

Table 8: Performance on SVG understanding dimension across different model types and difficulty levels. Accuracy scores are shown for Perceptual QA and Semantic QA tasks, with models marked as reasoning (★), code (★), open-source (★), or proprietary (★) variants.

Method	Perceptual QA(ACC↑)(%)			Semantic QA(ACC↑)(%)		
	Easy	Medium	Hard	Easy	Medium	Hard
DeepSeek-R1-Distill-Qwen-1.5B [12]★	43.01	26.92	17.78	31.18	20.51	26.67
DeepSeek-R1-Distill-Qwen-7B [12]★	53.09	32.89	35.56	34.57	39.47	28.89
DS-R1-Qwen-32B [12]★	64.20	43.42	42.22	51.85	52.63	37.78
DeepSeek-R1 [12]★	74.19	55.13	44.44	74.19	71.79	55.56
Llama3.2-3B-Instruct [11]★	47.31	26.92	26.67	44.09	43.59	46.67
Mistral-Small-3.1-24B-Instruct [21]★	62.37	34.62	28.89	54.84	62.82	44.44
Qwen2.5-Coder-3B [15]★	24.73	29.49	20.00	26.88	20.51	15.56
Qwen2.5-3B-Instruct [34]★	43.01	37.18	26.67	46.24	46.15	46.67
Qwen2.5-72B-Ins [34]★	67.74	38.46	24.44	50.54	47.43	51.11
QwQ-32B [59]★	62.37	34.62	33.33	53.76	57.69	37.78
Qwen3-1.7B [59]★	46.91	22.37	28.89	41.98	44.74	22.22
Qwen3-8B [59]★	70.96	43.59	22.22	44.09	46.15	42.22
Qwen3-32B [59]★	71.60	42.11	24.44	60.49	55.26	42.22
Gemini-2.0-Flash [43]★	77.78	40.79	31.11	62.96	55.26	51.11
GPT-4o [16]★	82.72	35.53	42.22	67.90	56.58	64.44
Claude-3.7-Sonnet [1]★	80.25	47.37	33.33	77.78	65.79	71.11

SSIM [51], or FID [45], which rely primarily on pixel-level comparisons, our approach leverages the structural properties of SVG, an XML-based vector graphics format, to capture both visual fidelity and underlying XML code consistency. SVG representations are not merely visual outputs; they are composed of structured commands, attribute sets, and rendering logic that encode information often missed by purely perceptual metrics.

Our metric incorporates both macroscopic visual alignment and microscopic structural correspondence. At the macroscopic level, we compute a global Intersection-over-Union (IoU) score to quantify the overlap of visual shapes. At the microscopic level, we perform a comprehensive analysis of SVG code. This begins with preprocessing steps such as center alignment and scale normalization, followed by path extraction and the construction of a multi-dimensional similarity matrix. The matrix integrates several components, including path-wise IoU, similarity to color attribute, and spatial relationship metrics. We employ the Hungarian algorithm to perform optimal path-level matching, ensuring robust structural alignment. Beyond direct path matching, we further introduce two structural consistency modules: path count consistency and path order scoring. The latter evaluates the semantic fidelity of rendering by assessing the alignment of drawing order, which in SVG inherently determines visual stacking and rendering priority.

The final PSS score is computed through weighted summation, where the macroscopic visual alignment score receives a weight of 0.6 and the microscopic structural correspondence score receives a weight of 0.4, balancing the importance of overall visual quality and structural precision.

E Generation

E.1 Text-to-SVG Generation

Construction Details. In the Text-to-SVG Generation task, each evaluation sample is constructed as a paired instance of a natural language caption and its corresponding SVG. To ensure that the

textual descriptions exhibit high semantic coverage and expressive fidelity, we employ a semi-automated pipeline as follows.

First, we render the original SVG into a high-resolution raster image using the Cairosvg library. The rendered image is then passed to the Claude-3.7-sonnet [1] model along with a minimal prompt, for example, "Describe this image in brief". The model generates an initial image caption that is then manually reviewed and refined. This human-in-the-loop step ensures that each caption accurately captures the core visual elements of the graphic, including structural layout, component composition, color attributes, and stylistic traits. To maintain quality, outputs exhibiting semantic ambiguity, structural inaccuracies, or excessive token length are systematically filtered.

This hybrid pipeline combines the scalability of automatic generation with the precision of manual validation, resulting in a high-quality dataset of paired captions and SVGs suitable for model evaluation.

