

HONEYBEE: Data Recipes for Vision-Language Reasoners

Supplementary Material

7. Related Work

Chain-of-Thought Reasoning Datasets. Prior work [27, 69] has shown that large language models (LLMs) can solve diverse reasoning tasks (e.g., math and logic) by generating chain-of-thoughts (CoTs) before providing their final answers at inference. This approach allows models to utilize additional computation at test time and further improve performance by aggregating decisions across several reasoning CoTs using methods such as self-consistency [66]. Notably, STaR [75] demonstrates that an LLM can be prompted to generate CoTs for novel questions, which can then be filtered based on final answer correctness. This synthetic process removes the need for expensive and time-consuming CoT collection from human annotators. Since then, several datasets [6, 29, 30, 45, 59, 60] have been proposed that bootstrap math CoT reasoning traces from strong LLMs [1, 19]. In particular, OpenThoughts [18] is a state-of-the-art CoT reasoning dataset, constructed through comprehensive ablation across several key data design decisions and ultimately scaled to 1.2M reasoning examples, predominantly centered on math problem-solving. In comparison, our work focuses on the more complex scenario of math reasoning in visual contexts, where vision-language (VL) models must integrate information from multiple modalities, image and text, instead of just text. Specifically, we create HONEYBEE, a high-quality VL CoT dataset that elucidates the data design process, including data sources, filtering strategies, and scaling properties at multiple model scales (e.g., 3B and 8B). Furthermore, we scale our dataset to 5M VL instances and observe consistent improvements with increased data size.

VL Reasoning Datasets. Math problem-solving serves as a critical testbed for assessing the reasoning capabilities of VL models, as it requires them to develop a deep understanding and interpretation skills across diverse visual categories such as geometry, functions, and charts [39, 65, 76]. To enable VL reasoning, prior work has proposed reasoning datasets, such as MathV360K [53], which selects and creates high-quality questions from a collection of existing image-question datasets. However, this dataset does not provide any CoT data, making it unsuitable for eliciting complex reasoning behaviors in VL models. Other datasets, such as MAVIS [77], focus on creating novel questions and CoTs for plane geometry and functions using an automatic data engine. However, such synthetic data is unsuitable for achieving strong performance on diverse real-world visual scenarios, such as tabular understanding, IQ tests, sci-

ence question-answering, and math reasoning on natural scenes. More recently, datasets such as LLaVA-CoT [71] and R1-OneVision [73] have sourced image-question data from diverse sources, including the web and textbooks, followed by CoT generation and multiple rounds of data curation from capable teacher VL models [22, 55]. However, it remains unclear whether the benefits of these datasets are attributable to differences in data sources, choice of teacher models, or specific data filtering strategies. In this work, we take a comprehensive approach by re-annotating the CoTs in each dataset with a given teacher model to understand the impact of diverse data sources, and we also study the effect of mixing the contexts in these datasets. Further, we curate a list of interventions (e.g., filtering, augmentation, and replacement) and test their impact on enhancing the performance of VL models of capacity. Subsequently, our approach examines the effect of scaling across several data dimensions, such as unique images, questions, and CoTs, which ultimately enables the development of HONEYBEE, one of the largest VL reasoning datasets to date.

VL Data Curation. Reasoning in visual contexts is a fundamental capability of modern AI systems [2, 11, 44, 48, 56, 57], enabling a wide range of real-world applications such as visual data analysis and scientific discovery. However, the training data for these systems are not publicly available, making it challenging to study the underlying science behind the development of state-of-the-art VL reasoning CoT datasets. In this work, we aim to investigate the science behind curating high-quality, large-scale VL reasoning datasets. To this end, we curate a list of several data interventions designed to improve the perception and reasoning capabilities of these models. Prior work [32] has shown that visual perturbations (e.g., rotation, distractors) can enhance perceptual robustness and lead to better mathematical reasoning. In addition, [76] highlights that VL models often rely primarily on the information about the visual content provided in the question, rather than truly integrating visual information. Further, [74] argues that augmenting the original questions to have 10 options instead of 4 reduces VL model performance on general reasoning tasks. [10] also demonstrates that VL reasoners benefit from exposure to unimodal (text) reasoning data. Other work [9] shows that perception-enhanced reasoning chains improve the ability of VL models to play games such as Shisen-Sho. However, all these potential data interventions use different training paradigms (e.g., instruction-tuning, RL), model families and sizes, and underlying evaluations. This makes it difficult to anticipate which strategies will lead to reliable

improvements in VL reasoning. To address this, we perform a comprehensive set of experiments in which several datasets are compared against each other using a fixed training algorithm, student models and their sizes, and a battery of VL reasoning evaluation datasets. Our findings help in creating the best VL reasoning dataset, HONEYBEE, and show that model performance improves with scale, along with efficiency benefits for test-time scaling.

8. Context Curation Pipeline

The quality of context i.e., image, question pair, is crucial for determining the knowledge, and reasoning skills imparted to the VL reasoner. For instance, a VL reasoner exposed to diverse geometric images and questions will excel in downstream geometry problem-solving [77]. Thus, we aim to curate contexts that allow the resulting VL reasoner to achieve good performance on several downstream reasoning tasks.

Sourcing. In the past few years, several CoT datasets have been proposed to enable strong vision-language (VL) reasoning capabilities [23, 53, 71]. However, the reported improvements are a combination of several factors, such as the contexts present in the dataset, custom training algorithms, and the quality of the generator model. Furthermore, the contexts in different datasets originate in diverse ways. In particular, the contexts in reasoning datasets like Math-LLaVA and R1-OneVision [53, 73] are composed of several existing expert-curated and task-specific datasets, such as Geometry-3K [37], IconQA [38], CLEVR-Math [34], while ViRL [64] integrates several VL datasets [15] and performs a customized cleaning process. Similar to prior data curation work [18, 31], we fix the training algorithm (e.g., supervised finetuning) and the CoT generator model to analyze the *direct effect* of the contexts in individual datasets on training performant VL reasoners.

First, we list several widely adopted VL reasoning datasets consisting of (image, question, final answer) tuples. Next, we remove instances from these datasets that contain images with identical perceptual hashes⁷ to any of the images in the evaluation VL reasoning tasks [79]. Second, we prompt our generator model to produce a CoT to solve the (image, question) pairs in these datasets. Importantly, we consider small, similarly sized subsets of these datasets to control for the impact of data quality, rather than data quantity, on model performance. Similar to [18], this approach also facilitates faster training iterations. Further, we also create a filtered version of each dataset by discarding the CoTs that do not lead to the correct final answer. This step ensures that we do not penalize a dataset when our generator produces an incorrect solution. Third, we train multiple

VLMs (e.g., 3B and 8B models) on different context sources (and their filtered versions) and evaluate them on several downstream VL reasoning tasks. Finally, we rank each individual source dataset based on the average performance across multiple downstream tasks and model trainings.

Mixing. Here, we consider mixing as an additional strategy to combine the strengths of the top-performing data sources. Specifically, prior work in LLM reasoning [18, 28] has shown that mixing can yield superior datasets compared to using individual sources alone. To this end, we mix the data from the top-2, top-4, and all datasets from the previous stage. Subsequently, we take an equally sized subset from each mixture to control for dataset size and focus solely on mixture quality. We then train multiple VLMs on these mixtures and evaluate them on a battery of VL reasoning tasks. Finally, we assess whether any of the mixtures achieve better average performance than the best-performing individual dataset.

9. Perception Enhancing Data Interventions

We will outline the various data interventions for enhancing the *perception* of the VL reasoner:

Visual Perturbation. This intervention creates a perturbed version of the original image in the reasoning dataset to enhance the perceptual robustness of the VLM [32]. Starting from an instance in the synthetic reasoning dataset, $\{I_j, Q_j, C_j\} \in \mathcal{D}_G$, we apply one of three transformations, $\mathcal{T} = \{\text{rotation, distractor concatenation, dominance-preserving mixup}\}$, randomly to the image in each instance such that $I'_j = h(I_j)$, where h is a transformation from \mathcal{T} . We test the impact of this method in two ways: (a) as a *replacement*, where we perturb the images in part of the dataset while leaving the other part untouched, and (b) as an *augmentation*, where we mix the transformed dataset with the original dataset.

