TIKZILLA: SCALING TEXT-TO-TIKZ WITH HIGH-QUALITY DATA AND REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly used to assist scientists across diverse workflows. A key challenge is generating high-quality figures from textual descriptions, often represented as TikZ programs that can be rendered as scientific images. Prior research has proposed a variety of datasets and modeling approaches for this task. However, existing datasets for Text-to-TikZ are too small and noisy to capture the complexity of TikZ, causing mismatches between text and rendered figures. Moreover, prior approaches rely solely on supervised fine-tuning (SFT), which does not expose the model to the rendered semantics of the figure, often resulting in errors such as looping, irrelevant content, and incorrect spatial relations. To address these issues, we construct DaTikZ-V4, a dataset more than four times larger and substantially higher in quality than DaTikZ-V3, enriched with LLM-generated figure descriptions. Using this dataset, we train TikZilla, a family of small open-source Qwen models (3B and 8B) with a two-stage pipeline of SFT followed by reinforcement learning (RL). For RL, we leverage an image encoder trained via inverse graphics to provide semantically faithful reward signals. Extensive human evaluations with over 1,000 judgments show that TikZilla improves by 1.5-2 points over its base models on a 5-point scale, surpasses GPT-4o by 0.5 points, and matches GPT-5 in the image-based evaluation, while operating at much smaller model sizes. Code, data, and models will be made available.

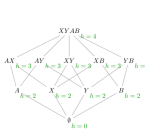
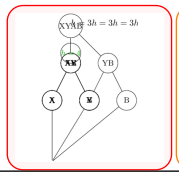
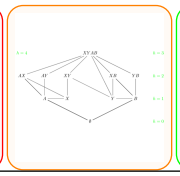
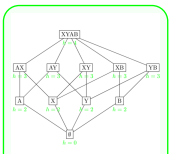
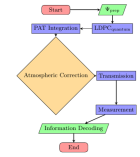
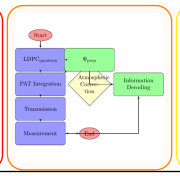
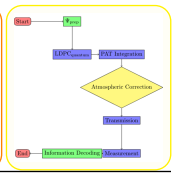
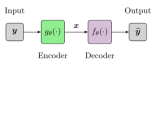
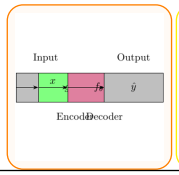
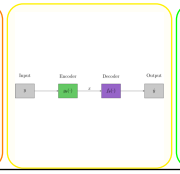
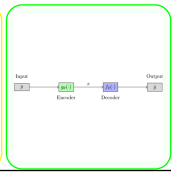
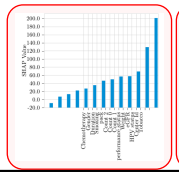
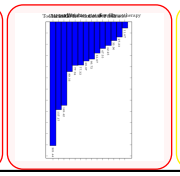
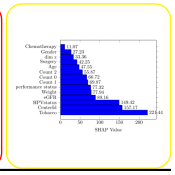
1 INTRODUCTION

In recent years, natural language processing has become an increasingly valuable tool for scientists across all domains (Bi et al., 2024; Eger et al., 2025). This progress is driven not only by continuous performance improvements for Large Language Models (LLMs), enabled by scaling model size, hardware, and data (Minaee et al., 2025), but also by research expanding their capabilities into the multimodal domain (Wu et al., 2023), and enabling advanced reasoning (Huang & Chang, 2023). As a result, an increasing number of tools have been developed to support scientists throughout the research process, which range from idea generation (Gottweis et al., 2025) to the full automation of scientific outputs (Lu et al., 2024). However, these fully autonomous tools are still far from meeting the high scientific standards required for practical use. Achieving such standards involves overcoming complex subtasks, such as generating accurate scientific images based on textual descriptions (Rodriguez et al., 2023; 2024; Zou et al., 2024).

Graphics programming languages such as TikZ are the de facto standard in academia due to their precision, interpretability and seamless integration in the LaTeX ecosystem. However, their steep learning curve and highly varied syntax make them difficult for both humans and LLMs to master (Belouadi et al., 2024a). Prior works have attempted to bridge this gap by finetuning LLMs on caption-TikZ pairs (Belouadi et al., 2024a; 2025). Due to the sparsely available data, Belouadi et al. (2025) leverage captioned images without the underlying graphics program available, therefore having access to a much richer dataset. However, these efforts remain limited by noisy captions, a lack of executable and standardized TikZ code, as well as a lack of direct visual feedback, leaving models prone to low compilation rates, hallucinations, overly long responses, and low-quality outputs.

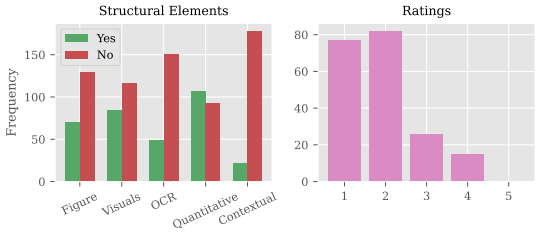
We address these limitations by constructing DaTikZ-V4, a dataset more than 1.5M instances larger than its predecessor, sourced from arXiv, GitHub, TeX StackExchange (TeX SE), and synthetic data.

Table 1: Exemplary scientific TikZ figures produced by one baseline LLM (GPT-4o) and two of our finetuned LLMs (TikZilla-3B and TikZilla-3B-RL) using the prompts from the first column which have been VLM augmented based on the Ground Truth figures in the second column. -boxed figures have been rated as very good, as good, as bad, and as very bad by human annotators. Additional examples are provided in the Appendix (Table 8, 9, 10, and 11)

Prompt	Ground Truth	GPT-4o	TikZilla-3B	TikZilla-3B-RL
A lattice diagram consists of nodes connected by thin black lines. At the top center, node "XYAB" is labeled with $h=4$ in green below it. Directly below, five nodes "AX", "AY", "XY", "XB", and "YB" are horizontally aligned, each labeled with $h=3$ in green below. Below these, nodes "A", "X", "Y", and "B" are horizontally aligned, each labeled with $h=2$ in green below. At the bottom center, node "$"emptyset $" is labeled with $h=0$ in green above. Lines connect "XYAB" to each of the nodes in the second row. "AX" connects to "A" and "X", "AY" connects to "A" and "Y", "XY" connects to "X" and "Y", "XB" connects to "X" and "B", and "YB" connects to "Y" and "B". Each node is connected to the node "$"emptyset $" at the bottom.				
A flowchart consists of various colored shapes connected by arrows. At the top left, a red rounded rectangle labeled "Start" connects via a rightward arrow to a green parallelogram labeled "$"Psi `--"text {prep}"$". Below, a blue rectangle labeled "LDPC$ `--"text {quantum}"$" connects leftward to another blue rectangle labeled "PAT Integration". This rectangle connect downward to a large yellow diamond labeled "Atmospheric Correction". From the diamond's right side, a rightward arrow leads to a blue rectangle labeled "Transmission", which connects downward to another blue rectangle labeled "Measurement". A downward arrow leads to a green parallelogram labeled "Information Decoding", which connects downward to a red rounded rectangle labeled "End". All arrows are black and connect the shapes in a logical sequence.				
A horizontal sequence of four rectangles is centered in the image. The first rectangle on the left is gray, labeled y in black at its center, and has the label "Input" above it. To its right, a black arrow points from the center of the first rectangle to the center of the second rectangle, which is green and labeled $g("theta ("cdot)$ in black. Below the green rectangle, the label "Encoder" is centered. A black arrow extends from the center of the green rectangle to the center of the third rectangle, which is purple and labeled $f("theta ("cdot)$ in black. Below the purple rectangle, the label "Decoder" is centered. A black arrow points from the center of the purple rectangle to the center of a fourth gray rectangle on the right, labeled $"hat `--y"$ in black. Above this fourth rectangle, the label "Output" is centered. The label x is placed above the arrow between the green and purple rectangles.				
A horizontal bar chart labeled 'SHAP Value' in the top right corner. Each bar is colored blue and corresponds to a specific feature, with the SHAP value displayed numerically at the end of each bar. The chart is sorted in ascending order of SHAP value, starting with 'Chemotherapy' at the top, which has the smallest value (11.07). This is followed by 'Gender' (27.23), 'dim z' (33.36), 'Surgery' (42.25), 'Age' (47.55), 'Count 2' (55.87), 'Count 0' (66.72), 'Count 1' (69.87), 'performance status' (77.32), 'Weight' (77.94), 'eGFR' (89.16), 'HPVstatus' (149.42), 'CenterId' (157.17) and 'Tobacco' with the largest value (221.44).				

To improve data quality, we introduce an LLM-based debugging pipeline that repairs uncompilable TikZ code, and employ Vision Language Models (VLMs) to generate accurate figure descriptions. Building on DaTikZ-V4, we develop TikZilla, a family of small Qwen-based models (3B and 8B) trained with a two-stage pipeline: Supervised Finetuning (SFT) for syntax alignment, followed by Reinforcement Learning (RL) with a reward model trained on the Image-to-TikZ task beforehand. We find that this approach substantially improves Text-to-TikZ generation quality, where even models as small as 3B parameters outperform GPT-4o across automatic metrics and over 1,000 human judgments spanning four baseline LLMs. Table 1 shows examples with corresponding human ratings. We summarize our key contributions as follows:

- **Caption Quality Analyisis:** We show that widely available captions are insufficient for reconstructing figures.
- **Scaling Dataset Size:** We introduce DaTikZ-V4 with over 2M unique TikZ samples, sourced from newer arXiv submissions and GitHub, quadrupling the scale of prior datasets.
- **Data Quality Enhancements:** We combine (1) improved rule-based filtering (e.g., dynamic package inclusion), (2) VLM-based scientific figure descriptions, and (3) an LLM debugging pipeline for uncompilable TikZ code.
- **Reward Model:** We finetune an image encoder on the Image-TikZ task using our larger TikZ corpus, providing more semantically meaningful rewards for RL optimization.
- **TikZilla Models:** We release TikZilla, a family of small open-source Qwen models (3B and 8B). TikZilla outperforms GPT-4o across automatic and human evaluation, and matches GPT-5 in image-based evaluation, despite operating at much smaller model sizes.



Variant	BLEU-4↑	ROUGE-L↑	STS↑	Length
Captions	0.003	0.098	0.355	34.0
Qwen2.5-VL-7B	0.068	0.276	0.744	126.3
Qwen2.5-VL-32B	0.047	0.242	0.719	177.8
InternVL3-8B	0.045	0.235	0.716	159.8
InternVL3-38B	0.057	0.264	0.743	141.6
GPT-4o-mini	0.073	0.281	0.761	140.9
GPT-4o	0.089	0.317	0.777	123.5

Figure 1: Left: human evaluation of caption quality by structural elements and usefulness ratings. Right: automatic metrics (BLEU-4, ROUGE-L, STS) and average length for captions and VLM-generated descriptions using human written descriptions as references.

2 Related Work

Text-Guided Graphics Program Generation for Scientific Figures Generating vector graphics such as SVG or TikZ is essential in scientific publishing due to their fidelity and interpretability. Early approaches relied on handcrafted heuristics or neural sequence models to approximate images with path primitives (Lopes et al., 2019; Carlier et al., 2020), but these struggled with complex scientific figures. More recently, LLM-based methods have emerged: AutomaTikZ (Belouadi et al., 2024a) finetunes on caption–TikZ pairs from arXiv and TeX SE, while StarVector (Rodriguez et al., 2024) focuses on SVG generation with a dedicated benchmark. Yet for TikZ, dataset sparsity remains a bottleneck. TikZero (Belouadi et al., 2025) partially addresses this by combining an inverse-graphics model (Belouadi et al., 2024b) with a modality-bridging adapter (Hu et al., 2023), distilling supervision from text–image pairs. However, TikZero still depends on noisy captions and cannot finetune its text decoder without paired graphics programs, limiting performance. In contrast, we construct a dataset over four times larger and of higher quality, pairing TikZ programs with VLM-generated descriptions, enabling small LLMs to be effectively finetuned for Text-to-TikZ.

Post-training with Reinforcement Learning Advances in RL such as Group Relative Policy Optimization (GRPO) (Zhihong Shao, 2024) allow to more efficiently align LLMs either with human preferences (Ouyang et al., 2022) or verifiable tasks (Lambert et al., 2025). For example, RLEF (Gehring et al., 2025) iteratively leverages execution feedback for code synthesis, Yoshihara et al. (2025) enhance LLM reasoning on math benchmarks, and VisionR1 (Huang et al., 2025) extends reasoning capabilities to the multimodal domain. Closest to our setting, RLRF (Rodriguez et al., 2025) optimizes SVG code generation via composite rewards assessing code efficiency, semantic alignment, and visual fidelity. Our work differs in two ways: we focus on TikZ generation for scientific figures, and we introduce a domain-specific reward model, trained through inverse-graphics (Image–TikZ), which better captures semantics than general-purpose metrics such as CLIPScore (Hessel et al., 2021) or DreamSIM (Fu et al., 2023).

3 Caption Quality Analysis

Accurate Text-to-TikZ generation requires captions that specify objects, attributes, and spatial relations (Zhang et al., 2025). To assess whether existing captions meet this need, we analyzed 200 samples from DaTikZ-V3 with three annotators (Figure 1, left). The annotators checked captions for missing structural elements (e. g. figure type, components, and labels) and judged usefulness on a 1–5 Likert scale. Two findings emerged: (i) key details such as figure types, components, and labels are often missing, and (ii) most captions received low usefulness scores (1–2). This indicates that raw captions are insufficient for faithfully reconstructing scientific figures.

To quantify this further, one annotator also wrote reference descriptions for all 200 figures. We then compared these human-written descriptions against both the original captions and VLM-generated descriptions using BLEU-4 (Papineni et al., 2002), ROUGE-L (Lin, 2004), and Semantic Textual Similarity (STS) (Reimers & Gurevych, 2019) (Figure 1, right). Across multiple VLMs (Qwen2.5-VL 7B/32B (Bai et al., 2025), InternVL3 8B/38B (Zhu et al., 2025), GPT-4o and GPT-4o-mini (OpenAI et al., 2024)), results show that VLMs produce richer and more faithful descriptions than raw

captions. For example, GPT-4o reaches 0.089 BLEU-4 compared to just 0.003 for captions. VLM outputs are also substantially longer (120–170 vs. 34 characters), indicating that they capture additional detail necessary for figure reconstruction. These results motivate our use of VLM-generated descriptions in DaTikZ-V4. For additional information, we refer to the Appendix A.1.

4 DATASET

Building on DaTikZ-V3, we introduce DaTikZ-V4, a significantly expanded and refined dataset designed to support the training and evaluation of Text-to-TikZ models. The development of DaTikZ-V4 addresses the growing need for both larger and higher-quality datasets, which are critical for surpassing not only proprietary state-of-the-art models like GPT-5 but also increasingly more capable open-source LLMs such as Qwen3.

Data Sourcing To enhance dataset scale, we first identify GitHub as a valuable large-scale source of high-quality graphics programs. With over one billion repositories, GitHub hosts a wealth of educational resources, tutorials, theses, books, and personal projects, many of which contain TikZ code. From this, we clone approximately 5,500 repositories containing `.tex` or `.pgf` files with TikZ content, resulting in over 400,000 unique TikZ samples. This GitHub-only subset is nearly as large as the entirety of DaTikZ-V3. To further expand coverage, we also extend sourcing from arXiv by including data post-2021 to mid 2025. The increasing amounts of arXiv submissions each year allows us to source 1M additional samples, resulting in over 2M TikZ graphics in total. A comparison of DaTikZ-V4 to previous releases is seen in Table 2.

Table 2: Unique TikZ graphics across all DaTikZ versions.