Evaluation Metrics. To systematically evaluate the performance of models on the Text-to-SVG task, we propose a three-dimensional evaluation framework encompassing perceptual quality, visual reconstructability, and semantic alignment:

Perceptual Quality. This dimension assesses subjective aesthetic appeal and user-level acceptability of generated SVG. We adopt two metrics:

- **Aesthetic Score [38]:** Computed using a pre-trained image aesthetics model, this score evaluates the visual pleasantness of the rendered SVG image.
- **Human Preference Score (HPS) [54]:** Derived from a series of A/B testing rounds, this metric captures human preferences by comparing model-generated SVGs against reference graphics. The final score reflects the frequency with which participants favor the generated output.

Table 9: Performance on SVG editing dimension across different model types and difficulty levels. Results are reported using task-specific metrics (ACC, rMSE, MSE, RLD, CCR) for Bug Fixing, Style Editing and Code Optimization tasks, with models marked as specialized (★), reasoning (★), code (★), open-source (★), or proprietary (★) variants.

Method	Bug Fixing	Style Editing			Code Optimization	
	ACC↑	ACC↑	rMSE↑	RLD↓	MSE↓	CCR↑
Easy						
DeepSeek-R1-Distill-Qwen-1.5B [12]★	3.85	56.06	28.29	234.41	11.50	13.04
DeepSeek-R1-Distill-Qwen-7B [12]★	3.80	69.69	54.51	136.09	15.56	18.35
DS-R1-Dis-Qwen-32B [12]★	62.63	84.81	75.16	1.45	5.63	23.16
DeepSeek-R1 [12]★	71.00	84.62	73.66	18.21	1.01	23.67
Llama3.2-3B-Instruct [11]★	33.70	83.54	57.04	82.61	7.96	19.83
Mistral-Small-3.1-24B-Instruct [21]★	54.00	74.36	62.39	25.84	12.05	23.51
Qwen2.5-Coder-3B [15]★	9.43	56.82	39.55	136.92	17.29	20.90
Qwen2.5-3B-Instruct [34]★	36.46	70.51	60.99	126.06	13.13	13.41
Qwen2.5-72B-Ins [34]★	71.00	75.95	64.25	2.08	2.98	20.57
QwQ-32B [59]★	71.43	91.14	81.70	2.61	3.20	20.20
Qwen3-1.7B [59]★	22.34	74.36	67.22	33.40	11.43	35.79
Qwen3-8B [59]★	53.00	87.34	80.40	221.88	2.84	18.61
Qwen3-32B [59]★	56.12	88.46	82.63	1.17	2.55	17.96
Gemini-2.0-Flash [43]★	69.00	86.07	79.70	4.78	0.72	20.30
GPT-4o [16]★	74.00	78.48	65.81	46.53	1.30	10.57
Claude-3.7-Sonnet [1]★	76.00	79.75	67.17	20.89	0.31	16.81
Medium						
DeepSeek-R1-Distill-Qwen-1.5B [12]★	0.00	12.00	9.56	4757.67	14.73	47.38
DeepSeek-R1-Distill-Qwen-7B [12]★	0.00	36.36	36.33	5811.5	19.05	44.55
DS-R1-Dis-Qwen-32B [12]★	46.00	56.41	53.11	140.68	8.14	33.10
DeepSeek-R1 [12]★	63.83	62.16	53.93	1227.70	2.02	36.70
Llama3.2-3B-Instruct [11]★	4.60	50.67	37.93	442.66	5.33	15.53
Mistral-Small-3.1-24B-Instruct [21]★	16.85	37.88	32.46	742.92	14.50	46.48
Qwen2.5-Coder-3B [15]★	0.00	37.00	29.96	3992.66	12.85	79.18
Qwen2.5-3B-Instruct [34]★	4.30	34.33	27.11	1616.04	8.42	19.81
Qwen2.5-72B-Ins [34]★	50.56	59.49	51.90	136.00	3.41	30.49
QwQ-32B [59]★	46.00	64.56	61.27	41.67	5.03	29.28
Qwen3-1.7B [59]★	1.22	33.90	25.55	1656.50	10.20	34.87
Qwen3-8B [59]★	35.42	62.34	52.10	826.21	2.63	21.69
Qwen3-32B [59]★	46.47	66.67	62.54	17.13	2.78	26.61
Gemini-2.0-Flash [43]★	60.00	65.82	60.43	113.44	1.59	27.49
GPT-4o [16]★	49.00	50.63	42.64	840.73	4.74	43.01
Claude-3.7-Sonnet [1]★	75.00	59.74	53.61	134.52	0.86	26.41
Hard						
DeepSeek-R1-Distill-Qwen-1.5B [12]★	0.00	28.57	6933.75	352.85	12.38	57.41
DeepSeek-R1-Distill-Qwen-7B [12]★	0.00	42.86	42.85	11559.67	8.80	61.35
DS-R1-Dis-Qwen-32B [12]★	34.02	55.26	41.68	226.50	5.89	36.41
Deepseek-R1 [12]★	61.80	51.56	41.83	2610.48	2.41	39.95
Llama3.2-3B-Instruct [11]★	1.37	40.35	24.03	3631.91	5.34	9.52
mistral-small-3.1-24b-instruct [21]★	3.95	25.86	20.25	2247.67	11.99	55.31
Qwen2.5-coder-3B [15]★	0.00	36.36	32.81	7347.00	12.39	82.32
Qwen2.5-3B-Instruct [34]★	0.00	24.53	15.90	4122.69	4.94	20.76
Qwen2.5-72B-Ins [34]★	40.00	54.67	43.80	722.00	3.24	36.40
QwQ-32B [59]★	10.42	50.00	44.04	533.85	5.82	12.98
Qwen3-1.7B [59]★	0.00	26.42	21.94	3650.71	5.60	32.42
Qwen3-8B [59]★	25.30	60.81	52.21	935.13	2.15	23.68
Qwen3-32B [59]★	40.86	55.26	48.06	58.79	1.50	26.75
Gemini-2.0-Flash [43]★	53.95	57.38	42.34	628.14	2.62	29.35
GPT-4o [16]★	51.02	42.67	35.43	843.81	5.14	46.91
Claude-3.7-Sonnet [1]★	69.00	52.56	40.75	27.00	0.88	28.27

