MathCoder-VL: Bridging Vision and Code for Enhanced Multimodal Mathematical Reasoning

Ke Wang[1] **Junting Pan**[1,2*] **Linda Wei**[1] **Aojun Zhou**[1] **Weikang Shi**[1] **Zimu Lu**[1]

Han Xiao[1] **Yunqiao Yang**[1] **Houxing Ren**[1] **Mingjie Zhan**[4†] **Hongsheng Li**[1,2,3†]

[1]Multimedia Laboratory (MMLab), The Chinese University of Hong Kong,
[2]CPII under InnoHK, [3]Shanghai AI Laboratory, [4]SenseTime

wangk@link.cuhk.edu.hk hsli@ee.cuhk.edu.hk

Abstract

Natural language image-caption datasets, widely used for training Large Multimodal Models, mainly focus on natural scenarios and overlook the intricate details of mathematical figures that are critical for problem-solving, hindering the advancement of current LMMs in multimodal mathematical reasoning. To this end, we propose leveraging code as supervision for cross-modal alignment, since code inherently encodes all information needed to generate corresponding figures, establishing a precise connection between the two modalities. Specifically, we co-develop our image-to-code model and dataset with model-in-the-loop approach, resulting in an image-to-code model, FigCodifier and `ImgCode-8.6M`, the largest image-code dataset to date. Furthermore, we utilize FigCodifier to synthesize novel mathematical figures and then construct `MM-MathInstruct-3M`, a high-quality multimodal math instruction fine-tuning dataset. Finally, we present MathCoder-VL, trained with `ImgCode-8.6M` for cross-modal alignment and subsequently fine-tuned on `MM-MathInstruct-3M` for multimodal math problem solving. Our model achieves a new open-source state-of-the-art across all six metrics. Notably, it surpasses GPT-4o and Claude 3.5 Sonnet in the geometry problem-solving subset of MathVista, achieving improvements of 8.9% and 9.2%.

1 Introduction

Recently, Large Language Models (LLMs) have outperformed humans in complex reasoning at the Olympiad competition level (OpenAI et al., 2024; DeepSeek-AI et al., 2025). However, the reasoning abilities of Large Multimodal Models (LMMs) still fall short of their potential, often struggling with even simple tasks, such as simple geometry problems (Wang et al., 2024b). Overcoming these limitations is essential for advancing toward Artificial General Intelligence (AGI).

In our efforts to enhance the mathematical capabilities of LMMs, we identify two key challenges that distinguish them from LLMs: (i) Aligning math-related visual and textual details accurately to enable effective problem-solving. (ii) Scaling the generation of diverse new math figures for multimodal math problem synthesis.

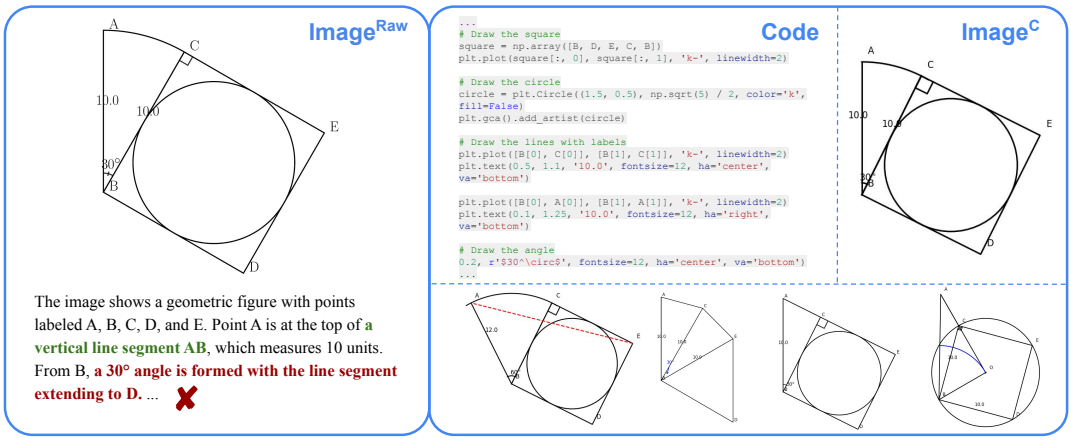
Despite significant advancements, LMMs still struggle with effective modality alignment, especially in the math field, primarily due to the scarcity of high-quality, error-free, math-specific cross-modal data. Traditional image caption datasets (Chen et al., 2023; Schuhmann et al., 2022) often focus on natural scenarios and lose details important for math problem-solving, and cannot guarantee correctness, as shown in Figure 1 (a).

In contrast, code inherently contains all information needed to render corresponding image and establish a strict correspondence between the two modalities. In light of this, we propose image-to-code mid-training to enhance math-related cross-modal alignment. We construct an image-to-code model, FigCodifier, which converts math-related images into detailed code capable of rendering new images, as shown in Figure 1 (b). By pairing the generated code with the rendered images, we create high-quality $\langle \mathbf{Image}^{\mathbf{C}}, \mathbf{Code} \rangle$ pairs that are inherently **always accurate and contain all details for cross-modal alignment**. Using this automated data engine, we construct `ImgCode-8.6M`, significantly enhancing LMMs' cross-modal ability.

Additionally, with a higher temperature, our FigCodifier can synthesize new images that are more different from the raw images, which enables the **synthesis of new diverse images** for problem-solving dataset construction. Synthetic data have proven effective for math reasoning (Wang et al., 2023a; Gou et al., 2024; Huang et al., 2024), and

*Project lead
†Corresponding author

Findings of the Association for Computational Linguistics: ACL 2025, pages 2505–2534



(a) Captioning example from MAVIS (b) <Code, Image^{C}> pair and corresponding synthetic images

Figure 1: (a) Natural language captions often struggle to convey all details in a image and guarantee correctness. (b) Our approach uses image-translated **Code** and code-generated $\mathbf{Image}^{\mathbf{C}}$ to create $\langle \mathbf{Image}^{\mathbf{C}}, \mathbf{Code} \rangle$ pairs. Since the $\mathbf{Image}^{\mathbf{C}}$ is rendered from the **Code**, the cross-modal alignment is always accurate and contains all the details. Below are four examples of new figures synthesized based on $\mathbf{Image}^{\mathbf{Raw}}$.

dataset quality and diversity are the most important factors. However, the construction of multimodal math problem-solving datasets still relies heavily on either question rewriting and generating new solutions (Guo et al., 2024; Luo et al., 2025), sourcing existing images (Shi et al., 2024; Peng et al., 2024), or manually designed figures (Zhuang et al., 2024; Zhang et al., 2025b). The diversity of images lags significantly behind the diversity of text, restricting the overall dataset variety. Unlike these methods, with our FigCodifier, generating new images becomes significantly easier, as shown in Figure 1 (b). This allows us to create diverse new math figures at low cost, which has the potential to improve LMMs' mathematical reasoning abilities substantially. Our main contributions are:

1. We co-develop our image-to-code model with model-in-the-loop approach, resulting in a FigCodifier model and `ImgCode-8.6M` dataset, the largest image-code dataset to date.

2. With our FigCodifier, we construct `MM-MathInstruct-3M`. To our knowledge, this is the first high-quality multi-modal problem-solving dataset with not only new questions but also diverse newly synthesized images.

3. We present MathCoder-VL, achieving SOTA results across all six metrics among comparable-size LMMs. We will open-source our models, code and datasets.

2 Related Works

Multimodal Math Reasoning The mathematical reasoning abilities of LMMs have garnered widespread attention (Gao et al., 2023; Li et al., 2024; Dong et al., 2024b; Hu et al., 2024; Yang et al., 2024c; Han et al., 2024; Guo et al., 2024; Shao et al., 2024; Zong et al., 2024b). Unlike mathematical reasoning tasks in traditional large language models (Zhou et al., 2024; Luo et al., 2023; Yu et al., 2023; Sharma et al., 2024), multimodal mathematical reasoning requires LMMs to extract information from the visual domain and perform cross-modal reasoning. Tasks such as geometric problem-solving are particularly challenging (Chen et al., 2021; Wang et al., 2024b). Several studies have attempted to enhance the input of visual mathematical signals by enhancing visual encoders (Liu et al., 2024a; Chen et al., 2024a). However, ensuring accurate correspondence between images and text remains a significant challenge. To address this, we propose using code and code-generated images, which inherently maintain precise and sufficient alignment between modalities.

Data Synthesis. Methods based on data synthesis are favored by academia and industry due to their demonstrated efficiency (Sprague et al., 2024; Lu et al., 2023b; Huang et al., 2024; Fu et al., 2024; Zong et al., 2024a; Ma et al., 2024). Numerous fine-tuning (Yu et al., 2024; Wang et al., 2023a; Lu et al., 2024b) and pretraining (Gunasekar et al., 2023; Wang et al., 2023b; Yang et al., 2024a) studies have explored training on synthetic data generated using language models or predefined templates. MathGLM (Yang et al., 2023) and InternLM-Math (Ying et al., 2024) use templates to generate synthetic numerical operation data, while Phi (Gunasekar

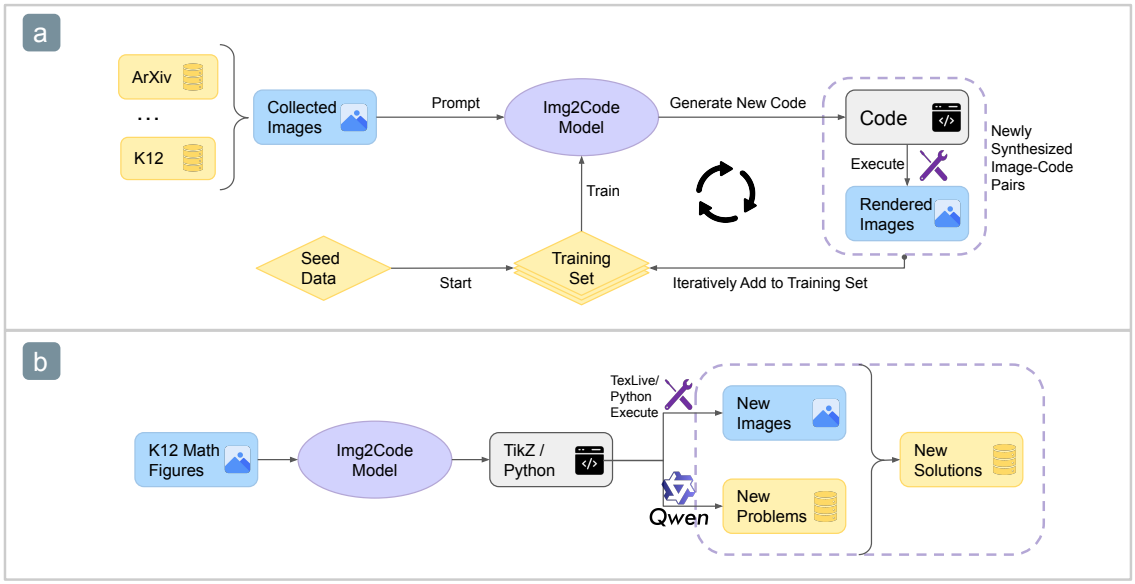


Figure 2: (a) The iterative training pipeline of our image-to-code model. We use DaTikZ-119K as seed data to train our first image-to-code model. We start by collecting 3 million math-related images and ultimately synthesize 8.6 million image-code pairs. Our final image-to-code model, FigCodifier, is based on InternVL2-8B (Chen et al., 2024b), with all model parameters being fully learnable. (b) The pipeline for generating new math problems with diverse new images. Using the final model from (a), we convert raw images into code and leverage Qwen models to generate new questions and step-by-step solutions based on the newly synthesized images.

et al., 2023) produces textbook-quality data with models. EntiGraph (Yang et al., 2024d) generates diverse text by drawing connections between sampled entities. However, efforts on the synthesis of multimodal mathematical reasoning data are primarily focused on the diversity and complexity of problem or solution text. Math-LLaVA (Shi et al., 2024) proposes the MathV360K dataset by classifying images based on complexity and enhancing questions accordingly. R-CoT (Deng et al., 2024), GeoGPT4V (Cai et al., 2024), MammoTH-VL (Guo et al., 2024), and Multimath (Peng et al., 2024) collect and enhance problems or solutions. MAVIS (Zhang et al., 2025b) generates new geometry and function images with code but lacks diversity, as the codes are design by humans and only contain three types. Our work proposes a novel method that can synthesize diverse new images automatically for crafting problems.

3 MathCoder-VL

We developed MathCoder-VL through a two-stage process: image-to-code mid-training using the `ImgCode-8.6M` dataset, followed by math instruction fine-tuning on `MM-MathInstruct-3M`. This section details the construction of the two datasets.

3.1 Image-to-Code Model and Data

To synthesize image-code pairs and new images, we need models that can generate code to render high-quality mathematical figures. However, even commercial models like Claude 3.5 and GPT-4 struggle to perform image-to-code conversion effectively (Belouadi et al., 2024b). Additionally, the largest TikZ dataset to date, DaTikZ (Belouadi et al., 2024a), contains only 119k TikZ graphics. To address these limitations, we build `ImgCode-8.6M` and develop our FigCodifier.

3.1.1 Collect Math-related Images

We start by collecting 3 million math-related images, of which 164K are paired with corresponding TikZ code. The data composition is as follows.

DaTikZ Training Set. DaTikZ is designed to facilitate the development of machine learning models capable of generating or manipulating vector graphics in LaTeX. We use the 119K image-TikZ code pairs from DaTikZ as our seed data.

K12 Problem-Solving Dataset. To diversify our dataset, we included math problems from K12 books, exercises, and exams with permission from the data providers. We gathered 4.6 million math problems, of which 996K include at least one image. This dataset contains 1.57 million images

from a wide range of math problems across all K12 grades, spanning 19 subjects, including Statistics, Probability, Algebra, Geometry, Functions, Permutations, Combinations, and more. See Sec. 3.2.1 for details on the curation process.

Mathematical Textbooks. Textbooks provide structured presentations of math concepts and are a valuable resource. We collected 8K PDFs of math-related textbooks from online sources, focusing on titles with keywords like algebra, geometry, and probability. These PDFs were converted into markdown format, and the images were extracted, resulting in 202K diverse math-related images.

ArXiv. We utilized bulk data from arXiv between September 2023 and October 2024, yielding 45K images with corresponding TikZ code and 681K images without code, many of which are statistical visualizations.