Source	DaTikZ	V2	V3	V4
arXiv	85,656	326,450	407,851	1,471,083
GitHub	0	0	0	413,178
TeX SE	29,238	30,609	42,654	97,909
Synthetic	1,957	1,958	2,256	13,514
Curated	981	1,566	3,646	5,196
Total	117,832	360,583	456,407	2,000,880

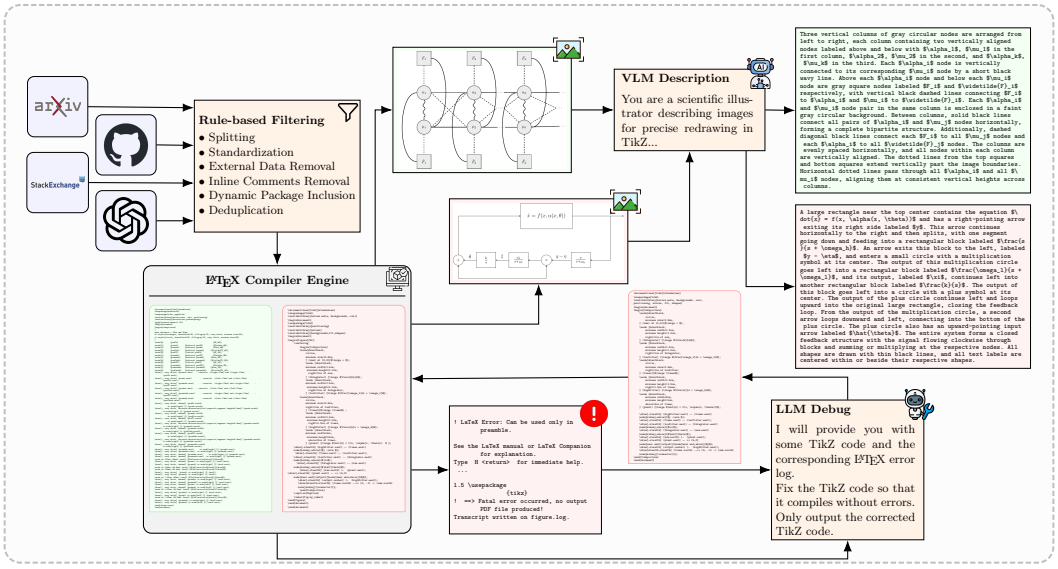


Figure 2: Overview of the data preprocessing workflow. We start by sourcing TikZ graphics programs primarily from arXiv, GitHub, TeX SE, as well as synthetic data. Next, rule-based filtering techniques are applied, and the TikZ code is rendered. Uncompilable code undergoes an iterative debugging process using LLMs alongside the error messages to attempt error correction. Finally, all compilable code images are described using VLMs.

Filtering Beyond traditional `tikzpicture` environments, we now extract from other environments such as `tikz-cd` (common in mathematical diagrams) and `circuitikz` (used in elec-

tronics). Since individual figures often contain multiple subfigures, we recursively split and extract all subfigure content. Furthermore, we enforce a standardized TikZ code by wrapping the code inside the `\documentclass[tikz]{standalone}` environment. Additionally, we implement a dynamic package detection approach by using regular expressions to include necessary LaTeX packages (e.g., recognizing `circuitikz` from context such as `resistor`). We also remove any code that depends on external files (e.g., `\input{...}`, `\includegraphics{...}`), as well as all inline comments, to improve compilation rates and reduce noise. Lastly, we apply exact deduplication and dismiss all samples where the number of characters is both smaller than 100 and larger than 4000.

LLM Debugging Due to the low compilation success rate, especially from arXiv (success rate 31.3%), we introduce an LLM-based debugging pipeline. Given a code snippet and its compiler error, an LLM is instructed to fix the TikZ code. Using Qwen-32B across our corpus of 1.3M uncompilable TikZ samples, we successfully repair 600K instances in the first pass. This approach substantially boosts the proportion of usable TikZ programs at scale.

VLM-based Image Description As shown in Section 3, raw captions are often unhelpful for figure reproduction, potentially leading to severe hallucinations. To mitigate this, we employ VLMs to generate precise descriptions of TikZ figures. Using Qwen2.5-VL-7B-Instruct, we annotate around 1.3M compilable samples, producing the first large-scale dataset of TikZ paired with semantically rich textual descriptions, providing stronger supervision for downstream model training. An overview of our dataset construction is illustrated in Figure 2. For ablations and further details about prompts and frameworks, we refer to A.2.

5 METHOD

We train Text-to-TikZ models in two stages: SFT to ground models in TikZ syntax and task-specific token distributions, followed by RL for incorporating feedback from rendered images to enforce enhanced visual alignment (Rodriguez et al., 2025). Similar two-stage paradigms have also proven effective in related domains such as code generation and mathematical reasoning, where surface-level syntax is complemented by execution-level accuracy (Le et al., 2022; Gehring et al., 2025).

Stage 1: Supervised Finetuning Given a figure description x_{desc} and ground-truth TikZ sequence $x_{\text{tikz}} = (x_1, \ldots, x_T)$, we minimize the standard autoregressive negative log-likelihood:

$$\mathcal{L}_{\text{SFT}}(\theta) = \mathbb{E}_{(x_{\text{desc}},\, x_{\text{tikz}})\sim\mathcal{D}}\left[-\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t},\, x_{\text{desc}}) \right] \tag{1}$$

This ensures syntactic validity and prompt alignment. At the same time, the model remains unaware of the rendered semantics of the figure, which leads to common errors such as loops, irrelevant content, or incorrect spatial relations.

Stage 2: Reinforcement Learning To address this, we reinterpret the SFT model $p_{\theta_{\text{SFT}}}$ as a stochastic policy and apply reinforcement learning with GRPO. For each description, G rollouts $\{o_1, \ldots, o_G\} \sim p_{\theta_{\text{old}}}(\cdot \mid x_{\text{desc}})$ are sampled, each of which is assigned a scalar reward $\{r_1, \ldots, r_G\}$ scored by a reward model, and updated with group-centered advantages $A_i = \frac{r_i - \text{mean}(\{r_j\})}{\text{std}(\{r_j\})}$. The GRPO objective we maximize is:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x_{\text{desc}}\sim\mathcal{D}}\left[\frac{1}{LG} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \min\left(\frac{p_\theta(o_i \mid x_{\text{desc}})}{p_{\theta_{\text{old}}}(o_i \mid x_{\text{desc}})} A_i, \right.\right.$$
$$\left.\left. \text{clip}\left(\frac{p_\theta(o_i|x_{\text{desc}})}{p_{\theta_{\text{old}}}(o_i|x_{\text{desc}})}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}} \right) A_i \right) - \beta\, D_{\text{KL}}\big(p_\theta \,\|\, p_{\theta_{\text{SFT}}}\big) \right]$$

where β regulates the KL penalty. We implement the "Dr.GRPO" variant (Liu et al., 2025), which replaces the response-level normalization by a token-level normalization with a constant divisor (the maximum completion length L). This removes the response length bias in TikZ sequences,

where longer responses are under-penalized. Furthermore, we apply the "Clip-Higher" strategy from DAPO (Yu et al., 2025), which decouples the clipping threshold ϵ into ϵ_{low} and ϵ_{high}. This allows more headroom for increasing the probability of low-probability exploration tokens (by raising ϵ_{high}), while still preventing collapse of high-probability exploitation tokens (by keeping ϵ_{low} smaller). As in DAPO, we set $\epsilon_{low} = 0.2$ and $\epsilon_{high} = 0.28$. Additionally, we remove scaling the advantages by the standard deviation of the group rewards to not introduce a bias towards more or less difficult prompts, and mask all samples whose completion was cut by the length cap as we find that it increases training stability. Finally, we disable the KL coefficient ($\beta = 0$) and sample with `temperature=1.0` and `top_p=0.9`.

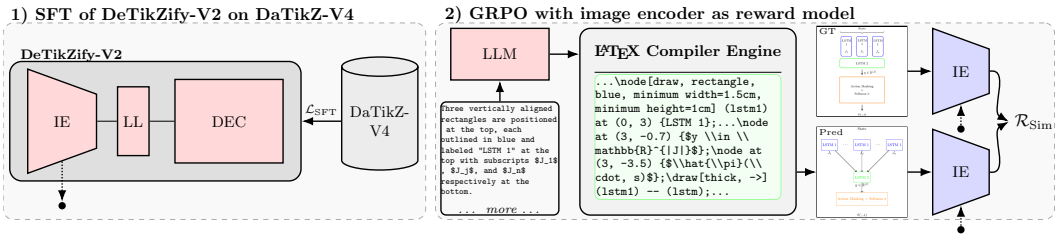


Figure 3: Overview of our post-SFT optimization steps. We first fully finetune DeTikZify-V2 consisting of an image encoder (IE), linear layer (LL) and LLM decoder (DEC) on our larger DaTikZ-V4 where we then use its enhanced IE to further finetune our LLMs based on the semantic similarity of the embeddings between ground truth and rendered image in an online RL setting using GRPO. The IE is kept frozen during RL optimization to mitigate reward hacking.

Rewards Designing reward signals for Text-to-TikZ is challenging: they must capture faithfulness, scientific style, attributes, and spatial relations. Recent work has shown that metrics such as CLIPScore or DreamSim correlate poorly with human judgments as they fail to represent nuances in scientific figures (Belouadi et al., 2025) and are prone to reward hacking (e.g., embedding text into figures) (Rodriguez et al., 2025).

To the best of our knowledge, we propose the first domain-specific reward model for Text-to-TikZ. It builds on the image encoder of DeTikZify-V2 (Belouadi et al., 2024b), which is a SigLIP (Zhai et al., 2023) vision encoder of `PaliGemma-3b-mix-448` (Beyer et al., 2024)), originally trained on DaTikZ-V3 for inverse graphics (image $\rightarrow$ TikZ). DeTikZify consists of an image encoder followed by a linear layer and an LLM decoder. By keeping the image encoder unfrozen during training, it incidentally learns to generate good low-dimensional representations of scientific figures in order to accurately reproduce the figure, allowing us to utilize it to measure semantic similarity between the embeddings of two scientific figures more accurately. With DaTikZ-V4 providing a much larger dataset, we retrain DeTikZify-V2 end-to-end, yielding a stronger encoder that produces richer, more generalizable embeddings of scientific diagrams. Subsequently, we use the retrained image encoder as our reward model in an online RL environment with GRPO. Both steps are illustrated in Figure 3. Training details are provided in A.3.

For reward computation, pooled cosine similarity is not available since DeTikZify-V2 outputs patch-level embeddings. We therefore adopt an Earth Mover's Distance (EMD) (Rubner et al., 1998; Kusner et al., 2015) formulation, inspired by test-time scaling approaches in TikZero (Belouadi et al., 2025). Given patch embeddings $\mathbf{x} = \{x_i\}_{i=1}^{|\mathbf{x}|}$ and $\mathbf{y} = \{y_j\}_{j=1}^{|\mathbf{y}|}$ from ground truth and predicted images, with distance matrix $D_{i,j} = 1 - \cos(x_i, y_j)$, the similarity reward is defined as

$$\mathcal{R}_{\text{Sim}}(\mathbf{x}, \mathbf{y}) \;=\; 1 - \frac{\sum_{i=1}^{|\mathbf{x}|} \sum_{j=1}^{|\mathbf{y}|} F_{i,j} D_{i,j}}{\sum_{i=1}^{|\mathbf{x}|} \sum_{j=1}^{|\mathbf{y}|} F_{i,j}}, \tag{2}$$

where $F \in \mathbb{R}_{\geq 0}^{|\mathbf{x}| \times |\mathbf{y}|}$ is the optimal flow matrix that minimizes the transport cost, subject to $\sum_i F_{i,j} = 1/|\mathbf{y}|$ and $\sum_j F_{i,j} = 1/|\mathbf{x}|$. This formulation yields a scalar reward in $[0, 1]$ capturing semantic alignment. Finally, we add a format reward to ensure that the TikZ code starts and ends with valid document environments (i.e., `\documentclass[tikz]{standalone}`, followed by `\begin{document}`, and ending with `\end{document}`). Non-conforming outputs receive a reward of zero.

6 EXPERIMENTS

Experimental Setup For evaluation, we construct a contamination-free test set of 1,047 samples from DaTikZ-V4. To prevent overlap with training data, we (i) restrict to post–May 2025 samples, (ii) enforce per-source uniqueness (e.g., one figure per arXiv paper or GitHub repo, removing the rest from training), (iii) filter with n-gram matching (OpenAI, 2023), and (iv) manual inspection to discard trivial or corrupted figures. To avoid model bias, all test descriptions are generated by GPT-4o. For RL-tuning, we create DaTikZ-V4-RL, a 160K-sample subset obtained by repairing uncompilable figures via a second LLM debugging step and re-describing them with Qwen2.5-VL-7B. This provides additional high-quality pairs beyond the training split.

Models We benchmark nine LLMs: (i) proprietary GPT-5[1] and GPT-4o (OpenAI et al., 2024), (ii) open-source Qwen3 (32B, 8B), Qwen3-Coder-30B-A3B (Yang et al., 2025), Qwen2.5 (14B, 3B) (Qwen et al., 2025), TikZero-Plus-10B (Belouadi et al., 2025), and Llama3.1-8B (Grattafiori et al., 2024), and (iii) our fine-tuned Qwen2.5-3B and Qwen3-8B models. We refer to our trained models as TikZilla, with the following variants: TikZilla-3B and TikZilla-8B (SFT only), and TikZilla-3B-RL and TikZilla-8B-RL (two-stage training). In addition, we also test RL-only training.

Evaluation Metrics We evaluate along four axes: (i) CLIPScore (CLIP) (Hessel et al., 2021) for text–image alignment, (ii) DreamSIM (DSim) (Fu et al., 2023) for perceptual fidelity, (iii) TeX Edit Distance (TED) (Kusner et al., 2015) for code similarity, and (iv) Compilation Rate (CR) for executability. We also report average tokens (AT) for efficiency. An aggregate score (AVG) is computed as the mean of CLIP, DSim, and 1-TED. Additional details are reported in A.4.

7 RESULTS

Main Results Table 3 reports results on automatic metrics. Our SFT+RL-tuned Qwen models achieve the best AVG performance, with TikZilla-3B-RL reaching 0.385 and TikZilla-8B-RL 0.384. Both surpass GPT-5 (0.365), despite it being recently released as one of the strongest reasoning LLMs, evaluated with no output length restrictions. Compared to the recently released TikZero-Plus-10B, TikZilla-3B-RL improves by +0.085 on CLIP and +0.334 on DSim, while achieving a 37% higher compilation rate and requiring 261 fewer tokens on average. Similar improvements hold for TikZilla-8B-RL. These results highlight the effectiveness of our two-stage training process, combining high-quality data with a domain-specific reward model. For qualitative examples with TikZ code, we refer to the Appendix (Figure 14, 15, 16, and 17).

MODEL SIZE AND TRAINING REGIME Interestingly, the smaller Qwen2.5-3B not only closes the gap with Qwen3-8B but even slightly outperforms it once trained with SFT+RL. However, its low baseline (0.202) indicates that it strongly relies on SFT before RL, whereas Qwen3-8B benefits from RL directly ($0.251 \rightarrow 0.357$). This suggests that SFT primarily provides syntax grounding for smaller models, while larger models already encode some TikZ knowledge that RL can amplify.

EMERGENT PROPERTIES RL consistently improves compilation rates to 95–98% and reduces token length, indicating more efficient code generation. Unlike prior SVG studies (Rodriguez et al., 2025), which required explicit code efficiency rewards, we observe a natural reduction in sequence length. We hypothesize this stems from our semantic reward model penalizing hallucinated or redundant elements, indirectly encouraging conciseness. A deeper comparison with explicit efficiency rewards is left for future work.

