A DATASET FOR UNDERSTANDING AND RECONSTRUCTING HANDWRITTEN COMPUTER SCIENCE DIAGRAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Handwritten technical diagrams are widely used in science, engineering, and education to communicate concepts, relationships, and processes. Automatic interpretation of such diagrams has applications in the digitization of handwritten material, evaluation of student work, error identification, and conversion of handwritten diagrams into editable digital representations. However, handwritten technical diagrams remain underrepresented in existing datasets, which limits the data available for training and evaluating models for diagram understanding and generation. We present *AutoDiagram-1000-v1*, a dataset of 1,000 handwritten diagrams from automata theory paired with natural language descriptions that describe visual content without requiring automata theory terminology. Each description is manually verified with respect to the handwritten diagram and also validated by diagram reconstruction. Specifically, the description is converted into TikZ code and rendered as a digital diagram, after which the reconstruction is compared with the original handwritten diagram to verify the correspondence between the description and the source diagram. In addition to verified descriptions, the dataset includes incorrect descriptions, their resulting reconstructions, and annotations of the observed errors. This combination provides examples of both correct and incorrect mappings between handwritten diagrams and their textual representations. We further fine tune and evaluate two models to establish an initial benchmark for handwritten diagram description and generation.

1 INTRODUCTION

Diagrams are a fundamental medium for representing information in science, engineering, and education. They are commonly used to communicate relationships, processes, algorithms, state transitions, and logical structures that are often difficult to express through text alone. Recent advances in vision-language models (VLMs) and multimodal foundation models have led to substantial progress in understanding natural images, scientific figures, charts, and documents. However, the understanding of handwritten technical diagrams remains relatively underexplored.

There are unique challenges associated with handwritten diagrams. Variations in handwriting style, symbol formation, spatial arrangement, drawing quality, and scanning artifacts can introduce ambiguity that is often absent in professionally created graphics. Furthermore, technical diagrams frequently contain graph-like structures, symbolic labels, arrows, loops, and domain-specific notation whose interpretation requires accurate understanding of both visual and structural relationships. Small recognition errors, such as an incorrect transition label or a missed connection, can significantly alter the meaning of a diagram.

In the past, datasets have been introduced to support scientific figure understanding, chart captioning, and diagram analysis. However, these datasets primarily focus on digitally created figures and charts. To the best of our knowledge, there is currently no dataset that provides handwritten technical diagrams from actual classrooms paired with both natural-language descriptions, executable diagram code representations and error annotations. To address this gap, we introduce *AutoDiagram-1000-v1*, a dataset consisting of handwritten, noisy technical diagrams collected from computer science instructional and assessment materials. Unlike breadth-first diagram describing datasets that pair a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

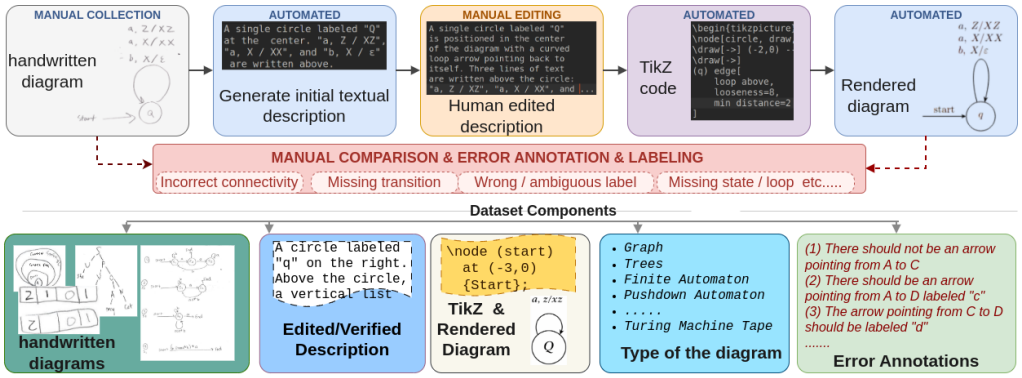


Figure 1: AutoDiagram-1000-v1 creation process and available annotations

large number of images with a single description each, AutoDiagram-1000-v1 is depth prioritized; every diagram is paired with 5 different types of human annotations including error annotations.This aligned representations could enable the study of multiple multimodal tasks, including diagram captioning, diagram-to-code generation, description-to-code generation, and diagram reconstruction. Though the raw data of 1000 images is relatively smaller, but the annotation work and its focus on an underexplored area is what makes it unique from others.

Young et al. Young et al. (2026) found that state-of-the-art vision-language models frequently make errors in symbols, labels, connectivity, and spatial relationships when interpreting handwritten technical diagrams, motivating the need for a dedicated benchmark.

We evaluated several state-of-the-art vision-language and language models on tasks supported by this novel dataset. Our evaluation highlights the gap between current model capabilities and human-level understanding of handwritten technical diagrams. The primary contributions of this work are as follows:

- We introduce *AutoDiagram-1000-v1*, a new dataset of approximately 1,000 handwritten technical diagrams paired with human-verified descriptions, executable TikZ code, rendered diagram images, type of the diagrams and error annotations.
- The dataset contains rich error annotation based on the comparison of the handwritten diagram and the rendered diagram. We further classify the errors in two distinct categories such as formatting issues and structural issues.
- We provide baseline evaluations using state-of-the-art vision-language and language models and identify common challenges in handwritten technical diagram understanding.

The dataset will be publicly released.

2 Related Work

Recent advances in vision-language models have motivated the development of datasets and benchmarks for scientific figure understanding. SciCap Hsu et al. (2021) and its extension SciCap+ Yang et al. (2024) introduced large-scale datasets for scientific figure captioning, enabling models to generate textual descriptions from figures extracted from research papers. Similarly, ChartCap Lim et al. (2025) focuses on chart understanding and caption generation, while AnaFig Yue et al. (2025) evaluates multimodal reasoning over scientific figures.

