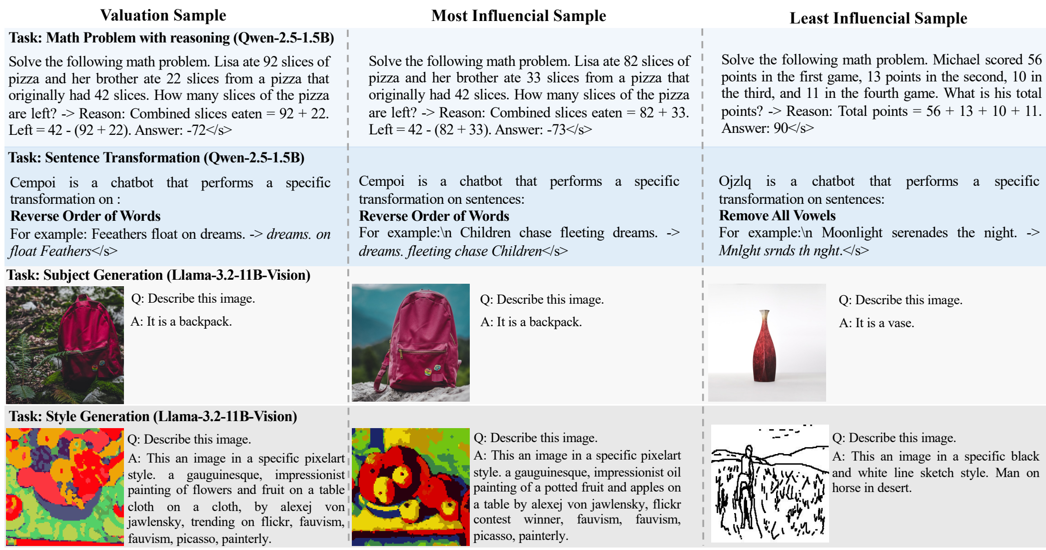

Figure 5: Qualitative examples of data influence identified by `For-Value`. For each target valuation sample (left column), the most influential (middle column) and least influential (right column) training samples are shown. `For-Value` correctly retrieves training samples that share relevant task characteristics (e.g., same reasoning type, sentence transformation rule, subject, or style) and filters out unrelated or mismatched examples.

for both the baselines and `For-Value`. To ensure consistency and performance, we also perform a one-epoch warm-up but with full-parameter finetuning on the entire dataset. Finally, we select the top 5% of data based on these influence scores to further finetune the model with learning rate $1e-5$ and batch size 64 on 4 H100 GPU for 4 epochs.
Medicine: Noise-Huatuo-Complex-CoT As the baseline methods utilize LoRA, we begin with a one-epoch training on Llama3-8B-Instruction Meta (2024) using the whole training set to avoid using gradients from randomly initialized LoRA modules (with a rank of $r = 16$). Next, we calculate influence scores for both the baselines and our approach. Considering the training data is noisy, we select the top 5% high value training data based on these scores and finetune the original pretrained model using full-parameter finetuning for 5 epochs, with a learning rate of 1×10^{-6}, a batch size of 16 and gradient accumulation 8 on 8 H100 GPUs. We follow Wu et al. (2025) using greedy decoding to evaluate the model on 5 held out datasets MedQA Jin et al. (2021), MedMCQA Pal et al. (2022), PubMedQA Jin et al. (2019), MMLU-Pro-Med Wang et al. (2024b), GPQA-Med Rein et al. (2024).

Medicine: Noise-Huatuo-Complex-CoT Similarly, we start with a one-epoch warm-up on the entire training set to prevent using gradients from randomly initialized LoRA modules (with a rank of $r = 16$). Then, we compute influence scores for the baseline methods. For our method, since the pretrained model already demonstrates sufficient medical knowledge (as shown by adequate test accuracy in Table 2), we directly use the original pretrained model to assess data value. Finally, we finetune the pretrained Qwen2.5-3B-VL model Bai et al. (2025a) with full-parameter finetuning for 3 epochs, using a learning rate of 1×10^{-5}, a batch size of 16, and gradient accumulation of 8 on 8 H100 GPUs. We evaluate the model with greedy decoding on 6 held out datasets: PMC Zhang et al. (2023), MMMU Yue et al. (2024), MedX-M Zuo et al. (2025), PathVQA He et al. (2020), SLAKE Liu et al. (2021), VQA-Rad Lau et al. (2018).