Human Evaluation We conduct a human evaluation with 9 expert annotators (6 PhD, 2 postdoc, 1 faculty member). Each annotator rated 30 randomized figures/descriptions across 4–5 models, using a 1–5 Likert scale (1 = uncompilable, 5 = publication-ready). Two criteria were considered: (i) textual alignment (does the output follow the provided description?) and (ii) image alignment (does the output match the original ground-truth figure?). Annotator agreement was strong (Cohen's $\kappa = 0.814$ for text, 0.794 for image). Full details are provided in A.5.

[1] https://openai.com/de-DE/index/gpt-5-system-card/

Table 3: Results of all models on the evaluation subset of DaTikZ-V4. Both of our models trained with SFT and RL perform best, while GPT-5 and Qwen3-32B are the best proprietary and open-source LLMs respectively. **Bold** denotes best-performing while underline is second-best.

LLM	CLIP↑	DSim↑	TED↓	AVG↑	CR↑	AT
GPT-5	0.181	0.679	0.765	0.365	88%	480
GPT-4o	0.147	0.580	0.767	0.320	78%	404
Qwen3-32B	0.149	0.583	0.765	0.322	79%	416
Qwen3-Coder-30B-A3B	0.140	0.566	0.778	0.309	77%	472
Qwen2.5-14B	0.132	0.511	0.765	0.293	71%	376
TikZero-Plus-10B	0.104	0.397	0.807	0.231	61%	742
Llama3.1-8B	0.088	0.339	0.786	0.214	50%	529
Qwen2.5-3B	0.081	0.315	0.789	0.202	52%	387
Qwen2.5-3B (+RL)	0.098	0.505	0.795	0.269	**98%**	234
TikZilla-3B	0.161	0.613	0.802	0.324	89%	672
TikZilla-3B-RL	**0.189**	**0.731**	0.766	**0.385**	**98%**	481
Qwen3-8B	0.106	0.421	0.775	0.251	63%	412
Qwen3-8B (+RL)	0.169	0.669	0.768	0.357	**98%**	393
TikZilla-8B	0.158	0.602	0.793	0.322	86%	729
TikZilla-8B-RL	0.185	0.727	**0.761**	0.384	95%	459

RESULTS Figure 4 shows that GPT-5 achieved the highest textual score (4.18) and tied with our TikZilla-8B-RL on image evaluation (3.48 vs. 3.46). TikZilla-3B-RL also performed competitively (3.40 text, 3.30 image). Reinforcement learning substantially boosted both Qwen models (+0.75 and +0.67 points), while base models lagged 1.5–2 points behind. Interestingly, most models (especially GPT-5) scored higher on the textual evaluation than on the image evaluation. We hypothesize two possible explanations: (i) if VLM-generated captions omit or misrepresent visual details, models may score highly on textual alignment (satisfying the description) but lower on image alignment (failing to match the true figure). (ii) Human annotators may apply stricter criteria when comparing against ground-truth images than when comparing against text. Disentangling these two factors remains an open question, which we leave for future work.

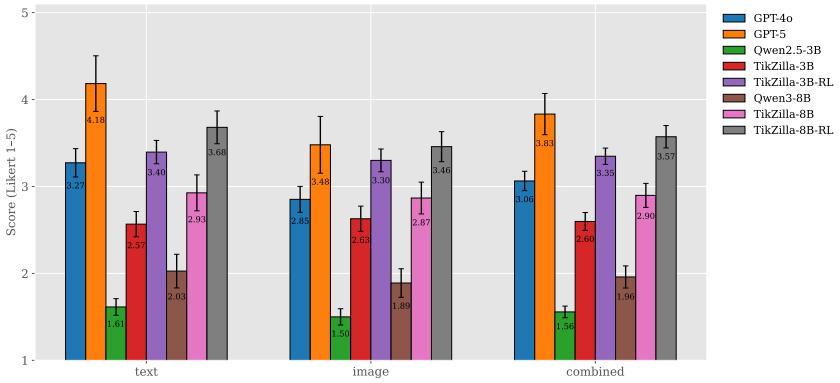


Figure 4: Average Likert-scale ratings (1–5, higher is better) with 95% confidence intervals for eight LLMs, evaluated under two settings: (i) alignment with textual descriptions and (ii) alignment with ground-truth images. Combined scores are shown as the average of both settings.

CORRELATION WITH METRICS Finally, we compute correlations between automatic metrics and human scores using Spearman's ρ. CLIP ($\rho_{CLIP} = 0.260$) and TED ($\rho_{1-TED} = 0.307$) show weak, DSim moderate ($\rho_{DSim} = 0.586$), and our reward model strong ($\rho_{\mathcal{R}_{Sim}} = 0.714$) correlation. This validates our design of a domain-specific reward model aligned with human judgment.

Ablations We conduct ablations to isolate the effects of input quality, LLM-based debugging and reward modeling (Table 4).

Table 4: Ablations on input data quality, debugging, and reward modeling. VLM-based descriptions consistently outperform captions, while mixing or oversampling captions brings no gains. Our LLM-based debugging step and retraining the reward model on DaTikZ-V4 both yield improvements.

LLM	CLIP↑	DSim↑	TED↓	AVG↑	CR↑	AT
GPT-4o$_{\text{cap.}}$	0.105	0.469	0.763	0.270	80%	337
GPT-4o$_{\text{desc.}}$	0.143	0.568	0.767	0.315	76%	416
Qwen2.5-3B (+SFT$_{\text{cap.}}$)	0.134	0.511	0.809	0.279	79%	768
Qwen2.5-3B (+SFT$_{\text{desc.}}$)	0.141	0.530	0.805	0.289	85%	651
Qwen2.5-3B (+SFT$_{\text{desc. } \vee \text{ cap.}}$)	0.154	0.589	0.804	0.313	85%	735
Qwen2.5-3B (+SFT$_{\text{desc. + cap.}}$)	0.157	0.599	0.799	0.319	89%	787
Qwen2.5-3B (+SFT$_{\text{no debug}}$)	0.138	0.534	0.809	0.288	79%	762
TikZilla-3B	0.161	0.613	0.802	0.324	89%	672
Qwen2.5-3B (+SFT+RL$_{\text{DaTikZ V3}}$)	0.183	0.712	0.770	0.375	97%	496
TikZilla-3B-RL	0.192	0.741	0.766	0.389	98%	481

CAPTIONS VS. DESCRIPTIONS VLM-generated descriptions consistently outperform raw captions. At inference, GPT-4o achieves 0.315 AVG with descriptions versus 0.270 with captions, confirming our earlier analysis that captions are often unhelpful for figure reproduction. Examples are shown in Figure 13 in the Appendix. For training, Qwen2.5-3B also benefits from descriptions (0.289 vs. 0.279), though the gap is smaller, likely due to the limited caption subset (468k samples). Mixing captions/descriptions (desc. $\vee$ cap.) and oversampling descriptions with captions (desc. + cap.) degrade performance, suggesting that low-quality captions dilute training even when more data is added.

LLM-BASED DEBUGGING Models trained only on first-try compilable code perform considerably worse than those trained on the full dataset (0.288 vs. 0.324), highlighting the necessity of our LLM-based debugging pipeline to increase the size of our usable TikZ corpus.

REWARD MODEL TRAINING Lastly, we demonstrate that retraining DeTikZify-V2 on DaTikZ-V4 yields a stronger reward model (0.389 vs. 0.375). Correlations with human judgments also improve ($\rho_{\mathcal{R}_{Sim}} = 0.714$ vs. 0.698), confirming that larger-scale scientific data produces more reliable image encoders for semantic evaluation. We did not ablate against general-purpose rewards such as CLIP or DreamSIM due to resource constraints. However, their weaker alignment with human judgments suggests they would be less effective in this setting.

8 CONCLUSION, LIMITATIONS, AND FUTURE WORK

We presented DaTikZ-V4, a large-scale, high-quality dataset for Text-to-TikZ, and a two-stage training framework combining SFT with RL. Our key contributions are a richer dataset sourced from arXiv and GitHub with LLM-based debugging to improve compilability, VLM-generated descriptions that overcome the low quality of raw captions, and a domain-specific reward model derived from an inverse-graphics image encoder, which correlates strongly with human judgments of figure quality. Building on these components, we introduced TikZilla, a family of small Qwen-based models that achieve near-perfect compilation rates and even surpass much larger commercial systems such as GPT-4o across automatic and human evaluation. Beyond technical performance, TikZilla demonstrates the feasibility of building reproducible, efficient, and ethical text-to-image generation systems with small-scale open models, reducing reliance on costly proprietary solutions.

A key limitation is that our figure descriptions are generated automatically by VLMs, which may introduce omissions or hallucinations. This can bias training and, in rare cases, reward optimization may reinforce errors when descriptions diverge from figures. More reliable annotation methods and fine-grained reward functions are therefore crucial directions for future work. Beyond addressing these issues, future work should focus on designing automatic metrics with stronger alignment to human perception, and extending our approach to other structured generation tasks (e.g., LaTeX tables, CAD, or flowcharts), where programmatic fidelity is critical.

REFERENCES

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Jonas Belouadi, Anne Lauscher, and Steffen Eger. AutomaTikZ: Text-guided synthesis of scientific vector graphics with TikZ. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=v3K5TVP8kZ.

Jonas Belouadi, Simone Paolo Ponzetto, and Steffen Eger. DeTikZify: Synthesizing graphics programs for scientific figures and sketches with TikZ. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=bcVLFQCOjc.

Jonas Belouadi, Eddy Ilg, Margret Keuper, Hideki Tanaka, Masao Utiyama, Raj Dabre, Steffen Eger, and Simone Paolo Ponzetto. Tikzero: Zero-shot text-guided graphics program synthesis, 2025. URL https://arxiv.org/abs/2503.11509.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. Paligemma: A versatile 3b vlm for transfer, 2024. URL https://arxiv.org/abs/2407.07726.

Zhenyu Bi, Minghao Xu, Jian Tang, and Xuan Wang. AI for science in the era of large language models. In Jessy Li and Fei Liu (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pp. 32–38, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-tutorials.5. URL https://aclanthology.org/2024.emnlp-tutorials.5/.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. Deepsvg: A hierarchical generative network for vector graphics animation. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 16351–16361. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/bcf9d6bd14a2095866ce8c950b702341-Paper.pdf.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

Steffen Eger, Yong Cao, Jennifer D'Souza, Andreas Geiger, Christian Greisinger, Stephanie Gross, Yufang Hou, Brigitte Krenn, Anne Lauscher, Yizhi Li, Chenghua Lin, Nafise Sadat Moosavi, Wei Zhao, and Tristan Miller. Transforming science with large language models: A survey on ai-assisted scientific discovery, experimentation, content generation, and evaluation, 2025. URL https://arxiv.org/abs/2502.05151.

Stephanie Fu, Netanel Y. Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: learning new dimensions of human visual similarity using synthetic data. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Jonas Gehring, Kunhao Zheng, Jade Copet, Vegard Mella, Quentin Carbonneaux, Taco Cohen, and Gabriel Synnaeve. Rlef: Grounding code llms in execution feedback with reinforcement learning, 2025. URL https://arxiv.org/abs/2410.02089.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R D Costa, José R Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an ai co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein,

Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7514–7528, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.595. URL https://aclanthology.org/2021.

emnlp-main.595/.

Ting-Yao Hsu, Chieh-Yang Huang, Shih-Hong Huang, Ryan Rossi, Sungchul Kim, Tong Yu, C Lee Giles, and Ting-Hao Kenneth Huang. Scicapenter: Supporting caption composition for scientific figures with machine-generated captions and ratings. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703317. doi: 10.1145/3613905.3650738. URL https://doi.org/10.1145/3613905.3650738.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models, 2023. URL https://arxiv.org/abs/2304.01933.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 1049–1065, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.67. URL https://aclanthology.org/2023.findings-acl.67/.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models, 2025. URL https://arxiv.org/abs/2503.06749.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 957–966. JMLR.org, 2015.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 21314–21328. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/8636419dea1aa9fbd25fc4248e702da4-Paper-Conference.pdf.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

Raphael Gontijo Lopes, David R Ha, Douglas Eck, and Jonathon Shlens. A learned representation for scalable vector graphics. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7929–7938, 2019. URL https://api.semanticscholar.org/CorpusID:102353397.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery, 2024. URL https://arxiv.org/abs/2408.06292.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey, 2025. URL https://arxiv.org/abs/2402.06196.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774. URL https://doi.org/10.48550/arXiv.2303.08774.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum,

Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin (eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

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

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Juan A Rodriguez, David Vazquez, Issam Laradji, Marco Pedersoli, and Pau Rodriguez. Figgen: Text to scientific figure generation. *arXiv preprint arXiv:2306.00800*, 2023.

Juan A. Rodriguez, Abhay Puri, Shubham Agarwal, Issam H. Laradji, Pau Rodriguez, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. Starvector: Generating scalable vector graphics code from images and text, 2024. URL https://arxiv.org/abs/2312.11556.

Juan A. Rodriguez, Haotian Zhang, Abhay Puri, Aarash Feizi, Rishav Pramanik, Pascal Wichmann, Arnab Mondal, Mohammad Reza Samsami, Rabiul Awal, Perouz Taslakian, Spandana Gella, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. Rendering-aware reinforcement learning for vector graphics generation, 2025. URL https://arxiv.org/abs/2505.20793.

Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 59–66, 1998. doi: 10.1109/ICCV.1998.710701.

Benny Tang, Angie Boggust, and Arvind Satyanarayan. VisText: A benchmark for semantically rich chart captioning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7268–7298, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.401. URL https://aclanthology.org/2023.acl-long.401/.

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S. Yu. Multimodal large language models: A survey, 2023. URL https://arxiv.org/abs/2311.13165.

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

Hiroshi Yoshihara, Taiki Yamaguchi, and Yuichi Inoue. A practical two-stage recipe for mathematical llms: Maximizing accuracy with sft and efficiency with reinforcement learning, 2025. URL https://arxiv.org/abs/2507.08267.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023. URL https://arxiv.org/abs/2303.15343.

Leixin Zhang, Steffen Eger, Yinjie Cheng, WEIHE ZHAI, Jonas Belouadi, Fahimeh Moafian, and Zhixue Zhao. Scimage: How good are multimodal large language models at scientific text-to-image generation? In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ugyqNEOjoU.

Qihao Zhu Runxin Xu Junxiao Song Mingchuan Zhang Y.K. Li Y. Wu Daya Guo Zhihong Shao, Peiyi Wang. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, Hao Li, Jiahao Wang, Nianchen Deng, Songze Li, Yinan He, Tan Jiang, Jiapeng Luo, Yi Wang, Conghui He, Botian Shi, Xingcheng Zhang, Wenqi Shao, Junjun He, Yingtong Xiong, Wenwen Qu, Peng Sun, Penglong Jiao, Han Lv, Lijun Wu, Kaipeng Zhang, Huipeng Deng, Jiaye Ge, Kai Chen, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models, 2025. URL https://arxiv.org/abs/2504.10479.

Bocheng Zou, Mu Cai, Jianrui Zhang, and Yong Jae Lee. Vgbench: Evaluating large language models on vector graphics understanding and generation. *arXiv preprint arXiv:2407.10972*, 2024.

A Appendix

A.1 Caption Quality Analysis

Our caption quality analysis involved three annotators: one bachelor's student, one PhD student, and one faculty member (all male). From our subset of DaTikZ-V3, 74% of samples originate from arXiv and 26% from TeX SE. One annotator completed the evaluation sheet in Figure 5, based on the taxonomy in Table 5. This annotator also manually described all 200 scientific figures, which we subsequently used as reference descriptions to compute BLEU-4, ROUGE-L, and STS with the `all-mpnet-base-v2` sentence encoder between human descriptions and VLM-generated descriptions.