Several recent works have explored broader forms of diagram and chart understanding. Chart-HQA Chen et al. (2025), ChartCoder Zhao et al. (2025), and MatCha Lai et al. (2025) evaluate multimodal reasoning, chart interpretation, and code generation from visual inputs. DiagramEval Liang & You (2025) investigates evaluation methodologies for diagram understanding systems.

A growing body of work has investigated the generation of diagram code from textual descriptions. TikZero Belouadi et al. (2025) studies the generation of TikZ code from natural language prompts, while TikZilla Learning (2025) introduces datasets and methods for learning TikZ representations

Table 1: Comparison of AutoDiagram-1000-v1 with existing figure understanding, chart captioning, and diagram-to-code datasets. AutoDiagram-1000-v1 contains handwritten technical diagrams paired with human-reviewed descriptions, executable TikZ representations, error annotations for Tikz compiled diagrams and label specifying whether the TikZ compiled image is acceptable or not.

Dataset	Handwritten	Technical	Desc.	Code	Rend. Images	Error Anno.	Rend. Accept.	Primary Task
SciCap	✗	✓	✓	✗	✗	✗	✗	Figure Captioning
SciCap+	✗	✓	✓	✗	✗	✗	✗	Figure Captioning
ChartCap	✗	✗	✓	✗	✗	✗	✗	Chart Captioning
AnaFig	✗	✓	✓	✗	✗	✗	✗	Figure Understanding
Chart2Code	✗	✓	✗	✓	✓	✗	✗	Chart-to-Code
TikZilla	✗	✓	✗	✓	✓	✗	✗	Text-to-TikZ
Ours	✓	✓	✓	✓	✓	✓	✓	**Diagram Understanding & Code Generation**

of diagrams. Related efforts such as Chart2Code-160K Zhao et al. (2025) and ScImage Zhang et al. (2024) explore image-to-code generation pipelines for scientific and chart-based visualizations.

Recent work on scientific figure understanding spans text-to-figure generation with Pluto Srinivasan et al. (2025); and diagram and chart reasoning with Chart-HQA Chen et al. (2025), . Evaluation and large-scale resources include DiagramEval Liang & You (2025), PatentDesc-355K Shukla et al. (2025), Chart2Code-160k Zhao et al. (2025), and DaTikZ-V4 Learning (2025). Table 1 summarizes the comparison with existing datasets.

Despite these advances, existing datasets largely focus on digitally created diagrams, charts, and figures. They do not address the problem of understanding handwritten technical diagrams that contain irregular layouts, noisy handwriting, incomplete drawings, and varying notation styles. Furthermore, few datasets provide paired handwritten images, natural language descriptions, and executable diagram code representations.

3 AUTODIAGRAM-1000-V1

AutoDiagram-1000-v1 contains 1,000 handwritten technical diagrams collected from real computer science instructional and assessment materials. Each sample consists of a handwritten diagram image, a human verified natural language description, verified TikZ code, a rendered digital diagram, label of the rendered diagram as either acceptable or not acceptable via human verification, error annotations produced during the verification process and type of the diagram. The dataset contains finite state automata, graphs, trees, pushdown automata, Turing machine tapes, state elimination diagrams, and related technical diagrams commonly used in undergraduate computer science courses. Exam diagrams generally have plain white backgrounds, whereas assignments and in-class work may contain notebook backgrounds, including ruled and graph paper.

Permission: The dataset is collected from actual computer science courses under an approved IRB. The students' permission is taken to ensure that the dataset can be published. All diagrams are anonymized before inclusion in the dataset. Student names, ID numbers, and identifying marks are removed from the scanned image, and the diagram region is cropped from the surrounding page.

Protocol: At each stage of the annotation pipeline, a protocol was established through discussions between the study directors (with experience of three dataset creation) and the primary author (PhD student with experience in two dataset creation). The primary author then explained the protocol to each annotator before they began the corresponding task. The protocols were discussed through online meetings, during which annotators were provided with representative examples and shown how the annotation process should be carried out. Each annotator also shared their initial annotations with the primary author for review and feedback and proceeded with the remaining annotations after receiving approval from the primary author. Annotator feedback and observations during this process informed subsequent revisions to the protocols. Separate protocols were established for image extraction, editing LLM-generated descriptions, error annotation, and error categorization.

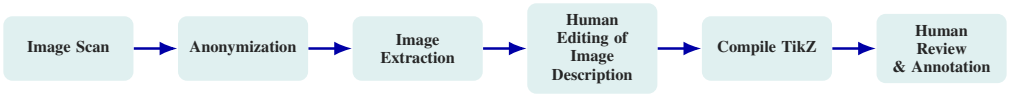


Figure 2: Dataset construction and annotation pipeline.

3.1 Data Collection and Annotation

The collection contains diagrams produced by multiple individuals with different handwriting styles and drawing conventions. Some diagrams are scanned from complete pages, whereas others are extracted as individual diagram regions. Exam diagrams generally have plain white backgrounds, whereas assignments and in-class work may contain notebook backgrounds, including ruled and graph paper. The images also exhibit variations in lighting and acquisition method, as diagrams were digitized using either professional scanners or cameras. The assignments were completed as take-home work in a relatively unconstrained setting, whereas the exams were conducted in person under timed conditions. The collection therefore contains variation in handwriting, symbol appearance, diagram layout, image resolution, and scan quality. Our dataset creation process has several stages as shown below: **1) Image Scan:** The diagrams are collected from examinations, assignments, lecture notes, instructional materials, and practice exercises from automata theory courses conducted during 2024 and 2025. The source material includes 8 examinations, 5 assignments, and 48 questions. Only questions containing significant diagram components are considered for inclusion in the dataset. The diagrams are contributed by 93.75% of the students who provided consent for the use of their diagrams in this study. **2) Anonymization:** All student materials are anonymized before inclusion in the dataset. An authorized member of the course staff who has legitimate access to student identities performs the anonymization before the materials are provided for subsequent dataset processing. Student names, identification numbers, and other identifying information are removed before individual diagram regions are extracted. **3) Image Extraction:** In the nest step, individual diagram regions are extracted from the source materials, assigned consistent filenames and question tags, and organized into the appropriate question folders.