Table 10: Performance on SVG generation dimension across different model types and difficulty levels. Results are reported using task-specific metrics (CLIP, AES, HPS, rCLIP, PSS, Cart., Pixel, Line, 3D) for Text-based Generation and Style Transfer tasks, with models marked as specialized (*), reasoning (*), code (*), open-source (*), or proprietary (*) variants.

Method	Text-based Generation					Style Transfer			
	CLIP↑	AES↑	HPS↑	rCLIP↑	PSS↑	Cart.	Pixel	Line	3D
Easy									
Iconshop [53]*	22.74	3.56	17.99	86.22	5.26	-	-	-	-
LLM4SVG(GPT-2 XL) [55]*	18.09	3.63	16.76	75.78	3.01	-	-	-	-
DeepSeek-R1-Distill-Qwen-1.5B [12]*	18.43	3.55	16.51	73.51	0.79	1.27	1.25	0.36	1.30
DeepSeek-R1-Distill-Qwen-7B [12]*	19.27	3.50	16.71	76.76	0.93	0.77	0.65	1.60	0.90
DS-R1-Qwen-32B [12]*	22.04	3.47	17.42	84.09	10.47	3.14	1.55	3.04	2.73
DeepSeek-R1 [12]*	24.34	3.61	20.37	91.39	18.07	3.13	1.70	2.68	2.87
Mistral-Small-3.1-24B-Instruct [21]*	21.15	3.25	16.64	81.81	8.31	2.07	1.65	2.28	1.30
Qwen2.5-Coder-3B [15]*	19.23	3.22	14.97	74.08	1.32	-	-	-	-
Qwen2.5-3B-Instruct [34]*	20.03	3.19	15.81	78.25	2.33	1.70	1.87	1.68	2.17
Qwen2.5-72B-Ins [34]*	20.86	3.59	17.38	91.18	15.77	2.83	1.85	2.72	2.90
QwQ-32B [59]*	23.01	3.51	19.19	89.30	16.78	3.56	2.24	3.30	3.12
Qwen3-1.7B [59]*	20.37	3.50	16.92	79.80	10.22	1.64	2.35	3.00	1.97
Qwen3-8B [59]*	22.39	3.48	18.15	85.53	12.73	-	-	-	-
Qwen3-32B [59]*	23.81	3.48	19.39	89.13	12.62	2.93	2.10	2.92	2.80
Gemini-2.0-Flash [43]*	24.20	3.70	19.82	90.96	15.94	3.17	2.32	3.16	3.25
GPT-4o [16]*	24.97	3.63	20.35	90.95	19.72	3.27	1.95	2.88	2.53
Claude-3.7-Sonnet [1]*	25.11	3.96	21.35	92.90	16.69	3.68	2.00	2.08	2.93
Medium									
Iconshop [53]*	20.76	3.46	14.68	77.35	3.24	-	-	-	-
LLM4SVG(GPT-2 XL) [55]*	17.98	3.55	14.88	68.84	2.70	-	-	-	-
DeepSeek-R1-Distill-Qwen-1.5B [12]*	17.23	3.49	14.99	66.43	0.23	1.13	0.53	0.00	0.00
DeepSeek-R1-Distill-Qwen-7B [12]*	18.21	3.53	15.00	69.20	0.41	0.00	0.65	0.62	0.52
DS-R1-Qwen-32B [12]*	20.04	3.52	15.76	75.52	7.55	1.27	1.43	2.38	3.16
DeepSeek-R1 [12]*	22.54	3.55	17.46	82.99	14.78	2.40	1.82	2.12	2.44
Mistral-Small-3.1-24B-Instruct [21]*	20.09	3.32	15.34	75.40	5.39	1.80	0.68	1.38	0.00