Text-Rich Images. It is crucial for VLMs to comprehend textual information embedded in images to perform accurate reasoning [63, 74]. To enhance the model’s ability to read and integrate textual information within images, we programmatically transform each (image, question) pair from the original data into a new image. Specifically, the new dataset \mathcal{D}_{TR} contains only the transformed text-rich image and the CoT from the original dataset. In particular, we embed the question in diverse font styles and colors, overlaying it with the original image on a blank background with various colors. Formally, $I'_j = \text{Render}(I_j, Q_j)$, thus, the new dataset is $\mathcal{D}_{TR} = \{I'_j, C_j\}_{j=1}^N$. Similar to visual perturbation, we assess the usefulness of this method as both a *replacement* and an *augmentation* strategy.

⁷<https://github.com/idealo/imagededup>

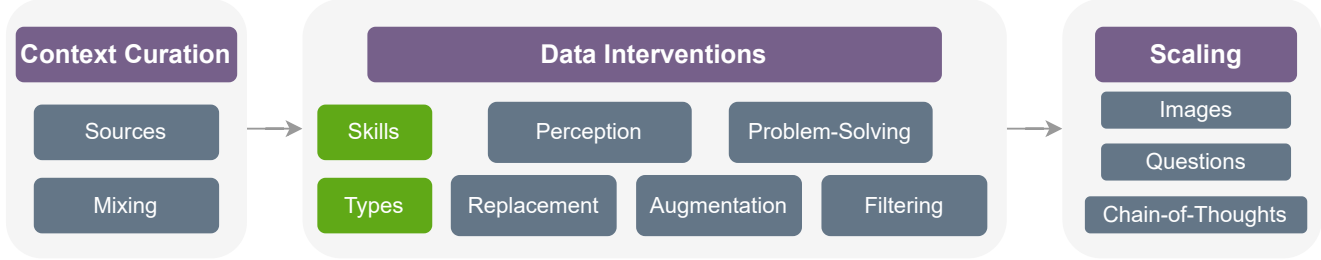


Figure 5. **Overview of the data curation pipeline.** In this work, we curate the context (image, question) from diverse sources and assess the impact of their mixing. Further, we curate a set of data interventions that target diverse skills and types. Subsequently, we study the impact of scaling along different data axes. These insights lead to the creation of our large-scale HONEYBEE dataset.

Perceptual Redundancy. Prior work [76] has shown that VLMs often rely on the visual content embedded within textual questions for answer deduction, rather than truly understanding the context image. Additionally, we observe that some textual questions can be answered accurately without access to the image, using only the model’s existing knowledge. For instance, a strong VLM can utilize its pre-trained knowledge to answer questions (e.g., What will lead to an increase in the population of deer? *i.* increase in lion, *ii.* decrease in plants, **iii. decrease in lion**, *iv.* increase in pika) without access to the food web image [26]. To encourage greater reliance on visual inputs, we *filter* the original synthetic dataset by removing instances where the generator model leads to the correct final answer without access to the image; that is, we exclude examples where the predicted answer $\hat{P}_j = \mathcal{G}(Q_j)$ matches the ground-truth answer A (i.e., $\hat{P}_j = A_j$).⁸

Shallow Perception. In the existing VL datasets, there are several instances that can be solved by querying the generator to solve them with access to the question and an image caption. Since image caption contains shallower information than all the visual content in the original image, such instances will not require deep visual insights in the reasoning process. Thus, we filter the original dataset where the generator model leads to the correct final answer with access to the question and image caption but not the image; that is, we exclude examples where the predicted answer $\hat{P}_j = \mathcal{G}(Q_j, I_j^{cap})$ matches the ground-truth answer A_j (i.e., $\hat{P}_j = A_j$). In our experiments, the image caption $I_j^{cap} = \mathcal{G}(I_j)$ is also generated by the same generator model.

Caption and Solve. We explore an active approach to enhancing the visual understanding capabilities of the VL rea-

⁸In our experimental setup, we use the same foundation model as CoT generator as well as caption generator. Thus, we take the liberty to denote image captioner as \mathcal{G} .

soner by providing auxiliary visual signals from the image caption I_j^{cap} from the stronger generator model. Specifically, we augment the original dataset’s CoT by including the image caption in the training data, i.e., $C'_j = [I_j^{cap}; C_j]$, where $;$ denotes concatenation in the raw text space. The transformed reasoning dataset is then $\mathcal{D}_{CS} = \{I_j, Q_j, C'_j\}_{j=1}^N$. By design, this method also increases the length of the CoT, $|C'_j| > |C_j|$, which facilitates the use of inference-time compute. We provide more information about different configurations in Appendix 15.

10. Problem-Solving Skill Enhancing Data Interventions

We will outline the various data interventions for enhancing the *problem-solving* skills of the VL reasoner:

Text-Only Reasoning. Through this intervention, we aim to enhance problem-solving skills by exposing the VLM to novel problems and solutions from a high-performing text-only reasoning dataset [18]. This strategy serves multiple purposes: (a) it allows the VLM to learn useful skills from existing unimodal reasoning data via cross-task transfer, and (b) it makes the VLM a more general-purpose reasoner, enabling it to solve textual reasoning problems too beyond VL reasoning problems. In practice, this method acts as an *augmentation*, where the final dataset is a concatenation of the original VL reasoning data and the high-performing text-only reasoning data, i.e., $\mathcal{D}_G^{VL+Text} = \mathcal{D}_G^{VL} \cup \mathcal{D}_G^{Text}$. For consistency, we re-annotate the problems from the text-only reasoning data to obtain unimodal CoTs from the same generator model.

Increased Distractors. Prior work [25, 68, 74] has shown that introducing distractors in reasoning tasks, specifically by increasing the number of options (e.g., from four to ten), makes them harder for reasoning models to solve. Motivated by this observation, we explore a strategy to transform the original questions to have ten options and modify the original CoT accordingly; that is, $(Q'_j, C'_j) =$

$LLM(Q_j, C_j)$. This method aims to encourage the VLM not to rely on chance when answering complex problems and to boost model robustness. We test the impact of this method in two ways: (a) as a *replacement*, where part of the original dataset is transformed with distractors while the other part remains untouched, and (b) as an *augmentation*, where we mix the transformed dataset with the original dataset.

Length. In this method, we aim to encourage a reasoner to generate longer chains of thought (CoTs). This approach allows the model to utilize additional test-time compute for more detailed visual comprehension, stepwise solutions, planning, and re-evaluation during problem-solving. To achieve this, we compute the distribution of CoT lengths in the original reasoning dataset and split it into two equal halves: those with less than and those with more than the median CoT length. Subsequently, we train the models on the longer half to promote extended reasoning behavior in downstream tasks.

Difficulty. Reasoning datasets are typically skewed toward easier problems, as these are easier to acquire at scale compared to more difficult problems, which only a limited number of experts can solve [20]. To increase the difficulty of our dataset, we first classify all instances into diverse difficulty levels (1: beginner, 2–3: AMC, 4: intermediate level AIME, > 4: Olympiad level) using an LLM and a predefined rubric.⁹ We then filter the dataset to ensure that the representation for each level is roughly balanced. This enables the model to learn from harder examples and tackle more complex VL reasoning problems in downstream tasks.

We present additional details about the data intervention strategies in Appendix 15. In theory, all the data intervention strategies listed above should enhance VL reasoning behaviors. However, an intervention strategy may steer the model toward behaviors that benefit a specific evaluation dataset at a particular model scale. Thus, outperforming a simple baseline of training on the original reasoning dataset across various downstream VL reasoning tasks and multiple model scales remains a non-trivial challenge. Therefore, it is essential to compare these strategies under identical conditions (e.g., training and evaluation) to identify those that consistently improve performance across the board.