4) Image Description and Human Review: Writing accurate descriptions for a large number of handwritten diagrams is time consuming and error prone. Descriptions written independently by multiple annotators may also differ in wording even when the same visual content is represented. For this reason, an initial reference description is generated for each handwritten diagram using a publicly available multimodal large language model (GPT4-o). Then the human annotators subsequently reviewed each initial description against the corresponding handwritten diagram and modified the description when necessary. This procedure provided a common starting point for the annotators and reduces differences caused solely by independent wording choices. The description task focused on the visual representation rather than the theoretical correctness of the automaton or domain knowledge although the initial LLM-generated descriptions were found to use domain-specific terms such as `accepting state`, `start state`, and `final state`.

5) TikZ Generation and Compilation. The human-verified descriptions are then converted into TikZ source code and compiled to produce rendered digital diagrams. Here the TikZ code could not be compiled many times directly and human intervention was need to fix the errors.

6) Human Review and Error Annotation. In this phase another round of human review was conducted. After reviewing the human-edited descriptions, it was found that some diagrams that were considered acceptable still had minor formatting or spacing issues. These issues were listed separately in error annotation. For diagrams that were found to be unacceptable, some had correct descriptions, while others had incorrect descriptions. During the error annotation phase, these errors were identified and documented.

The complete data preparation, annotation, and processing procedure required an estimated 373 hours. We have provided the time associated with each stage in Table 8 in Appendix.

Each of the 1,000 handwritten diagrams is associated with five components: (i) a human-verified natural language description, (ii) corrected compilable TikZ code together with the corresponding rendered diagram, (iii) an assessment of whether the rendered diagram is an acceptable representation of the handwritten diagram, (iv) an error annotation describing structural or formatting errors in the rendered diagram, and (v) diagram type. The 1,000 diagrams therefore correspond to 5,000

annotation fields in total. Five annotators participated in image extraction, eight annotators reviewed and edited the initial descriptions, and seven annotators review the error annotations.

3.2 Annotation Methodology

Seven different annotators of different experience level independently reviewed and edited the GPT-4o generated descriptions of subsets from the collection. One annotator additionally verified a random sample of each of the other annotator's work for consistency against a shared annotation guideline covering diagram components, arrow directions, labels, connectivity, and structural relationships. A final full-dataset verification pass is performed by one author with experience in dataset ceration process.

Each handwritten diagrams contains five aligned modalities: (1) a handwritten diagram image, (2) a human-verified natural language description, (3) TikZ source code together with the rendered digital diagram, (4) an annotation of the diagram label/type, and (5) an annotation of errors in the rendered diagram compared with the original diagram. We treat the rendered image as part of the TikZ component, since it is produced automatically by compiling the TikZ code rather than annotated independently. In this pass, the annotator reviewed the GPT-4o-generated description corresponding to the handwritten diagram, edited the description when necessary to match the diagram, and marked `needs_edit` label as True or False. In this pass, the annotator compares the diagram rendered from the corrected description's TikZ code with the original handwritten diagram, records any remaining mismatches as structural or formatting errors, and marks the rendered diagram as `Acceptable` or `Not Acceptable`.

This multimodal structure would be beneficial to the evaluation of several tasks, including diagram captioning, diagram-to-code generation, description-to-code generation, and diagram reconstruction. These error annotations can be used to help models identify structural and formatting errors and improve the correction of generated TikZ code.

Statistics: The distribution of diagram categories is presented in Table 2. The image in Fig. 3 shows the distribution of the descriptions in terms of acceptability and whether the description needs editing.

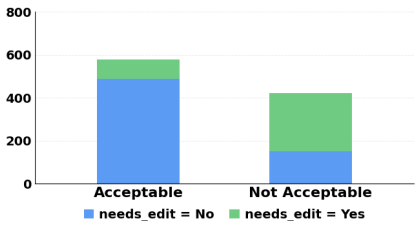


Figure 3: Distribution of diagram acceptability and required edits.

Table 2: Distribution of diagram categories in AutoDiagram-1000-v1.

Type	Count	Percentage
Finite-State Automata	307	30.67
Graphs	241	24.08
Trees	125	12.49
State-Elimination Diagrams	114	11.39
Turing Machine Tape	101	10.09
Push Down Automaton	97	9.69
Subset	16	1.60

Error Annotation We classify annotation errors into two broad categories: structural issues and formatting issues. Structural issues affect the semantic correctness of the generated diagram, including incorrect or missing nodes, edges, labels, transitions, and graph connectivity. Formatting issues preserve the underlying semantics but affect the visual presentation, such as overlapping text, misaligned labels, incorrect arrow placement, spacing inconsistencies, and other rendering artifacts. This taxonomy enables independent analysis of semantic correctness and visual quality.

In the annotation process, we distinguish between two types of errors: machine errors and human errors. A machine error occurs during the description-to-code step. A human error occurs while an annotator writes or edits the description. For example, an annotator may miss a double circle or a closing brace. The error annotations include these and other types of issues.