The other two annotators each described 30 figures to measure agreement, yielding unweighted $\kappa = 0.35$ and weighted $\kappa = 0.63$. The structural elements for scientific figure captions were adapted from best practices in academic writing and prior research taxonomies (Tang et al., 2023; Hsu et al., 2024).

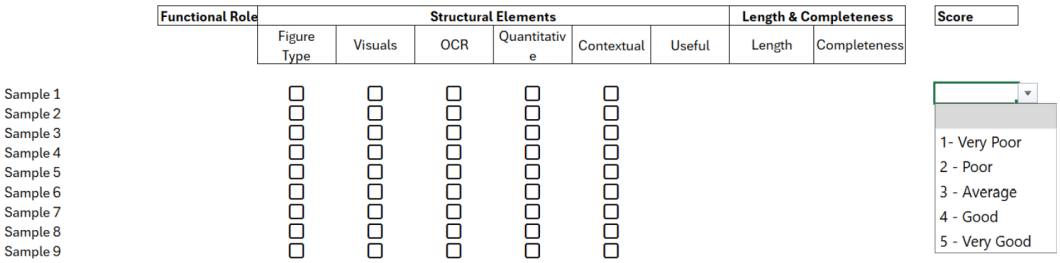


Figure 5: Screenshot of our evaluation form for the first nine scientific figures.

Table 5: Caption analysis taxonomy for structural elements and usefulness scores.

Structural Elements	**Figure type:** names the high-level type (e.g., graph, tree, workflow). **Visual details:** mentions colors, shapes, axes, layout/spatial relations. **OCR:** includes textual elements visible in the figure (axis labels, annotations, math), aiding correct labeling. **Contextual reference:** points outside the figure (e.g., "see Sec. 3"). Useful but reduces standalone utility. **Quantitative content:** numbers, formulas, code. Adds technical substance (often paired with OCR).
Usefulness Scores	**Very Poor:** not meaningfully descriptive. May be only a label or irrelevant text. **Poor:** somewhat relevant but vague/incomplete. Mentions topic/elements without adequate clarity or context. **Average:** describes the main content but lacks depth/specifics. States what it is without highlighting key details. **Good:** clear, specific, and near-complete. Covers important visual/quantitative details and structure. **Very Good:** precise, insightful, and largely self-contained. Explains key elements so the figure is almost unnecessary.

A.2 Dataset

To create synthetic data, we follow a strategy similar to SciImage (Zhang et al., 2025). We first generate 2,000 templates with varied terms, each used to produce 10 queries that generate TikZ code. All steps are performed using GPT-4o with minimal human intervention.

LLM Debugging For LLM-based debugging, we use the prompt in Figure 6. We first tested this on a subset of 753 samples spanning all sources, manually evaluating the percentage of compilable, non-empty, and non-corrupted outputs. As shown in Table 6, Qwen3-32B (non-thinking) was the best-performing model, recovering 49.40% of errors in a single pass and 59.04% after three repair rounds. Smaller Qwen variants and Qwen2.5-7B-Instruct (Qwen et al., 2025) performed considerably worse. We therefore applied Qwen3-32B for large-scale debugging, which took 14 days on 4 × A100 40GB GPUs using the vLLM framework (Kwon et al., 2023). Examples of the debugging process are shown in Figure 7 and 8.

Table 6: Accuracy of different LLMs in debugging TikZ code from error logs over three refinement iterations. Bold indicates the best-performing model.

LLM	Iteration 1	Iteration 2	Iteration 3
Qwen2.5-7B	17.49%	28.03%	34.08%
Qwen3-4B	14.17%	18.56%	21.98%
Qwen3-8B	35.11%	39.36%	41.49%
Qwen3-32B	**49.40%**	**55.42%**	**59.04%**
GPT-4o-mini	36.82%	41.36%	43.62%
GPT-4o	48.10%	53.73%	58.89%

LLM Debug Prompt

```
I will provide you with some TikZ code and the corresponding LaTeX error log.  Fix the
TikZ code so that it compiles without errors.  Only output the corrected TikZ code.\n
Original TikZ Code:  {tikz_code}\n
Compilation Error Log:  {log_message}
```

VLM-based Image Description The prompt for image description is shown in Figure 9. We use few-shot in-context learning (Brown et al., 2020) with two high-STS human descriptions as exemplars. We run Qwen2.5-VL-7B-Instruct, which was the strongest open-source VLM in our evaluation, to describe all figures in DaTikZ-V4. Processing required 2 days on 4 × A100 40GB GPUs.

VLM Description Prompt

```
You are a scientific illustrator describing images for precise redrawing in TikZ.\n
Your task is to describe the image in precise, continuous prose without bullet points,
lists, or line breaks.\n
Start directly with the main object or scene.  Avoid introductory phrases like
'Certainly!', 'The image depicts...', 'Here is a precise description.'.\n
Use clear, active language focused on geometry, labels, colors, spatial relationships,
coordinates, and other visible properties.\n
Describe all visible elements such as shapes, lines, arrows, and labels, including
their relative or absolute positions, dimensions, and orientation.\n
Use consistent, minimal naming for objects (e.g., 'circle A', 'line L1') and specify
label positions relative to shapes precisely.\n
Only describe exact, concrete visual elements that enable precise image reconstruction
in TikZ.\n
Avoid vague, interpretive, or inferential language, and exclude summaries,
conclusions, or commentary about the image's meaning, function, or aesthetics.\n
Here are a few examples:\n
A thin black horizontal line centered in the middle, containing nine evenly spaced
black dots, and labeled x_2 at the left.  Each dot is connected by a thin black line in
an alternating pattern to either x_0 (placed at the top middle) or x_1 (placed at the
bottom middle).\n
A line chart has different instruction scales of 1/10, 1/4, 1/2, and 1 on the x-axis.
On the y-axis it shows BLEU scores between 20 and 50, with steps of 5.  The chart
contains three lines with Zh-En in blue, De-En in red, and Fr-En in brown.  All BLEU
scores are initially 20 at the lowest instruction scale.  As the instruction scale
increases, BLEU scores improve for all pairs.  De-En is the highest, closely followed
by Fr-En and then Zh-En far below.  The increase is largest from 1/10 to 1/4 and only
marginally above an instruction scale of 1/4.  The legend is placed inside the chart
at the top left.\n
Write a description in this exact style for the given image.
```

A.3 METHOD

For finetuning DeTikZify-V2, we use the training split of DaTikZ-V4 consisting of 1.3M Image–TikZ pairs. Inputs are 448×448-pixel images with a maximum output length of 2048 tokens. Training runs for two epochs with a learning rate of 5e-5, AdamW (Loshchilov & Hutter, 2019), cosine scheduler, and 3% linear warmup. The batch size is 128, trained on 4 × H200 140GB GPUs for 12 days.

LLM-based TikZ Debugging

Original TikZ Code:

```
\documentclass[tikz]{standalone}
\usepackage[utf8]{inputenc}
\usepackage{circuitikz}
\usepackage{float}
\usepackage{calc}
\begin{document}
\begin{circuitikz}[american, straight voltages]
   \draw (-1,0)
   to [american voltage source, v=$V_P$, invert, voltage shift=1] (-1,4)
   to [R, R=$R_p$, i^>=$i_p$] (2,4)
   to [R=$R_L$] (4,4)
   to [L, l_=$L$, v^<=$v_L$, i=$i_L$, voltage shift=1.5] (7,4)
   to [Tnigbt,bodydiode] (10,4)
   to [short] (12,4)
   to [american voltage source, v^<=$V_{out}$, voltage shift=1] (12,0)
   to [short] (-1,0)
   (2.0,4) to [R=$R_Ci$, i=$i_{Ci}$] (2.0,1.5)
   to [C, l_=$C_i$, v^<=$v_{Ci}$] (2.0,0)
   (7.2,4) to [Tnigbt,bodydiode, invert] (7.2,0)
   (10.0,4) to [R=$R_Co$, i=$i_{Co}$] (10.0,1.5)
   to [C, l_=$C_o$, v^<=$v_{Co}$] (10.0,0)
   (8.5,5) node[align=center]{$G_2$}
   (6.1,2) node[align=center]{$G_1$}
   (7.2,0) node[circ, scale=1.5]{$1$}
   (7.2,4) node[circ, scale=1.5]
   (2,0) node[circ, scale=1.5]
   (2,4) node[circ, color=red, scale=1.5]
   (10,4) node[circ, color=red, scale=1.5]
   (10,0) node[circ, color=red, scale=1.5]
   ;
\end{circuitikz}
\end{document}
```

Compiler Error Log:

```
! Package tikz Error: A node must have a (possibly empty) label text.
See the tikz package documentation for explanation.
Type H <return> for immediate help.
 ...
l.26 (2,0) node[circ, scale=1.5]
! ==> Fatal error occurred, no output PDF file produced!
Transcript written on figure.log.
```

Corrected TikZ Code (Changed Parts):

```
...
(7.2,4) node[circ, scale=1.5]{}
(2,0) node[circ, scale=1.5]{}
(2,4) node[circ, color=red, scale=1.5]{}
(10,4) node[circ, color=red, scale=1.5]{}
(10,0) node[circ, color=red, scale=1.5]{}
...
```

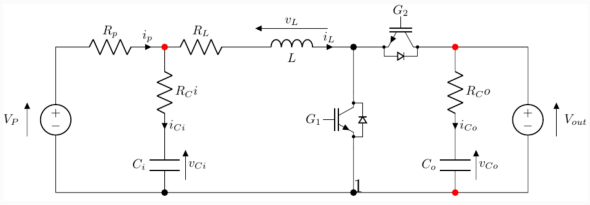


Figure 7: An example of the LLM debugging pipeline. The original TikZ code failed to compile. The compiler error log was passed to the LLM, which generated corrected TikZ code. The fixed code produces the valid figure shown above.

LLM-based TikZ Debugging

Original TikZ Code:

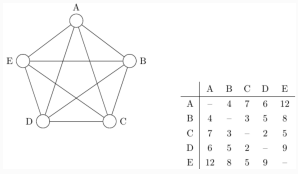
```
\documentclass[tikz]{standalone}
\usepackage{tikz}
\usetikzlibrary{automata,shapes.geometric}
\usepackage{array}
\begin{document}
\begin{figure}[h]
\begin{tabular}{*{2}{>{\centering\arraybackslash}b{\dimexpr0.5\textwidth-2\tabcolsep\relax}}}
\legend{Weighted, complete graph $K_H$}
\begin{tikzpicture}[state/.append style={minimum size=5mm}]
    \node [state] (0) at (-2, 3) [label=left:E] {};
    \node [state] (1) at ( 2, 3) [label=right:B]{};
    \node [state] (2) at (-1.25, 0.75)[label=left:D] {};
    \node [state] (3) at ( 1.25, 0.75) [label=right:C]{};
    \node [state] (4) at ( 0, 4.5) [label=above: A]{};
    \draw (0) to (4);
    \draw (4) to (1);
    \draw (1) to (3);
    \draw (0) to (2);
    \draw (3) to (2);
    \draw (2) to (4);
    \draw (4) to (3);
    \draw (0) to (1);
    \draw (0) to (3);
    \draw (2) to (1);
\end{tikzpicture} &
\renewcommand{\arraystretch}{1.3}
\legend{\textbf{THIS IS TABLE LEGEND}}
\begin{tabular}{c|ccccc}
 & A & B & C & D & E \\
\hline
A & -- & 4 & 7 & 6 & 12 \\
B & 4 & -- & 3 & 5 & 8 \\
C & 7 & 3 & -- & 2 & 5 \\
D & 6 & 5 & 2 & -- & 9 \\
E & 12 & 8 & 5 & 9 & --
\end{tabular}
\end{tabular}
\end{figure}
\end{document}
```

Compiler Error Log:

```
! LaTeX Error: Not allowed in LR mode.
See the LaTeX manual or LaTeX Companion for explanation.
Type H <return> for immediate help.
 ...
l.6 \begin{figure}[h]
! ==> Fatal error occurred, no output PDF file produced!
Transcript written on figure.log.
```

Corrected TikZ Code (Changed Parts):

```
...
\begin{document}
\begin{tabular}{*{2}{>{\centering\arraybackslash}b{\dimexpr0.5\textwidth-2\tabcolsep\relax}}}
\begin{tikzpicture}[state/.append style={minimum size=5mm}]
...
\renewcommand{\arraystretch}{1.3}
\begin{tabular}{c|ccccc}
...
\end{tabular}
\end{tabular}
\end{document}
```



	A	B	C	D	E
A	–	4	7	6	12
B	4	–	3	5	8
C	7	3	–	2	5
D	6	5	2	–	9
E	12	8	5	9	–

Figure 8: An example of the LLM debugging pipeline. The original TikZ code failed to compile. The compiler error log was passed to the LLM, which generated corrected TikZ code. The fixed code produces the valid figure shown above.

A.4 EXPERIMENTS

The prompt template for all models is shown in Figure 10. We also experimented with templates without the standalone environment but found that this reduced performance and compilation rates.

Models Except for GPT-5, decoding uses `temperature=1.0`, `top_p=0.9`, and max length 2048. For GPT-5, we set `reasoning=medium`, `verbosity=medium`, and evaluate a random subset of 100 samples due to cost. For TikZero, trained on caption–TikZ pairs, we only provide the figure description as prompt. For SFT, Qwen2.5-3B is finetuned on DaTikZ-V4 for two days with a learning rate of 1e-4, warmup ratio 3%, cosine scheduler, and batch size 128. Qwen3-8B is trained for four days with a reduced learning rate of 5e-5. For RL on DaTikZ-V4-RL, TikZilla-3B is trained with GRPO for 4,000 iterations (batch size 144, 8 rollouts) using learning rate 5e-6 and weight decay 1e-2. TikZilla-8B uses learning rate 2e-6. RL-only runs were also tested. Training took 5 days for TikZilla-3B and 10 days for TikZilla-8B, all on 4 × H200 140GB GPUs.

Metrics CLIPScore (**CLIP**) is computed with `siglip-so400m-patch14-384`. DreamSIM (**DSim**) uses CLIP, DINO (Caron et al., 2021), and OpenCLIP (`ViT-B/16`). TeX Edit Distance (**TED**) uses `TexLexer`. Average tokens (**AT**) are measured with `o200k_base` tokenizer.

Prompt Template

```
Generate a complete LaTeX document that contains a TikZ figure according to the
following requirements:
{figure_description}
Wrap your code using \documentclass[tikz]{standalone}, and include
\begin{document}...\end{document}.  Only output valid LaTeX code with no extra text.
```

A.5 RESULTS

Human Evaluation We split 9 annotators (6 male, 3 female) into two groups. Group 1 (5 annotators) evaluated GPT-5, GPT-4o, Qwen2.5-3B, TikZilla-3B, and TikZilla-3B-RL. Group 2 (4 annotators) evaluated GPT-5, GPT-4o, Qwen3-8B, TikZilla-8B, and TikZilla-8B-RL. Each annotator received two Excel sheets (textual vs. image alignment), each with 30 randomized samples. We ensured at least five overlapping samples for inter-annotator agreement and five GPT-5 samples (scarcer due to cost). Annotation interfaces are shown in Figures 11 and 12. Likert scale definitions are shown below:

- **5 Excellent**: Figure fulfills all requirements. Few minor issues (e.g., slightly imperfect layout, one or two mislabeled/extra elements) are acceptable. Think about it as publication or almost publication ready where only small tweaks needs to be made.
- **4 Good**: Figure broadly fulfills the requirements and contains no major errors, but it is clearly not perfect. Typical cases include multiple minor flaws (e.g., clutter, small inaccuracies, awkward design) or one moderate issue.
- **3 Fair**: The figure has about one to two major issues (e.g., important elements missing, wrong trends in charts, ...) and/or some minor issues. It is still usable with corrections as parts of the figure are clearly correct.
- **2 Poor**: Several major issues and/or many minor ones. The figure no longer meaningfully reflects the description or GT image (e.g., severe overlaps, high amounts of hallucinated content, ...).
- **1 Failed**: Non-compilable code (already auto-assigned).