Open-Source Datasets. MathV360K (Shi et al., 2024) consists of 360K question-answer pairs and 40K images from 24 previous datasets. MultiMath (Peng et al., 2024) contains 300K newly collected math problems with 280K images, mostly consisting of geometry diagrams.

3.1.2 Iteratively Build Image-to-Code Model

We train our first image-to-code model using 119K image-TikZ pairs sourced from DaTikZ, leveraging InternVL-Chat-V1-2-40B (Chen et al., 2024b). As the dataset scales beyond one million samples, we adopt InternVL2-8B (Chen et al., 2024a) as the base model after comprehensively weighing the image-to-code performance and cost. The complete training pipeline is illustrated in Figure 2 (a).

Synthesis of Image-Code Pairs. To scale the size of Image-Code pairs, we used the image-to-code model to translate the 3M collected images into corresponding code. We then run the generated code to render new images, and only the successfully generated $\langle \mathbf{Image}^{\mathbf{C}}, \mathbf{Code} \rangle$ pairs were included in our dataset. This iterative process allowed us to continually generate fresh $\langle \mathbf{Image}^{\mathbf{C}}, \mathbf{Code} \rangle$ pairs and refine the model with each new version. Ultimately, we get FigCodifier and the `ImgCode-8.6M`.

TikZ to Python Conversion. In addition to TikZ code, we also leverage GPT-4o mini to translate TikZ code into Python code, which is then executed to generate new images. This step significantly expands our dataset, further enhancing the model's capabilities. By diversifying the types of code used, the model can generate a broader range of images, as different code structures produce distinct visual outputs. Through this process, we curate 3.1 million image-Python pairs.

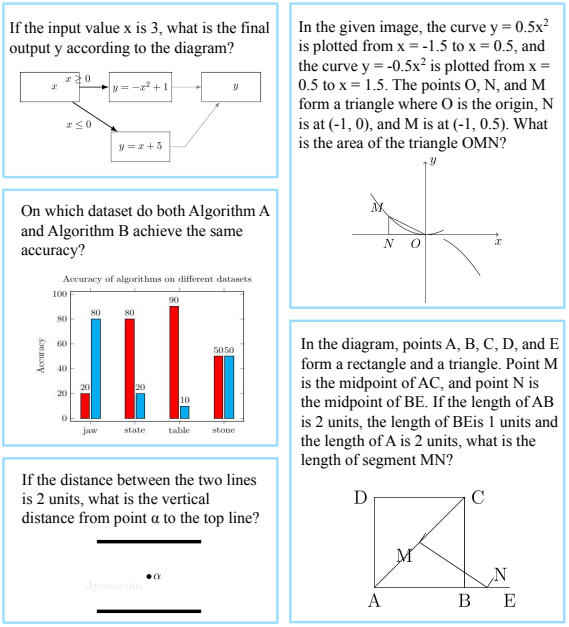


Figure 3: Sample questions paired with newly synthesized images, as generated in Figure 2 (b).

Data Cleaning and Deduplication. We implement a rigorous cleaning and deduplication process to ensure data quality: 1. Code Validation: We only retain code that generates a valid image. Over the course of the iterative process, the code success rate improves, rising from 46.5% for TikZ to 81.2% for TikZ and 84.5% for Python on the DaTikZ test set. 2. Deduplication: We apply carefully designed rules to eliminate duplicate or highly similar code, removing 4.4% of the dataset. 3. Quality Filtering: Through keyword matching, we filter out low-quality data, such as randomly generated or irrelevant images, which accounts for 3.7% of the data. 4. Code Length: We remove code that is excessively long, which can introduce unnecessary complexity. 5. Image Quality: Images that are almost entirely white—identified through standard deviation and mean pixel value analysis—are removed, accounting for approximately 0.5% of the data. Details of this process can be found in Appendix A. After cleaning, we retain 4.3M image-TikZ pairs and 4.3M image-Python pairs.

3.2 Math Instruction Fine-tuning Data

In this section, we introduce the construction of our `MM-MathInstruct-3M` as shown in Figure 2 (b).

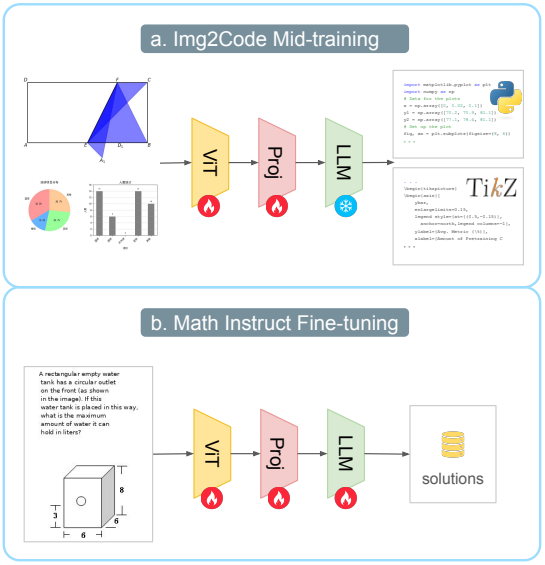


Figure 4: Two training stages of MathCoder-VL.

3.2.1 Construction of K12-2M Dataset

We collected 4.6 million math problems with simple solutions, where the equations are in image format. First, we distinguish math figures from equations based on their size, as equations tend to be much smaller. Next, we convert the equations into LaTeX text using MinerU (Wang et al., 2024a). This process results in 2 million samples containing at least one actual image. To enhance data quality, we then use GPT-4o mini to translate the original simple solutions into detailed, step-by-step CoT solutions, ultimately resulting in K12-2M.

3.2.2 Synthetic Math Data with New Images

To generate new multi-modal math problems, we follow a structured approach:

Newly Synthesized Images. We leverage the 1.57 million raw images from K12-2M, using our FigCodifier with a temperature of 0.7 to generate new math figures. With a higher temperature, the model can produce images that diverge more from the raw dataset. More examples of the newly synthesized images are shown in Appendix B.5.

Questions Based on New Images. From the 1.1 million newly generated image-code pairs, we use Qwen2.5-72B-Instruct (Team, 2024b) to craft math reasoning questions appropriate for a K12 audience. These questions are based on the visual elements (such as patterns, shapes, and numbers) present in each image. The questions are designed to be concise, self-contained, and to engage the reasoning skills of the reader. At this stage, the model is not required to provide answers to the questions. Details can be found in Appendix B.5.

Synthesize Solutions. For generating solutions, we employ both Qwen2.5-Math-72B-Instruct (Yang et al., 2024b) and Qwen2.5-72B-Instruct (Team, 2024b). Each model independently attempts to solve the question, taking both the question and image code as inputs. We retain a solution only if both models produce consistent answers, assuming that there is typically one correct answer and multiple possible incorrect ones. The solution pass rate is 51%. Following the data cleaning procedure outlined in Section 3.1.2, we remove duplicates and overly long samples. The final output consists of 1 million new samples, some of which are illustrated in Figure 3.

4 Experiments

In this section, we introduce our two-stage training approach: image-to-code mid-training with `ImgCode-8.6M`, followed by math instruction fine-tuning with `MM-MathInstruct-3M`.

4.1 Training Stages

As illustrated in Figure 4, the training process for a single MathCoder-VL model consists of two stages aimed at improving the model's math-related visual perception and multimodal reasoning capabilities.

Image-to-Code Mid-training. In this stage, we use `ImgCode-8.6M` to improve cross-modal alignment between mathematical diagrams and language embedding spaces. Both the vision encoder and MLP projector are trainable during this phase. The primary objective is to enhance the vision encoder's ability to extract mathematic visual features. Since the correspondence between code and image is highly accurate and contains all the detailed information, this stage allows the model to capture intricate patterns, especially those related to mathematics. These math-related patterns, including geometric shapes, process flows, and other mathematical representations, are underrepresented in large web-scale datasets like LAION-5B (Schuhmann et al., 2022). Importantly, we freeze the LLM backbone during this stage to preserve its general language abilities, as we do not require it to generate code for downstream tasks.

Math Instruction Fine-tuning. In this stage, as shown in Figure 4, the entire model is fine-tuned on our high-quality multimodal math problem-solving dataset, `MM-MathInstruct-3M`. This dataset includes 3 million samples, with 1 million gener-

Model	#Params	MATH-Vision (Test)	MathVerse (Testmini)	MathVista (GPS)	GAOKAO-MM (Math)	We-Math (S1)	We-Math (S2)	We-Math (S3)
Random Chance	-	7.2	12.4	21.6	-	-	-	-
Human	-	68.8	64.9	48.4	-	-	-	-
Closed-source LMMs								
Qwen-VL-Plus (Bai et al., 2023)	-	10.8	21.3	35.5	33.8			
Qwen-VL-Max (Bai et al., 2023)	-	15.6	35.9	46.1	-	40.8	30.3	20.6
GPT-4V (OpenAI, 2023)	-	22.8	39.4	50.5	45.0	65.5	49.2	38.2
GPT-4-turbo (OpenAI, 2024a)	-	30.3	43.5	58.3	50.0	-	-	-
GPT-4o (OpenAI, 2024b)	-	30.4	50.8	64.7	-	72.8	58.1	43.6
Claude3-Opus (Anthropic, 2024)	-	27.1	31.8	52.9	-	-	-	-
Claude3.5-Sonnet (Anthropic, 2024)	-	37.9	49.0	64.4	-	-	-	-
Gemini-1.5-Pro (Team, 2024a)	-	19.2	51.1	58.9	-	56.1	51.4	33.9
Open-source LMMs								
LLaVA-1.5-13B (Liu et al., 2024b)	13B	11.0	12.7	22.7	16.3	35.4	30.0	32.7
SPHINX-V2-13B (Lin et al., 2023)	13B	-	16.1	16.4	-	-	-	-
IXC-2-VL (Dong et al., 2024a)	7B	14.5	25.9	63.0	-	47.0	33.1	33.3
Deepseek-VL (Lu et al., 2024a)	8B	-	19.3	28.4	20.0	32.6	26.7	25.5
Qwen2-VL (Wang et al., 2024c)	8B	19.2	33.6	40.9	25.0	59.1	43.6	26.7
InternVL-Chat-2B-V1-5 (Gao et al., 2024)	2B	15.3	23.1	37.5	17.5	34.3	26.1	20.0
InternVL2-8B (Chen et al., 2024a)	8B	20.0	35.9	62.0	32.5	59.4	43.6	35.2
InternVL2-26B (Chen et al., 2024a)	26B	23.1	40.0	54.3	33.4	51.0	39.2	46.1
InternVL2-76B (Chen et al., 2024a)	76B	23.6	42.8	67.8	41.2	65.2	49.4	49.1
IXC-2.5-Reward (Zang et al., 2025)	7B	19.0	18.8	63.5	-	44.4	35.3	27.9
Open-source Math LMMs								
G-LLaVA-7B (Gao et al., 2023)	7B	-	16.6	48.7	-	32.4	30.6	32.7
Math-LLaVA-13B (Shi et al., 2024)	13B	15.7	22.9	57.7	-	38.7	34.2	34.6
InfiMM-Math (Han et al., 2024)	7B	-	34.5	-	-	-	-	-
MathGLM-Vision-9B (Yang et al., 2024c)	9B	19.2	44.2	64.4	-	-	-	-
Math-PUMA-Qwen2 (Zhuang et al., 2024)	8B	14.0	33.6	48.1	-	53.3	39.4	36.4
Math-PUMA-DS (Zhuang et al., 2024)	7B	-	31.8	39.9	-	45.6	38.1	33.9
Multimath-7B (Peng et al., 2024)	7B	16.3	27.7	66.8	-	-	-	-
MAVIS-7B (Zhang et al., 2025b)	7B	19.2	35.2	64.1	-	57.2	37.9	34.6
MathCoder-VL-2B	2B	21.7	35.4	66.4	37.5	52.0	42.2	38.8
Δ *Over Base Model*		+6.4	+12.3	+28.9	+20.0	+17.7	+16.1	+18.8
MathCoder-VL-8B	8B	26.1	46.5	73.6	51.2	65.4	58.6	52.1
Δ *Over Base Model*		+6.1	+10.6	+11.6	+18.7	+6.0	+15.0	+16.9

Table 1: Comparison of model performances across various math benchmarks. MATH-Vision (Wang et al., 2024b), MathVerse (Zhang et al., 2025a), MathVista (Lu et al., 2023a), and We-Math (Qiao et al., 2024) are in English, while GAOKAO-MM (Zong and Qiu, 2024) is in Chinese. The best results of closed-source LMMs are highlighted in red. The best and second-best results of open-source LMMs are highlighted in blue and green respectively. (GPS: geometry problem solving, S1: one-step problems, S2: two-step problems, S3: three-step problems)

ated by our image-to-code model-based data engine. To the best of our knowledge, this is the first data engine capable of generating multimodal math problem-solving data that includes not only new textual content but also new diverse math figures.

4.2 Experimental Setup

We use InternVL-Chat-2B-V1-5 (Gao et al., 2024) and InternVL2-8B (Chen et al., 2024a) as the base models for our experiments.

Implementation Details. We train the model for one epoch across two stages. In the first stage, we use a batch size of 1024 and a learning rate of 2e-5. In the second stage, we use a batch size of 512 and a learning rate of 4e-5. To efficiently train the computationally intensive models, we utilize DeepSpeed at ZeRO-1 stage (Rajbhandari et al., 2020) and flash attention (Dao et al., 2022). The 2B and 8B models are trained on 32 and 64 NVIDIA A800 80GB GPUs, respectively. To ensure reproducibility, we fix the random seed and employ greedy decoding during testing.