Qwen2.5-Coder-3B [15]*	16.75	3.38	13.48	63.74	1.89	-	-	-	-
Qwen2.5-3B-Instruct [34]*	18.08	3.32	13.40	69.46	1.47	1.00	1.18	1.48	1.12
Qwen2.5-72B-Ins [34]*	20.40	3.50	16.35	79.91	13.11	3.07	2.13	2.12	3.52
QwQ-32B [59]*	21.39	3.57	16.73	77.87	17.56	3.00	1.47	2.31	3.48
Qwen3-1.7B [59]*	19.26	3.59	16.00	70.66	11.97	1.87	2.14	1.89	1.84
Qwen3-8B [59]*	21.65	3.45	16.70	80.37	11.26	-	-	-	-
Qwen3-32B [59]*	21.80	3.51	17.17	80.88	14.38	3.33	1.67	1.56	2.76
Gemini-2.0-Flash [43]*	21.77	3.65	17.00	80.36	13.70	3.13	1.87	2.13	3.00
GPT-4o [16]*	23.03	3.49	17.51	84.72	11.97	1.67	1.69	2.00	1.56
Claude-3.7-Sonnet [1]*	24.18	3.78	85.71	87.62	16.60	3.67	2.61	1.94	3.28
Hard									
Iconshop [53]*	19.34	3.48	12.95	72.55	2.12	-	-	-	-
LLM4SVG(GPT-2 XL) [55]*	16.19	3.61	14.62	62.98	0.02	-	-	-	-
DeepSeek-R1-Distill-Qwen-1.5B [12]*	16.19	3.46	14.01	61.25	0.12	0.00	0.08	0.32	0.00
DeepSeek-R1-Distill-Qwen-7B [12]*	17.51	3.48	14.10	65.33	0.25	0.24	0.35	1.04	0.00
DS-R1-Qwen-32B [12]*	19.22	3.63	14.56	72.71	6.56	2.20	1.50	2.08	2.40
DeepSeek-R1 [12]*	22.73	3.63	16.86	83.06	10.07	2.16	1.95	2.00	2.13
Mistral-Small-3.1-24B-Instruct [21]*	18.61	3.33	14.35	70.60	4.11	2.07	1.65	2.28	1.30
Qwen2.5-Coder-3B [15]*	16.07	3.35	12.50	59.91	1.31	-	-	-	-
Qwen2.5-3B-Instruct [34]*	17.38	3.30	13.31	65.59	2.08	0.96	1.15	1.24	1.20
Qwen2.5-72B-Ins [34]*	20.08	3.65	15.51	75.22	9.57	2.64	2.00	1.96	3.20
QwQ-32B [59]*	22.03	3.51	16.36	82.24	8.45	1.92	1.40	1.92	2.40
Qwen3-1.7B [59]*	18.37	3.50	14.52	68.91	4.46	1.32	1.30	2.24	1.20
Qwen3-8B [59]*	20.60	3.43	15.34	77.47	8.64	-	-	-	-
Qwen3-32B [59]*	22.06	3.52	16.02	81.84	9.88	2.96	1.27	1.60	3.33
Gemini-2.0-Flash [43]*	20.95	3.73	16.01	78.61	9.89	1.32	1.39	1.08	1.26
GPT-4o [16]*	22.57	3.64	16.69	82.81	10.39	1.84	1.92	1.96	2.73
Claude-3.7-Sonnet [1]*	24.54	3.96	18.74	87.97	10.19	2.64	2.36	1.80	3.33

Table 11: Performance on SVG generation dimension across different model types and difficulty levels. Results are reported using task-specific metrics (CP, DF, SC, CH, CB) for Style Transfer tasks, with models marked as reasoning (★), open-source (★), or proprietary (★) variants.