Ablations For inference, we ablate input quality by comparing GPT-4o on the evaluation subset where captions are available (GPT-4o$_{\text{cap.}}$) versus the same subset with VLM-generated descriptions instead (GPT-4o$_{\text{desc.}}$). For training, we finetune Qwen2.5-3B on different input variants: (i) Qwen2.5-3B (SFT$_{\text{cap.}}$), using only caption–TikZ pairs (468k samples), (ii) Qwen2.5-3B (SFT$_{\text{desc.}}$), using the same subset but replacing captions with VLM descriptions, (iii) Qwen2.5-3B (SFT$_{\text{desc.} \vee \text{cap.}}$), using the full DaTikZ-V4 dataset, but preferring captions whenever they exist, and (iv) Qwen2.5-3B (SFT$_{\text{desc.} + \text{cap.}}$), oversampling by including both descriptions and captions for all

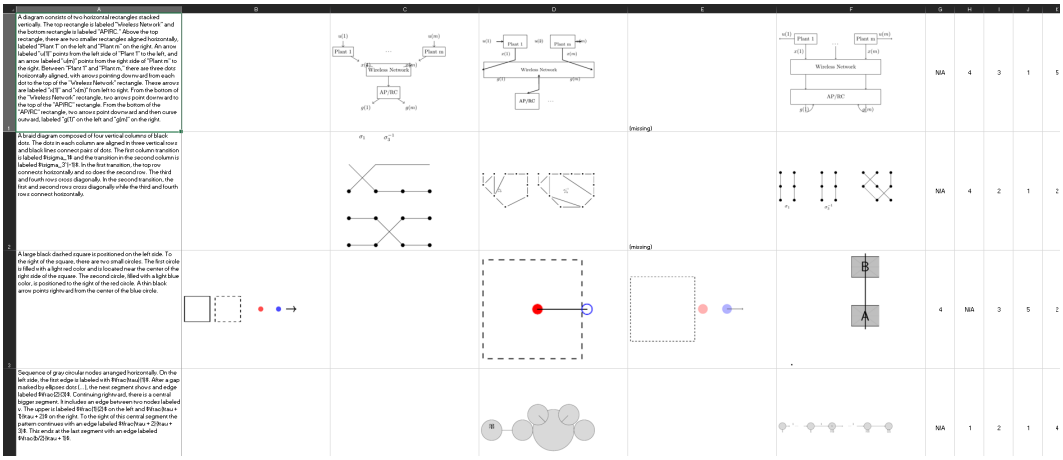


Figure 11: Example of text-image annotations.

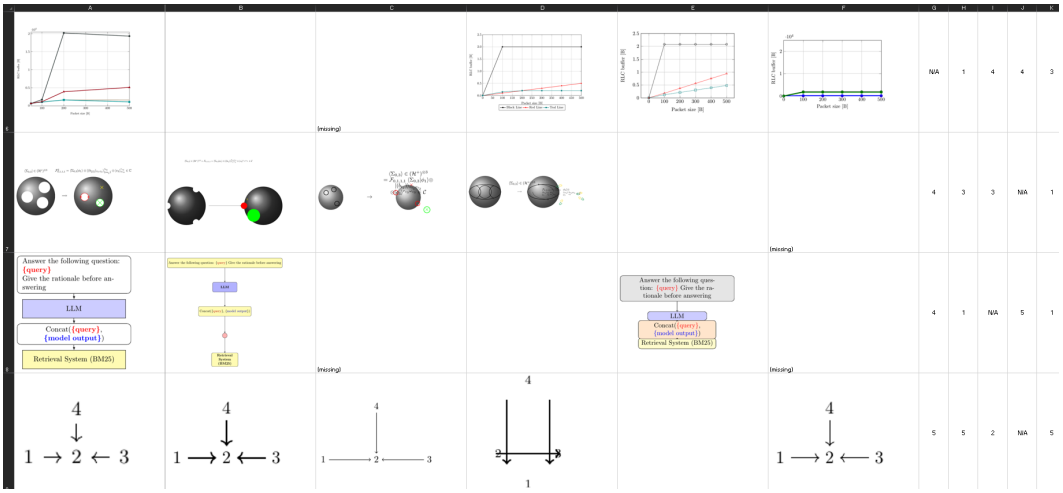


Figure 12: Example of image-image annotations.

samples with paired captions. This setup isolates whether captions add robustness or simply dilute supervision from richer descriptions.

Table 7 shows that arXiv data alone achieves strong results (0.305 AVG). Adding GitHub yields further gains (0.320), while TeX SE and synthetic data provide marginal benefits. This highlights that large-scale, naturally occurring TikZ from arXiv and GitHub are the most valuable sources.

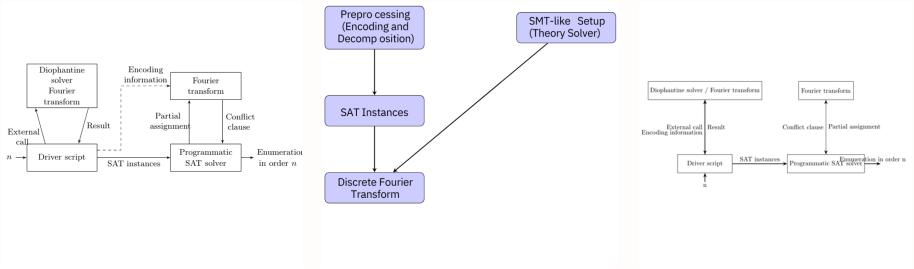
Table 7: Ablation study of different data sources. Using only data from arXiv already leads to very good performances and arXiV + GitHub almost reaches its full potential.

Source	CLIP↑	DSim↑	TED↓	AVG↑	CR↑	AT
arXiv	0.152	0.568	0.805	0.305	84%	550
+ GitHub	0.158	0.605	0.802	0.320	88%	548
+ TeX SE	0.159	0.608	0.806	0.320	88%	569
All	0.161	0.613	0.802	0.324	89%	529

Captions vs. Descriptions

Caption: Outline of our algorithm for enumerating Williamson sequences of order n. The boxes on the left correspond to the preprocessing which encodes and decomposes the original problem into SAT instances. The boxes on the right correspond to an SMT-like setup where the system that computes the discrete Fourier transform takes on the role of the theory solver.
Description: A block diagram illustrating with several components. There are four main labeled rectangular blocks connected by arrows indicating the direction. At the bottom left, there is an input labeled n entering a rectangular block titled 'Driver script', which sends an arrow labeled 'External call' upward to a block titled 'Diophantine solver / Fourier transform'. From this block another arrow labeled 'Result' points downwards back to the 'Driver script'. From the 'Driver script' a horizontal black arrow point to the right and is labeled 'SAT instances' connected to a block titled 'Programmatic SAT solver'. It outputs a horizontal black arrow labeled 'Enumeration in order n' pointing to the right out of the diagram. Above the 'Programmatic SAT solver' is another block labeled 'Fourier transform' and connected with an upward arrow labeled 'Partial assignment' and a downward arrow labeled 'Conflict clause'. A dashed arrow labeled 'Encoding information' points from the 'Driver script' block back to the ~~'Diophantine solver / Fourier transform'~~ to the 'Fourier transform'.

Ground Truth **Caption (GPT-4o)** **Description (GPT-4o)**



Caption: A set $\sigma \in PW+$ inside a rectangle R. The blue region $\frac{R}{(\sigma \cup \partial R)}$ can always be triangulated.
Description: A blue rectangle labeled R in the top-left corner. Inside the rectangle, there are two black geometric figures. At the lower-left side, is a layered square pattern composed of three squares, a small black square at the center, surrounded by a blue square matching the background color of the rectangle, surrounded by a larger black square. Diagonally toward the upper-right is an irregular black polygon labeled σ. Inside the polygon two shapes have the ~~black~~ background color of the rectangle, one is hexagonal at the top and the other is diamond shaped at the bottom.

Ground Truth **Caption (GPT-4o)** **Description (GPT-4o)**

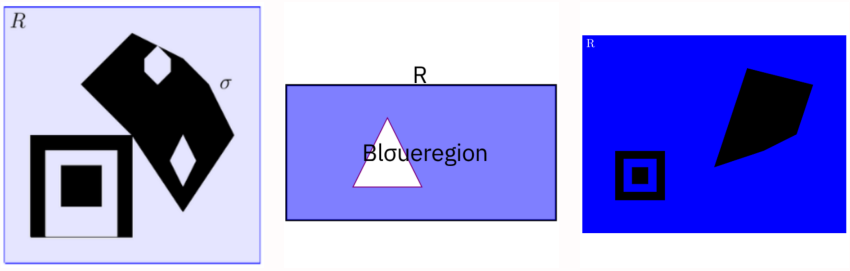
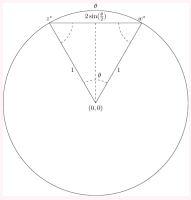


Figure 13: Captions versus VLM-generated descriptions and their produced figures by GPT-4o. ~~Striked out~~ text indicates incorrect VLM-descriptions and red text indicates improved descriptions made by human annotators.

Examples with TikZ Code

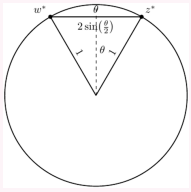
Description & GT Image:

```
A large circle centered around the point (0,0). Two points labeled $z^*$ and $w^*$ are placed on
the circle near the top. A straight black horizontal line connects both points and is labeled
$\theta$ and $2 \sin\left(\frac{\theta}{2}\right)$ and its also exactly aligned with the vertical
axes of the circle. From each point $z^*$ and $w^*$ a straight solid line extends downward to the
center of the circle. These lines form an isosceles triangle with the lines labeled 1. A vertical
dashed line, which is the vertical axes of both the triangle and circle goes from the point (0,0)
upwards to the circles circumference and is labeled $\theta$.
```



GPT-5:

```
\documentclass[tikz]{standalone}
\usepackage{amsmath}
\usetikzlibrary{calc}
\begin{document}
\begin{tikzpicture}[scale=1]
\def\\R{3}
\def\\thetadeg{60}
\coordinate (O) at (0,0);
\coordinate (Z) at ({\\R*cos(90-\\thetadeg/2)},{\\R*sin(90-\\thetadeg/2)});
\coordinate (W) at ({\\R*cos(90+\\thetadeg/2)},{\\R*sin(90+\\thetadeg/2)});
\coordinate (M) at ($(Z)!0.5!(W)$);
\draw[thick] (O) circle (\\R);
\draw[dashed] (O) -- (0,\\R) node[midway, right] {$\\theta$};
\draw[thick] (O) -- (Z) node[midway, sloped, above] {$1$};
\draw[thick] (O) -- (W) node[midway, sloped, above] {$1$};
\draw[thick] (Z) -- (W);
\node[above] at (M) {$\\theta$};
\node[below] at (M) {$2\\sin\\!\\left(\\frac{\\theta}{2}\\right)$};
\fill (Z) circle (2pt) node[above right] {$z^{*}$};
\fill (W) circle (2pt) node[above left] {$w^{*}$};
\end{tikzpicture}
\end{document}
```



TikZilla-8B-RL:

```
\documentclass[tikz]{standalone}
\usepackage{tikz}
\usepackage{pgfplots}
\usetikzlibrary{arrows.meta, decorations.pathreplacing, positioning, shapes, calc}
\begin{document}
\begin{tikzpicture}
\draw (0,0) circle (3cm);
\fill (0,0) circle (0.05cm) node[below left] {$0$};
\fill (1.5, 2.598) circle (0.05cm) node[above right] {$z^*$};
\fill (-1.5, 2.598) circle (0.05cm) node[above left] {$w^*$};
\draw[thick] (0,0) -- (1.5, 2.598) node[midway, above left] {$1$};
\draw[thick] (0,0) -- (-1.5, 2.598) node[midway, above right] {$1$};
\draw[dashed] (0,0) -- (0, 3) node[midway, right] {$\\theta$};
\node at (0.8, 3.2) {$2\\sin\\left(\\frac{\\theta}{2}\\right)$};
\end{tikzpicture}
\end{document}
```

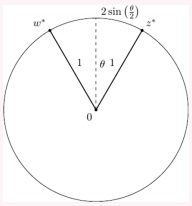
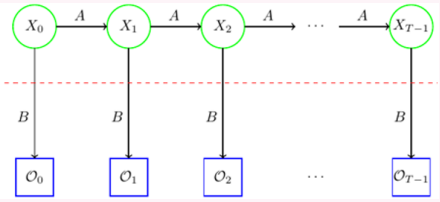


Figure 14: TikZ code and rendered figures shown for GPT-5 and TikZilla-3B-RL using the description above. The code produced by GPT-5 more accurately follows the description compared to TikZilla-3B-RL.

Examples with TikZ Code

Description & GT Image:

```
A sequence of green circles labeled $X_0$, $X_1$, $X_2$, and $X_{T-1}$ is arranged horizontally
from left to right. Each circle is connected to the next by a rightward-pointing arrow labeled
$A$. Below each circle, there is a corresponding blue square labeled $O_0$, $O_1$, $O_2$, and
$O_{T-1}$, respectively. Each circle is connected to its corresponding square by a vertical black
arrow labeled $B$. A dashed red horizontal line runs across the image, intersecting the vertical
arrows. The sequence continues with ellipses between $X_2$ and $X_{T-1}$, and after $X_{T-1}$,
indicating continuation.
```



GPT-4o:

```
\documentclass[tikz]{standalone}
\usepackage{amsmath}
\begin{document}
\begin{tikzpicture}[node distance=2cm, auto, >=stealth']
\node[circle, draw=green!50!black, fill=green!20, minimum size=10mm] (X0) {$X_0$};
\node[circle, draw=green!50!black, fill=green!20, minimum size=10mm, right of=X0] (X1) {$X_1$};
\node[circle, draw=green!50!black, fill=green!20, minimum size=10mm, right of=X1] (X2) {$X_2$};
\node[circle, draw=green!50!black, fill=green!20, minimum size=10mm, right of=X2, node distance=3
    cm] (XT1) {$X_{T-1}$};
\node[right of=X2, node distance=1.5cm] (dots) {$\\cdots$};
\node[right of=XT1, node distance=1.5cm] (dots2) {$\\cdots$};
\node[rectangle, draw=blue!50!black, fill=blue!20, minimum size=10mm, below of=X0] (O0) {$O_0$};
\node[rectangle, draw=blue!50!black, fill=blue!20, minimum size=10mm, below of=X1] (O1) {$O_1$};
\node[rectangle, draw=blue!50!black, fill=blue!20, minimum size=10mm, below of=X2] (O2) {$O_2$};
\node[rectangle, draw=blue!50!black, fill=blue!20, minimum size=10mm, below of=XT1] (OT1) {$O_{T
    -1}$};
\draw[->] (X0) -- node[midway, above] {$A$} (X1);
\draw[->] (X1) -- node[midway, above] {$A$} (X2);
\draw[->] (X2) -- node[midway, above] {$A$} (dots);
\draw[->] (dots) -- node[midway, above] {$A$} (XT1);
\draw[->, thick] (X0) -- node[right] {$B$} (O0);
\draw[->, thick] (X1) -- node[right] {$B$} (O1);
\draw[->, thick] (X2) -- node[right] {$B$} (O2);
\draw[->, thick] (XT1) -- node[right] {$B$} (OT1);
\draw[dashed, red] ($(O0)!0.5!(O1)$) -- ($(OT1)!0.5!(dots2)$);
\end{tikzpicture}
\end{document}
```