For example, a description may not specify whether a line is straight or curved or whether one circle is larger than another and by how much. These differences are treated as differences in annotation style rather than diagram errors. We do not consider unspecified visual details to be errors when they appear differently in the rendered diagram.

To provide a quantitative characterization of the annotation errors, we further divide structural and formatting errors into fine-grained subtypes. Table 3 reports the number of occurrences of each subtype in the error-analysis annotation. For severity, we operationalize structural errors as *major* errors because they alter the semantic or graph structure of the diagram, whereas formatting errors are treated as *minor* errors because they primarily affect visual presentation without necessarily changing the intended semantics. Instances containing both structural and formatting errors are therefore considered to contain errors at both severity levels. The table provides an overall measure of the extent to which annotated examples required an additional correction pass.

Table 3: Counts of error subtypes in the error analysis.

Error subtype	**Count**
Structural errors	
Missing arrow/loop/label	140
Extra arrow/circle/label added	103
Wrong label or text	97
Arrow points the wrong way	59
Label/circle/arrow in the wrong place	50
Wrong start/end circle	28
Tree hierarchy not preserved	16
Arrow/label overlaps a circle	15
Wrong coloring	11
Label split/merged incorrectly	5
Formatting errors	
Text overlapping/merging	151
Spacing issues	84
Extra circle/box around a label	25
Arrow shape looks wrong	24
Misplaced arrow/label	24

4 EXPERIMENTS

We measured the usefulness of the dataset by testing whether current vision language models when finetuned with this data can produce descriptions that are closer to human-reviewed descriptions.

Experiment 1: Benchmarking

For fine-tuning, we used the 640 images whose descriptions did not require further editing, comprising 490 `Acceptable` and 150 `Not Acceptable` diagrams. Importantly, some compiled TikZ diagrams were classified as Not Acceptable even though the corresponding human-verified descriptions contained no errors; these correct descriptions were therefore included in fine-tuning. We split these 640 images into training, validation, and test as shown in Table 4. Benchmarking Task: We evaluated the task of diagram description generation with our dataset. Given a handwritten technical diagram, a model generates a natural language description of the diagram. The generated description is compared with two references: the original GPT-4o generated description and the final human reviewed description. The use of both references measures whether a model is closer to the initial automatic description or to the final human reviewed description.

Experiment 2: Comparing the Human reviewed description with Other models

We evaluated three vision language models were evaluated: Qwen2.5-VL-72B, Claude 4 Sonnet, and Gemini 2.5 Pro. Each model generated descriptions of handwritten diagrams in AutoDiagram-1000-v1 are compared with two references: (a) original GPT-4o generated descriptions and (b) the final version of the description which is reviewed and edited by human. The comparison is performed using the following metrics: BLEU Papineni et al. (2002), METEOR Banerjee & Lavie (2005) , ROUGE-1 Lin (2004), ROUGE-L, and GPT-4o similarity.

Table 4: Train/validation/test split by category.

Split	Acceptable, does not need further edit	Not acceptable, does not need edited	Total
Train	440	100	540
Validation	20	20	40
Test	30	30	60

4.1 RESULTS & DISCUSSION

Fine-tuning Results: We fine-tuned two open-weight vision-language models, Qwen2.5-VL-7B-Instruct Qwen Team (2025) and PaliGemma2-3B PaliGemma Team (2024), using Low-Rank Adaptation (LoRA) Hu et al. (2021) on the 540-image training split described above, and evaluate each

model before and after fine-tuning on the 60-image held-out test split against the human-reviewed description.

Our results demonstrated that the finetuning improved the performance of both the models. For Qwen2.5-VL-7B-Instruct, median BLEU increased from 1.57 to 26.93, and median ROUGE-L increased from 19.07 to 46.03. For PaliGemma2-3B, the pretrained model's scores are close to zero, because the base checkpoint is not instruction-tuned and does not generate a description in response to the prompt without fine-tuning; after fine-tuning, its median BLEU reached 19.65 and its median ROUGE-L reaches 44.97. After fine-tuning, Qwen2.5-VL-7B-Instruct scored highest on every metric, and PaliGemma2-3B scored second on every metric.

We report the BLEU, METEOR, ROUGE-1, and ROUGE-L scores before and after LoRA fine-tuning on AutoDiagram-1000-v1 in Table 5 . Each model is tasked to generate a text description of a handwritten technical diagram, and the description is compared against the human-reviewed reference description for the same image. Scores are computed on the 60-image held-out test split and reported as medians. Higher values indicate better performance for all four metrics. The best score in each column is shown in bold, and the second-best score is underlined.

Table 5: Benchmarking results before and after fine-tuning on the evaluation set. Both models are evaluated on 60 images.

Model	**Stage**	**BLEU ↑**	**METEOR ↑**	**ROUGE-1 ↑**	**ROUGE-L ↑**
Qwen2.5-VL-7B-Instruct	Baseline	1.57	24.49	32.62	19.07
Qwen2.5-VL-7B-Instruct	Fine-tuned	**26.93**	**45.30**	**65.24**	**46.03**
PaliGemma2-3B	Baseline	0.00	0.42	1.94	1.88
PaliGemma2-3B	Fine-tuned	<u>19.65</u>	<u>40.37</u>	<u>60.61</u>	<u>44.97</u>

To evaluate handwritten diagram description generation, we compared descriptions generated by Qwen2.5-VL-72B, Claude 4 Sonnet, and Gemini 2.5 Pro against both the original GPT-4o-generated descriptions and the final human-reviewed descriptions. We report complementary lexical and semantic metrics, including BLEU, METEOR, ROUGE-1, ROUGE-L, and GPT-4o similarity. As shown in Table 6 and Figure 4, Gemini 2.5 Pro achieves the strongest overall performance across all reported metrics, with the highest median BLEU, METEOR, ROUGE-1, ROUGE-L, and GPT-4o similarity scores.