Method	Cartoon Style					Pixel art					Line art					3D style				
	CP	DF	SC	CH	CB	CP	DF	SC	CH	CB	CP	DF	SC	CH	CB	CP	DF	SC	CH	CB
Easy																				
DeepSeek-R1-Distill-Qwen-1.5B [12]★	1.17	0.67	0.67	1.17	2.67	0.75	0.75	0.75	1.25	2.75	0.40	0.20	0.40	0.40	0.40	1.17	1.00	0.67	1.83	1.83
DeepSeek-R1-Distill-Qwen-7B [12]★	0.33	0.33	0.83	0.67	1.67	0.50	0.50	0.50	0.75	1.00	1.20	1.20	1.60	1.60	2.40	0.67	0.67	0.67	1.17	1.33
DS-R1-Qwen-32B [12]★	3.17	2.67	2.67	2.17	5.00	1.25	1.25	1.25	1.75	2.25	2.80	2.60	3.00	2.40	4.40	2.83	2.00	2.33	3.00	3.50
DeepSeek-R1 [12]★	3.67	2.00	2.83	2.67	4.50	1.50	1.25	1.25	2.00	2.50	2.20	2.20	3.00	1.80	4.20	3.17	2.33	2.33	3.00	3.50
Llama3.2-3B-Instruct [11]★	0.83	1.00	0.83	1.33	1.83	1.75	1.75	1.50	2.25	2.75	1.00	0.80	0.40	0.60	1.40	0.83	0.67	0.67	1.17	1.33
Mistral-Small-3.1-24B-Instruct [21]★	2.00	1.67	1.83	1.50	3.33	1.25	1.25	1.00	2.00	2.75	1.60	2.40	2.00	1.40	4.00	1.50	1.17	0.83	1.00	2.00
Qwen2.5-3B-Instruct [34]★	1.50	1.33	1.17	1.50	3.00	1.50	1.00	1.17	2.17	3.50	1.60	1.00	1.20	2.20	2.40	2.33	1.67	1.00	2.83	3.00
Qwen2.5-72B-Ins [34]★	3.17	2.00	1.83	2.67	4.50	1.25	1.00	1.50	2.50	3.00	2.20	2.60	2.80	1.40	4.60	3.50	2.33	2.17	2.83	3.67
QwQ-32B [59]★	3.80	2.80	3.80	2.80	4.60	1.60	1.80	1.40	2.80	3.60	3.00	2.50	3.75	2.75	4.50	4.00	2.00	2.60	3.00	4.00
Qwen3-1.7B [59]★	1.17	1.00	1.17	1.67	3.17	2.00	2.00	1.75	2.50	3.50	3.00	3.00	2.80	1.80	4.40	2.67	1.50	1.00	2.17	2.50
Qwen3-4B [59]★	3.17	2.00	2.67	2.50	4.33	2.75	3.00	2.75	3.25	3.50	1.80	2.20	2.40	0.80	3.60	3.50	2.67	2.33	3.00	3.83
Qwen3-32B [59]★	3.17	2.33	2.67	2.50	4.00	1.50	1.50	1.75	2.75	3.00	3.60	2.80	1.80	1.80	4.60	3.00	2.33	2.67	2.83	3.17
Gemini-2.0-Flash [43]★	3.17	2.67	3.17	2.67	4.17	1.75	1.50	1.75	2.25	4.33	3.60	3.00	2.20	2.20	4.80	3.67	2.67	3.40	3.00	3.50
GPT-4o [16]★	3.67	2.67	2.67	2.83	4.50	1.50	1.50	1.50	2.25	3.00	3.20	2.00	3.40	1.80	4.00	3.00	2.00	1.83	2.50	3.33
Claude-3.7-Sonnet [1]★	4.00	2.75	3.83	3.00	4.83	1.50	1.50	1.75	2.25	3.00	2.00	1.40	1.40	1.20	4.40	3.33	2.50	2.33	2.67	3.83
Medium																				
DeepSeek-R1-Distill-Qwen-1.5B [12]★	0.67	0.67	0.67	1.33	2.33	0.42	0.58	0.33	0.50	0.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
DeepSeek-R1-Distill-Qwen-7B [12]★	0.00	0.00	0.00	0.00	0.00	0.58	0.50	0.42	0.75	1.00	0.50	0.50	0.60	0.60	0.90	0.40	0.40	0.40	0.60	0.80
DS-R1-Qwen-32B [12]★	1.00	1.33	1.00	1.33	1.67	1.25	1.17	1.17	1.58	2.00	2.50	1.80	1.60	1.70	4.30	3.20	3.00	3.20	2.80	3.60
DeepSeek-R1 [12]★	2.00	1.67	2.00	2.67	3.67	1.17	1.42	1.83	2.00	2.67	1.90	1.90	1.80	1.60	3.40	2.40	2.80	2.00	2.60	2.40
Llama3.2-3B-Instruct [11]★	0.67	0.67	0.67	1.00	0.67	0.75	0.75	0.64	1.17	1.25	1.00	0.90	0.17	0.70	1.50	1.40	1.20	1.40	1.20	1.60
Mistral-Small-3.1-24B-Instruct [21]★	1.67	1.33	1.67	1.67	2.67	0.42	0.42	0.50	1.00	1.08	1.10	1.10	1.50	0.90	2.30	0.00	0.00	0.00	0.00	0.00
Qwen2.5-3B-Instruct [34]★	0.75	0.75	1.00	1.25	1.25	1.00	0.92	0.83	1.33	1.82	1.30	1.20	1.10	1.80	2.00	0.80	0.60	0.80	1.80	1.60