TikZilla-3B-RL:

```
\documentclass[tikz]{standalone}
\usepackage{tikz, pgfplots}
\usetikzlibrary{arrows.meta, decorations.pathreplacing, positioning, shapes, calc}
\begin{document}
\begin{tikzpicture}
\node[draw, circle, fill=green!30, minimum size=1cm] (x0) at (0, 0) {$X_0$};
\node[draw, circle, fill=green!30, minimum size=1cm] (x1) at (2.5, 0) {$X_1$};
\node[draw, circle, fill=green!30, minimum size=1cm] (x2) at (5, 0) {$X_2$};
\node at (7.5, 0) {$\\cdots$};
\node[draw, circle, fill=green!30, minimum size=1cm] (xT) at (10, 0) {$X_{T-1}$};
\node[draw, rectangle, fill=blue!30, minimum size=0.8cm] (o0) at (0, -2) {$O_0$};
\node[draw, rectangle, fill=blue!30, minimum size=0.8cm] (o1) at (2.5, -2) {$O_1$};
\node[draw, rectangle, fill=blue!30, minimum size=0.8cm] (o2) at (5, -2) {$O_2$};
\node at (7.5, -2) {$\\cdots$};
\node[draw, rectangle, fill=blue!30, minimum size=0.8cm] (oT) at (10, -2) {$O_{T-1}$};
\draw[->] (x0) -- (x1) node[midway, above] {$A$};
\draw[->] (x1) -- (x2) node[midway, above] {$A$};
\draw[->] (x2) -- (7, 0) node[midway, above] {$A$};
\draw[->] (8, 0) -- (xT) node[midway, above] {$A$};
\draw[->] (x0) -- (o0) node[midway, left] {$B$};
\draw[->] (x1) -- (o1) node[midway, left] {$B$};
\draw[->] (x2) -- (o2) node[midway, left] {$B$};
\draw[->] (xT) -- (oT) node[midway, left] {$B$};
\draw[dashed, red] (-1, -1) -- (11, -1);
\end{tikzpicture}
\end{document}
```

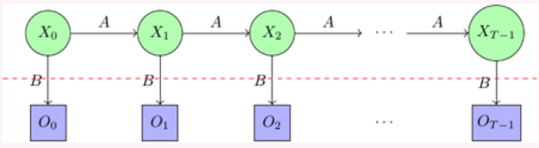
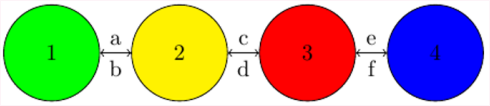


Figure 15: TikZ code and rendered figures shown for GPT-4o and TikZilla-3B-RL using the description above. The code produced by GPT-4o could not be compiled while TikZilla-3B-RL produced correct code.

Examples with TikZ Code

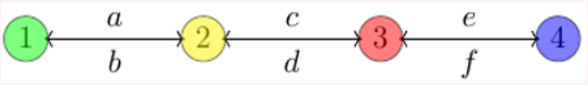
Description & GT Image:

```
A sequence of four colored circles is arranged horizontally, each connected by bidirectional
arrows. Circle 1 is green and positioned on the far left, labeled with the number "1" centered
inside. To its right, circle 2 is yellow, labeled "2" inside. A bidirectional arrow labeled
"a" above and "b" below connects circle 1 and circle 2. Next, circle 3 is red, labeled "3" inside,
connected to circle 2 by a bidirectional arrow labeled "c" above and "d" below. Finally, circle 4
is blue, labeled "4" inside, connected to circle 3 by a bidirectional arrow labeled "e" above
and "f" below. The circles are evenly spaced along the horizontal axis.
```



TikZilla-8B:

```
\documentclass[tikz{standalone}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{colortbl}
\usepackage{tikz}
\usetikzlibrary{calc,matrix}
\usetikzlibrary{decorations.pathmorphing,decorations.markings,decorations.pathreplacing,arrows,
    shapes}
\usetikzlibrary{knots}
\usetikzlibrary{arrows.meta}
\usepackage[bookmarks=true, bookmarksopen=true]{hyperref}
\begin{document}
\begin{tikzpicture}[baseline=-0.5ex]
\node (1) at (0,0) [] {$1$};
\node (2) at (2,0) [] {$2$};
\node (3) at (4,0) [] {$3$};
\node (4) at (6,0) [] {$4$};
\draw [->] (1) -- node [above] {$a$} (2);
\draw [->] (2) -- node [above] {$c$} (3);
\draw [->] (3) -- node [above] {$e$} (4);
\draw [<-] (1) -- node [below] {$b$} (2);
\draw [<-] (2) -- node [below] {$d$} (3);
\draw [<-] (3) -- node [below] {$f$} (4);
\draw [fill=green, opacity=0.5] (1) circle (0.25);
\draw [fill=yellow, opacity=0.5] (2) circle (0.25);
\draw [fill=red, opacity=0.5] (3) circle (0.25);
\draw [fill=blue, opacity=0.5] (4) circle (0.25);
\end{tikzpicture}
\end{document}
```



TikZilla-8B-RL:

```
\documentclass[tikz]{standalone}
\usepackage{tikz, amsmath, amssymb, pgfplots, pgffor}
\usetikzlibrary{automata, positioning, backgrounds, external}
\tikzset{every state/.style={minimum size=2pt}}
\begin{document}
\begin{tikzpicture}
\node[circle, draw, fill=green!30] (1) at (0,0) {$1$};
\node[circle, draw, fill=yellow!30] (2) at (2,0) {$2$};
\node[circle, draw, fill=red!30] (3) at (4,0) {$3$};
\node[circle, draw, fill=blue!30] (4) at (6,0) {$4$};
\draw[->] (1) -- node[above] {$a$} (2);
\draw[->] (2) -- node[above] {$c$} (3);
\draw[->] (3) -- node[above] {$e$} (4);
\draw[<-] (1) -- node[below] {$b$} (2);
\draw[<-] (2) -- node[below] {$d$} (3);
\draw[<-] (3) -- node[below] {$f$} (4);
\end{tikzpicture}
\end{document}
```

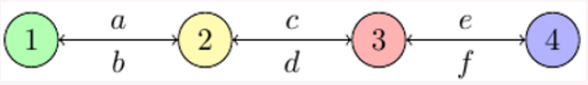
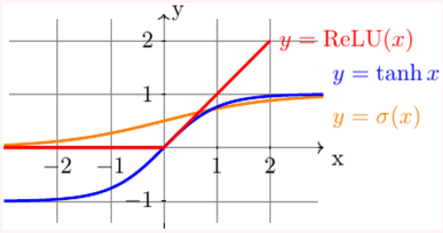


Figure 16: TikZ code and rendered figures shown for TikZilla-8B and TikZilla-8B-RL using the description above. Despite both figures being correct, the code produced by TikZilla-8B-RL is much shorter compared to TikZilla-8B.

Examples with TikZ Code

Description & GT Image:

```
A Cartesian coordinate system with x-axis labeled 'x' and y-axis labeled 'y' contains three
distinct curves. The x-axis ranges from -3 to 3, and the y-axis ranges from -2 to 2, with grid
lines at each integer. The first curve, labeled 'y = \text{ReLU}(x)' in red, is a piecewise
linear function starting at the origin, extending horizontally along the x-axis for negative
x-values, and rising linearly with a slope of 1 for positive x-values. The second curve,
labeled 'y = \tanh x' in blue, is an S-shaped curve passing through the origin, asymptotically
approaching y = 1 as x approaches positive infinity and y = -1 as x approaches negative
infinity. The third curve, labeled 'y = \sigma(x)' in orange, is another S-shaped curve, similar
to the blue curve but with a more gradual slope, also passing through the origin and
asymptotically approaching y = 1 and y = 0 for positive and negative x-values, respectively.
The labels for the curves are positioned to the right of the graph, with the red label at the
top, followed by the blue label, and the orange label at the bottom.
```



TikZilla-8B:

```
\documentclass[tikz]{standalone}
\usepackage[utf8{inputenc}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage[dvipsnames,table,xcdraw]{xcolor}
\usepackage{tikz}
\usetikzlibrary{cd, positioning, arrows.meta, backgrounds, calc, 3d}
\usepackage[many,breakable]{tcolorbox}
\begin{document}
\begin{tikzpicture}[domain=-2.75:2.75]
\draw[very thin,color=gray] (-2.9,-1.2) grid (3,2.9);
\draw[->] (-3,0) -- (3,0) node[right] {$x$};
\draw[->] (0,-2.2) -- (0,3) node[above] {$y$};
\draw[color=RedOrange,thick] plot (\\x,\\x) node[anchor=south west,text width=1.5cm] {$y=\\
    operatorname{ReLU} (x)$};
\draw[color=Blue,thick] plot (\\x,{0.5*(exp(\\x)-exp(-\\x))}) node[below right,text width=1.5cm] {
    $y=\\tanh x$};
\draw[color=Orange,thick] plot (\\x,{0.5*(1+sgn(\\x))}) node[below right,text width=1.5cm] {$y=\\
    sigma(x)$};
\end{tikzpicture}
\end{document}
```

TikZilla-8B-RL:

```
\documentclass[tikz]{standalone}
\usepackage{tikz, pgfplots, amsmath}
\usetikzlibrary{arrows.meta, decorations.pathreplacing, positioning, shapes, calc}
\begin{document}
\begin{tikzpicture}
\draw[->] (-3, 0) -- (3, 0) node[right] {$x$};
\draw[->] (0, -2) -- (0, 2) node[above] {$y$};
\draw[help lines, gray!30] (-3,-2) grid (3,2);
\draw[thick, red] (-3, 0) -- (0, 0) -- (3, 3);
\node[red, right] at (3.2, 1.5) {$y = \\text{ReLU}(x)$};
\draw[thick, blue] plot[domain=-3:3, samples=100] (\\x, {tanh(\\x)});
\node[blue, right] at (3.2, -0.5) {$y = \\tanh x$};
\draw[thick, orange] plot[domain=-3:3, samples=100] (\\x, {1/(1 + exp(-\\x))});
\node[orange, right] at (3.2, 0.5) {$y = \\sigma(x)$};
\node[below] at (-2, 0) {-2};
\node[below] at (-1, 0) {-1};
\node[below] at (1, 0) {1};
\node[below] at (2, 0) {2};
\node[left] at (0, -1) {-1};
\node[left] at (0, 1) {1};
\end{tikzpicture}
\end{document}
```

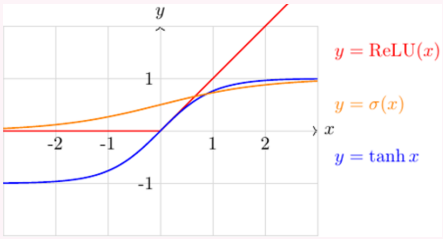


Figure 17: TikZ code and rendered figures shown for TikZilla-8B and TikZilla-8B-RL using the description above. The code produced by TikZilla-8B could not be compiled while TikZilla-8B-RL produced correct code.

Table 8: Exemplary scientific TikZ figures produced by one baseline LLM (GPT-4o) and two of our finetuned LLMs (TikZilla-3B and TikZilla-3B-RL) using the prompts from the first column which have been VLM augmented based on the Ground Truth figures in the second column. ■-boxed figures have been rated as very good, ■ as good, ■ as bad, and ■ as very bad by human annotators. Empty cells indicate non-compilable TikZ code.