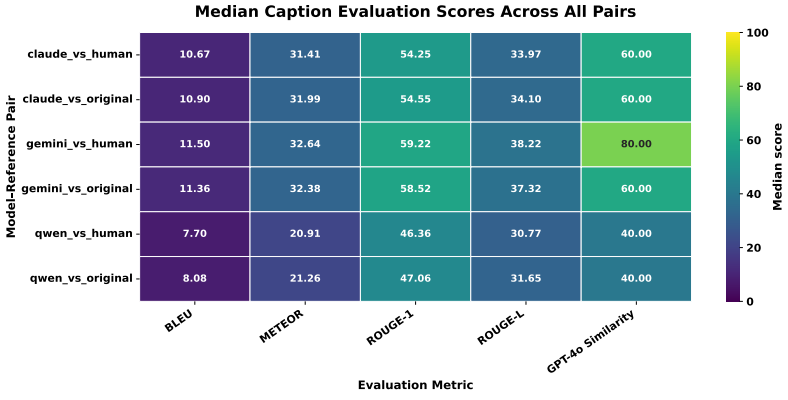


Figure 4: Median diagram description evaluation scores for all model-reference pairs. The heatmap summarizes BLEU, METEOR, ROUGE-1, ROUGE-L, and GPT-4o similarity.

Gemini is closer to the human-reviewed descriptions than to the original automatically generated descriptions, suggesting that its outputs better align with the structural and semantic corrections introduced during human verification. In contrast, Qwen2.5-VL-72B and Claude 4 Sonnet are slightly closer to the original automatically generated descriptions than to the human-reviewed references as indicated by most lexical metrics. We show the individual model performance in Figure 5.

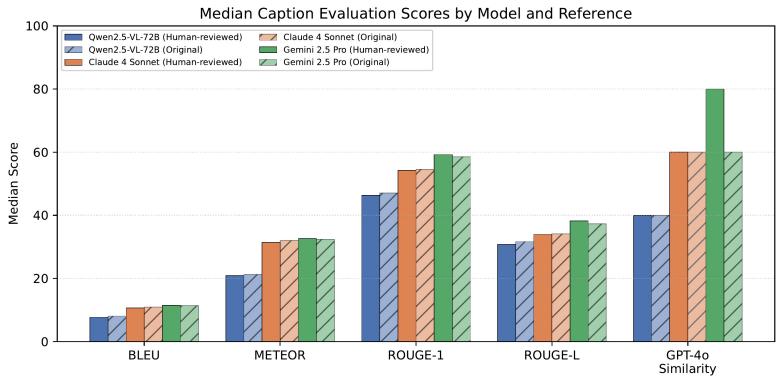


Figure 5: Median caption evaluation results for Qwen 2.5-VL-72B, Claude 4 Sonnet, and Gemini 2.5 Pro using the original descriptions and the final human-reviewed descriptions as references.

Table 6: Diagram description performance on AutoDiagram-1000-v1. Models generate textual descriptions from handwritten technical diagrams are evaluated against both the original generated descriptions and the final human-reviewed descriptions. Results report median scores from the caption evaluation heatmap. Higher values indicate better performance for all metrics. The best score in each column is shown in bold; second-best values are underlined.

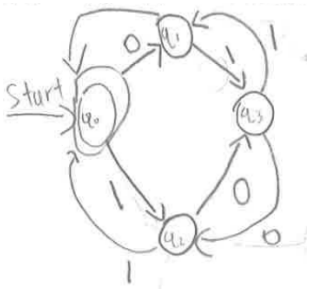
Model	**Reference**	**BLEU ↑**	**METEOR ↑**	**ROUGE-1 ↑**	**ROUGE-L ↑**	**GPT-4o Similarity ↑**
Gemini 2.5 Pro	Human-reviewed	**11.50**	**32.64**	**59.22**	**38.22**	**80.00**
Gemini 2.5 Pro	Original	<u>11.36</u>	<u>32.38</u>	<u>58.52</u>	<u>37.32</u>	<u>60.00</u>
Claude 4 Sonnet	Human-reviewed	10.67	31.41	54.25	33.97	<u>60.00</u>
Claude 4 Sonnet	Original	10.90	31.99	54.55	34.10	<u>60.00</u>
Qwen2.5-VL-72B	Human-reviewed	7.70	20.91	46.36	30.77	40.00
Qwen2.5-VL-72B	Original	8.08	21.26	47.06	31.65	40.00

The dataset also contains diagrams with different levels of visual and technical complexity. Figure 7 shows examples labeled as easy, medium, and complex. The examples differ in diagram size, number of components, connectivity, handwriting quality, and visual density.

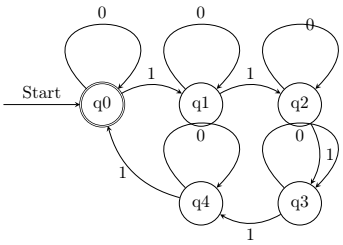
The results indicate that handwritten diagram description generation remains difficult for current vision language models. The main errors occur in handwritten symbol recognition, state labels, transition labels, and graph connectivity. These errors are important because a single incorrect label or missing transition can change the meaning of a technical diagram as we can see in the representative example in Figure 6. The GPT4-o generated description incorrectly identifies five states, while the human-reviewed description identifies four states. The model generated description also include incorrect transitions and labels. The generated description, when corrected by human, matches the handwritten diagram better.