Benchmarks. We assess our models across a diverse set of widely recognized mathematical benchmarks. The MATH-Vision (Wang et al., 2024b) dataset includes 3,040 visually contextualized math problems sourced from real-world competitions. MathVista (Lu et al., 2023a) is a well-known dataset designed for evaluating reasoning in visual contexts. MathVerse (Zhang et al., 2025a) emphasizes core mathematical skills such as plane geometry, solid geometry, and functions. GAOKAO-MM (Zong and Qiu, 2024) is based on the Chinese College Entrance Examination. Many tasks in MathVista require more emphasis on natural image recognition rather than math reasoning abilities (Wang et al., 2024b), so we only report results on the Geometry Problem Solving (GPS) sub-

Model	MATH-V Geometry			Average
	angle	area	length	
Closed-source LMMs				
GPT-4o	17.3	29.8	30.1	25.7
GPT-4V	22.0	22.2	20.9	21.7
Gemini-1.5-Pro	14.5	14.4	16.5	15.1
Open-source LMMs				
Qwen2-VL-8B	19.1	22.4	22.5	21.3
InternVL2-8B	20.8	22.4	20.5	21.2
InternVL2.5-8B	22.0	19.4	15.4	18.9
Open-source Math LMMs				
Math-LLaVA-13B	20.2	18.4	17.6	18.7
Multimath-7B	20.1	16.4	21.3	19.3
Math-PUMA-8B	11.7	15.8	12.2	13.2
MathCoder-VL-8B	48.6	32.2	32.1	37.6

Table 2: Comparison of model performances on the three plane geometry subsets of MATH-Vision (Wang et al., 2024b). The best and second-best results are highlighted in red and blue respectively.

set. Collectively, these datasets cover a wide spectrum of mathematical challenges, ranging from elementary word problems to advanced college-level exercises in both English and Chinese, providing a comprehensive evaluation of model capabilities.

Baselines. We compare our approach against a range of base models with strong mathematical capabilities and similar sizes. Our selected baselines include both closed-source and open-source LMMs. Both general LMMs and math-focused LMMs are incorporated. For general LMMs, we include powerful models like GPT-4o (OpenAI, 2024b), Qwen2-VL (Wang et al., 2024c) and IXC-2.5-Reward (Zang et al., 2025). For math-focused LMMs, we choose recent models such as MathGLM-Vision (Yang et al., 2024c), Math-PUMA (Zhuang et al., 2024), Multimath (Peng et al., 2024), and MAVIS (Zhang et al., 2025b).

4.3 Main Results

We evaluate MathCoder-VL across several benchmarks, analyzing its performance from the perspectives of mathematical subjects and input modalities.

Overall Performances. As shown in Table 1, MathCoder-VL demonstrates strong performance across multiple mathematical benchmarks, particularly in comparison to other open-source models. MathCoder-VL-8B achieves the highest accuracy among open-source LMMs of similar sizes, with 26.1% on MATH-Vision, 46.5% on MathVerse, and an impressive 73.6% on the MathVista (GPS). These results show a notable improvement over its base model, InternVL2-8B, by 6.1%, 10.6%, and 11.6% on the respective benchmarks. The smaller model also demonstrates strong capabilities, with MathCoder-VL-2B outperforming MathGLM-Vision-9B by 2.5% and Multimath-7B by 5.4% on MATH-Vision. MathCoder-VL-8B significantly outperforms InternVL2-76B, with a gap of 2.5% on MATH-Vision, 3.7% on MathVerse, 5.8% on MathVista (GPS), and 10.0% on GAOKAO-MM Math. The model's performance in Chinese is also noteworthy, with MathCoder-VL-8B reaching 51.2% on GAOKAO-MM, outperforming all other open-source LMMs.

Compared to closed-source models, MathCoder-VL-8B remains competitive, outperforming several proprietary models. It surpasses GPT-4V on all four benchmarks and exceeds GPT-4-turbo by 3.0% on MathVerse. It also outperforms the newest Claude3.5-Sonnet (64.4% vs 73.6%) and GPT-4o (64.7% vs 73.6%) on MathVista (GPS). However, it still falls short of top-tier closed-source LMMs in some areas. For example, it lags behind GPT-4o by 3.0% on MATH-Vision.

Performance on multi-step problems. MathCoder-VL-8B exhibits robust performance on multi-step problems, outperforming GPT-4o on both two-step (58.6% vs 58.1%) and three-step problems (52.1% vs 43.6%) on We-Math (Qiao et al., 2024). Our `MM-MathInstruct-3M`, which provides step-by-step solutions for every problem, enhances the model's Chain-of-Thought (Wei et al., 2022) reasoning ability. Notably, MathCoder-VL-8B surpasses InternVL2-76B by a significant margin, achieving a 20.7% improvement on two-step problems and a 3.0% improvement on three-step problems, while only slightly edging it out by 0.2% on one-step problems. This demonstrates that, as a math-specific language model, MathCoder-VL excels over general open-source models, particularly on complex problems.

Outstanding Ability in Geometry. When evaluating MathCoder-VL's capabilities in geometry, its performance on the MathVista (GPS) stands out. Additionally, we present the detailed accuracy of the model on the plane geometry subsets from MATH-Vision, as shown in Table 2. MathCoder-VL excels across all three plane geometry subsets in MATH-V, achieving an impressive average score of 37.6%, which surpasses GPT-4o by 11.9%. Notably, the model scored exceptionally well in each of the three subsets—angle, area, and length—with scores of 48.6%, 32.2%, and 32.1%, respectively.

Model	Image-to-Code Mid-training	Math Instruction Fine-tuning	MATH-Vision (Test)	MathVerse (Testmini)	MathVista (Testmini)	MathVista (GPS)	GAOKAO-MM (Math)
InternVL-Chat-2B-V1-5	✗	✗	15.3	23.1	41.1	37.5	17.5
	✗	K12-2M	20.3 +5.0	27.2 +4.1	37.0 -4.1	45.7 +8.2	30.0 +12.5
	✓	K12-2M	22.0 +1.7	33.0 +5.8	39.4 +2.4	64.4 +18.7	33.8 +3.8
MathCoder-VL-2B	✓	K12-2M + New-1M	21.7 -0.3	35.4 +2.4	44.4 +5.0	66.4 +2.0	37.5 +3.7

Table 3: Ablation study of image-to-code mid-training and math instruction fine-tuning dataset on MathCoder-VL-2B. K12-2M + New-1M dataset is our `MM-MathInstruct-3M`.

Mid-training	Fine-tuning	MathVerse TD	TL	VD	VO	All
✗	✗	27.5	25.8	20.1	18.1	23.1
✗	2M	36.7	30.7	25.3	15.9	27.2
✓	2M	40.9	34.5	31.1	26.9	33.0
		+4.2	+3.8	+5.8	+11.0	+5.8
✗	2M+1M	40.7	32.4	30.1	19.8	30.8
✓	2M+1M	43.7	36.9	34.1	27.2	35.4
		+3.0	+4.5	+4.0	+7.4	+4.6

Table 4: Effects of image-to-code mid-training on model performances with varying degrees of input content in multi-modality on MathVerse (Zhang et al., 2025a).

This superior performance can be attributed to MathCoder-VL's enhanced understanding of geometry figures, enabling it to effectively process and interpret geometric shapes and measurements.

4.4 Ablation Study

In this session, we analyze the impact of various components of the training pipeline.

Ablation on Impact of Image-to-Code Mid-training. From Table 3, we can observe the impact of image-to-code mid-training on the model's reasoning ability. Comparing the results without mid-training to those with mid-training, performance improvements are noted in MATH-Vision (+1.7%), MathVerse (+5.8%), MathVista (GPS) (+18.7%), and GAOKAO-MM (Math) (+3.8%), highlighting its contribution to enhanced multi-modal mathematical reasoning. The most significant gain is observed in MathVista (GPS), suggesting that image-to-code mid-training strengthens spatial and graphical problem-solving capabilities and improves understanding of geometry figures.

Ablation on Impact of Input Modality. Table 4 illustrates the impact of image-to-code mid-training on MathVerse across different modality dominance levels: Text-Dominant (TD), Text-Lite (TL), Vision-Dominant (VD), and Vision-Only (VO). Across all categories, mid-training with image-to-code leads to improved performance, with an overall gain of 5.8% and 4.6%. Notably, the largest improvement is seen in the VO setting, where performance increases by 11.0% and 7.4%, indicating that image-to-code mid-training significantly enhances the model's ability to process purely visual inputs, while the smallest improvements are observed in TD (+3.8%) and TL (+3.0%). This suggests that image-to-code mid-training effectively enhances multi-modal reasoning, particularly in scenarios where vision plays a more dominant role.

Ablation on Impact of Newly Synthesized Images. As shown in Table 3, the MathCoder-VL-2B model generally benefits from the math instruction fine-tuning dataset based on newly synthesized images. Performance improvements are observed across multiple benchmarks: MathVerse (+2.4%), MathVista-Testmini (+5.0%), MathVista-GPS (+2.0%), and GAOKAO-MM (Math) (+3.7%), with only a slight decrease on MATH-Vision of 0.3%. Notably, MathVista shows a significant increase of 5.0%, suggesting that the new synthetic math problems contribute to a broader diversity of instructions. This enhanced diversity likely improves the model's generalization capabilities, particularly as many tasks in MathVista differ substantially from traditional math problem-solving.

5 Conclusion

In this paper, we propose a model-based multi-modal data engine. Using this data engine, we construct two datasets: `ImgCode-8.6M` for accurate cross-modal alignment and `MM-MathInstruct-3M`, a math problem-solving dataset featuring diverse newly synthesized images. Leveraging these datasets, we develop MathCoder-VL-2B and 8B models trained with image-to-code mid-training and math instruction fine-tuning. MathCoder-VL achieves a new state-of-the-art among open-source models for multi-modal mathematical reasoning.

6 Limitations

One limitation of our work is that `MM-MathInstruct-3M` focuses primarily on mathematics and does not intentionally include other STEM subjects, such as physics and chemistry. Additionally, our dataset consists entirely of English text and does not incorporate math-related content in other languages, such as Chinese. Due to computational resource constraints, we only trained 2B and 8B models. Future work could address these limitations by expanding the dataset to include other subjects and languages and by training larger language models. Furthermore, this paper primarily focuses on image-to-code mid-training and math instruction fine-tuning, so we did not apply reinforcement learning methods, such as GRPO, in the post-training phase, which could further improve performance on mathematical reasoning tasks. In the future, we plan to explore these methods with MathCoder-VL.

7 Acknowledgements

This project is supported in part by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd under the Innovation and Technology Commission (ITC)'s InnoHK, and in part by NSFC-RGC Project N_CUHK498/24. Hongsheng Li is a PI of CPII under the InnoHK.

References

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. `https://www.anthropic.com/claude-3-model-card`. Claude-3 Model Card.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*.

Jonas Belouadi, Anne Lauscher, and Steffen Eger. 2024a. AutomaTikZ: Text-guided synthesis of scientific vector graphics with TikZ. In *The Twelfth International Conference on Learning Representations*.

Jonas Belouadi, Simone Paolo Ponzetto, and Steffen Eger. 2024b. DeTikZify: Synthesizing graphics programs for scientific figures and sketches with TikZ. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Shihao Cai, Keqin Bao, Hangyu Guo, Jizhi Zhang, Jun Song, and Bo Zheng. 2024. Geogpt4v: Towards geometric multi-modal large language models with geometric image generation. *arXiv preprint arXiv:2406.11503*.

Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric Xing, and Liang Lin. 2021. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 513–523.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. 2023. Sharegpt4v: Improving large multimodal models with better captions. *Preprint*, arXiv:2311.12793.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. 2024a. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 67(12):220101.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024b. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198.

Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Re. 2022. Flashattention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu,

Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Linger Deng, Yuliang Liu, Bohan Li, Dongliang Luo, Liang Wu, Chengquan Zhang, Pengyuan Lyu, Ziyang Zhang, Gang Zhang, Errui Ding, et al. 2024. R-cot: Reverse chain-of-thought problem generation for geometric reasoning in large multimodal models. *arXiv preprint arXiv:2410.17885*.

Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Bin Wang, Linke Ouyang, Xilin Wei, Songyang Zhang, Haodong Duan, Maosong Cao, et al. 2024a. Internlm-xcomposer2: Mastering free-form text-image composition and comprehension in vision-language large model. *arXiv preprint arXiv:2401.16420*.

Yuhao Dong, Zuyan Liu, Hai-Long Sun, Jingkang Yang, Winston Hu, Yongming Rao, and Ziwei Liu. 2024b. Insight-v: Exploring long-chain visual reasoning with multimodal large language models. *arXiv preprint arXiv:2411.14432*.

Chaoyou Fu, Haojia Lin, Zuwei Long, Yunhang Shen, Meng Zhao, Yifan Zhang, Shaoqi Dong, Xiong Wang, Di Yin, Long Ma, et al. 2024. Vita: Towards open-source interactive omni multimodal llm. *arXiv preprint arXiv:2408.05211*.

Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, et al. 2023. G-llava: Solving geometric problem with multi-modal large language model. *arXiv preprint arXiv:2312.11370*.

Zhangwei Gao, Zhe Chen, Erfei Cui, Yiming Ren, Weiyun Wang, Jinguo Zhu, Hao Tian, Shenglong Ye, Junjun He, Xizhou Zhu, et al. 2024. Mini-internvl: A flexible-transfer pocket multimodal model with 5% parameters and 90% performance. *arXiv preprint arXiv:2410.16261*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2024. Tora: A tool-integrated reasoning agent for mathematical problem solving. *Preprint*, arXiv:2309.17452.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need. *Preprint*, arXiv:2306.11644.

Jarvis Guo, Tuney Zheng, Yuelin Bai, Bo Li, Yubo Wang, King Zhu, Yizhi Li, Graham Neubig, Wenhu Chen, and Xiang Yue. 2024. Mammoth-vl: Eliciting multimodal reasoning with instruction tuning at scale. *arXiv preprint arXiv:2412.05237*.

Xiaotian Han, Yiren Jian, Xuefeng Hu, Haogeng Liu, Yiqi Wang, Qihang Fan, Yuang Ai, Huaibo Huang, Ran He, Zhenheng Yang, et al. 2024. Infimm-webmath-40b: Advancing multimodal pre-training for enhanced mathematical reasoning. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, and Ranjay Krishna. 2024. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. *arXiv preprint arXiv:2406.09403*.

Yiming Huang, Xiao Liu, Yeyun Gong, Zhibin Gou, Yelong Shen, Nan Duan, and Weizhu Chen. 2024. Key-point-driven data synthesis with its enhancement on mathematical reasoning. *arXiv preprint arXiv:2403.02333*.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. 2024. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Ziyi Lin, Chris Liu, Renrui Zhang, Peng Gao, Longtian Qiu, Han Xiao, Han Qiu, Chen Lin, Wenqi Shao, Keqin Chen, et al. 2023. Sphinx: The joint mixing of weights, tasks, and visual embeddings for multi-modal large language models. *arXiv preprint arXiv:2311.07575*.