Qwen2.5-72B-Ins [34]★	3.00	2.67	2.67	3.00	4.00	2.00	1.75	1.50	2.25	3.17	1.90	1.60	2.30	1.20	3.60	3.60	2.80	3.60	3.40	4.20
QwQ-32B [59]★	2.67	3.00	3.33	2.67	3.33	1.36	1.21	1.21	1.43	2.14	2.09	1.91	2.45	1.64	3.45	3.80	3.00	3.60	3.40	3.60
Qwen3-1.7B [59]★	2.00	2.00	0.67	2.00	2.67	1.91	2.20	1.83	2.25	2.50	2.20	1.44	1.30	1.40	3.10	2.20	1.80	1.20	1.80	2.20
Qwen3-4B [59]★	3.33	3.00	2.33	2.33	3.33	3.00	2.67	2.92	2.58	3.17	2.20	2.00	2.40	1.20	3.70	4.00	2.60	2.60	3.60	3.60
Qwen3-32B [59]★	3.33	3.00	2.67	3.00	4.67	1.00	1.25	1.50	2.00	2.58	1.60	1.30	1.10	1.20	2.58	2.60	2.40	2.80	2.60	3.40
Gemini-2.0-Flash [43]★	3.00	3.00	3.33	3.00	3.33	1.64	1.42	1.58	2.25	2.45	1.67	1.90	2.10	1.30	3.70	3.00	2.80	3.20	2.80	3.20
GPT-4o [16]★	1.67	2.00	1.00	1.33	2.33	1.45	1.55	1.45	1.91	2.09	2.00	1.30	1.30	1.40	4.00	1.60	1.60	1.40	1.60	1.60
Claude-3.7-Sonnet [1]★	3.00	3.00	4.00	3.33	5.00	2.50	2.33	2.58	2.83	2.83	1.60	1.40	1.30	1.50	3.90	3.60	3.00	3.00	2.80	4.00
Hard																				
DeepSeek-R1-Distill-Qwen-1.5B [12]★	0.00	0.00	0.00	0.00	0.00	0.06	0.06	0.06	0.12	0.12	0.20	0.20	0.20	0.20	0.80	0.00	0.00	0.00	0.00	0.00
DeepSeek-R1-Distill-Qwen-7B [12]★	0.20	0.20	0.20	0.20	0.40	0.19	0.19	0.25	0.50	0.62	0.60	1.00	1.40	0.60	1.60	0.00	0.00	0.00	0.00	0.00
DS-R1-Qwen-32B [12]★	2.20	1.80	1.60	2.20	3.20	1.25	1.19	1.44	1.69	1.94	1.00	1.40	2.60	1.20	4.20	2.67	2.00	1.67	2.67	3.00
DeepSeek-R1 [12]★	1.40	1.60	1.80	2.60	3.40	1.60	1.25	1.88	2.19	2.81	1.00	1.40	3.20	1.20	3.20	2.00	1.67	2.00	2.67	2.33
Llama3.2-3B-Instruct [11]★	0.83	1.00	0.83	1.33	1.83	1.75	1.75	1.50	2.25	2.75	1.00	0.80	0.40	0.60	1.40	0.83	0.67	0.67	1.17	1.33
Mistral-Small-3.1-24B-Instruct [21]★	2.0	1.67	1.83	1.5	3.33	1.25	1.25	1.0	2.0	2.75	1.6	2.4	2.0	1.4	4.0	1.5	1.17	0.83	1.0	2.0
Qwen2.5-3B-Instruct [34]★	0.80	0.80	0.80	0.80	1.60	0.79	0.79	0.95	1.32	1.89	1.00	1.00	1.00	1.60	1.60	1.00	1.00	1.00	1.33	1.67
Qwen2.5-72B-Ins [34]★	3.40	2.60	1.60	2.00	3.60	1.94	1.69	1.62	2.12	2.62	1.20	1.40	2.40	1.00	3.80	3.33	3.33	2.00	3.33	4.00
QwQ-32B [59]★	1.80	1.60	1.60	1.80	2.80	0.94	1.00	1.31	1.62	2.12	1.40	1.60	2.20	1.80	2.60	2.33	2.00	2.67	2.33	2.67
Qwen3-1.7B [59]★	1.00	1.00	1.00	1.40	2.20	1.27	1.12	1.12	1.19	1.81	2.20	1.80	2.80	1.60	2.80	0.67	0.67	1.00	1.67	2.00
Qwen3-4B [59]★	2.60	2.20	2.60	2.20	4.20	3.06	2.44	2.81	3.00	3.69	1.80	1.80	3.40	1.60	4.20	2.67	2.33	2.67	2.00	3.33
Qwen3-32B [59]★	3.20	2.60	2.00	2.80	4.20	1.12	1.00	1.06	1.25	1.94	1.00	1.00	1.60	1.00	3.40	4.00	3.00	3.33	2.67	3.67
Gemini-2.0-Flash [43]★	1.20	1.00	1.20	1.40	1.80	1.00	0.94	1.25	1.56	2.19	0.60	1.00	1.20	0.60	2.00	1.33	1.00	1.33	1.33	1.33
GPT-4o [16]★	1.40	1.20	1.40	1.60	3.60	1.69	1.44	1.81	2.12	2.56	1.00	1.40	2.20	1.40	3.80	3.00	3.00	2.00	2.33	3.33
Claude-3.7-Sonnet [1]★	2.60	2.40	2.20	2.20	3.80	2.25	2.06	2.06	2.44	3.00	1.00	1.40	1.60	1.20	3.80	3.67	3.00	3.00	3.00	4.00