5 Conclusion

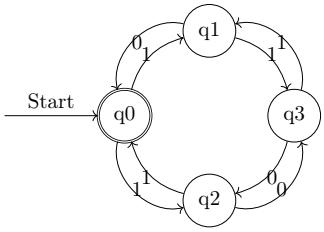
We present AutoDiagram-1000-v1, a dataset of 1,000 handwritten technical diagrams collected from computer science instructional and assessment materials. Each sample contains a handwritten diagram, a human-verified description, verified TikZ source code, a rendered digital diagram, together with a structural/formatting error annotation, an acceptable/not acceptable label for the rendered diagram, and the type of the diagram. The dataset provides a benchmark for handwritten diagram description generation. Baseline evaluation using state-of-the-art vision-language models indicates that handwritten technical diagrams remain challenging because of errors in handwritten symbol recognition, graph connectivity, and structural relationships. We expect AutoDiagram-1000-v1 to serve as a common benchmark for future research on handwritten technical diagram understanding and diagram code generation. Future releases of AutoDiagram-1000-v1 will incorporate corrections identified through continued use of the dataset and may include additional diagrams, annotation categories, and benchmark tasks.



(a) Hand-drawn diagram



(b) Raw TikZ output from LLM generated description



(c) TikZ output from human edited description

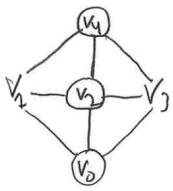
Vision Language Model Generated Description

The diagram consists of five circles labeled q0, q1, q2, q3, and q4 arranged in a clockwise circular pattern. The circle labeled q0 is at the top left and has a double outline, with an arrow labeled "Start" pointing to it from the left. A curved arrow connects q0 to q1, which is at the top center, and another curved arrow connects q1 to q2, which is at the top right. A curved arrow connects q2 to q3, located at the bottom right, and another curved arrow connects q3 to q4, which is at the bottom center. A curved arrow connects q4 back to q0. Each circle has a looped arrow pointing back to itself, and each of these loops is labeled with 0. The connections between q0 to q1, q1 to q2, q2 to q3, q3 to q4, and q4 to q0 are labeled with 1.

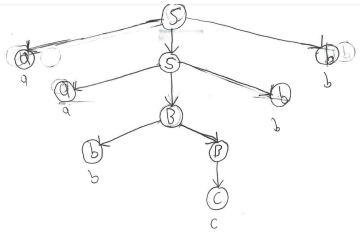
Human-Edited Description

The diagram consists of four circles labeled q0, q1, q3, and q2 arranged in a clockwise circular pattern. The circle labeled q0 is at the top left and has a double outline, with an arrow labeled "Start" pointing to it from the left. A curved arrow connects q0 to q1, which is at the top center, and another curved arrow connects q1 to q3, which is at the top right. A curved arrow connects q3 to q2, which is at the bottom center. A curved arrow connects q2 back to q0. The connections from q0 to q2, q2 to q1, q3 to q1, and q1 to q3 are labeled with 1, and the arrows from q0 to q1, q2 to q3, and q3 to q2 are labeled with 0.

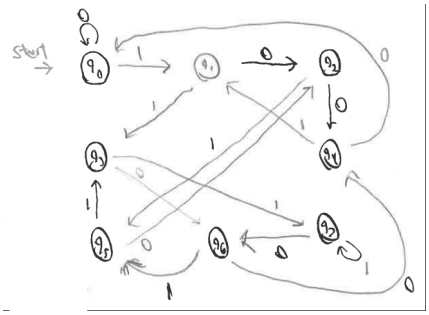
Figure 6: Example from AutoDiagram-1000-v1 showing a student-drawn diagram, raw TikZ output, edited TikZ output, a model-generated description, and a human-edited description.



(a) Easy graph example



(b) Medium parse tree example



(c) Complex finite automaton example

Figure 7: Examples from AutoDiagram-1000-v1 with different levels of diagram complexity.

AI USE DISCLAIMER

Generative AI is used to check grammar, LaTeX syntax, and sentence construction. Each suggestion is reviewed by the authors before inclusion. Generative AI is not used for study design, experimental design, or protocol design and did not influence the scientific procedures. Generative AI is also used to adapt existing fine-tuning codebases for different models. As described in the paper, the initial text descriptions of the handwritten images are generated using AI as part of the study design and annotation protocol.

REFERENCES

Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pp. 65–72, 2005.

Jonas Belouadi, Eddy Ilg, Margret Keuper, Hideki Tanaka, Masao Utiyama, Raj Dabre, Steffen Eger, and Simone Ponzetto. Tikzero: Zero-shot text-guided graphics program synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17793–17806, 2025.

Xiangnan Chen, Yuancheng Fang, Juncheng Li, Qian Xiao, Jun Lin, Siliang Tang, and Yueting Zhuang. Chart-hqa: A benchmark for hypothetical question answering in charts. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 13297–13303, 2025.

Ting-Yao Hsu, C Lee Giles, and Ting-Hao Huang. Scicap: Generating captions for scientific figures. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3258–3264, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Zhengzhao Lai, Youbin Zheng, Zhenyang Cai, Haonan Lyu, Jinpu Yang, Hongqing Liang, Yan Hu, and Benyou Wang. Can multimodal llms see materials clearly? a multimodal benchmark on materials characterization. *arXiv preprint arXiv:2509.09307*, 2025.

Reinforcement Learning. Tikzilla: Scaling text-to-tikz with high-quality data and reinforcement learning. *OpenReview*, 2025.

Chumeng Liang and Jiaxuan You. Diagrameval: Evaluating llm-generated diagrams via graphs. *arXiv preprint arXiv:2510.25761*, 2025.

Junyoung Lim, Jaewoo Ahn, and Gunhee Kim. Chartcap: Mitigating hallucination of dense chart captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13171–13182, 2025.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pp. 74–81, 2004.