Dongyang Liu, Renrui Zhang, Longtian Qiu, Siyuan Huang, Weifeng Lin, Shitian Zhao, Shijie Geng, Ziyi Lin, Peng Jin, Kaipeng Zhang, et al. 2024a. Sphinx-x: Scaling data and parameters for a family of multi-modal large language models. *arXiv preprint arXiv:2402.05935*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024b. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306.

Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. 2024a. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2023a. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Yingzhou Lu, Minjie Shen, Huazheng Wang, Xiao Wang, Capucine van Rechem, Tianfan Fu, and Wenqi Wei. 2023b. Machine learning for synthetic data generation: a review. *arXiv preprint arXiv:2302.04062*.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024b. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. *Preprint*, arXiv:2402.16352.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Ruilin Luo, Zhuofan Zheng, Yifan Wang, Yiyao Yu, Xinzhe Ni, Zicheng Lin, Jin Zeng, and Yujiu Yang. 2025. Ursa: Understanding and verifying chain-of-thought reasoning in multimodal mathematics. *Preprint*, arXiv:2501.04686.

Bingqi Ma, Zhuofan Zong, Guanglu Song, Hongsheng Li, and Yu Liu. 2024. Exploring the role of large language models in prompt encoding for diffusion models. *arXiv preprint arXiv:2406.11831*.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

OpenAI. 2023. GPT-4V(ision) system card.

OpenAI. 2024a. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

OpenAI. 2024b. GPT-4o system card.

Shuai Peng, Di Fu, Liangcai Gao, Xiuqin Zhong, Hongguang Fu, and Zhi Tang. 2024. Multimath: Bridging visual and mathematical reasoning for large language models. *Preprint*, arXiv:2409.00147.

Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Zhuoma GongQue, Shanglin Lei, Zhe Wei, Miaoxuan Zhang, et al. 2024. We-math: Does your large multimodal model achieve human-like mathematical reasoning? *arXiv preprint arXiv:2407.01284*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. *Preprint*, arXiv:1910.02054.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. 2022. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. 2024. Visual cot: Unleashing chain-of-thought reasoning in multi-modal language models. *arXiv e-prints*, pages arXiv–2403.

Aditya Sharma, Aman Dalmia, Mehran Kazemi, Amal Zouaq, and Christopher J. Pal. 2024. Geocoder: Solving geometry problems by generating modular code through vision-language models. *Preprint*, arXiv:2410.13510.

Wenhao Shi, Zhiqiang Hu, Yi Bin, Junhua Liu, Yang Yang, See-Kiong Ng, Lidong Bing, and Roy Ka-Wei Lee. 2024. Math-llava: Bootstrapping mathematical reasoning for multimodal large language models. *Preprint*, arXiv:2406.17294.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *arXiv preprint arXiv:2409.12183*.

Gemini Team. 2024a. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *Preprint*, arXiv:2403.05530.

Qwen Team. 2024b. Qwen2.5: A party of foundation models.

Bin Wang, Chao Xu, Xiaomeng Zhao, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Rui Xu, Kaiwen Liu, Yuan Qu, Fukai Shang, Bo Zhang, Liqun Wei, Zhihao Sui, Wei Li, Botian Shi, Yu Qiao, Dahua Lin, and Conghui He. 2024a. Mineru: An open-source solution for precise document content extraction. *Preprint*, arXiv:2409.18839.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. 2024b. Measuring multimodal mathematical reasoning with MATH-vision dataset. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2023a. Mathcoder: Seamless code integration in llms for enhanced mathematical reasoning. *Preprint*, arXiv:2310.03731.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024c. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Zengzhi Wang, Rui Xia, and Pengfei Liu. 2023b. Generative ai for math: Part i – mathpile: A billion-token-scale pretraining corpus for math. *Preprint*, arXiv:2312.17120.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024a. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024b. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Zhen Yang, Jinhao Chen, Zhengxiao Du, Wenmeng Yu, Weihan Wang, Wenyi Hong, Zhihuan Jiang, Bin Xu, and Jie Tang. 2024c. Mathglm-vision: Solving mathematical problems with multi-modal large language model. *Preprint*, arXiv:2409.13729.

Zhen Yang, Ming Ding, Qingsong Lv, Zhihuan Jiang, Zehai He, Yuyi Guo, Jinfeng Bai, and Jie Tang. 2023. Gpt can solve mathematical problems without a calculator. *Preprint*, arXiv:2309.03241.

Zitong Yang, Neil Band, Shuangping Li, Emmanuel Candès, and Tatsunori Hashimoto. 2024d. Synthetic continued pretraining. *Preprint*, arXiv:2409.07431.

Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, Yudong Wang, Zijian Wu, Shuaibin Li, Fengzhe Zhou, Hongwei Liu, Songyang Zhang, Wenwei Zhang, Hang Yan, Xipeng Qiu, Jiayu

Wang, Kai Chen, and Dahua Lin. 2024. Internlm-math: Open math large language models toward verifiable reasoning. *Preprint*, arXiv:2402.06332.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. *Preprint*, arXiv:2309.12284.

Yuhang Zang, Xiaoyi Dong, Pan Zhang, Yuhang Cao, Ziyu Liu, Shengyuan Ding, Shenxi Wu, Yubo Ma, Haodong Duan, Wenwei Zhang, Kai Chen, Dahua Lin, and Jiaqi Wang. 2025. Internlm-xcomposer2.5-reward: A simple yet effective multi-modal reward model. *Preprint*, arXiv:2501.12368.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. 2025a. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer.

Renrui Zhang, Xinyu Wei, Dongzhi Jiang, Ziyu Guo, Yichi Zhang, Chengzhuo Tong, Jiaming Liu, Aojun Zhou, Shanghang Zhang, Peng Gao, and Hongsheng Li. 2025b. MAVIS: Mathematical visual instruction tuning with an automatic data engine. In *The Thirteenth International Conference on Learning Representations*.

Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2024. Solving challenging math word problems using GPT-4 code interpreter with code-based self-verification. In *The Twelfth International Conference on Learning Representations*.

Wenwen Zhuang, Xin Huang, Xiantao Zhang, and Jin Zeng. 2024. Math-puma: Progressive upward multimodal alignment to enhance mathematical reasoning. *arXiv preprint arXiv:2408.08640*.

Yi Zong and Xipeng Qiu. 2024. Gaokao-mm: A chinese human-level benchmark for multimodal models evaluation. *CoRR*, abs/2402.15745.

Zhuofan Zong, Dongzhi Jiang, Bingqi Ma, Guanglu Song, Hao Shao, Dazhong Shen, Yu Liu, and Hongsheng Li. 2024a. Easyref: Omni-generalized group image reference for diffusion models via multimodal llm. *arXiv preprint arXiv:2412.09618*.

Zhuofan Zong, Bingqi Ma, Dazhong Shen, Guanglu Song, Hao Shao, Dongzhi Jiang, Hongsheng Li, and Yu Liu. 2024b. Mova: Adapting mixture of vision experts to multimodal context. *arXiv preprint arXiv:2404.13046*.

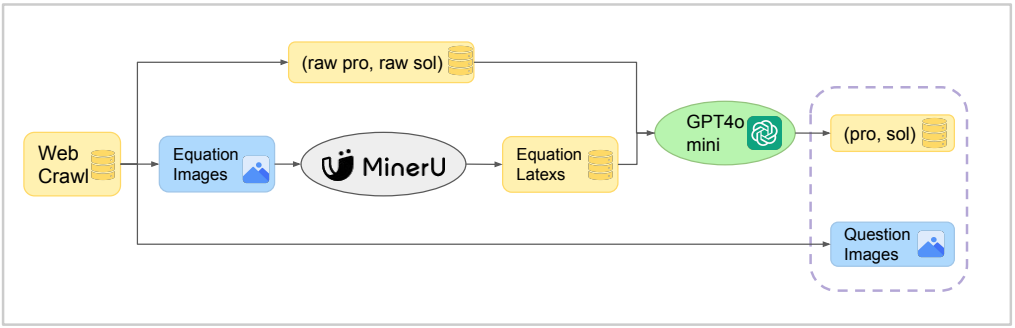


Figure 5: The pipeline for processing the K12 math problem-solving dataset.

A Details of K12 Data Process

In this section, we provide additional details about processing the newly collected K12 math problem-solving dataset. The overall pipeline for data processing is illustrated in Figure 5.

A.1 Data Cleaning

The primary objective of the data cleaning process is to curate a dataset that consists exclusively of multi-modal math problems. These problems should include both textual descriptions and mathematical expressions represented in LaTeX code and math figures. In the raw dataset, a significant number of equations were provided solely as images, as shown in Figure 6. To address this, we employed the MinerU tool to convert these equation images into LaTex-formatted equations, ensuring a consistent and standardized representation of mathematical content. Furthermore, problems that contained only equation images are excluded from the dataset. This cleaning process ensures that the final dataset is rich, diverse, and appropriately structured for addressing K12 math problems that require multi-modal reasoning.

A.2 Data Augmentation

Figure 7 presents a structured system prompt designed for processing K-12 mathematical problems. It outlines a comprehensive workflow for translating, solving, and formatting math problems from a JSON object. The prompt includes explicit instructions for translation into English, step-by-step solution generation, and concise answer presentation, ensuring clarity and correctness in the output. One example of the GPT4o-mini's output is shown in Figure 8.

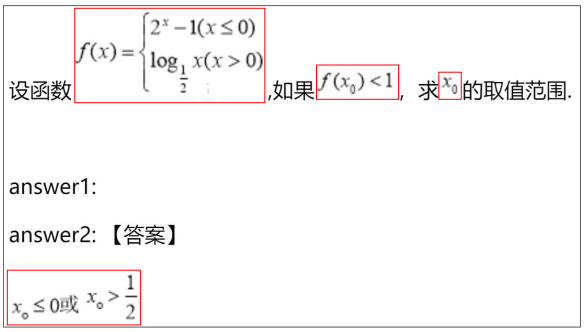


Figure 6: Example of a raw math problem that only contains equation images. Such problems are filtered out after converting the images into LaTeX equations using MinerU.

B Details of Image-to-Code

B.1 Code Ability

TikZ is a powerful and flexible package for creating vector graphics in LaTeX. It is based on the PGF (Portable Graphics Format) system and is known for its high-quality output and extensive customization options. TikZ allows users to create a wide range of graphics, from simple shapes and diagrams to complex illustrations and plots. Its strength lies in its ability to seamlessly integrate with LaTeX documents, ensuring that the fonts, styles, and layout of the graphics match the document's overall design. TikZ is particularly useful for creating precise, technical illustrations, flowcharts, and scientific figures. The syntax of TikZ is based on a series of commands that define paths, nodes, and styles, making it highly programmable and suitable for generating graphics algorithmically. Some examples of images generated by TikZ are shown in Figure 10.

On the other hand, Matplotlib is a popular plotting library in Python that provides a wide range

K12 Process Prompt

System Prompt:

You are an expert in mathematical problem-solving, LaTeX formatting, and structured data extraction. Please present all results in English and well-formatted LaTeX, converting HTML to LaTeX as needed. You will be provided with a JSON object containing the following fields: ["question", "option_a", "option_b", "option_c", "option_d", "option_e", "answer1", "answer2", "parse"].

User Prompt:

Please process the provided JSON object by following these steps:

1. **Translation:**
   - Translate the math problem and any accompanying options into English.
   - If the problem includes multiple-choice options, format them as a bulleted list.
   - If no options are available, return an empty option list (‘[]‘).
   - For problems with multiple sub-questions, separate each sub-question as an individual item in another list.

2. **Step-by-Step Solution:**
   - Provide a detailed, step-by-step solution to the problem, referencing "answer1", "answer2", and "parse".
   - Adhere to the solution process provided by "answer1", "answer2", and "parse", as they are correct.

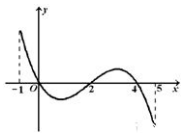
3. **Short Answer:**
   - Specify the answer(s) in a list format, where each item is a single word or phrase.
   - Answer(s) should adhere to that provided by "answer1", "answer2", and "parse".
   - For multiple-choice questions, return one of A, B, C, D, or E.
   - For proof-based questions, return "proven".
   - For problems with sub-questions, provide the answer for each sub-question in the same order as the sub-question list.

Input JSON:

```json
[Raw Json Data]
```

Figure 7: Prompt for processing, solving, and formatting K-12 math problems from structured JSON input.

x	-1	0	4	5
$f(x)$	1	2	2	1




```
{
    "question": "Given that the domain of the function $f(x)$ is $[-1,5]$, and some corresponding values are shown in the table, the graph of the derivative $y=f^{\\\\prime}(x)$ is shown in the image. The following statements about $f(x)$ are made: ① The local maximum points of the function $f(x)$ are at $0$ and $4$; ② The function $f(x)$ is decreasing on the interval $[0,2]$; ③ If the maximum value of $f(x)$ is $2$ when $x \\\\in [-1,t]$, then the maximum value of $t$ is $4$; ④ The function $y=f(x)$ has at most $2$ zeros. Which of the following statements are correct? (       )",
    "options": [
        "①②",
        "③④",
        "①②④",
        "②③④.",
        "[]"
    ],
    "sub_questions": [],
    "solution": "Analysis of the problem: From the graph of the derivative, we can see that the function $y=f(x)$ is increasing on the intervals $(-1,0)$ and $(2,4)$ where the derivative is positive. It is decreasing on the intervals $(0,2)$ and $(4,5)$ where the derivative is negative. Therefore, statement ① that the local maximum points of the function $f(x)$ are at $0$ and $4$ is correct. Statement ② that the function $f(x)$ is decreasing on the interval $[0,2]$ is also correct. Statement ③ is incorrect because if the maximum value of $f(x)$ is $2$ when $x \\\\in [-1,t]$, then the maximum value of $t$ is actually $5$, not $4$. Statement ④ is correct because the function $y=f(x)$ can have at most $2$ zeros, especially when $f(2)<0$, which allows for two zeros. Thus, the correct statements are ①, ②, and ④. Therefore, the correct answer is C.",
    "final_answers": ["C"]
}
```


Figure 8: Example output generated by GPT4o-mini, showcasing the translation, step-by-step solution, and short answer extraction for a given math problem.

```python
import matplotlib.pyplot as plt
import numpy as np
# Data for the plots
x = np.array([0, 0.02, 0.1])
y1 = np.array([70.2, 70.9, 81.1])
y2 = np.array([77.1, 78.4, 81.1])
# Set up the plot
fig, ax = plt.subplots(figsize=(8, 6))
# Plot the bars
bar_width = 0.15
bar_positions = np.arange(len(x))
ax.bar(bar_positions - bar_width/2, y1, bar_width, label='Plot
1')
ax.bar(bar_positions + bar_width/2, y2, bar_width, label='Plot
2')
# Set the xticks and labels
ax.set_xticks(bar_positions)
ax.set_xticklabels(x)
# Customize the plot
      . . .
# Display the values on top of the bars
      . . .
# Layout so plots do not overlap
fig.tight_layout()
```

```latex
. . .
\begin{tikzpicture}
\begin{axis}[
    ybar,
    enlargelimits=0.15,
    legend style={at={(0.5,-0.15)},
      anchor=north,legend columns=-1},
    ylabel={Avg. Metric (\%)},
    xlabel={Amount of Pretraining Corpus (Millions)},
    symbolic x coords={0,0.02,0.1},
    xtick=data,
    ytick={65,70,75,80,85},
    ymajorgrids=true,
    grid style=dashed,
    bar width=15pt,
    nodes near coords,
    every node near coord/.append style={font=\tiny},
    nodes near coords align={vertical},
    ]
\addplot coordinates {(0,70.2) (0.02,70.9) (0.1,81.1)};
\addplot coordinates {(0,77.1) (0.02,78.4) (0.1,81.1)};
\end{axis}
\end{tikzpicture}
. . .
```

Figure 9: Comparison between Python code and TiKZ code.