11. Mixing Text and VL reasoning data

We find that the base models PLM-3B and PLM-8B do not perform well on the MATH500 y achieving an accuracy of 32.2% and 34.0%, respectively. We find that

⁹https://artofproblemsolving.com/wiki/index.php/AoPS_Wiki:Competition_ratings

training these models with VL reasoning CoT data (baseline data in Table 8) improves the average performance to 39.2%, indicating cross-modal reasoning transfer (vision-language to language-only). However, this performance still lags behind training the VLMs with text-only reasoning data (OpenThoughts3). But, training with the mixture of VL and text-only reasoning data improves the accuracy from 39.2 to 59.7 while improving the VL reasoning performance too. Thus, we will apply *caption and solve* and *text-only reasoning* data intervention strategies to create our final VL reasoning data. Interestingly, we find that most methods lead to poorer performance than the baseline, which suggests that it is non-trivial to improve over the best-chosen data source using the targeted data interventions.

12. Correlation Analysis

For our data curation, we train PLM-3B and PLM-8B and evaluate them on five VL reasoning tasks for each experiment to make robust decisions that genuinely improve reasoning performance across multiple model scales and evaluations. Here, we aim to understand the correlation between the performance achieved by PLM-8B and PLM-3B on average (and individual dataset) accuracy across all experiments conducted in this work (Figure 8), and observe a strong correlation in average accuracy. This indicates that data curation methods have similar impacts on average accuracy across different model scales. Similar to [40] for LLM pretraining, a broader implication of this analysis is that future research can perform optimal data selection strategies with a small VLM, and it is likely to be optimal for a larger VLM.

13. Reasoning Actions in HONEYBEE

We analyze the occurrence of several keywords in our CoTs that are crucial for diverse reasoning behaviors, including visual understanding (perception), question comprehension, planning, and reflection (Table 11). This suggests that the reasoning CoTs in our dataset encourage VL reasoners to elicit more complex reasoning (such as reflection) behaviors than existing VL reasoning datasets.

14. Additional Context Dataset Details and Results

We present additional details about the context datasets in Table 12. Specifically, we report the total number of examples originally present in each dataset. Subsequently, we perform decontamination by removing instances in which the image exactly matches any image in the validation dataset, as determined by the phash algorithm. For the context curation experiments, we randomly sample at most 50K examples from each dataset, as shown in the third column. Furthermore, we create a filtered version of these datasets

Context Sources	Models	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
ViRL [†] [64]	3B	38.8	32.9	58.6	25.7	25.0	52.0
	8B	41.3	34.5	64.5	23.4	28.2	55.8
	Avg.	40.1	33.7	61.6	24.5	26.6	53.9
Math-LLaVA [53]	3B	36.3	30.0	56.3	23.6	21.7	50.1
	8B	39.2	33.7	62.2	23.4	26.1	50.5
	Avg.	37.7	31.8	59.2	23.5	23.9	50.3
R1-OneVision [73]	3B	35.7	30.6	55.2	20.1	22.8	49.7
	8B	38.8	32.5	60.5	22.0	26.9	52.2
	Avg.	37.3	31.6	57.9	21.1	24.9	51.0
ThinkLite-VL-Hard [†] [67]	3B	34.6	25.5	53.7	25.3	20.6	48.1
	8B	39.5	31.2	61.2	24.0	27.7	53.1
	Avg.	37.1	28.4	57.5	24.7	24.2	50.6
LLaVA-CoT [71]	3B	34.6	28.5	54.2	19.0	22.6	48.6
	8B	37.9	32.7	60.3	18.7	26.5	51.5
	Avg.	36.3	30.6	57.3	18.9	24.5	50.1
MMK12 [43]	3B	34.6	28.6	58.1	19.0	26.4	40.6
	8B	37.3	29.2	55.1	24.3	26.4	51.5
	Avg.	36.0	28.9	56.6	21.7	26.4	46.1

Table 6. **Ranking the quality of the context from diverse datasets.** We train PLM-3B and PLM-8B on CoTs generated on the context (image, question) pairs from diverse dataset. Then, we rank them based on the average performance of these models on several VL reasoning downstream tasks. For each context source, we report the best results achieved from one of two scenarios: unfiltered CoT data and filtered CoT data based on final answer correctness. We find that ViRL and ThinkLite-VL-Hard (marked in [†]) benefit from the final answer correctness filter (Appendix Table 13).

Mixing Sources	Models	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
Top 1	3B	38.8	32.9	58.6	25.7	25.0	52.0
	8B	41.3	34.5	64.5	23.4	28.2	55.8
	Avg.	40.1	33.7	61.6	24.5	26.6	53.9
Top 2	3B	37.8	33.4	58.9	21.1	25.6	50.2
	8B	39.3	32.7	60.9	20.7	29.2	53.0
	Avg.	38.6	33.1	59.9	20.9	27.4	51.6
Top 4	3B	36.7	31.5	57.6	20.1	24.0	50.5
	8B	39.7	34.9	61.6	23.4	28.2	50.3
	Avg.	38.2	33.2	59.6	21.7	26.1	50.4
All	3B	36.5	33.1	58.3	17.8	23.2	49.9
	8B	40.5	35.8	61.7	25.3	28.2	51.3
	Avg.	38.5	34.5	60.0	21.5	25.7	50.6

Table 7. **Impact of mixing source datasets.** We mix the VL reasoning CoT data from diverse datasets to assess whether it leads to better performance than the individual datasets itself. We find that the individual dataset source, ViRL, achieves the best performance.

based on final answer correctness, with the resulting sizes reported in the last column.

Next, we train PLM-3B and PLM-8B on both the original and filtered versions of the source datasets, where CoTs are generated by our generator model (Llama-4-Scout). We evaluate the trained models on five downstream VL tasks and present their average accuracy in Table 13. We find that ViRL and ThinkLite-VL-Hard benefit from final answer correctness filtering, while the original (unfiltered) datasets produce better reasoners. The best-performing ver-

sion of each context source dataset is presented in the main Table 6.

15. Additional Data Intervention Details and Results

Here, we present additional implementation details regarding the diverse data interventions. Where relevant, we also highlight experimental results across various configurations.

Method	Skill	Type	Models	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
Original Data	-	-	3B	38.8	32.9	58.6	25.7	25.0	52.0
			8B	41.8	35.3	65.8	22.4	29.8	55.6
			Avg.	40.1	33.7	61.6	24.5	26.6	53.9
Caption and Solve	Perception (Auxiliary Signal)	Augment	3B	39.7	35.7	56.6	24.7	26.7	55.0
			8B	43.0	38.3	63.2	26.3	30.4	56.9
			Avg.	41.4 (+3.3pp)	37.0	59.9	25.5	28.6	56.0
Visual Perturb	Perception (Robustness)	Replace	3B	36.5	31.2	57.5	18.1	25.8	49.9
			8B	40.5	31.9	63.4	22.7	29.5	54.8
			Avg.	38.5	31.6	60.5	20.4	27.7	52.4
Text-Rich Images	Perception (Synthetic Images)	Replace	3B	37.1	30.2	55.5	22.7	27.1	49.9
			8B	40.5	35.3	61.0	21.1	30.6	54.5
			Avg.	38.8	32.8	58.3	21.9	28.9	52.2
Perceptual Redundancy	Perception (Feasibility w/ image)	Filter	3B	36.0	31.2	54.6	21.4	24.6	48.4
			8B	36.9	32.7	58.0	18.1	27.8	48.1
			Avg.	36.5	32.0	56.3	19.8	26.2	48.3
Shallow Perception	Perception (Feasibility w/ caption)	Filter	3B	34.7	29.3	53.4	20.7	23.8	46.2
			8B	36.5	29.2	57.3	21.4	28.1	46.3
			Avg.	35.6	29.3	55.4	21.1	26.0	46.3
Text-Only Data	Problem-Solving (Cross-modal transfer)	Augment	3B	41.9	38.1	60.1	25.7	26.8	58.9
			8B	44.2	41.2	63.4	27.3	31.4	57.6
			Avg.	43.1 (+7.5pp)	39.7	61.8	26.5	29.1	58.3
Increased Distractors	Problem-Solving (Robustness)	Replace	3B	33.1	23.4	51.3	19.7	22.3	48.8
			8B	36.2	27.4	53.2	20.1	27.5	52.6
			Avg.	34.6	25.4	52.3	19.9	24.9	50.7
Length	Problem-Solving (Long Thinking)	Filter	3B	33.9	28.3	50.3	19.7	24.6	46.8
			8B	38.8	32.6	60.7	19.4	28.3	53.2
			Avg.	36.4	30.5	55.5	19.6	26.5	50.0
Uniform Difficulty	Problem-Solving (Hardness)	Filter	3B	33.1	25.1	52.9	17.4	24.1	45.8
			8B	36.1	28.6	56.8	21.4	24.9	48.8
			Avg.	34.6	26.9	54.9	19.4	24.5	47.3

Table 8. **Results for data intervention experiments.** We compare the performance of VL reasoners trained on datasets created using diverse data intervention strategies. Our results show that achieving better performance than the original data, which is obtained by selecting the right data sources and their mixtures, is non-trivial. We also report the results for the best-performing configuration from each intervention strategy. We find that augmenting the original CoT with image captions (*caption and solve*), and mixing with *text-only reasoning* data reliably improve VL reasoning performance.