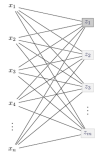
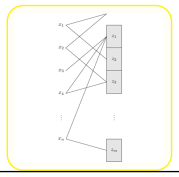
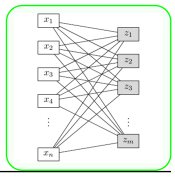
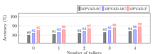
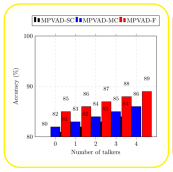
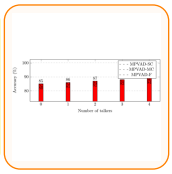
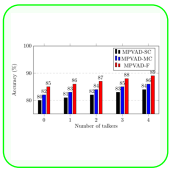
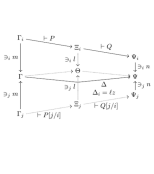
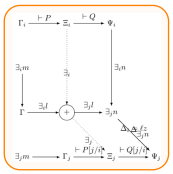
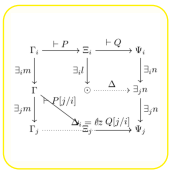
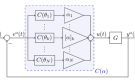
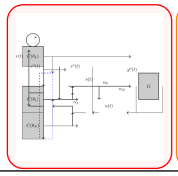
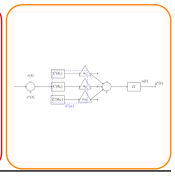
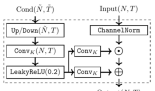
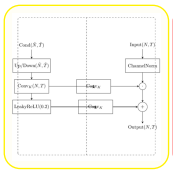
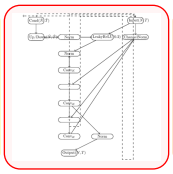
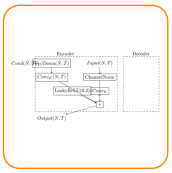
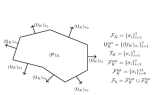
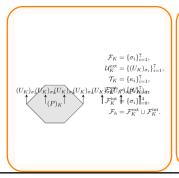
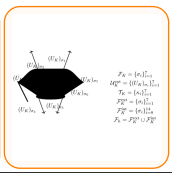
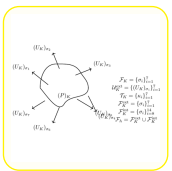
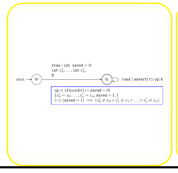
Prompt	Ground Truth	GPT-4o	TikZilla-3B	TikZilla-3B-RL
A series of black lines connect two vertical columns of elements. The left column contains labels $x˙1$, $x˙2$, $x˙3$, $x˙4$, and $x˙n$, arranged vertically from top to bottom with equal spacing. The right column contains shaded rectangles labeled $z˙1$, $z˙2$, $z˙3$, and $z˙m$, also arranged vertically from top to bottom with equal spacing. Each label in the left column is connected by straight black lines to multiple rectangles in the right column, forming a network of intersecting lines. Dotted ellipses are placed vertically between $x˙4$ and $x˙n$ and between $z˙3$ and $z˙m$, indicating continuation. The labels $x˙1$, $x˙2$, $x˙3$, $x˙4$, and $x˙n$ are positioned to the left of their respective lines, while the labels $z˙1$, $z˙2$, $z˙3$, and $z˙m$ are centered within their rectangles.				
The bar chart displays accuracy percentages on the y-axis ranging from 80“% to 100“% with increments of 10“%, labeled ”Accuracy (“%)” on the left. The x-axis is labeled ”Number of talkers” and includes five categories: 0, 1, 2, 3, and 4. Each category contains three vertical bars. The first bar is black, representing ”MPVAD-SC,” the second bar is blue, representing ”MPVAD-MC,” and the third bar is red, representing ”MPVAD-F.” Above each bar, there is a numerical label indicating the exact accuracy percentage. For category 0, the black bar is labeled 80, the blue bar 82, and the red bar 85. For category 1, the black bar is labeled 81, the blue bar 83, and the red bar 86. For category 2, the black bar is labeled 82, the blue bar 84, and the red bar 87. For category 3, the black bar is labeled 83, the blue bar 85, and the red bar 88. For category 4, the black bar is labeled 84, the blue bar 86, and the red bar 89. A legend is positioned at the top right corner of the chart, indicating the color and label for each bar type. The chart background includes horizontal dashed lines at each 10“% increment on the y-axis.				
A diagram consists of several labeled arrows and nodes arranged in a structured format. At the top left, node $“Gamma ˙i$ is connected by a rightward arrow labeled $“vdash P$ to node $“Xi ˙i$. From $“Xi ˙i$, a rightward arrow labeled $“vdash Q$ leads to node $“Psi ˙i$. Below $“Gamma ˙i$, node $“exists ˙i m$ is connected by a downward arrow to node $“Gamma $. From $“Gamma $, a rightward arrow labeled $“exists ˙i I$ leads to a central node marked with a circle containing a plus sign. This central node is connected by a rightward arrow labeled $“exists ˙j I$ to node $“exists ˙j n$. From $“exists ˙j n$, a rightward arrow labeled $“Delta $ leads to node $“Psi ˙j$. Below $“Gamma $, node $“exists ˙j m$ is connected by a downward arrow to node $“Gamma ˙j$. From $“Gamma ˙j$, a rightward arrow labeled $“vdash P[j/i]$ leads to node $“Xi ˙j$. From $“Xi ˙j$, a rightward arrow labeled $“vdash Q[j/i]$ leads to node $“Psi ˙j$. A dotted arrow labeled $“exists ˙i$ connects $“Xi ˙i$ to the central node, and another dotted arrow labeled $“exists ˙j$ connects the central node to $“Xi ˙j$. A vertical arrow labeled $“exists ˙i n$ connects $“Psi ˙i$ to $“exists ˙j n$, and a vertical arrow labeled $“exists ˙j n$ connects $“exists ˙j n$ to $“Psi ˙j$. A horizontal dotted arrow labeled $“Delta ˙i = “ell z$ connects $“exists ˙j n$ to $“Psi ˙j$.				
A control system diagram features a horizontal line starting from the left with a label $r(t)$, leading to a summation circle. The summation circle has a minus sign on the left and is labeled $e˙o(t)$ on the right. From the summation circle, a horizontal line extends rightward into a dashed blue rectangle labeled $C(“alpha)$ at the bottom right. Inside the rectangle, there are three vertically aligned blocks labeled $C(“theta ˙1)$, $C(“theta ˙k)$, and $C(“theta ˙N)$ from top to bottom. Each block has a horizontal line extending rightward to a corresponding triangular amplifier labeled $“alpha ˙1$, $[“alpha ˙]˙k$, and $“alpha ˙N$. The outputs of these amplifiers converge at a summation circle on the right side of the rectangle. From this circle, a horizontal line labeled $u(t)$ extends rightward to a block labeled G. A horizontal line continues from G to the right, labeled $y˙o(t)$. A feedback line loops from $y˙o(t)$ back to the summation circle, completing the system.				
A rectangular diagram is enclosed by a dashed border with rounded corners. Inside, there are two main vertical paths. The left path begins with a downward arrow labeled ”Cond$(˙N˙, ˙T)$” leading to a rectangle labeled ”Up/Down$(“tilde –N˙, “tilde –T˙)$”. Below, another downward arrow connects to a rectangle labeled ”Conv$ K(N, T)$”, followed by another downward arrow leading to a rectangle labeled ”LeakyReLU(0.2)”. The right path starts with a downward arrow labeled ”Input(N, T)” leading to a rectangle labeled ”ChannelNorm”. Below, a downward arrow connects to a circle with a dot inside, representing a multiplication operation. The left path has a rightward arrow from ”Conv$ K(N, T)$” connecting to a rectangle labeled ”Conv$˙K$”. This rectangle has a rightward arrow leading to the multiplication circle on the right path. Below the multiplication circle, a downward arrow leads to a circle with a plus inside, representing an addition operation. The left path continues with a rightward arrow from ”LeakyReLU(0.2)” connecting to another rectangle labeled ”Conv$˙K$”. This rectangle has a rightward arrow leading to the addition circle on the right path. Below the addition circle, a downward arrow leads to a label ”Output(N, T)”. The entire diagram is divided into two sections by a vertical dashed line, with the left section containing the ”Cond$(“tilde –N˙, “tilde –T˙)$” path and the right section containing the ”Input(N, T)” path.				
A black irregular polygon labeled $(P)˙K$ is centered in the image. Seven black arrows labeled $(U˙K)–“sigma ˙1“$ through $(U˙K)–“sigma ˙7“$ point outward from each vertex of the polygon, with labels positioned near the arrowheads. To the right of the polygon, a set of equations is displayed in black text. The equations are vertically aligned and read as follows: $“mathcal –F“˙K = “–“sigma ˙i“”–i=1“˙7$, $“mathcal –U“˙K = “–“text –ext“”K = “–(U˙K)’–“sigma ˙i“”–i=1“˙7$, $“mathcal –T“˙K = “–“kappa ˙i“”–i=1“˙7$, $“mathcal –F“”–“text –ext“˙K = “–“sigma ˙i“”–i=1“˙7$, $“mathcal –F“”–“text –int“˙K = “–“sigma ˙i“”–i=8“˙14“$, and $“mathcal –F“˙h = “mathcal –F“”–“text –ext“˙K “cup “mathcal –F“”–“text –int“˙K$. The text is right-aligned and positioned to the right of the polygon.				
State diagram with two circles labeled $q˙0$ and $q˙2$. Circle $q˙0$ is on the left, connected to circle $q˙2$ on the right by a horizontal arrow labeled ”true — int saved = 0; int x˙0, “ldots , int x˙n; — B” with the label ”init” above the arrow. Circle $q˙2$ has a loop arrow on its right side labeled ”cond — assert(“pi); op; — A” with the label ”loop˙head” above the loop. Below the diagram, a blue rectangular box contains two lines of text. The first line reads ”op “equiv ”text –if”(”text –nonlet”() ”wedge ”text –saved˙” = 0)”– x˙0 = x˙0; “ldots ; x˙n = x˙n; ”text –saved˙” = 1;”” and the second line reads ””“pi “equiv (”text –saved˙” = 1) ”implies (x˙0 “neq x˙0 “lor x˙1 “neq x˙1 “lor “cdots “lor x˙n “neq x˙n)”.				

Table 9: Exemplary scientific TikZ figures produced by one baseline LLM (GPT-5) and two of our finetuned LLMs (TikZilla-8B, and TikZilla-8B-RL) using the prompts from the first column which have been VLM augmented based on the Ground Truth figures in the second column. ■-boxed figures have been rated as very good, ■ as good, ■ as bad, and ■ as very bad by human annotators. Empty cells indicate non-compilable TikZ code.

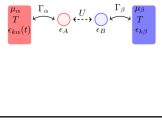
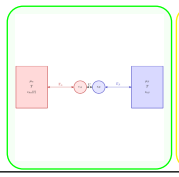
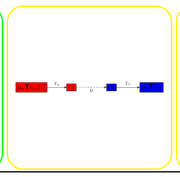
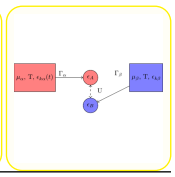
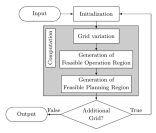
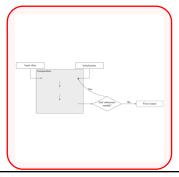
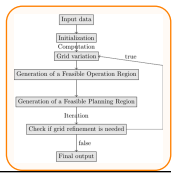
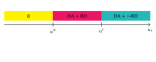
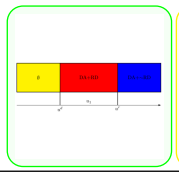
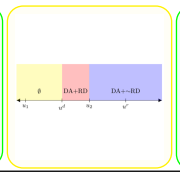
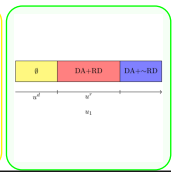
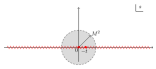
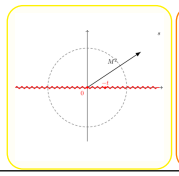
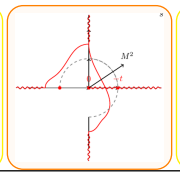
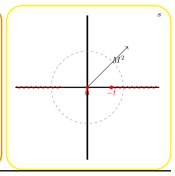
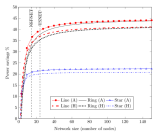
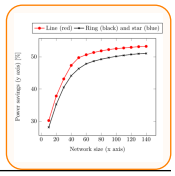
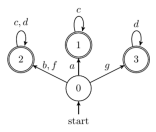
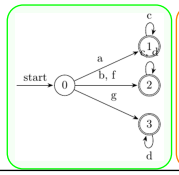
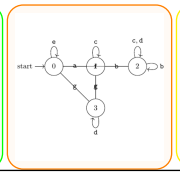
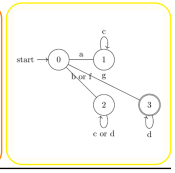
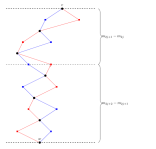
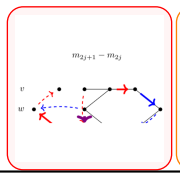
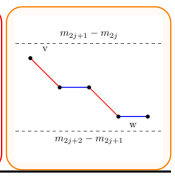
Prompt	Ground Truth	GPT-5	TikZilla-8B	TikZilla-8B-RL
A red rectangle on the left labeled with $“”mu¨””alpha$ at the top, T in the middle, and $“”epsilon¨–k””alpha¨(t)$ at the bottom. A blue rectangle on the right labeled with $“”mu¨””beta$ at the top, T in the middle, and $“”epsilon¨–k””beta¨$ at the bottom. Between the rectangles, a red circle labeled $“”epsilon¨A$ is on the left, and a blue circle labeled $“”epsilon¨B$ is on the right. A black arrow labeled $“”Gamma¨””alpha$ points from the red rectangle to the red circle, and another black arrow labeled $“”Gamma¨””beta$ points from the blue circle to the blue rectangle. A dashed black line labeled U connects the red circle to the blue circle, with arrows pointing in both directions.				
A flowchart with a series of connected shapes. At the top left, an oval labeled “”Input”” connects with a arrow to a rectangle labeled “”Initialization.”” This rectangle connects with a downward arrow to another rectangle labeled “”Grid variation,”” which is inside a larger gray rectangle. The gray rectangle is labeled “”Computation”” on the left side. Below “”Grid variation,”” a downward arrow leads to a rectangle labeled “”Generation of Feasible Operation Region,”” followed by another downward arrow leading to a rectangle labeled “”Generation of Feasible Planning Region.”” A downward arrow from this rectangle points to a diamond labeled “”Additional Grid?”” with two arrows branching from it. The leftward arrow labeled “”False”” leads to an oval labeled “”Output.”” The rightward arrow labeled “”True”” loops back to the top of the gray rectangle, connecting to the rectangle labeled “”Grid variation.””.				
A horizontal black arrow extends from left to right, labeled $u¨i$ at the tip. Above the arrow, three adjacent colored rectangles are aligned horizontally. The first rectangle on the left is yellow, labeled $“”emptyset$ in black at its center. The second rectangle is magenta, labeled DA + RD in black at its center. The third rectangle is cyan, labeled DA + $“”sim$RD in black at its center. Below the arrow, three vertical black tick marks intersect the arrow. The first tick mark is labeled $u¨d$ directly below the yellow rectangle, the second tick mark is labeled $u¨r$ directly below the magenta rectangle, and the third tick mark is at the base of the arrowhead.				
A Cartesian coordinate system with a horizontal red zigzag line along the x-axis and a vertical black arrow along the y-axis. The origin is marked with a black dot labeled 0. A gray shaded circle with a dashed outline is centered at the origin, intersecting the x-axis. A red dot labeled t is placed on the x-axis to the right of the origin, inside the circle. A black line extends from the origin to the red dot, forming an angle with the x-axis. This line is labeled $M¨2$ near the red dot. The label s is positioned in the top right corner of the image, outside the coordinate system.				
A line chart with the x-axis labeled “”Network size”” ranging from 0 to 140 in increments of 20 and the y-axis labeled “”Power savings “”%”” ranging from 0 to 50 in increments of 10. The chart contains six lines: a solid red line with circular markers labeled “”Line (A)”” and a solid black line labeled “”Ring (A)”” both starting at the origin and curving upwards, a solid blue line with triangular markers labeled “”Star (A)”” starting at the origin and remaining mostly horizontal around 10“”%, a dashed red line labeled “”Line (H)”” and a dotted black line labeled “”Ring (H)”” both following a similar upward curve to their (A) counterparts, and a dashed blue line labeled “”Star (H)”” remaining mostly horizontal around 10“”%. Vertical dashed lines are drawn at x=14 labeled “”NSFNET”” and x=24 labeled “”USNET””. The legend is placed inside a white box with a black border at the bottom right corner of the chart.				
An automaton with four states labeled as 0, 1, 2, and 3. The initial state is 0, as indicated by the incoming arrow labeled ‘start’. From state 0, the automaton transition to state 1 on input a, to state 2 on input b or f and to state 3 on input g. State 1 has a self-loop on input c, state 2 loops back to itself on inputs c or d and state 3 has a self loop on input d. States 1, 2 and 3 are indicated by double circles around it.				
A zigzag pattern composed of alternating red and blue lines connects a series of black dots and red squares vertically. The pattern starts at the top with a black dot connected to a red square by a blue line, followed by a red line connecting the red square to the next black dot. This alternating pattern continues downwards, with each black dot connected to a red square by a blue line and each red square connected to the next black dot by a red line. The sequence ends at the bottom with a black dot labeled w. Two dashed horizontal lines are placed above the topmost black dot and below the bottommost black dot, with another dashed line in the middle. Curly braces on the right side of the pattern span the sections between the dashed lines, with the top brace labeled $(m¨–2j+2¨ - m¨–2j+1¨)$ and the bottom brace labeled $(m¨–2j+1¨ - m¨–2j¨)$.				

Table 10: Exemplary scientific TikZ figures produced by one baseline LLM (GPT-4o) and two of our finetuned LLMs (TikZilla-8B and TikZilla-8B-RL) using the prompts from the first column which have been VLM augmented based on the Ground Truth figures in the second column. -boxed figures have been rated as very good, as good, as bad, and as very bad by human annotators. Empty cells indicate non-compilable TikZ code.