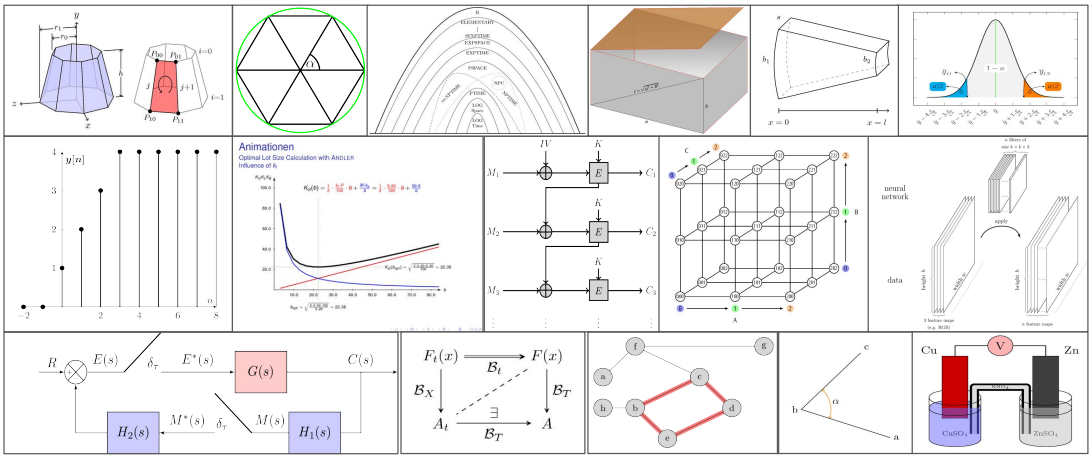


Figure 10: Some example of images generated by TikZ.

of tools for creating static, animated, and interactive visualizations. It is widely used in scientific computing, data analysis, and machine learning for generating publication-quality figures. Matplotlib supports various types of plots, including line plots, scatter plots, bar charts, histograms, and more. One of its key strengths is its flexibility and ease of use, allowing users to quickly generate visualizations with a few lines of code. Matplotlib also offers extensive customization options, enabling users to adjust every aspect of a plot, from line styles and colors to axis labels and legends. Additionally, it integrates well with other Python libraries such as NumPy and Pandas, making it a versatile tool for data visualization in the Python ecosystem. Some examples of images generated by Python are shown in Figure 11.

When comparing the syntax of Python's Matplotlib and LaTeX's TikZ for creating plots and graphics, the differences are quite pronounced as shown in Figure 9. Matplotlib, being a Python library, follows a procedural programming style, where functions are called to add elements to a plot. In contrast, TikZ, which is part of the LaTeX ecosystem, uses a declarative style, where user describe the elements of the graphic in a more structured, often nested, manner. While Matplotlib's syntax is more straightforward and easier to learn for those familiar with Python, TikZ offers greater control over the visual details of the plot, making it a preferred choice for complex, publication-quality graphics.

B.2 Prompt Templates

To facilitate the generation of code from images, we designed two structured prompt templates that

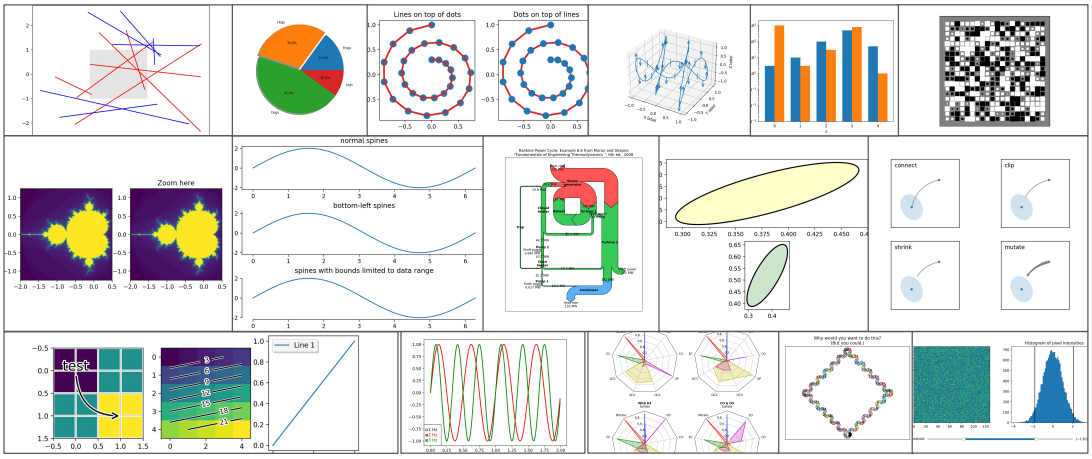


Figure 11: Some example of images generated by Python.

guide the process of converting visual elements into executable code as shown in Figure12.

B.3 TikZ to Python

To enhance the capabilities of our image-to-code model, we use GPT4o-mini to translate TikZ code into Python code. Figure 13 illustrates the detailed prompt used for this translation. The prompt in Figure 13 is designed to guide the conversion of LaTeX TikZ code into Python code using popular plotting libraries like Matplotlib. It ensures that the resulting Python code is executable, accurately reproduces the visual details of the TikZ diagram, and avoids overlaps between elements such as points, labels, and text for better readability. The prompt also emphasizes the correct formatting of LaTeX mathematical expressions to maintain visual clarity and precision in the generated plots. This structured approach helps bridge the gap between LaTeX-based graphics and Python-based visualization.

In Figure 5, we compare images generated from the original TikZ code with those generated from the translated Python code. The results demonstrate that the images produced by the Python code are highly similar to the original images, showcasing the effectiveness of our translation approach.

B.4 Data Cleaning

We remove low-quality image-code pairs from our dataset. Figure 14 illustrates four types of low-quality samples: (a) Almost blank images: We remove images with a standard deviation (std) of pixel values less than five. (b) Images with random lines or shapes: These are filtered out by analyzing

(a) Image-to-TikZ Prompt:

Please generate the corresponding TikZ code that accurately represents the visual elements in the image. TikZ is a powerful tool for creating vector graphics within LaTeX documents. Your generated code should be precise, well-structured, and should recreate the image as faithfully as possible.
<image>

The image can be generated using the following TikZ code:

```tikz
[code]
```

(b) Image-to-Python Prompt:

Please provide the Python code needed to reproduce this image.
<image>

The image can be generated using the following Python code:

```python
[code]
```

Figure 12: Prompt templates of our Image-to-Code Dataset.

TikZ-to-Python Prompt

System Prompt:

You are an expert in both LaTeX (specifically TiKZ) and Python (specifically Matplotlib).

User Prompt:

Translate the provided TiKZ code into Python code using appropriate plotting libraries, such as Matplotlib. Pay close attention to the following requirements:

1. **Avoid Overlapping**: Ensure that points, labels and text elements have different positions to avoid any overlap, enhancing readability.

2. **LaTeX Formatting**: Accurately interpret and format any LaTeX equations or mathematical expressions to ensure they render correctly in the image.

3. **Executable Code**: Ensure that the Python code is complete and can be executed directly without errors.

Hereś the TiKZ code:

```latex
[TiKZ Code]
```

Make sure to wrap your resulting Python code in the following format:

```python
[your python code here]
```

Figure 13: Prompt for translating LaTeX TiKZ code into Python Matplotlib code with a focus on accuracy, readability, and executability.

and filtering the corresponding code. (c) Images with black squares: This issue arises when images with blank backgrounds are converted incorrectly during preprocessing, resulting in completely black images. We addressed this by removing the affected data and optimizing the conversion logic. (d) Images with externally loaded content: We identify and remove such data by detecting commands in the code that access local files.

B.5 Performance of img2code model

The img2code model aims to bridge the gap between visual data and code generation by translating images into accurate and meaningful code representations. This section evaluates the model's progression through iterative training and highlights its ability to synthesize new, diverse images. By comparing the performance of the initial and final versions of the model and exploring its capabilities with high-temperature synthesis, we demonstrate its advancements in accuracy and creative output.

Comparison Between Initial and Final Models. Our img2code model was trained iteratively, culminating in a final version trained on 8.6 million image-code pairs. The performance improvements from the initial to the final model are demonstrated in Figures 6, 7, 8, and 9. These figures highlight the significant advancements in accuracy and the quality of the generated code and corresponding images as the model evolved through successive training cycles.

Synthesize New Images with High Temperature. Using the final iteration of the Img2Code-8B model, we synthesized new images from 1.57 million raw images in the foundational dataset. By setting a temperature of 0.7, the model was able to generate more diverse and creative outputs, deviating meaningfully from the original dataset. The results of this high-temperature synthesis are illustrated in Figures 10, 11, 12, 13, 14, and 15. These figures demonstrate the model's ability to produce innovative and varied image outputs suitable for diverse applications.

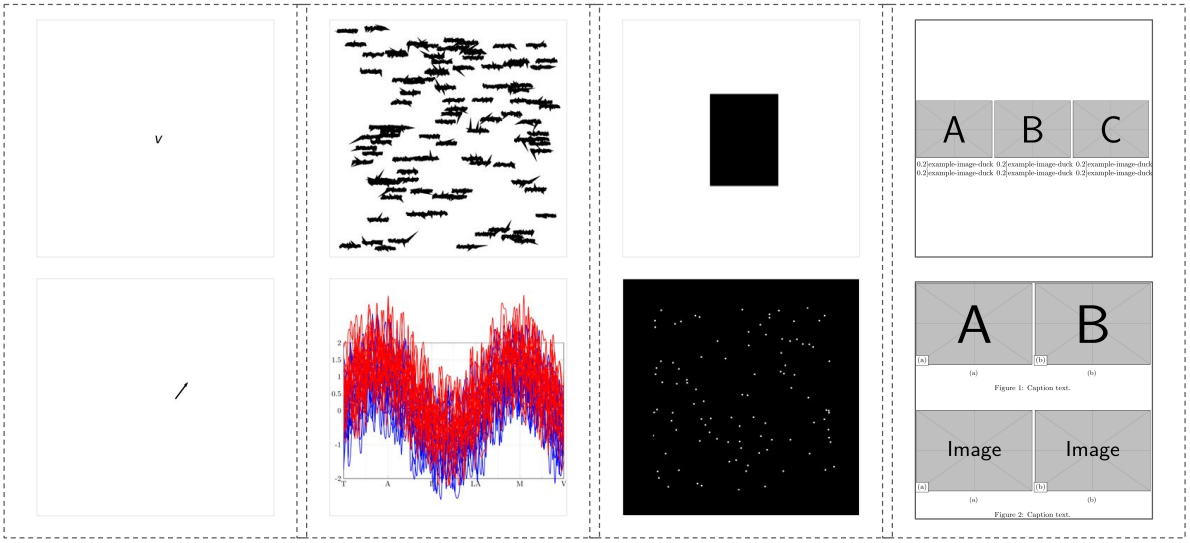


Figure 14: Examples of low-quality image-code pairs removed from the dataset. (a) Almost blank images with very low pixel variation. (b) Images containing random lines or shapes. (c) Images with black squares caused by incorrect preprocessing. (d) Images generated using external files accessed through the code.

Synthesize New Problems Based on New images. The Problem Synthesis Prompt shown in Figure 15 is designed to encourage creative and meaningful engagement with visual data by crafting math reasoning questions that are both accessible and challenging for a K-12 audience. This process involves analyzing patterns, shapes, and numerical relationships present in an image, then constructing a single, concise question that stimulates analytical thinking. The prompt ensures that the generated question is self-contained, solvable using the visible information in the image, and includes any essential details that may not be immediately apparent. By adhering to these guidelines, educators and content creators can develop visually engaging problems that promote critical reasoning and mathematical exploration, fostering a deeper connection between visual interpretation and problem-solving skills.

Problem Synthesis Prompt

Please create a **math reasoning question** for a K-12 audience based on the image generated by the following code. The question must adhere to these criteria:

1. **Image Engaging**: The question must utilize visible patterns, shapes, numbers, or other elements present in the image to engage reasoning skills.

2. **Single Question**: Write a single, standalone question. The question should be concise and self-contained, without any subparts. You do not need to provide an answer to the question.

3. **Self-Sufficiency**: The recipient will only see the image, not the code. Include any essential details from the code (e.g., coordinates, hidden axes, specific data points, or labels) that are necessary for solving the question but may not be visible in the image.

4. **Solvability**: Ensure the question can be solved using only the visible information in the image and the question text.

Below is the code that generates the image:

```python/tikz
[Image Code]
```

Question:

Figure 15: Prompt for synthesizing math reasoning problems based on synthesized images.