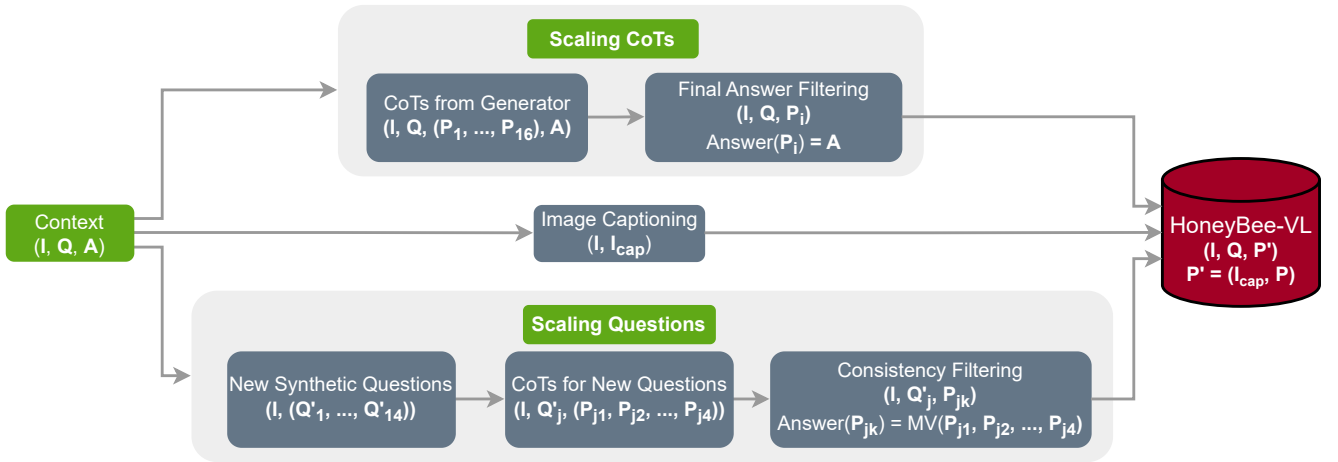


Figure 6. **HONEYBEE-VL scaling pipeline.** We present the pipeline to scale the number CoTs per (real image, question) pair (**top**), and scale the number of questions per real image (**bottom**) from the context datasets. We denote majority voting over the several predicted answers as MV. Ultimately, this pipeline leads to the creation of 1.5M VL reasoning instances.

Visual Perturbation. In this intervention, we apply one of three transformations *rotation*, *distractor concatenation*, *dominance-preserving mixup* to every image in the

dataset. In *rotation*, we rotate the image by any angle (in degrees) in this list [0, 15, 30, 45, 60, 75, 90, 105, 120, 135, 150, 165, 180, 195,

Data	Avg.	PLM-3B	PLM-8B
Baseline	32.2	30.4	34.0
VL	39.2	34.6	43.8
Text	62.7	60.2	65.2
VL + Text	59.7	57.0	62.4

Table 9. **Generalization to MATH500.** We present the performance of VL reasoners on text-only MATH500 reasoning task. The baseline refers to the base models performance, VL refers to exposure to just VL reasoning data, Text refers to training with re-annotated OpenThoughts3, and VL+Text refers to mixture of VL and text reasoning CoT data.

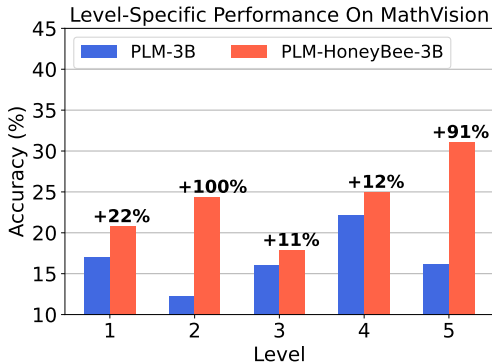


Figure 7. PLM-3B trained on HONEYBEE outperforms the base model across all MathVision difficulty levels.

210, 225, 240, 255, 270, 285, 300, 315, 330, 345]. In *distractor concatenation*, we treat any image from the other dataset as a distractor, and concatenate it with the original image in the width dimension to form a new image. In *dominance-preserving mixup*, we form a new image in the input space by mixing the original image with a distractor image using mixup image = $\alpha \times \text{original image} + (1 - \alpha) \times \text{distractor image}$ where $\alpha \sim \mathcal{U}[0.8, 1.0]$.

We test two approaches for applying visual perturbation to the original data: (a) *replacement*, where a 50% of the original data is replaced with its perturbed version, and (b) *augmentation*, where the perturbed version of the original data is concatenated with the original data itself. For (b), we compare the performance of training VLMs on 50% of the original data versus training them on the combination of this data and its visually perturbed version, effectively doubling the amount of training data at no additional cost. The results are shown in Table 14. We find that augmenting the original data with its visually perturbed version leads to worse reasoning performance, which may be attributed to increased redundancy in the training data and the noise introduced by visual perturbations.

16. Data Curation Pipeline: Scaling

Scaling the amount of reasoning CoT data has consistently helped in enhancing the LLM performance on several downstream tasks [18, 59]. However, there has been limited exploration on the impact of scaling data for VL reasoning. In addition, the introduction of a new modality (image) in VL space adds a new dimension to scaling training data. Hence, we study the impact of scaling diverse axes (e.g., image, question, and CoT) on the reasoning capabilities of the VLM. Here, we explain the method to study the scaling behaviors starting from a dataset containing N unique images, and corresponding question and CoT $\mathcal{D}_N = \{I_j, Q_j, C_j\}_{j=1}^N$. We drop the subscript \mathcal{G} for simplicity.

Scaling Images. We create three additional subsets of the original dataset, each of increasing size, resulting in four datasets in total: $[\mathcal{D}_{N/8}, \mathcal{D}_{N/4}, \mathcal{D}_{N/2}, \mathcal{D}_N]$. Here, $\mathcal{D}_{N/8}$ indicates that we randomly select $N/8$ instances from the original dataset. As we increase the data size, we are effectively scaling the number of unique images in the dataset. Subsequently, we train several VLMs on these datasets and study whether the average performance on downstream tasks improves with increased data scale.

Scaling Questions. We study the impact of synthesizing novel questions by using the existing questions as seeds. In particular, we prompt the question generator to create new questions that are reasonable, solvable, and of similar difficulty to the original questions, conditioned on the image and question [59]; i.e., we generate $Q'_{j,i} = \mathcal{G}(I_j, Q_j)$, where $Q'_{j,i}$ is the i^{th} synthetic question for (I_j, Q_j) using the question generator \mathcal{G} .¹⁰ Subsequently, we create a CoT for the newly synthesized questions using the generator model. We then merge this synthetic data with the original data, thereby scaling the dataset by having multiple questions for a given image. In our experiments, we construct four subsets of synthetic reasoning data: $[\mathcal{D}_{N/8}(n_q = 1), \mathcal{D}_{N/8}(n_q = 2), \mathcal{D}_{N/8}(n_q = 4), \mathcal{D}_{N/8}(n_q = 8)]$, where $\mathcal{D}_N(n_q = k) = \{I_j, Q_j, C_j\}_{j=1}^N \cup \{I_j, \{Q'_{j,i}, C_{j,i}\}_{i=1}^{k-1}\}_{j=1}^N$ and $C_{j,i} = \mathcal{G}(I_j, Q'_{j,i})$. If $n_q = 4$, then three questions are newly synthesized, and the remaining question comes from the original data. We present the prompt used to generate new questions and provide qualitative examples in Appendix 23 and Appendix 20, respectively.