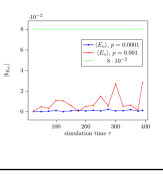
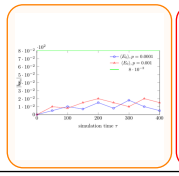
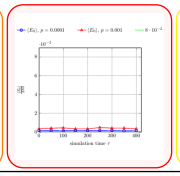
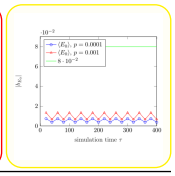
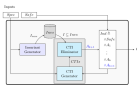
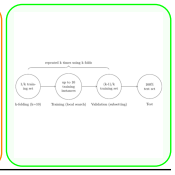
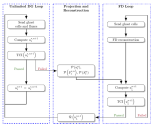
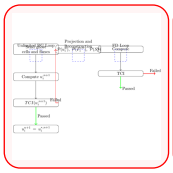
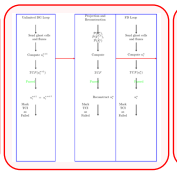
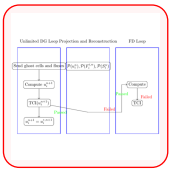
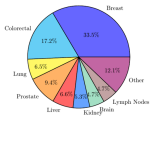
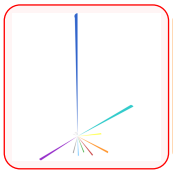
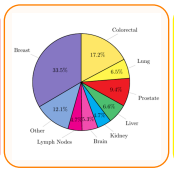
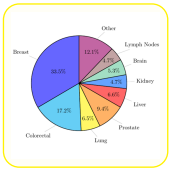
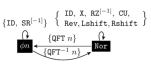
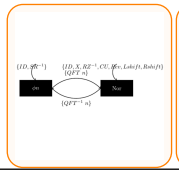
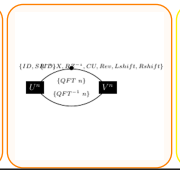
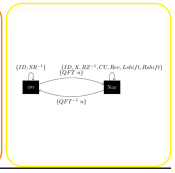
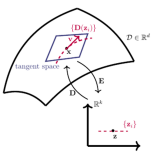
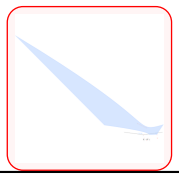
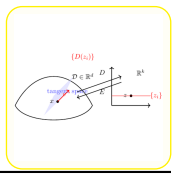
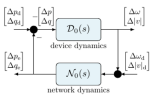
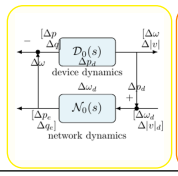
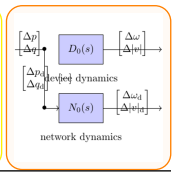
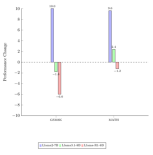
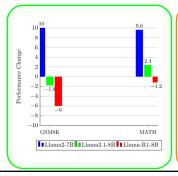
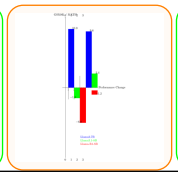
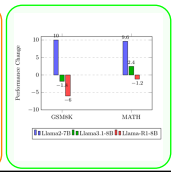
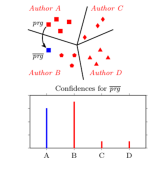
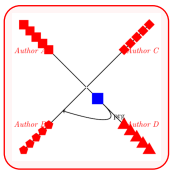
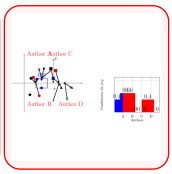
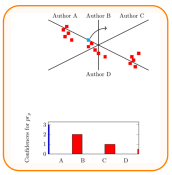
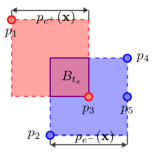
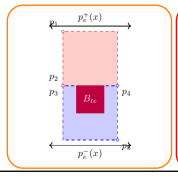
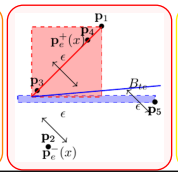
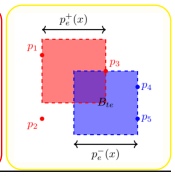
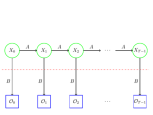
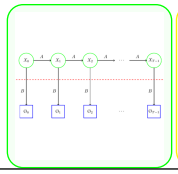
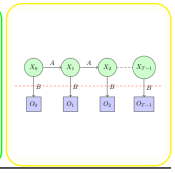
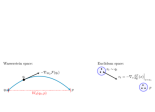
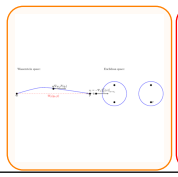
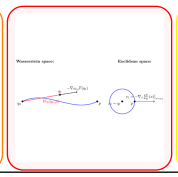
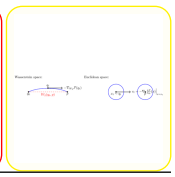
Prompt	Ground Truth	GPT-4o	TikZilla-8B	TikZilla-8B-RL
The image is a line chart with the x-axis labeled "simulation time $"tau $" ranging from 0 to 400, marked at intervals of 100. The y-axis is labeled $—b˙–E˙0"—$ and ranges from 0 to 8, with a scale factor of $10"—2"$ indicated at the top left. The chart contains two lines and a horizontal reference line. The first line is blue with circular markers, representing $"langle E˙0 "rangle , p = 0.0001$, and it fluctuates between 0 and approximately 2. The second line is red with triangular markers, representing $"langle E˙0 "rangle , p = 0.001$, and it also fluctuates between 0 and approximately 2, with more pronounced peaks. A green horizontal line is drawn at $y = 8$, representing the value $8 "cdot 10"—2"$. The legend is located inside the chart at the top right, containing three entries: a blue line with circular markers labeled $"langle E˙0 "rangle , p = 0.0001$, a red line with triangular markers labeled $"langle E˙0 "rangle , p = 0.001$, and a green line labeled $8 "cdot 10"—2"$.				
A block diagram for a verification workflow. It takes two inputs, labeled 'Spec' and 'Safe', which enter the system from the top-left. Inside the main box, the process begins with a purple block labeled 'Invariant Generator', which receives the Spec input and produces$I–new"$. This output is stored in a cylinder labeled 'Invs'. From Invs, a set of invariants I $"subseteq $ Invs is passed to the next component, the 'CTI Eliminator' shown as a blue rectangle. Directly below is another blue rectangle labeled 'CTI Generator', which also receives the Spec input and outputs 'CTIs' for the CTI Eliminator. Both blue rectangles are inside a big gray rectangle. On the right side of the diagram is a white rectangle labeled $"text –Ind" "triangleq "bigwedge "text –Safe" "wedge A˙1 "wedge "cdots "wedge A˙k "wedge A˙–k+1"$. It receives two inputs: Safe and $A˙–k+1"$, the latter coming from the CTI Eliminator. An arrow points to the CTI Generator and another arrow exits this block to the right, labeled 'Output'.				
A workflow for cross-validation using k-folds. It consists of four circular stages that are connected by arrows and the entire process is repeated k times. The first circle is labeled '1/k training set' and annotated beneath as 'k-folding (k=10)'. An arrow leads to the second circle, which is labeled 'up to 10 training instances' and annotated beneath as 'Training '(local search)'. The process continues to a third circle labeled '(k-1)/k training set' and annotated 'Validation (subsetting)' beneath. From there, a final arrow lead to a circle labeled '100"% test set' and annotated 'Test' beneath. A horizontal bracket across the top first three circles notes that its 'repeated k times using k folds'.				
A flowchart divided into three vertical sections, each outlined with a blue dashed border. These sections are labeled: 'Unlimited DG Loop' on the left, 'Projection and Reconstruction' in the center and 'FD Loop' on the right. Each section contains a sequence of boxes connected by arrows that indicate the computational flow. In the Unlimited DG Loop, the flow begins at the top with a rounded rectangle labeled 'Send ghost cells and fluxes'. A downward arrow leads to a rectangle labeled 'Compute $u˙i"–,n+1"$'. Another downward arrow connects to a block labeled '$TCI(u˙i"–,n+1"$'. This splits into two branches: an arrow labeled 'Passed' in green continues downward to a rectangle that is labeled $u˙i"–n+1" = u˙i"–*,n+1"$, while an arrow labeled 'Failed' in red exits to the right, connecting to the middle section. In the Projection and Reconstruction section, the incoming red arrow leads to a rectangle labeled $"mathcal –P"(u˙i"n),“ “mathcal –P"(F˙i"–i,n"),“ “mathcal –P"(S˙i"n)$. A arrow to the right connects it to the FD Loop section with a rectangle labeled 'Compute'. This rectangle connects to another rectangle labeled 'TCI', which then splits into two branches labeled 'Passed' in green and 'Failed' in red.				
A circular pie chart is divided into eight colored segments with distinct labels and percentages. Starting from the top and moving clockwise, the first segment is labeled "Breast" with a percentage of 33.5"% and is colored blue. The second segment is labeled "Other" with a percentage of 12.1"% and is colored purple. The third segment is labeled "Lymph Nodes" with a percentage of 4.7"% and is colored gray. The fourth segment is labeled "Brain" with a percentage of 5.3"% and is colored light blue. The fifth segment is labeled "Kidney" with a percentage of 4.7"% and is colored green. The sixth segment is labeled "Liver" with a percentage of 6.6"% and is colored red. The seventh segment is labeled "Prostate" with a percentage of 9.4"% and is colored orange. The eighth segment is labeled "Lung" with a percentage of 6.5"% and is colored yellow. The ninth segment is labeled "Colorectal" with a percentage of 17.2"% and is colored cyan. Each label is placed outside the corresponding segment, with lines connecting the labels to the segments.				
Two black rectangles are positioned horizontally in the center of the image. The left rectangle contains the label $"phi n$ in white, while the right rectangle contains the label Nor in white. Above the left rectangle, there is a curved arrow pointing downwards, labeled $"–ID, SR˙—1"˙"$. Above the right rectangle, there is a similar curved arrow pointing downwards, labeled $"–ID, X, RZ˙—1˙, CU, Rev, Lshift, Rshift"˙"$. A curved arrow connects the left rectangle to the right rectangle, labeled $"–QFT"˙ n"˙"$, and another curved arrow connects the right rectangle back to the left rectangle, labeled $"–QFT"˙—1"˙ n"˙"$.				
A large black curved shape occupies the top left, resembling a section of a circle, with a blue parallelogram labeled "tangent space" inside it. The parallelogram is oriented diagonally, with a red dashed arrow labeled "v" pointing from the bottom left to the top right, ending at a point labeled "x". Above the parallelogram, the red text "– D(z˙i)"˙" is positioned. Two black arrows labeled "D" and "E" point downward from the curved shape to a smaller coordinate system at the bottom right. This coordinate system has two black axes, with the vertical axis labeled "˙"mathbb –R"˙k" and the horizontal axis extending to the right. A red dashed horizontal line labeled "–z˙i"˙" extends from a black dot labeled "z" on the horizontal axis. The entire diagram is annotated with "˙"mathcal –D" "in "mathbb –R"˙d" in black text near the top right of the curved shape.				

Table 11: Exemplary scientific TikZ figures produced by one baseline LLM (GPT-4o) and two of our finetuned LLMs (TikZilla-8B, and TikZilla-8B-RL) using the prompts from the first column which have been VLM augmented based on the Ground Truth figures in the second column. -boxed figures have been rated as very good, as good, as bad, and as very bad by human annotators. Empty cells indicate non-compilable TikZ code.

Prompt	Ground Truth	GPT-4o	TikZilla-8B	TikZilla-8B-RL
A block diagram features two light blue rectangular blocks labeled $D˙0(s)$ and $N˙0(s)$, positioned vertically with $D˙0(s)$ on top and $N˙0(s)$ below. The block $D˙0(s)$ is labeled "device dynamics" beneath it, while $N˙0(s)$ is labeled "network dynamics" below. To the left of $D˙0(s)$, a vertical bracket labeled $["Delta p˙d ", "Delta q˙d]$ points to a black dot, which connects horizontally to $D˙0(s)$ with a line labeled $[-]$. Above this line, another vertical bracket labeled $["Delta p ", "Delta q]$ points downward. To the right of $D˙0(s)$, a horizontal arrow labeled $["Delta "omega ", "Delta —v—]$ points rightward. Below $D˙0(s)$, a horizontal line connects to a black dot to the left of $N˙0(s)$. From this dot, a vertical bracket labeled $["Delta p˙e ", "Delta q˙e]$ points leftward. The block $N˙0(s)$ has a horizontal arrow extending from its right side, labeled $["Delta "omega ˙d ", "Delta —v—˙d]$, pointing rightward.				
The bar chart contains two groups of vertical bars labeled "GSM8K" and "MATH" on the x-axis, with the y-axis labeled "Performance Change" ranging from -10 to 10 in increments of 2. Each group contains three bars. In the "GSM8K" group, the first bar is blue, extending from 0 to 10, labeled "10.0" at the top. The second bar is green, extending from 0 to -1.8, labeled "-1.8" at the bottom. The third bar is red, extending from 0 to -6.0, labeled "-6.0" at the bottom. In the "MATH" group, the first bar is blue, extending from 0 to 9.6, labeled "9.6" at the top. The second bar is green, extending from 0 to 2.4, labeled "2.4" at the top. The third bar is red, extending from 0 to -1.2, labeled "-1.2" at the bottom. A dashed horizontal line crosses the y-axis at 0. Below the chart, a legend identifies the colors: blue for "Llama2-7B," green for "Llama3.1-8B," and red for "Llama-R1-8B."				
In the upper section, there is a central point from which four black lines radiate outward, dividing the space into four quadrants. Each quadrant is labeled in red with italicized text: "Author A" in the top left, "Author C" in the top right, "Author B" in the bottom left, and "Author D" in the bottom right. In the "Author A" quadrant, there are five red squares. In the "Author C" quadrant, there are five red diamonds. In the "Author B" quadrant, there are five red pentagons. In the "Author D" quadrant, there are five red triangles. A blue square is located near the intersection of the lines, with a black curved arrow pointing from the blue square to the "Author B" quadrant, labeled "prg" in black. Below this, a horizontal bar chart is present. The x-axis is labeled with "A", "B", "C", and "D" corresponding to the authors, and the y-axis is labeled "c" on the left side. The chart title "Confidences for prg" is centered above the bars. The bar for "A" is blue and tall, the bar for "B" is red and equally tall, while the bars for "C" and "D" are red and significantly shorter.				
A red dashed square labeled $p˙1$ at the top left corner and $p˙2$ at the bottom left corner is positioned above a blue dashed square labeled $p˙3$ at the top left corner, $p˙4$ at the top right corner, and $p˙5$ at the bottom right corner. The red square is filled with a light red color, and the blue square is filled with a light blue color. The red square overlaps the blue square at the top left corner of the blue square. A smaller solid purple square labeled $B ˙-te"$ is centered at the overlapping region. The label $p˙e"+(x)$ is placed above the red square with a double-headed arrow indicating the width of the red square. The label $p˙e"-(x)$ is placed below the blue square with a double-headed arrow indicating the width of the blue square. The points $p˙1$ and $p˙3$ are marked with red circles, while points $p˙2$, $p˙4$, and $p˙5$ are marked with blue circles.				
A sequence of green circles labeled $X˙0$, $X˙1$, $X˙2$, and $X˙–T-1"$ is arranged horizontally from left to right. Each circle is connected to the next by a rightward-pointing arrow labeled A. Below each circle, there is a corresponding blue square labeled $O˙0$, $O˙1$, $O˙2$, and $O˙–T-1"$, respectively. Each circle is connected to its corresponding square by a vertical black arrow labeled B. A dashed red horizontal line runs across the image, intersecting the vertical arrows. The sequence continues with ellipses between $X˙2$ and $X˙–T-1"$, and after $X˙–T-1"$, indicating continuation.				
On the left, a blue curved line connects two black dots labeled $q˙0$ at the left end and p at the right end. Above the curve, a black dot labeled $q˙t$ is positioned slightly to the right of center. A black arrow extends from $q˙t$ pointing rightward, labeled $-"nabla ˙–W˙2"˙"mathcal –F"(q˙t)$. Below the curve, a red dashed line connects $q˙0$ and p, labeled $W˙2(q˙0, p)$ in red. The label "Wasserstein space:" is positioned above the curve. On the right, a blue circle contains two black dots, with a black dot labeled $x˙t "sim q˙t$ positioned outside and to the left of the circle. A black arrow points from $x˙t "sim q˙t$ to the circle, labeled $v˙t = -"nabla ˙x "frac –"delta "mathcal –F"˙–"delta q˙t"(x) "big — –x=x˙t"$. The label "Euclidean space:" is positioned above the circle. Another blue circle containing two black dots is positioned to the right, labeled p.				
A box plot comparing the performance of six different models based on their "Test NMAE (%)', expressed on the vertical axis, which is labeled from 1 to 3. The horizontal axis lists the model names, which are, from left to right: 'Reg-Unet', 'Reg', 'Reg-VGG', 'Residual' and 'IncDice'. Each box represents the interquartile range of NMAE values for a model, with the horizontal line inside the box indicating the median value. Whiskers extend from each box to show the range of non-outlier data and individual diamond markers indicate outliers. The Reg-Unet model has the highest NMAE values with a median close to 2.5"% and a range from just above 2.0"% to over 3.0"%. Reg shows a significantly lower median around 1.1"%, with a very small spread and an outlier below 1.0"% and one around 1.2"%. Reg-VGG shows a wider interquartile range from about 1.0"% to 1.4"% and a median close to 1.2"%. The Residual model has a small spread, similar to Reg, with a slightly higher median and includes an outlier above 1.3"%. Finally, IncDice shows a slightly wider spread, with a median near 1.4"% and one upper outlier approaching 1.6"%.