TiKZ	Python	TiKZ	Python	TiKZ	Python

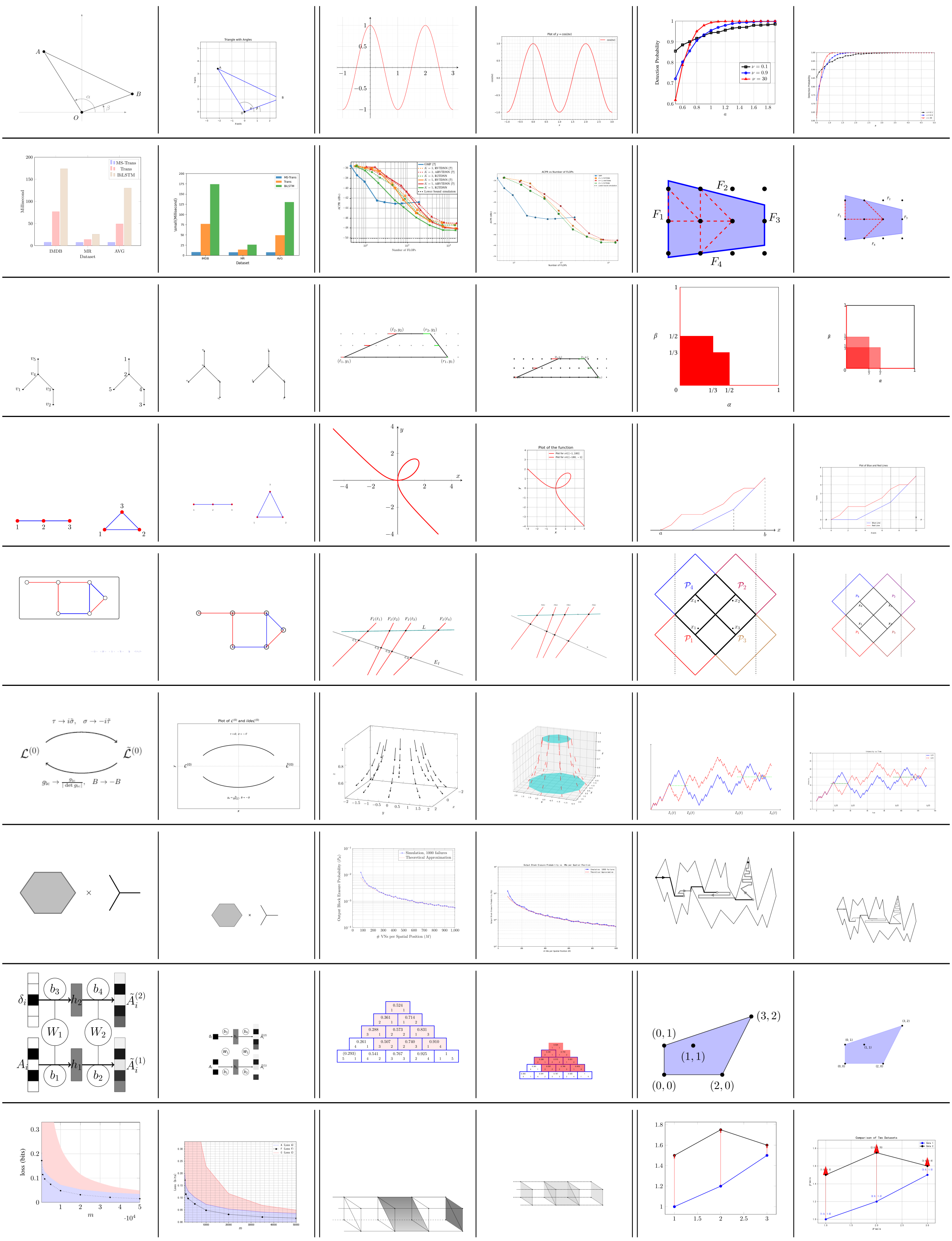


Table 5: Comparison of images generated from the original TiKZ code and the translated Python code.

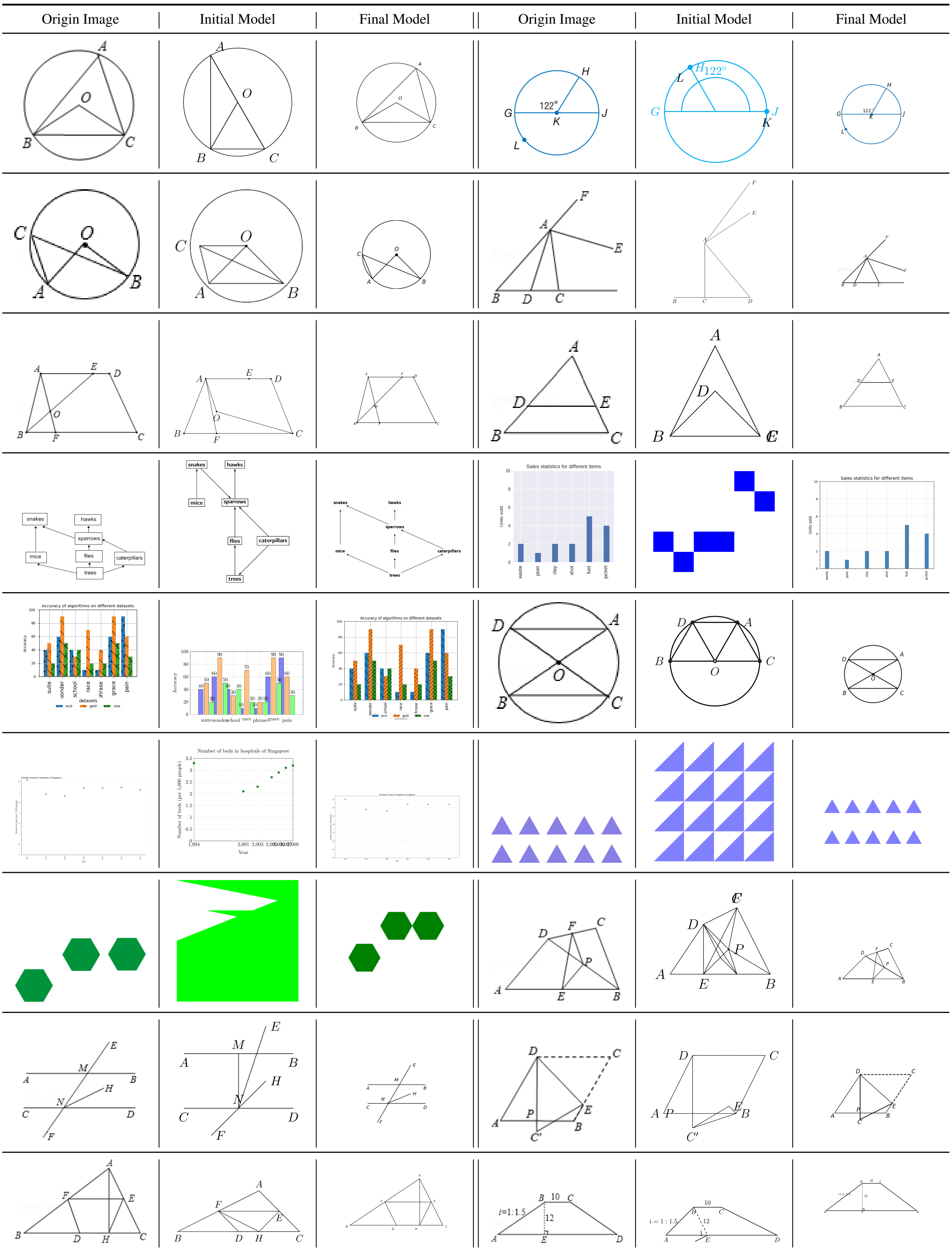


Table 6: Comparison of image-to-code performance between the initial and final models.

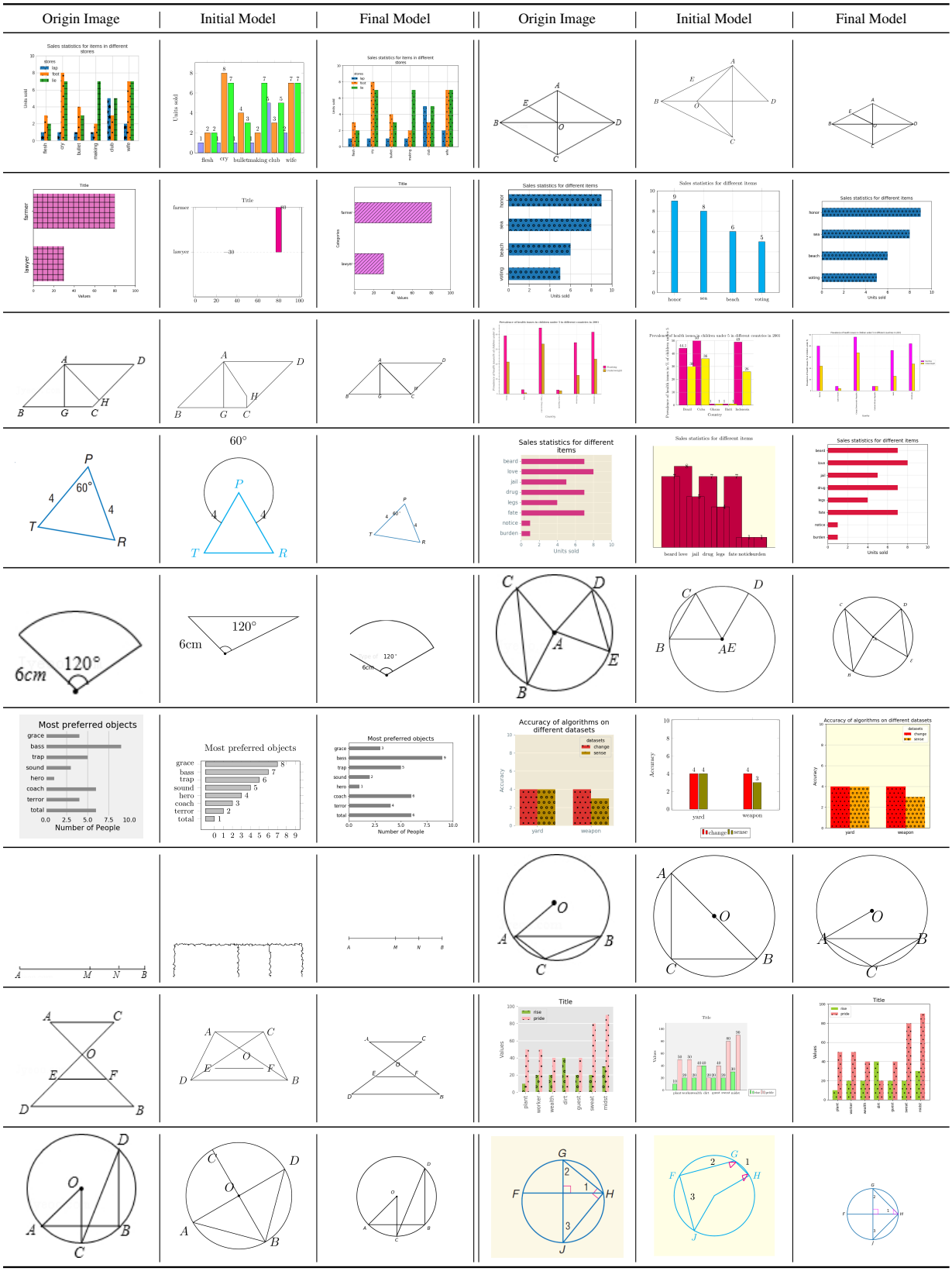


Table 7: Comparison of image-to-code performance between the initial and final models.

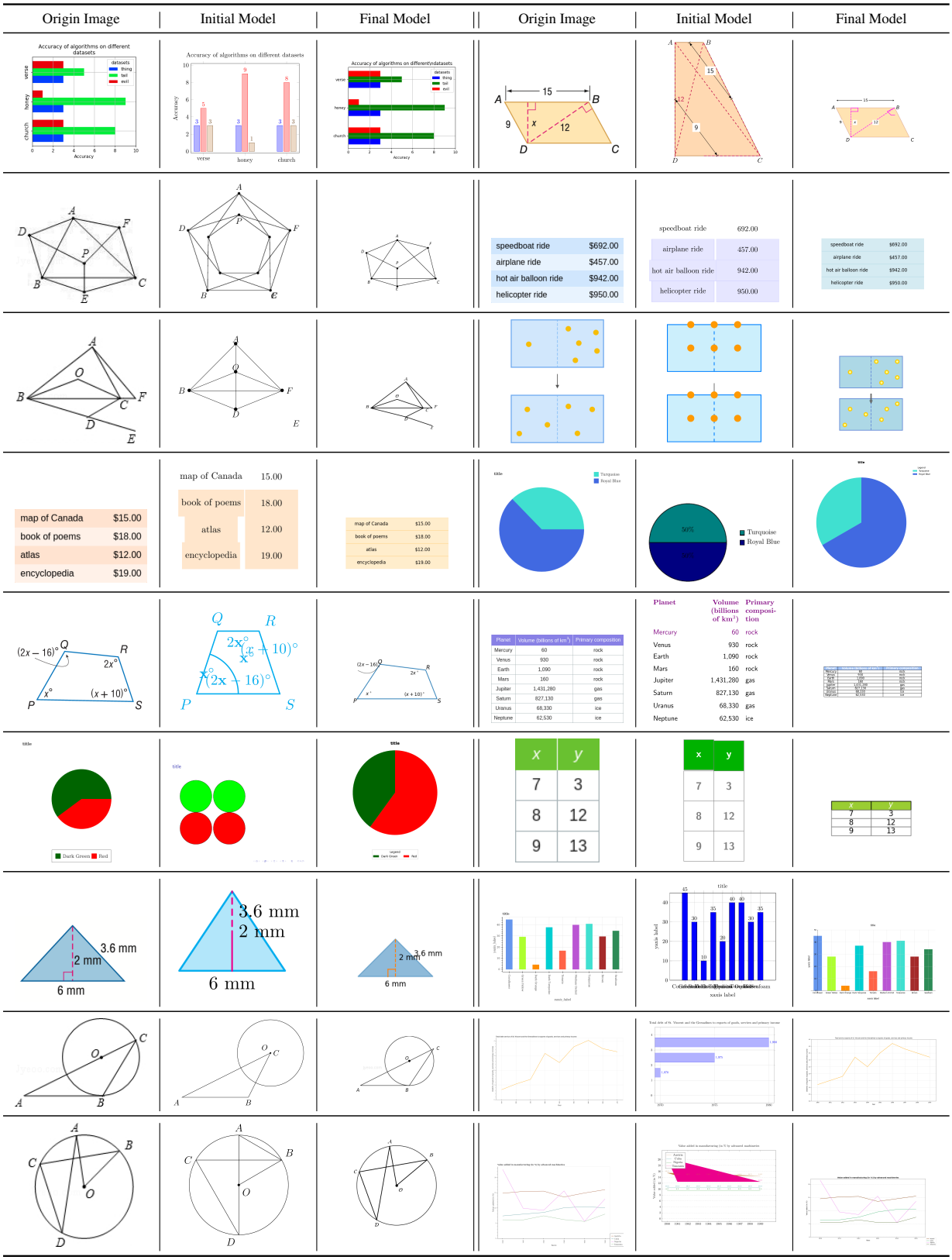


Table 8: Comparison of image-to-code performance between the initial and final models.