Scaling CoTs. We study the impact of increasing the number of CoT traces for a given image and question pair in the reasoning dataset. This approach exposes

¹⁰In our experiments, we use the same foundation model for both CoT generation and question generation; hence, we denote the question generator as \mathcal{G} .

Correlation between PLM-3B and PLM-8B Runs

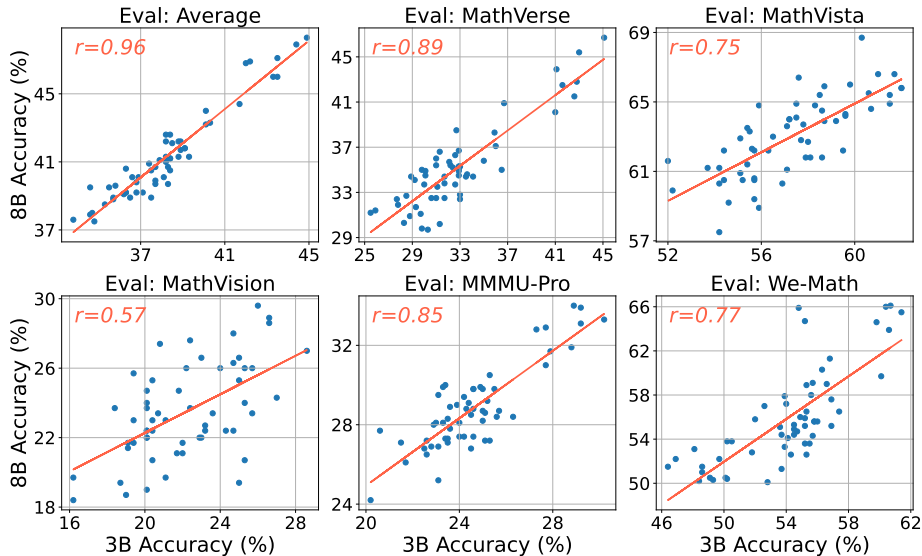


Figure 8. **Model Correlation Analysis.** We present the correlation between the performance of PLM-8B and PLM-3B, each trained with various data distributions throughout this work, across diverse downstream evaluation tasks. A higher correlation implies greater predictability between model performances; that is, a data curation method that is optimal for a 3B VLM is more likely to be optimal for an 8B VLM.

Method	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
OpenVLThinker-v1.2-3B [14]	42.3	34.1	63.9	25.0	28.7	59.8
PLM-HONEYBEE-3B	44.3	42.8	61.2	29.9	28.4	59.3
PLM-HONEYBEE-3B-GRPO	46.2	43.5	64.9	28.4	28.3	65.8

Table 10. **RL training on top of HONEYBEE trained VLM.** We present results for training PLM-3B with HONEYBEE data using supervised finetuning, followed by a round of RL training with the GRPO algorithm on a verifiable VL dataset. Furthermore, we compare its performance to that of a state-of-the-art RL-tuned VL reasoner, OpenVLThinker-v1.2-3B, which has a similar model capacity.

KeyWord	Category	Number (in K)
Identify	Perception	1950
Recall	Perception	1127
Provided	Comprehension	1040
First	Planning	2388
Wait	Reflection	32

Table 11. Keyword Analysis

the VL reasoner to diverse problem-solving strategies for a given problem. Specifically, we create four subsets of the dataset with an increasing number of CoTs per (image, question) pair: $[\mathcal{D}_{N/8}(n_c = 1), \mathcal{D}_{N/8}(n_c = 2), \mathcal{D}_{N/8}(n_c = 4), \mathcal{D}_{N/8}(n_c = 8)]$, where $\mathcal{D}_N(n_c = k) = \{I_j, Q_j, \{C_{j,i}\}_{i=1}^k\}_{j=1}^N$. Ultimately, we combine the findings from context curation, data interventions, and scaling diverse modalities experiments to create one of the largest

VL reasoning datasets, HONEYBEE (see §5.3 for more details).

Text-Rich Images. In this intervention, we render the original image and question from the dataset into a new in the raw space on randomly sampled background colors, text fonts and colors. We use functions from the PIL python library to achieve this task. In particular, we provide our rough implementation in Listing 23. Similar to visual perturbations, the *text-rich images* intervention can be used in two ways: (a) *replacement* on 50% of original data, and (b) *augmentation*. We highlight the results for *augmentation* strategy in Table 14. In particular, we find that augmenting the data with text-rich images does not improve the reasoning performance over the half the original data (row 1 vs row 3).

Context Sources	Total Examples (in K)	# Contaminated	# Context Curation (Original, in K)	# Context Curation (Filtered, in K)
ViRL	38.9	80	38.9	23.6
Math-LLaVA	338.7	856	50.0	29.0
R1-OneVision	154.6	508	50.0	22.5
ThinkLite-VL-Hard	11.0	129	11.0	4.0
LLaVA-CoT	98.6	336	50.0	27.4
MMK12	17.6	1	17.6	9.0

Table 12. **Context source dataset statistics.** We present the total number of examples, the number of instances found to be contaminated, and the number of instances originally used in context curation experiments compared to their filtered versions.

Context Sources		Original	Filtering
ViRL	3B	38.1	38.8
	8B	40.1	41.3
	Avg.	39.1	40.1
Math-LLaVA	3B	36.3	36.2
	8B	39.2	39.1
	Avg.	37.7	37.6
R1-OneVision	3B	35.7	34.8
	8B	38.8	37.5
	Avg.	37.3	36.2
ThinkLite-VL-Hard	3B	34.7	34.6
	8B	38.0	39.5
	Avg.	36.4	37.1
LLaVA-CoT	3B	34.6	33.8
	8B	37.9	37.6
	Avg.	36.3	35.7
MMK12	3B	34.6	35.6
	8B	37.3	36.0
	Avg.	36.0	35.8

Table 13. **Impact of filtering on final answer correctness.** We present the results of training PLM-3B and PLM-8B on both the original and filtered versions of the context source datasets. Specifically, we report the average accuracy across the five evaluation datasets and bold the best-performing setup.

Perceptual Redundancy. This strategy filters the instances in the original data if the blind generator model can solve the question correctly i.e., without access to the image. We observe that this strategy removes 42% of the instances in the data curation experiments.

Shallow Perception. This strategy is a stricter version of the perceptual redundancy where the instances that can be solved correctly using a blind generator model with access to image captions are filtered. In our experiments, we observe that this strategy removes 55% of the data.

Caption and Solve. In this strategy, the chain-of-thought (CoT) is augmented to first generate an image caption, providing auxiliary perceptual signals before addressing the

given question. Specifically, we explore three approaches for creating such data using our generator model: (a) synthesizing the image caption and problem-solving steps independently i.e., $I_j^{cap} = \mathcal{G}(I_j)$ and $C_j = \mathcal{G}(I_j, Q_j)$, (b) generating the image caption first, followed by problem-solving ($I_j^{cap}, C_j = \mathcal{G}(I_j, Q_j)$), and (c) generating the image caption and then providing it as additional context for problem-solving i.e., $C_j = \mathcal{G}(I_j, Q_j, I_j^{cap})$. Ultimately, the VLMs are trained in an identical fashion across all three strategies; that is, the input consists of (image, question), and the CoT comprises (caption, solution). We present the results from these three strategies in Table 15. We observe that the first strategy produces CoTs that train the strongest VL reasoners. This approach is also the most efficient in terms of scalable data generation, as the caption and solution can be

Augmentation Expts.	Models	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
50% Original Data	3B	35.1	28.7	52.7	23.0	25.0	46.1
	8B	40.0	32.6	61.5	24.3	28.4	53.2
	Avg.	37.6	30.6	57.1	23.7	26.7	49.7
50% Original Data + 50% Visual Perturbation	3B	34.6	28.3	52.6	19.1	24.5	48.5
	8B	39.2	31.9	62.0	23.4	27.4	51.4
	Avg.	36.9	30.1	57.3	21.2	26.0	50.0
50% Original Data + 50% Text-Rich Images	3B	34.8	28.7	55.4	16.1	25.4	48.2
	8B	39.4	32.9	59.5	22.7	28.1	53.7
	Avg.	37.1	30.8	57.5	19.4	26.8	51.0

Table 14. Data intervention augmentation results.