PaliGemma Team. Paligemma 2: A family of versatile vlms for transfer learning. *arXiv preprint arXiv:2407.07726*, 2024.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Qwen Team. Qwen2.5-vl, 2025.

Shreya Shukla, Nakul Sharma, Manish Gupta, and Anand Mishra. Patentlmm: large multimodal model for generating descriptions for patent figures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20488–20496, 2025.

Arjun Srinivasan, Vidya Setlur, and Arvind Satyanarayan. Pluto: Authoring semantically aligned text and charts for data-driven communication. In *Proceedings of the 30th International Conference on Intelligent User Interfaces*, pp. 1123–1140, 2025.

Zhishen Yang, Raj Dabre, Hideki Tanaka, and Naoaki Okazaki. Scicap+: A knowledge augmented dataset to study the challenges of scientific figure captioning. *Journal of Natural Language Processing*, 31(3):1140–1165, 2024.

Ethan Young, Zichun Wang, Aiden Taylor, Chance William Jewell, Julian Carter Myers, Satya Sri Rajiteswari Nimmagadda, Anthony T White, Aniruddha Maiti, and Ananya Jana. Text to automata diagrams: Comparing tikz code generation with direct image synthesis. In *2026 ASEE North Central Section Conference*, 2026.

Tan Yue, Xuzhao Shi, Rui Mao, Zilong Song, Zonghai Hu, and Dongyan Zhao. Anafig: A human-aligned dataset for scientific figure analysis. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 12837–12843, 2025.

Leixin Zhang, Steffen Eger, Yinjie Cheng, Weihe Zhai, Jonas Belouadi, Christoph Leiter, Simone Paolo Ponzetto, Fahimeh Moafian, and Zhixue Zhao. Scimage: How good are multimodal large language models at scientific text-to-image generation? *arXiv preprint arXiv:2412.02368*, 2024.

Xuanle Zhao, Xianzhen Luo, Qi Shi, Chi Chen, Shuo Wang, Zhiyuan Liu, and Maosong Sun. Chartcoder: Advancing multimodal large language model for chart-to-code generation. *arXiv preprint arXiv:2501.06598*, 2025.

A Appendix

A sample dataset has been provided in this supplementary material. Annotators compared the stated components and relationships with the handwritten diagram. Assessment of the theoretical correctness of the automaton is not part of the annotation task. Each annotator spends approximately six to eight hours reviewing and modifying the assigned set of descriptions.

After human verification, each description is converted into TikZ code. The TikZ code is compiled to produce a digital diagram. The rendered diagram, TikZ code, verified description, and original handwritten diagram are then reviewed together. Inconsistencies found during this comparison are recorded as error annotations. Human errors introduced during description editing are corrected because the human-verified description serves as the reference description for subsequent processing. Errors that arise during conversion from the verified description to the rendered diagram are recorded for error analysis.

Seven human annotators participate in description verification and error annotation. Annotators who participate in the description verification phase also participate in the error annotation phase and review samples that are edited by other annotators. Before each annotation stage, a common protocol is established for image extraction, initial description review, error annotation, and final error categorization. The annotation protocol is explained to each annotator before the corresponding stage begins.

The initial error annotations are written in free text and consequently contain differences in terminology among the seven annotators. A final review is therefore conducted on the complete set of error annotations. During this review, each error annotation and its corresponding description are examined individually. Error descriptions that refer to the same type of problem are assigned to broader error categories. The reviewer responsible for this final categorization has prior experience with dataset creation and annotation. Approximately 100 hours are spent on error annotation, based on an average of approximately 6 minutes per diagram for comparison of the handwritten image, rendered diagram, and human-verified description. An additional 15 hours are spent on final error categorization.

B Annotation Details and Quality after Fine Tuning

We summarize the number of annotators involved at each stage of the annotation process in Table 7.

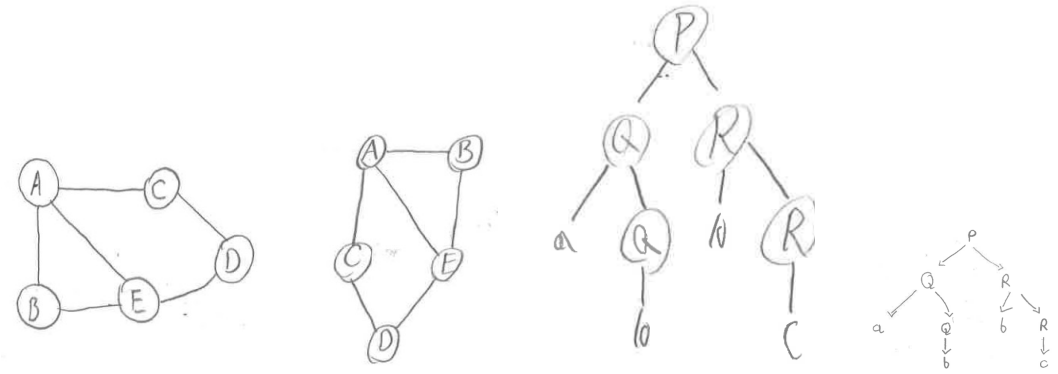
Table 7: Number of annotators involved at different stages of dataset creation.

Stage	**Annotators**
Image Extraction	5
LLM-Generated Description Editing	8
Error Annotation and Review	7
Final Error Categorization	1

We provide qualitative examples comparing the outputs of Qwen2.5-VL-7B-Instruct and PaliGemma2-3B before and after fine-tuning on AutoDiagram-1000-v1. For each example, we compare the baseline and fine-tuned model outputs with the human-reviewed description. The examples include diagrams whose rendered representations were labeled as "Acceptable and not Edited" and "Not Acceptable and not Edited."