Visual Reconstructability. This dimension evaluates how well the generated SVG replicates the structural code characteristics of the reference SVG. We design a suite of custom metrics PSS(cf. Appendix D for more details) that compare both SVG code and

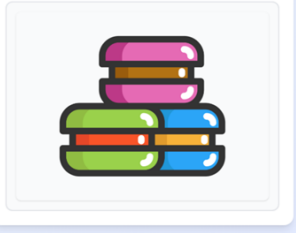
rendered output across subdimensions including path topology, geometric layout, and attribute encoding.

SVG Benchmark Quality Evaluation

Thank you for participating in this SVG benchmark quality evaluation. Please evaluate the following image in our dataset through three aspects.

1 2 3 4 5

Image for Evaluation



Complex

Image Description

a stack of colorful macarons

Difficulty: Complex

⚠ Please complete the following required fields:

- SVG Quality assessment is required
- Alignment Between SVG and Descriptions assessment is required
- Difficulty Grading assessment is required

SVG Quality

How satisfied are you with the quality of the SVG image(clarity, readability, aesthetics, etc.) in our benchmark dataset?

☐ Very satisfied

☐ Satisfied

☐ Neutral

☐ Somewhat dissatisfied

☐ Dissatisfied

Alignment Between SVG and Descriptions

To what extent does the SVG picture align with the description below?

☐ Perfectly aligned

☐ Mostly aligned

☐ Neutral

☐ Partially misaligned

☐ Completely misaligned

Difficulty Grading

To what extent do you think the difficulty level of this image is appropriately categorized?

☐ Very reasonable

☐ Reasonable

Figure 4: Screenshot of Our Questionnaires.

Semantic Alignment. This dimension measures the degree to which the generated SVG semantically aligns with the input caption. We employ two complementary metrics:

- **CLIP Score:** The generated SVG is rendered and embedded alongside the input caption using a CLIP model. The cosine similarity between their embeddings quantifies semantic alignment.
- **Relative CLIP Consistency (rCLIP):** We propose this novel metric to evaluate the semantic degradation of the generated image relative to the ground truth. It is defined as:

$$rCLIP = 1 - \max\left(0, \frac{CLIP(caption, GT) - CLIP(caption, GEN)}{CLIP(caption, GT)}\right) \quad (1)$$

Here, $CLIP(caption, GT)$ denotes the CLIP similarity between the input caption and the ground-truth rendering, while $CLIP(caption, GEN)$ measures similarity with the model-generated output. Higher rCLIP values indicate stronger semantic preservation.