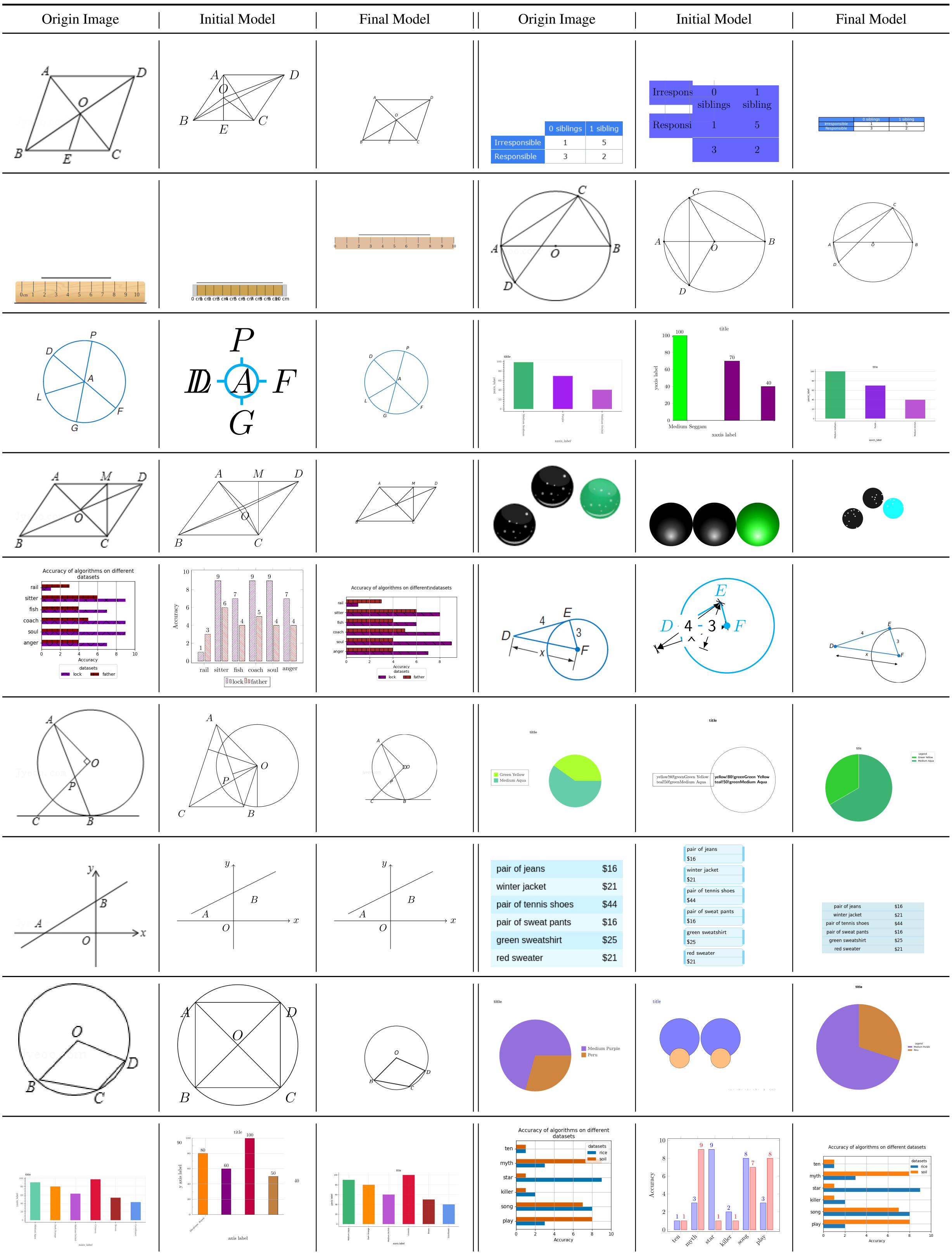


Table 9: Comparison of image-to-code performance between the initial and final models.

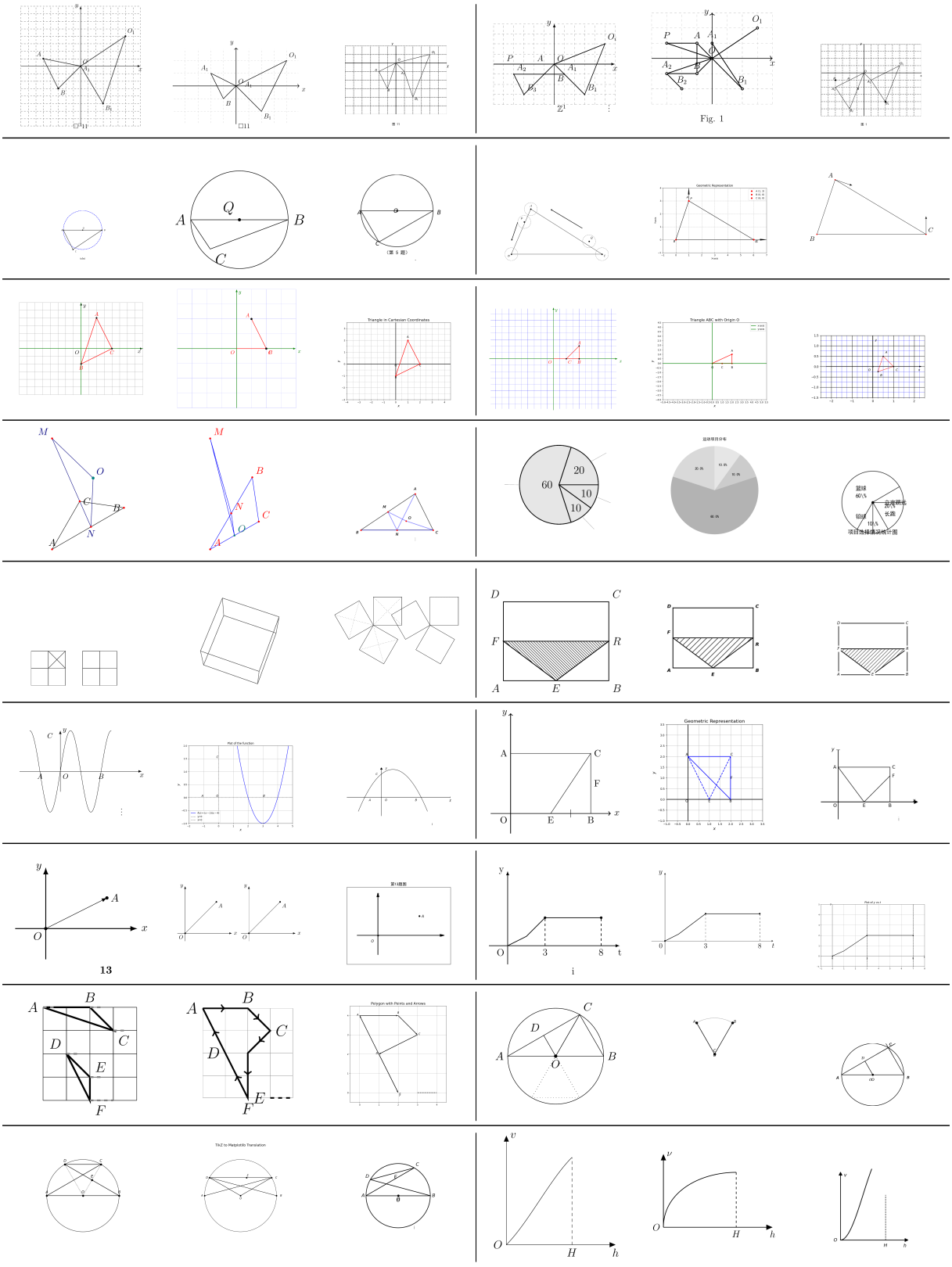


Table 10: New images synthesized with seed images form K12-2M.

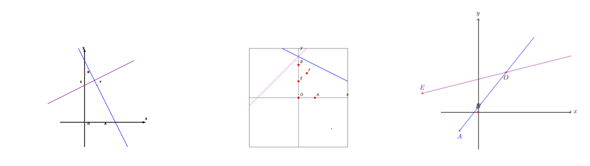
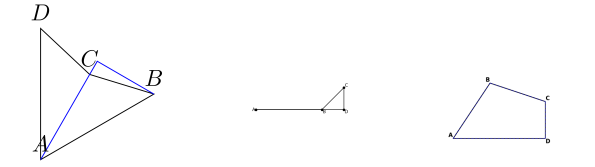
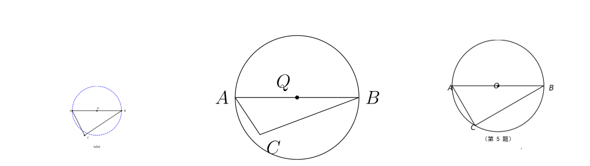
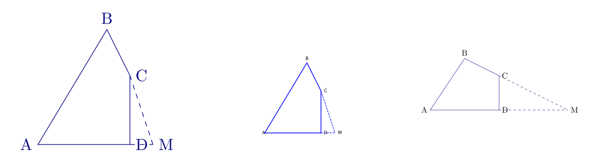
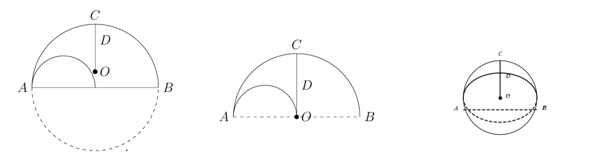
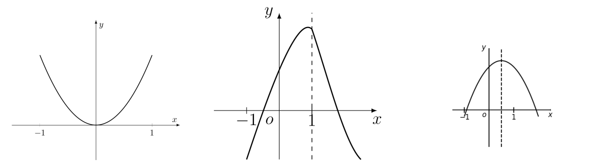
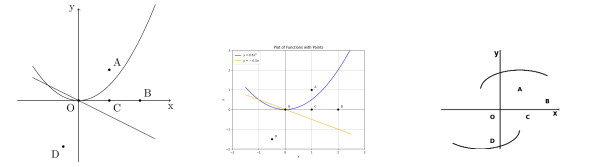
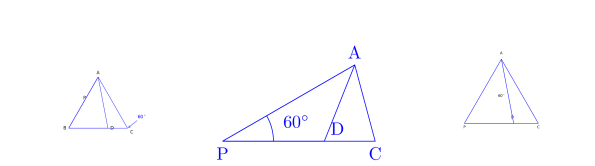
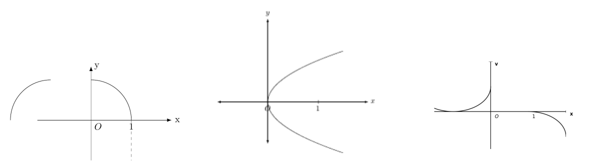
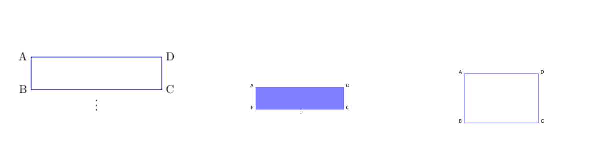
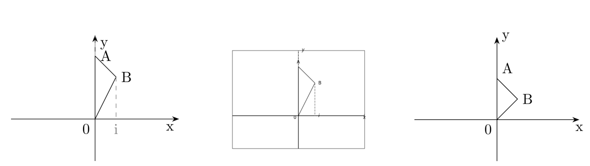
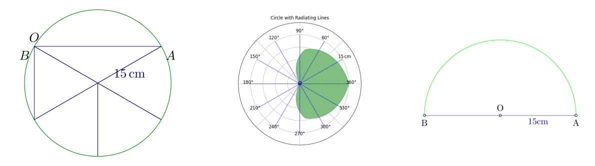
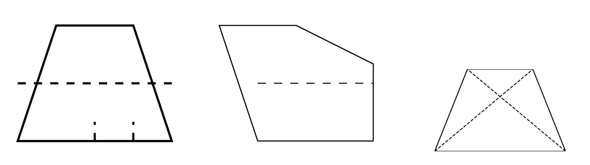
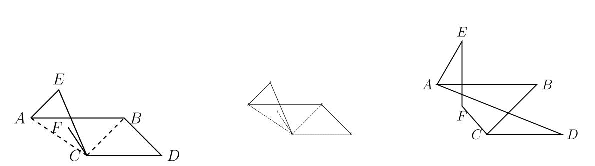
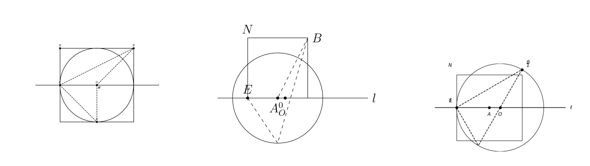
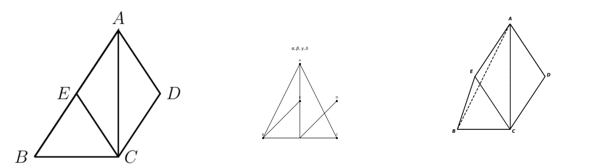
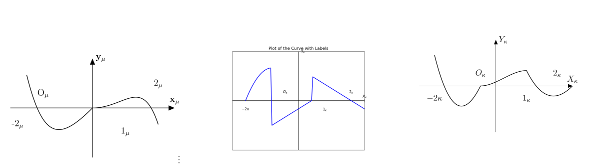
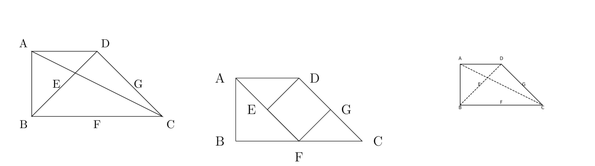
New Images from One Seed Image	New Images from One Seed Image
	