For example `in the example image 1` as shown in 8, the handwritten diagram contains five labeled circles and six undirected connections. Before fine-tuning, Qwen2.5-VL-7B-Instruct interpreted the diagram as a directed graph or finite-state machine, introduced directed edges, and described the structure as a cycle. Its fine-tuned output correctly identified the five labeled components and described the connections as straight lines rather than directed transitions, although some connectivity differences remained. The baseline PaliGemma2-3B output consisted only of the single character "A." After fine-tuning, the model identified all five labeled circles, their approximate positions, and most of the connections.

A similar pattern is observed for `in the example image 2` as shown in 8. The baseline Qwen2.5-VL-7B-Instruct output interpreted the undirected graph using directed transitions and



(a) image 1 (b) image 2 (c) image 3 (d) image 4

Figure 8: Example images from AutoDiagram-1000-v1.

listed substantially more edges than were present in the reference description. After fine-tuning, the model produced a description containing the five labeled circles and most of the correct connections while removing the additional directed transitions. The baseline PaliGemma2-3B output was limited to "A simple graph," whereas the fine-tuned model produced a detailed description of the diagram's nodes, positions, and connections.

The tree examples further illustrate differences between the baseline and fine-tuned outputs. For `in the example image 3` as shown in 8, baseline Qwen2.5-VL-7B-Instruct incorrectly interpreted the tree as a finite-state machine and introduced node names and transition labels that were not present in the human-reviewed description. After fine-tuning, its output instead identified the input as a tree and described its hierarchical structure using the visible node labels. Similarly, baseline PaliGemma2-3B generated only the character "Q," while the fine-tuned model generated a substantially more complete description of the tree structure.

For `example image 4` as shown in 8, baseline Qwen2.5-VL-7B-Instruct misidentified the handwritten "P" as the Greek letter ρ and interpreted the tree as a finite automaton with transition labels. After fine-tuning, the model correctly described the input as a tree, identified the node labels, and represented the hierarchical branching structure. PaliGemma2-3B also changed from a single-character baseline output to a substantially more detailed description after fine-tuning, although its fine-tuned description still contained omissions and repeated information.

B.1 Summary of Qualitative Observations

Across these examples, the baseline outputs frequently exhibit incorrect assumptions about diagram type, directedness, node labels, and connectivity. In particular, Qwen2.5-VL-7B-Instruct sometimes interprets graphs and trees as finite-state machines and consequently introduces directed transitions or transition labels that are not present in the diagram. The baseline PaliGemma2-3B outputs in these examples contain little descriptive information. After fine-tuning, both models produce descriptions that more closely reflect the diagram type, visible labels, spatial organization, and connectivity represented in the human-reviewed descriptions. However, the fine-tuned outputs are not consistently error-free and may still omit connections, misidentify individual nodes, or repeat information.

Table 9 summarizes the dataset statistics. AutoDiagram-1000-v1 contains 1,000 handwritten diagrams, 1,000 human verified descriptions, 1,000 TikZ files, 1,000 rendered digital diagrams, 1,000 diagran labels/type and 1,000 error annotations.

The study protocol is submitted to the Institutional Review Board and is determined to be exempt from further review under the applicable human subject research requirements. Students receive written information describing the data collection and anonymization procedures and are asked to provide their consent. Only diagrams from students who provide consent are included in the dataset. Of the 48 students, 45 provide consent. Consent is requested after the semester results are released. Students are given 15 days after providing consent to withdraw their data. Students are informed that the diagrams are used for research and that only the diagram regions are extracted from their submitted work. Students are also informed that participation has no relation to their course performance

Table 8: Estimated time spent on data preparation, annotation, and processing.

Task	Description	Time (hours)
Image scanning	Scanning student assignments and examinations	8
Image anonymization and source organization	Anonymizing student materials; downloading individual images and camera photographs; sorting student submissions; matching submissions with anonymous student codes; organizing files with inconsistent naming conventions; and extracting individual examination files from combined multi-student scans	60
Image extraction, naming, and question tagging	Extracting individual diagram regions, assigning consistent filenames and question tags, and sorting diagrams into the appropriate question folders	30
Round 1 TikZ Error Fixing and compilation	Initial compilation and validation of generated TikZ figures, including identification and correction of compilation errors	30
Image description	Human review and modification of GPT-4o generated descriptions, requiring approximately 8 minutes per image	130
Error annotation	Comparing the handwritten image, rendered diagram, and human-verified description and recording observed errors, requiring approximately 6 minutes per image	100
Final error categorization	Reviewing free-text error annotations and assigning common error categories	15
Total		**373**

Table 9: Summary statistics of AutoDiagram-1000-v1.

Statistic	Value
Number of diagrams	1,000
Number of diagram descriptions	1,000
Number of TikZ files	1,000
Number of rendered diagrams	1,000
Error Annotations	1,000
Diagram Labels	1,000

or standing in the department. Contact information is provided for questions concerning the study. Consent materials are then sent by email so that students have sufficient time to review the information before making a decision. For the 2024 course offering, consent is obtained retrospectively by email. For the 2025 course offering, written consent is obtained directly.

The preparation of the source material requires multiple stages of manual processing. Approximately 8 hours are spent scanning student assignments and examinations. Approximately 60 hours are spent anonymizing pages and organizing the source material. This work includes downloading submitted images and photographs, organizing student submissions, assigning anonymous student codes, standardizing file names, and separating individual examination files from scanned examination records. Approximately 30 additional hours are spent extracting individual diagram regions, assigning consistent filenames and question tags, and sorting the extracted diagrams into the appropriate question folders.