Caption and Solve Variants	Models	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
I \rightarrow C (I, Q) \rightarrow S	3B	39.7	35.7	56.6	24.7	26.7	55.0
	8B	43.0	38.3	63.2	26.3	30.4	56.9
	Avg.	41.4	37.0	59.9	25.5	28.6	56.0
(I, Q) \rightarrow (C, S)	3B	37.8	32.5	55.1	22.4	25.8	53.4
	8B	42.3	38.4	60.4	27.6	29.0	55.8
	Avg.	40.0	35.5	57.7	25.0	27.4	54.6
I \rightarrow C (I, Q, C) \rightarrow S	3B	36.6	31.2	52.3	23.1	24.4	52.1
	8B	38.9	30.2	60.5	22.0	29.4	52.6
	Avg.	37.8	30.7	56.4	22.5	26.9	52.3

Table 15. **Caption and Solve variants.** We present the accuracy of models trained with different data generation variants for the *caption and solve* intervention. Here, I \rightarrow C means that the teacher model takes image I as input and outputs the caption C. Furthermore, (I, Q) \rightarrow S indicates that the teacher model takes the (image, question) pair as input and outputs the solution S. We note that the student VL reasoner model is always trained to generate the caption C and solution S jointly, conditioned on the image and question pair: (I, Q) \rightarrow (C, S).

generated in parallel and later merged for training VL reasoners.

Text-Only Reasoning. In this strategy, we augment the original VL reasoning data with the text-only reasoning data. In particular, we choose a state-of-the-art reasoning data, OpenThoughts3 [18]. The CoTs in the original dataset are collected the QwQ-32B reasoning model.¹¹ To ensure consistency between the generator models, we re-annotate the entire 1.04 million samples with Llama-4-Scout.

Increased Distractors. In this strategy, we re-write the dataset to increase the number of distractors (Listing 17). In our experiments, we consider the *replacement* strategy with this intervention on 50% of original data.

Length. In this strategy, we bias our dataset to elicit long CoTs. Specifically, we compute the median CoT length and divide the dataset into two equal halves i.e., instances with

¹¹<https://qwenlm.github.io/blog/qwq-32b/>

CoTs lesser than the median and the ones more than the median.

Difficulty. In this strategy, we classify each instance in the dataset according to its difficulty level. In our dataset of size 50K, the distribution of difficulty levels is {Level 2: 15.1K, Level 1.5: 14.4K, Level 1: 11.8K, Level 4: 3.6K, Level 3: 3.5K, Level 4+: 0.5K}. To increase the impact of harder examples, we select 3500 samples from each difficulty level and train on this uniform-difficulty dataset.

17. Generalization to New Model Families

We trained Qwen2.5-VL and Qwen3-VL models on the HoneyBee data (Table 16). The HoneyBee models outperform their base models by a large margin across multiple evaluation datasets.

18. Self-Improvement

Here, we aim to study the impact of training the teacher model, Llama-4-Scout (109A17B MoE), on the self-generated reasoning data, HONEYBEE. Specifically, we

Model	Avg.	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math
Qwen2.5-VL-3B	39.3	35.0	58.9	23.7	29.8	49.2
Qwen2.5-VL-3B-HoneyBee	45.1	39.7	62.3	24.0	34.4	65.2
Qwen3-VL-2B	43.5	36.3	61.6	27.3	32.7	59.8
Qwen3-VL-3B-HoneyBee	47.2	43.9	63.0	28.0	34.8	66.2

Table 16. Performance (%) for new model families like Qwen-VL.

take 50K subset of this data and finetune this model using low-rank adaptation (LoRA) with the `Llama-factory` library.¹² We present the results across downstream evaluation datasets in Table 17. We find that parameter-efficient finetuning of Llama-4 on the HONEYBEE dataset leads to a relative gain of 3.7pp over the base model. Importantly, we observe improvements across most of the evaluation datasets indicating the strong self-improvement capabilities. Future work can focus on training much stronger models on using our high-quality and large-scale dataset.

19. Additional Training and Evaluation Setup

PLM setup. In this work, we predominantly train Perception Language Models (PLMs) [12] for our data curation and large-scale experiments. Specifically, the PLMs use a Llama-3.2 LLM backbone that processes visual input via a perception encoder [7]. We train PLMs using their publicly available codebase, starting with the stage-3 configurations.¹³ In particular, we perform full finetuning of all parameters of the PLMs including the vision encoder and the LLM backbone. Due to the sheer number of experiments, we use the default peak learning rate (LR) of $4e-5$ for PLM-1B and PLM-3B, and $1e-5$ for PLM-8B. The warmup ratio is set to 0.1 followed by cosine decay upto 10% of the peak learning rate. Furthermore, we train our models for 5 epochs across all experiments. In our data curation experiments, we set a global batch size of 2048 for PLM-1B and PLM-3B, and 1024 for PLM-8B, while we double the global batch sizes for the large-scale runs. All experiments are performed on multiple nodes i.e., either 8 or 16 A100 (80GB) nodes.

RL Training. In Table 10, we present a round of RL training using the GRPO algorithm. Specifically, we use the default implementation from the TRL library.¹⁴ Since this work is focused on supervised finetuning, we conduct lightweight GRPO training (1 epoch) to demonstrate its effectiveness as a cold start for RL. In particular, we use a

¹²<https://github.com/hiyouga/LLaMA-Factory/pull/7611>

¹³https://github.com/facebookresearch/perception_models

¹⁴https://github.com/huggingface/trl/blob/main/examples/scripts/grpo_vlm.py

verifiable multimodal dataset.¹⁵ We set the number of generations per prompt in the GRPO algorithm to four. Further, we use the peak learning rate of $1e-6$ works well for PLM-HONEYBEE SFT models in the TRL configuration.

Evaluation. To ensure consistent prompting across all benchmarks, we query our model with ‘Your final answer MUST BE put in `boxed`.’ Furthermore, we score our models using the rule-based verifier `mathruler` for scalability and fast iteration following [14].¹⁶ We notice that the original `MathVerse` free-form questions often contain numerics followed by units (e.g., celsius, cm), which makes rule-based verification more challenging. To address this, we use an LLM to extract only the final answer values and consider them as the ground-truth answers.

20. Qualitative Examples

We show several qualitative examples of newly generated questions for a given image in Figure 10 and Figure 11. These examples highlight that the synthetic questions can be quite diverse and reasonable for training VL reasoners. Furthermore, we present a few qualitative examples from the HONEYBEE dataset in Figure 12 and Figure 13. These examples demonstrate that the CoTs in our dataset are formed by concatenating the captions and problem solutions in the raw text space.

21. Reasoning Faithfulness

Evaluating CoT quality is far more challenging and time-consuming than verifying final answers. Our setup is therefore consistent with prior reasoning work such as DeepSeek-R1 [19]. We also assess CoT faithfulness by asking GPT-5.1-mini to judge HoneyBee-8B CoTs on MathVista. We find that CoTs are faithful in 80% of cases, showing that filtering by final-answer correctness still yields strong faithfulness.