A.6 Additional Analysis on Select Data for Finetuning

Medicine: Noise-Huatuo-Complex-CoT. As indicated in Tab. 4, baseline methods struggle to effectively select high-quality data from noisy training datasets. This is primarily because these methods rely on assumptions of uniqueness or convergence to an optimal solution Bae et al. (2024), which are difficult to satisfy in the presence of noisy data. To illustrate this, we evaluated the proportion of high-quality data within the top 10% of high-value data, as shown in Tab. 7. The results reveal

Llama-3.1-8B	Detection Accuracy
`Hessian-free`	48.2
`HyperINF`	15.1
`DataInf`	33.2
`For-Value`	84.4

Table 7: High quality data detection accuracy

that baseline methods generally lack the capability to accurately identify noisy data, whereas our proposed method (`For-Value`) achieves significantly higher accuracy in detecting clean data.

Table 8: Description of the sentence transformation task templates. We consider 10 different types of sentence transformations. For each sentence transformation, unique identifying "chatbot" names were additionally prepended to the task prompt to assist the model in training.

Sentence transformations	**Example transformation of "Sunrises herald hopeful tomorrows":**
Reverse Order of Words	tomorrows. hopeful herald Sunrises
Capitalize Every Other Letter	sUnRiSeS hErAlD hOpEfUl tOmOrRoWs.
Insert Number 1 Between Every Word	Sunrises 1herald 1hopeful 1tomorrows.
Replace Vowels with *	S*nr*s*s h*r*ld h*p*f*l t*m*rr*ws.
Double Every Consonant	SSunrriisseess hheraldd hhopefull ttomorrows.
Capitalize Every Word	Sunrises Herald Hopeful Tomorrows.
Remove All Vowels	Snrss hrld hpfl tmrrws.
Add 'ly' To End of Each Word	Sunrisesly heraldly hopefully tomorrows.ly
Remove All Consonants	uie ea oeu ooo.
Repeat Each Word Twice	Sunrises Sunrises herald herald hopeful hopeful tomorrows. tomorrows.

A.7 Detailed Task Description

A.7.1 LLM Influence Evaluation Tasks

Following (Kwon et al., 2024), we evaluate the performance of `For-Value` on three text generation tasks for large language models (LLMs) to identify influential data points:

- **Sentence Transformations:** This task requires transforming input sentences into alternative forms while preserving meaning (e.g., active to passive voice). The dataset comprises 10 distinct classes (e.g., declarative to interrogative), each with 100 examples, split into 90 training and 10 test examples per class. Data examples see Tab. 8.
- **Math Word Problems (Without Reasoning):** These problems involve direct numerical computation from textual descriptions (e.g., basic arithmetic). The dataset has 10 classes based on operation types, with 100 examples per class (90 training, 10 test). Data examples see Tab. 9.
- **Math Word Problems (With Reasoning):** These require multi-step reasoning (e.g., solving word problems involving algebra or logic). Similar to the previous task, the dataset includes 10 classes with 100 examples each (90 training, 10). Data examples see Tab. 9.

A.7.2 VLM Influence Evaluation Tasks

For VLMs, we adapt text-to-image generation tasks from (Kwon et al., 2024) into image-to-text (captioning) tasks to evaluate influence:

- **Style Generation:** This task involves generating captions for images in specific styles: cartoons (Norod78, 2023), pixel art (Jainr3, 2023), and line sketches (Zoheb, 2023). Each style dataset contains 200 training and 50 test image-text pairs, totaling 600 training and 150 test samples across three styles. Data examples see Fig. 5.
- **Subject Generation:** Using the DreamBooth dataset (Ruiz et al., 2023), this task generates captions for images of 30 distinct subjects (e.g., specific objects or animals). Each subject provides 3 training samples, with the remaining samples used for valuation. Data examples see Fig. 5.

Table 9: Description of the math problem task templates. We consider 10 different types of math word problems.