	
	
	
	
	
	
	
	

Table 11: New images synthesized with seed images form K12-2M

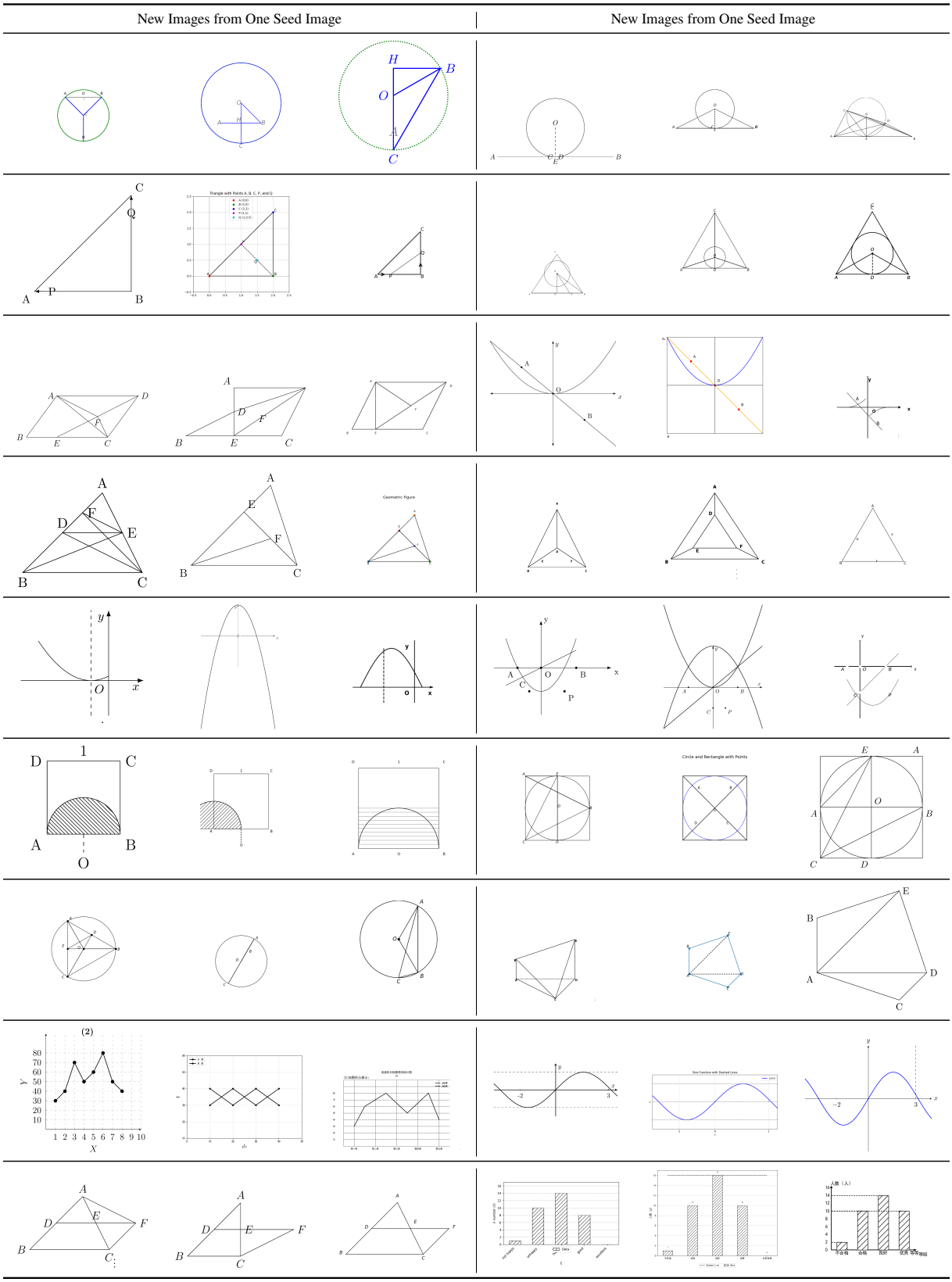


Table 12: New images synthesized with seed images form K12-2M

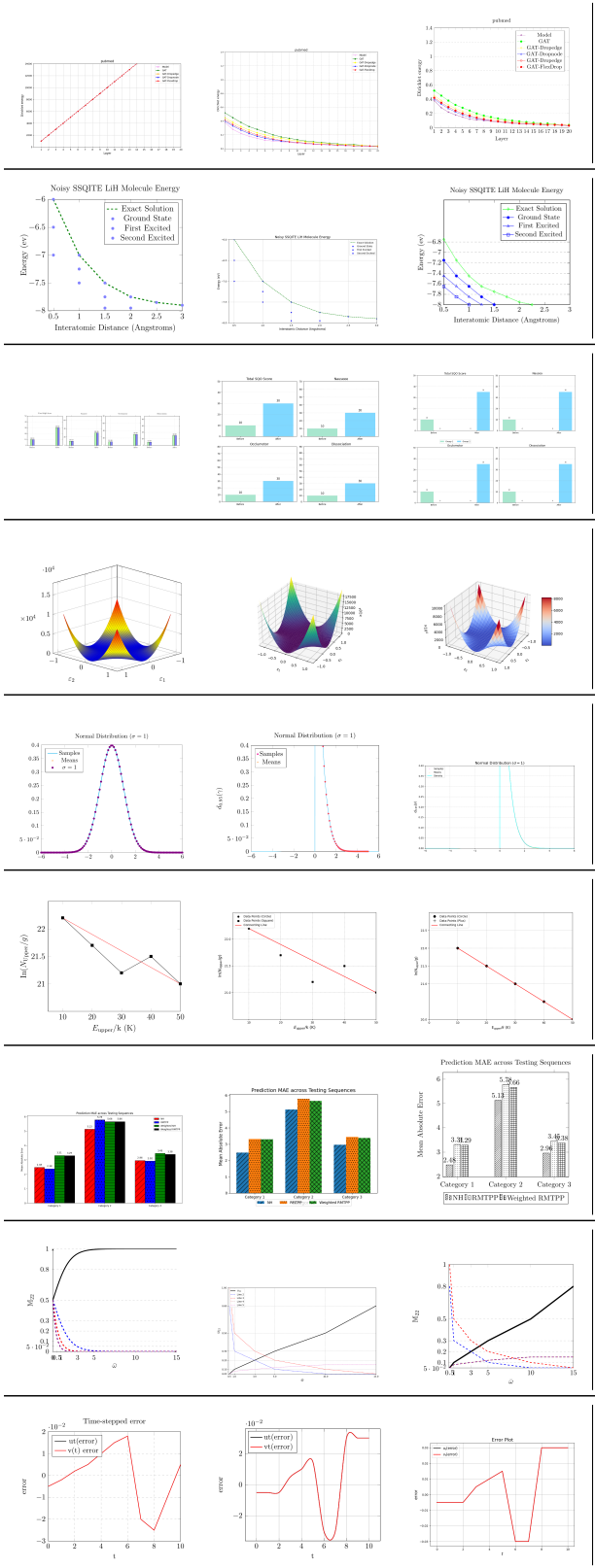
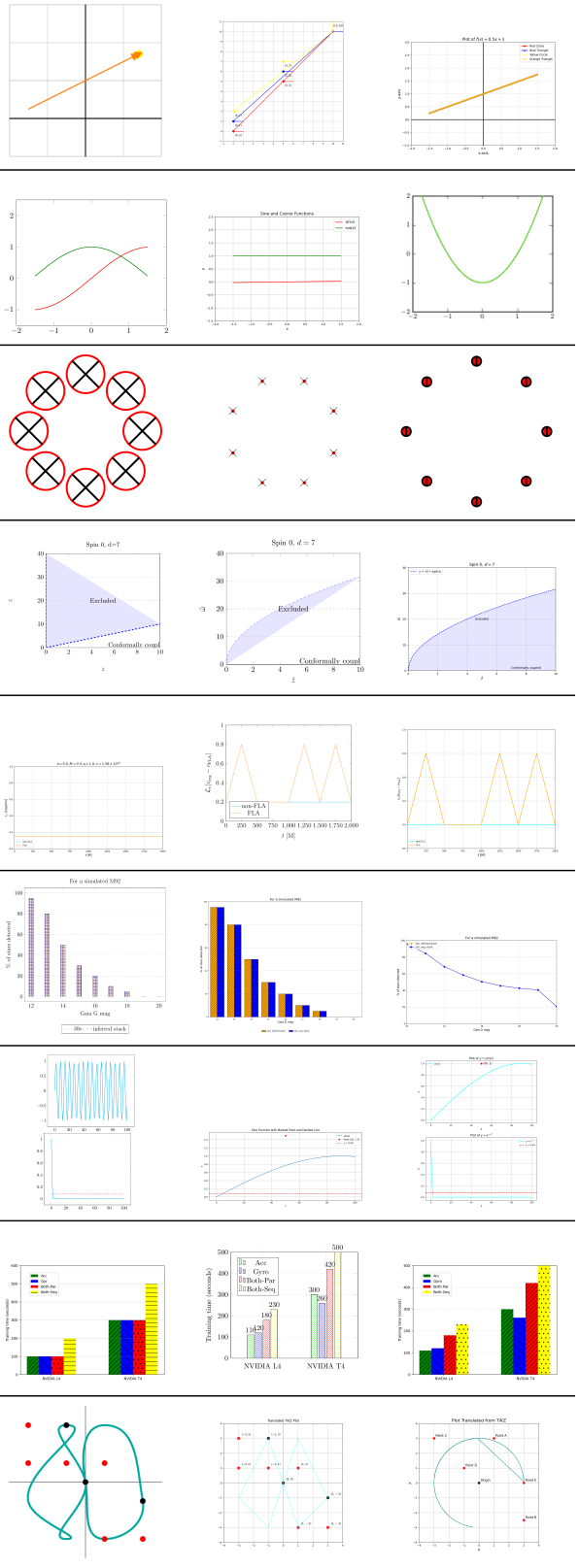
New Images from One Seed Image	New Images from One Seed Image
	

Table 13: New images synthesized with seed images form arXiv

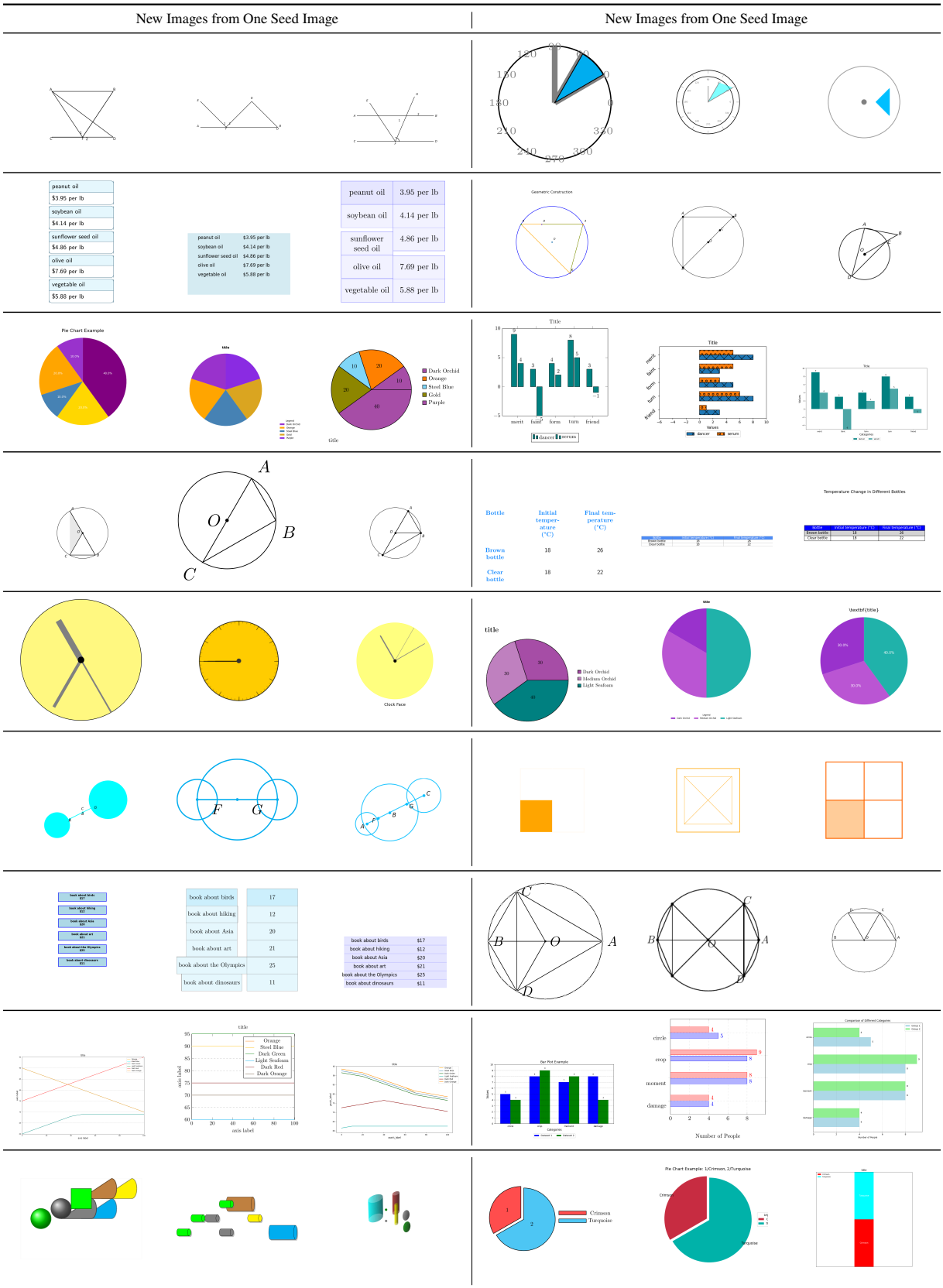


Table 14: New images synthesized with seed images form MathV360k

Table 15: New images synthesized with seed images form MathV360k