¹⁵[lmms-lab/multimodal-open-r1-8k-verified](https://github.com/lmms-lab/multimodal-open-r1-8k-verified)

¹⁶<https://github.com/hiyouga/MathRuler>, <https://github.com/yihedeng9/OpenVLThinker>

	Average	MathVerse	MathVista	MathVision	MMMU-Pro	We-Math	MATH500	GPQA
Llama-4-Scout	59.3	53.9	69.3	32.6	51.2	72.9	81.2	54.0
Llama-4-Scout (HONEYBEE)	61.5	54.3	70.9	37.5	51.7	72.8	83.0	60.6

Table 17. **Self-improvement results.** We train the generator model, Llama-4-Scout, on a subset of the HONEYBEE dataset and report its accuracy on diverse evaluation datasets.

Train schedule			
Rollercoaster Land	9:00 A.M.	2:45 P.M.	9:30 P.M.
Chickenville	9:30 A.M.	3:15 P.M.	10:00 P.M.
City of Green Trees	10:15 A.M.	4:00 P.M.	10:45 P.M.
Floral Gardens	10:45 A.M.	4:30 P.M.	11:15 P.M.
Comfy Pillows Resort	11:15 A.M.	5:00 P.M.	11:45 P.M.

Synthetic Question 1: What is the travel time from Rollercoaster Land to Chickenville?
 Choices:
 (A) 15 minutes
 (B) 30 minutes
 (C) 45 minutes
 (D) 1 hour

Synthetic Question 2: What is the travel time from City of Green Trees to Floral Gardens?
 Choices:
 (A) 30 minutes
 (B) 45 minutes
 (C) 1 hour
 (D) 15 minutes

Synthetic Question 3: What is the most common time interval between consecutive train stops according to the schedule?
 Choices:
 (A) 15 minutes
 (B) 30 minutes
 (C) 45 minutes
 (D) 1 hour

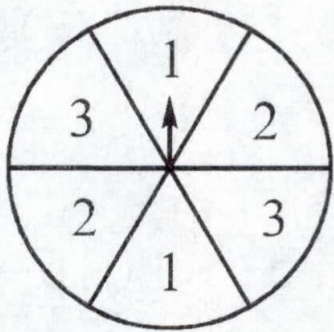
Figure 10. **Qualitative example to show diverse and reasonable synthetic images for a given image.**

22. MCQ-style Evaluations

About 50% of MathVista, 50% of MathVision, 60% of DynaMath, and 100% of MATH500 consist of free-form (non-MCQ) questions. We observe comparable performance improvements on both MCQ and free-form subsets across these benchmarks, reflecting improved problem-solving ability. In addition, the teacher-model CoTs do not rely on elimination strategies for MCQs (Appendix Figures 12 and 11).

23. Prompts

We present the prompts used in our experiments (Listing 14, 15, 17, and 16).



Synthetic Question 1: What is the probability that the pointer lands on a number greater than 1 when the spinner is spun once?

Synthetic Question 2: If the spinner is spun twice, what is the probability that the pointer lands on a number greater than 1 on both spins?

Synthetic Question 3: What is the probability that the pointer lands on a sector labeled with the number 2 when the spinner is spun once?

Figure 11. **Qualitative example to show diverse and reasonable synthetic images for a given image.**

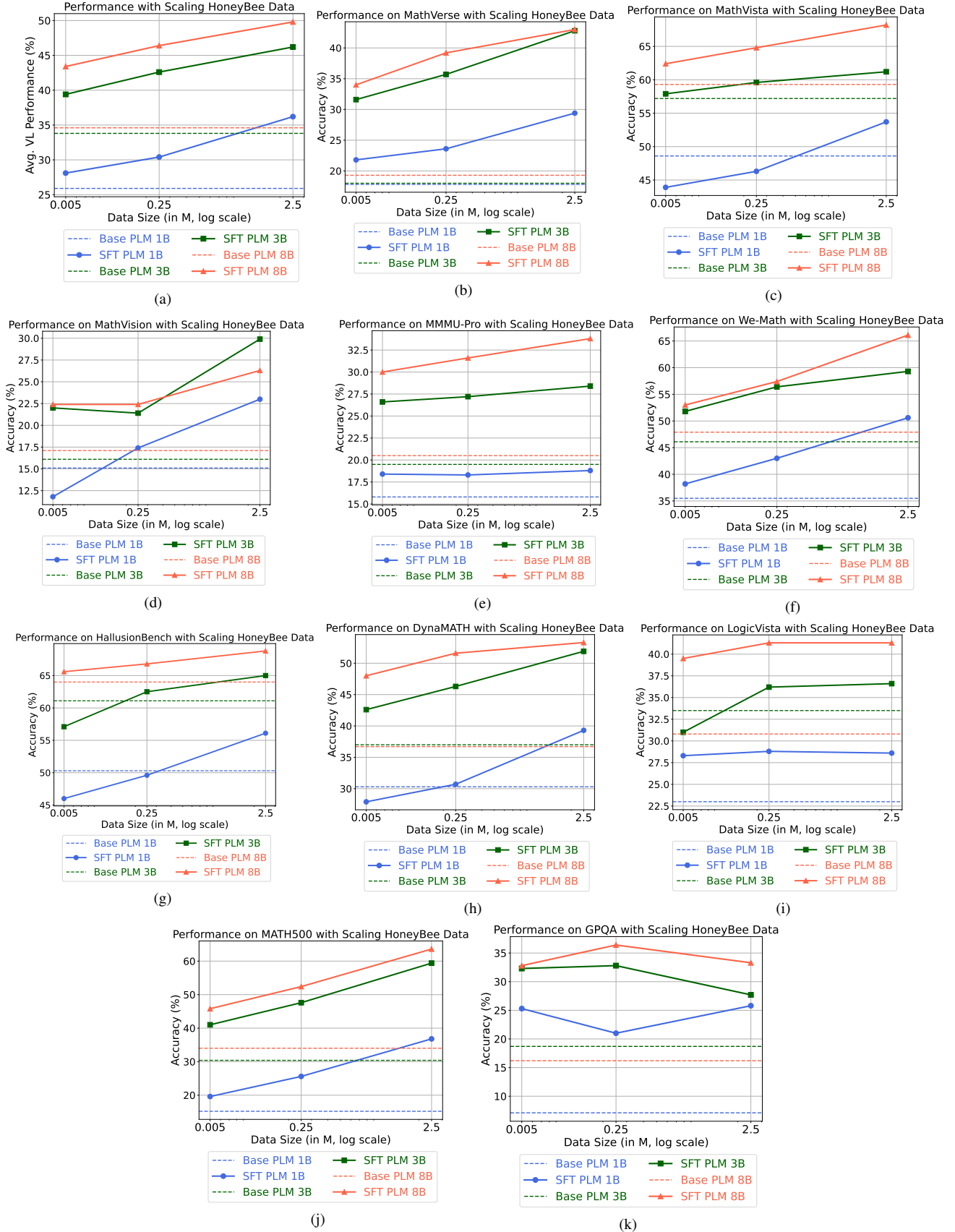


Figure 9. **Scaling trends across individual evaluation datasets.** We train PLMs (1B-8B) on diverse subsets of the HONEYBEE data and evaluate them on individual datasets. The first plot (first row, first column) is the average performance across the ten evaluation datasets.

Time Cody spent on homework	
Day	Minutes
Sunday	38
Monday	38
Tuesday	35
Wednesday	34
Thursday	31

Question: What is the total time Cody spent on homework from Sunday to Tuesday?

- A) 105 minutes
- B) 111 minutes
- C) 106 minutes
- D) 108 minutes.

Chain-of-Thought:

The image presents a table illustrating the time Cody spent on homework each day, with data for five days.

Table Structure:

The table consists of two columns and six rows.

The first row is a header row, with the left column labeled Day and the right column labeled Minutes.

The remaining five rows represent individual days of the week, from Sunday to Thursday.

Data:

Sunday: 38 minutes

Monday: 38 minutes

Tuesday: 35 minutes

Wednesday: 34 minutes

Thursday: 31 minutes

In summary, the table provides a clear and concise overview of Cody's daily homework time, allowing for easy comparison and analysis of his study habits over the course of a week.

think

Step 1: Identify the relevant data

To find the total time Cody spent on homework from Sunday to Tuesday, we need to identify the time spent on each of these days. According to the table, Cody spent 38 minutes on Sunday, 38 minutes on Monday, and 35 minutes on Tuesday.