Math Word Problems	**Template prompt question**
Remaining pizza slices	Lisa ate A slices of pizza and her brother ate B slices from a pizza that originally had C slices. How many slices of the pizza are left? Reason: Combined slices eaten = A + B. Left = C - (A + B).
Chaperones needed for trip	For every A students going on a field trip, there are B adults needed as chaperones. If C students are attending, how many adults are needed? Reason: Adults needed = (B * C) // A.
Total number after purchase	In an aquarium, there are A sharks and B dolphins. If they bought C more sharks, how many sharks would be there in total? Reason: Total sharks = A + C.
Total game points	Michael scored A points in the first game, B points in the second, C in the third, and D in the fourth game. What is his total points? Reason: Total points = A + B + C + D.
Total reading hours	Emily reads for A hours each day. How many hours does she read in total in B days? Reason: Total hours read = A * B.
Shirt cost after discount	A shirt costs A. There's a B-dollar off sale. How much does the shirt cost after the discount? Reason: Cost after discount = A - B.
Area of a garden	A rectangular garden has a length of A meters and a width of B meters. What is its area? Reason: Area = A * B.
Total savings	If Jake saves A each week, how much will he save after B weeks? Reason: Total savings = A * B.
Number of cupcake boxes	A bakery sells cupcakes in boxes of A. If they have B cupcakes, how many boxes can they fill? Reason: Boxes filled = B // A.
Interest earned	John invests A at an annual interest rate of B%. How much interest will he earn after C years? Reason: Interest = (A * B * C) // 100.

A.7.3 Influential Data Detection Metrics

We adopt two metrics from (Kwon et al., 2024) to assess influence:

- **AUC Score:** For each test data point, we assign pseudo labels to training points (1 if the training point's label matches the test point's, 0 otherwise). We compute the Area Under the Curve (AUC) between data values (influence scores) and pseudo labels, averaging across all test points. A higher AUC indicates better identification of influential points.
- **Recall:** For each test point, we calculate the percentage of influential training points (top-ranked by influence score) that share the same class as the test point. This measures the relevance of identified influential points.

A.7.4 Mislabeled Data Detection Data & Metrics

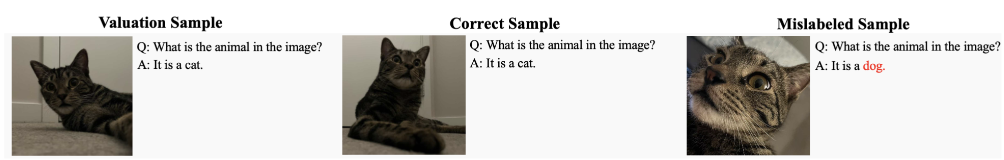

Figure 6: Description of the mislabeled data detection task. We utilize a cat versus dog classification dataset and intentionally introduce noise by randomly swapping the labels of 50% of the data.

For mislabeled detection, we transform the dataset into a visual-language question answering task with the template "What is the animal in the image? It is a [label]" with demonstration[3] in Fig. 6. We then select the first 400 images for both dogs and cats, flipping 50% of the labels to introduce noise. For valuation, we use 200 images, with each class containing 100 images. For evaluation, we also calculate the AUC and Recall but with the pseudo labels to training points being 1 if the training point's label matches the test point's and it is clean data, 0 otherwise.

A.8 Noise-Huatuo-CoT Data Example

We construct the Noise-Huatuo-Complex-CoT dataset by randomly sampling 5,000 examples without replacement and injecting noise into 40% of them through random insertion or deletion of irrelevant words, as illustrated in Fig. 7.

Random Drop

Question: A nucleic acid was analyzed and found to contain 32% adenine, 18% guanine, 17% cytosine and 33deg/o thymine. The nucleic acid must be: A. Single-stranded RNA B. Single-stranded DNA C. Double-stranded RNA D. Double stranded DNA

Alright, let's figure out what kind of nucleic acid we're with here. that's adenine, guanine, 17% cytosine, and 33% first things first, in double-stranded DNA, you'd expect adenine to pair with pair with cytosine. That means their should looking at is 32% while really close. Guanine is 18% and 17%. close. This of makes me think of double-stranded DNA because that's where see this kind of pairing. wait, not jump conclusions too quickly. Single-stranded DNA doesn't really care about matching percentages, so technically, it could have any mix of bases. be single-stranded DNA? It's possible, since there's no rule the strands. Now, out instead of at 33%. If it were RNA, we'd be seeing instead, So, it can't be neither We're definitely dealing with DNA here. Hmm, circling those matching percentages of A and T, G and really at a characteristic balance. show was a reason for such balance, which seems unlikely here without any it. considering all of this, the nucleic acid we're looking DNA. The complementary just too that seems to make the most sense given we've got. ## D. Double-stranded DNA. double-stranded Additionally, presence of thymine rather uracil rules out any RNA options.

Random Insert

Question: What physiological process in the collecting tubules is most likely to be decreased upon administration of a drug that blocks epithelial sodium channels in the cortical collecting duct?

Thinking bar bar Alright, let's think about these epithelial sodium bar foo bar bar channels. They're bar sitting there in the foo foo apical membrane of those epithelial cells – specifically in the cortical collecting duct foo baz foo bar foo – doing the job of pulling sodium foo bar bar foo bar foo bar bar from the bar tubular fluid back foo baz into bar the bloodstream. bar Super important foo for foo baz bar sodium balance, right? So, if we throw a wrench in that foo system by blocking these channels, what happens? Hmm, yeah, sodium can't baz baz bar get reabsorbed, that's for sure. Less sodium going back into the blood – that's foo baz foo significant! Now, let's consider the water side foo of things. ... of a drug that foo bar bar blocks epithelial sodium channels in baz the cortical collecting duct is sodium reabsorption. When these channels foo foo are inhibited, sodium cannot be reabsorbed from the tubular fluid back into the bloodstream. Consequently, this disruption also affects water bar foo bar reabsorption due to the osmotic relationship between bar baz sodium baz and water, and it can alter potassium handling, although foo the primary and most immediate effect is on baz baz sodium reabsorption.

Figure 7: Examples of two types of noisy data. (Left) Random word deletion, where tokens are dropped from the reasoning, for instance, 'Thinking' is removed after ##. (Right) Random word insertion, where irrelevant tokens such as 'bar,' 'foo,' and 'baz' are injected into the reasoning. Red dashes means omitted reasoning.

A.8.1 Baseline Checkpoints Selection

For baseline methods, we select the model checkpoint with the highest test AUC, as influence function-based methods exhibit significant performance variability across training checkpoints. Notably, this variability does not correlate with validation loss, posing challenges for practical deployment. We compare `For-Value` against these baselines to ensure robust evaluation.

A.8.2 Dataset Statistics

We present dataset statistics in Tab. 10

A.9 Usage of Large Language Model

In preparing this paper, we made limited use of ChatGPT to support writing and editing. Specifically, LLMs were employed for language polishing, grammar refinement, and rephrasing sentences to improve clarity and readability. Importantly, all technical content, including theoretical analysis, algorithm design, and experimental results, was conceived, implemented, and validated by the

[3] To prevent any licensing issues, the images shown are not from the original dataset; they were personally captured for demonstration purposes.

Table 10: Dataset statistics for LLM and VLM tasks.

Task	Training Samples	Valuation Samples
Sentence Transformations	900 (90 × 10 classes)	100 (10 × 10 classes)
Math Word Problems (No Reasoning)	900 (90 × 10 classes)	100 (10 × 10 classes)
Math Word Problems (With Reasoning)	900 (90 × 10 classes)	100 (10 × 10 classes)
Style Generation	600 (200 × 3 styles)	150 (50 × 3 styles)
Subject Generation	90 (3 × 30 subjects)	Variable (1-3) per subject
Mislabel Detection	800 (400 × 2 subjects 50% noise)	200 (100 × 2 subjects)
GSM8K	7470	1319
Noise-Huatuo-Complex-CoT	5000 (2981 clean, 2019 noise)	5000 (clean)
PMC-Reasoning (subset)	10000	5000

authors. LLM outputs were always critically reviewed, verified, and revised before inclusion. No LLM-generated text, figures, or tables were incorporated without careful human oversight.

A.10 License Clarification

The Dreambooth images have been either taken by the authors of the paper or obtained from Unsplash[4]. The file located at this link[5] includes a list of all reference links to the images on Unsplash, along with the photographers' attributions and the image licenses. The sketch images are sourced from FS-COCO Chowdhury et al. (2022). Data attributions and image licenses can be found in the file provided at the following link[6].

[4] https://www.unsplash.com/

[5] https://huggingface.co/datasets/google/dreambooth/blob/main/dataset/references_and_licenses.txt

[6] https://github.com/pinakinathc/fscoco