Step 2: Add up the time spent on each day

Now, we add up the time spent on Sunday, Monday, and Tuesday.

$$\text{total_time} = 38 + 38 + 35$$

Step 3: Calculate the total time

Perform the addition to find the total time.

$$\text{total_time} = 38 + 38 + 35 = 111 \text{ minutes}$$

Step 4: Evaluate the answer choices

Compare the calculated total time to the given answer choices to select the correct one. The choices are A: 105 minutes, B: 111 minutes, C: 106 minutes, D: 108 minutes.

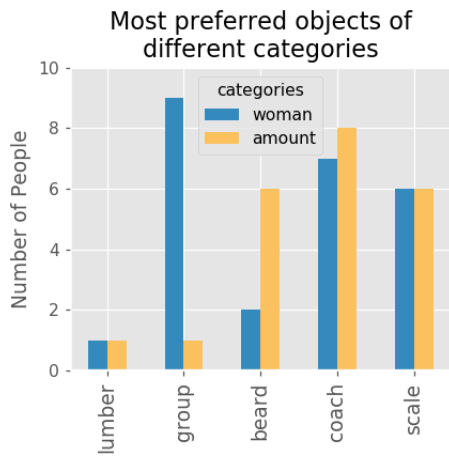
Step 5: Select the correct answer

Based on the calculation, the total time Cody spent on homework from Sunday to Tuesday is 111 minutes, which matches option B. The steps taken to solve this problem involved directly adding up the time Cody spent on homework from Sunday to Tuesday. The process was straightforward, and the calculation was simple. There were no alternative solutions needed since the question only required a basic arithmetic operation. The reflection on the steps confirms that the approach was correct and efficient for this type of problem.

think

B

Figure 12. Qualitative example from HONEYBEE dataset.



Question: How many total people preferred the object coach across all the categories?

Chain-of-Thought:

caption The image presents a bar graph illustrating the most preferred objects across different categories, with two distinct groups: "woman" and "amount." The graph features five categories on the x-axis: lumber, group, beard, coach, and scale.

Key Observations:

The y-axis represents the number of people, ranging from 0 to 10.

The graph displays two sets of bars for each category, one in blue (representing "woman") and the other in orange (representing "amount").

Notably, the "group" category has the highest value for "woman," while the "coach" category has the highest value for "amount."

Category-Specific Data:

Lumber: Woman: 1 Amount: 1

Group: Woman: 9 Amount: 1

Beard: Woman: 2 Amount: 6

Coach: Woman: 7 Amount: 8

Scale: Woman: 6 Amount: 6

Conclusion: The graph provides a clear visual representation of the preferred objects across different categories, allowing for easy comparison between the "woman" and "amount" groups.

think

Step 1: Identify the relevant data The image shows a bar graph with different categories on the x-axis and the number of people on the y-axis. The categories are lumber, group, beard, coach, and scale. We are interested in the category "coach".

To solve this question, I will first identify the values associated with the "coach" category for both woman and amount. Then, I will add these values to find the total number of people who preferred the object "coach" across all categories.

Step 2: Extract the values for the coach category For the "coach" category, the value for woman is 7 and the value for amount is 8.

I will now evaluate if these values are correctly identified and if they align with the question asked. The question is about the total number of people who preferred the object "coach", so I need to ensure I'm considering both woman and amount as they represent different aspects of preference.

Step 3: Calculate the total number of people To find the total, I add the number of woman who preferred coach (7) and the amount of people who preferred coach (8).

$$\text{total_people} = 7 + 8 = 15$$

I will reflect on my approach to ensure it's correct. The approach involves directly adding the values for woman and amount in the coach category. This seems straightforward and aligns with basic principles of data analysis. There are no obvious alternative solutions since the question directly asks for a sum.

Step 4: Provide the final answer Given the calculation, the total number of people who preferred the object "coach" across all categories is 15.

Figure 13. Qualitative example from HONEYBEE dataset.

Think about the reasoning process required to solve the question in detail based on the given image and then provide the answer. The reasoning process should be enclosed within `<think>` and `</think>` tags followed by the summary of the answer. In your reasoning, incorporate the following elements:

- Planning: Systematically outline the steps before executing them.
- Evaluation: Critically assess each intermediate step.
- Reflection: Reconsider and refine your approach as needed.
- Exploration: Consider alternative solutions.

If the question is multiple-choice, present the final answer as the option letter A, B, C, ... F. Enclose the final answer in `boxed`.
Question: `<Question>`

Figure 14. CoT generation prompt for a given (image, question) pair.

Help the user create a new math problem that inspired by the original one related to the provided image.

- Make sure the new problem is logical and can be solved.
- The new question should be directly based on the image, not an independent question.
- Do not solve the original question.
- Conclude your response with "New Question: ". If necessary, please include options or choices in the new question.
- Do not provide the answer in your new question.

Original Question: `<Question>`

Figure 15. Prompt to generate new question for a given (image, question) pair.

Provide a detailed caption of the given image.
Caption:

Figure 16. Prompt to generate a detailed caption for a given image.

Please rewrite the question shown in the image to include a total of 10 options.

- Rewrite the given question to include a total of 10 options. If the original question does not have any options, create and add 10 options to it.
- If the original question has options then the original options should be shuffled (e.g., content of option A goes to option D etc.)
- Rewrite the predicted answer in its entirety, maintaining the same step-by-step structure and content as the original answer.
- Ensure that the final answer should be in the format of `boxedOption Letter`, where "Option Letter" corresponds to the correct choice.
- Do not attempt to solve the question at any time, just focus on rewriting.

Question: `<Question>`
Answer: `<Answer>`
Respond with "Rewrite Question: " and "Rewrite Answer: ".

Figure 17. Prompt to rewrite the original (question, CoT) pair with distractors.

Code snippet to create text-rich images.

```
1 from PIL import Image, ImageDraw, ImageFont
2 import random
3
4 def wrap_text(draw, text, font, max_width):
5     """Wrap text to fit within max_width pixels"""
6     lines, line = [], []
7     for word in text.split():
8         test_line = ' '.join(line + [word])
9         if draw.textlength(test_line, font) <= max_width:
10            line.append(word)
11        else:
12            if line: lines.append(' '.join(line))
13            line = [word]
14        if line: lines.append(' '.join(line))
15    return lines
16
17 def get_text_rich_image(image_path, question, save_path, bg_colors, text_colors, fonts,
18 img_widths=[512]):
19     img_width, img_height = random.choice(img_widths), 2048
20     bg_color, text_color = random.choice(bg_colors), random.choice(text_colors)
21     font = ImageFont.truetype(random.choice(fonts), 16)
22     img = Image.new('RGB', (img_width, img_height), bg_color)
23     draw = ImageDraw.Draw(img)
24     margin, spacing = 20, 5
25     max_width = img_width - 2 * margin
26     wrapped = wrap_text(draw, question.replace("\n", " "), font, max_width)
27     -, -, -, h = draw.textbbox((0, 0), "Sample", font=font)
28
29     y = margin
30     for line in wrapped:
31         draw.text((margin, y), line, fill=text_color, font=font)
32         y += h + spacing
33
34     overlay = Image.open(image_path)
35     w = min(overlay.width, int(0.8 * img_width))
36     h = int(w * overlay.height / overlay.width)
37     overlay = overlay.resize((w, h), Image.Resampling.LANCZOS)
38
39     # Convert to RGBA if has transparency
40     if overlay.mode in ('RGBA', 'LA', 'P'):
41         overlay = overlay.convert('RGBA')
42         img.paste(overlay, (int(0.1 * img_width), y + margin), overlay)
43     else:
44         img.paste(overlay, (int(0.1 * img_width), y + margin))
45
46     final_height = y + h + 2 * margin
47     img = img.crop((0, 0, img_width, final_height))
48     img.save(save_path)
```