Prompt for Text-based Generation

You are a professional SVG designer with extensive experience in vector graphics creation. Please generate the corresponding SVG code based on the user's description.

Provide your answer strictly in the following format: Answer: {SVG code}, providing the complete fixed code only in the SVG code position, without adding any explanations, comments, or other content.

Important Notes:

Your SVG must ONLY include `<path>` elements with "fill" and "d" attributes. Do not use any other SVG elements or attributes, and must use exactly this opening tag: `<svg class="icon" viewBox="0 0 1024 1024" version="1.1" xmlns="http://www.w3.org/2000/svg">`

Please generate an SVG code of: {prompt}

E.2 Image-to-SVG Generation

Construction Details. In the Image-to-SVG Generation task, we aim to evaluate a model's capability to generate structured SVG conditioned on both image and textual inputs. Each evaluation sample comprises three components: (1) a ground-truth SVG file, (2) its corresponding rendered raster image (generated via the CairoSVG library), and (3) a paired natural language description. The image-caption pairs are constructed following the same pipeline as in the Text-to-SVG task.

By leveraging both visual and textual modalities, this task simulates realistic usage scenarios in which users express design intent through a combination of imagery and language. Consequently, it imposes a more challenging multimodal grounding requirement on generative models.

Evaluation Metrics. To comprehensively assess the performance of multimodal large language models (MLLMs) on this task, we propose two complementary categories of evaluation metrics, each targeting a distinct aspect of generation quality: perceptual similarity and visual reconstructability.

Perceptual Similarity These metrics quantify the visual fidelity of the generated SVGs relative to the reference graphics, both globally and locally:

- **Learned Perceptual Image Patch Similarity (LPIPS) [62]:** Measures perceptual differences in deep feature space, capturing consistency in local texture and edge structures.
- **Structural Similarity Index Measure (SSIM) [51]:** Evaluates similarity in terms of luminance, contrast, and structural alignment, with an emphasis on holistic visual reconstruction.

- **DINO Score [32]**: Leverages semantic embeddings from the self-supervised DINO model to assess structural alignment between generated and reference images, particularly suited to evaluating contour preservation and compositional fidelity.

Visual Reconstructability Given the structural and syntactic nature of SVG graphics, we further introduce metrics that analyze both image-level and code-level alignment:

- **MSE**: Computes pixel-wise differences between the rendered outputs of generated and reference SVGs within a spatially aligned viewport, incorporating boundary-aware weighting to emphasize contour fidelity.
- **Path-Structure Similarity Score (PSS)**: Extracts and compares SVG path commands, attributes, and hierarchical organization to assess fine-grained geometric and syntactic alignment. (cf. Appendix D for more details)

Together, these perceptual and structural metrics provide a robust and reproducible evaluation framework for multimodal generative tasks, capturing both the visual plausibility and syntactic correctness of generated SVGs.

Prompt for Image-based Generation

You are a professional SVG designer with extensive experience in vector graphics creation. Please generate the corresponding SVG code based on the user's description and the provided image reference.

Provide your answer strictly in the following format: Answer: {SVG code}, providing the complete fixed code only in the SVG code position, without adding any explanations, comments, or other content.

Important Notes:

Your SVG must ONLY include <path> elements with "fill" and "d" attributes. Do not use any other SVG elements or attributes, and must use exactly this opening tag: `svg class="icon" viewBox="0 0 1024 1024" version="1.1" xmlns="http://www.w3.org/2000/svg">`

Please generate an SVG code based on this description: {prompt} and the reference image I've provided.

E.3 Style Transfer

Construction Details. In the Style Transfer Generation task, we aim to evaluate the model's ability to perform cross-style transformations of SVG graphics while preserving both the underlying structure and semantic content. Given the context-length limitations of large multimodal models when processing long SVG sequences, we prioritize the selection of compact and semantically clear SVG samples during dataset construction. This ensures that the transformation process focuses on stylistic expression rather than structural reconstruction.

We define four representative target styles that span key axes of variation across mainstream visual aesthetics:

- **3D Style**: Enhances depth and lighting effects, assessing the model's ability to abstract and reconstruct complex rendering semantics.
- **Line art**: Emphasizes outlines and linear representations, requiring the model to simplify visual elements while retaining structural integrity.
- **Pixel art**: Mimics low-resolution rasterized appearances, testing the model's capacity for detail compression under strict structural constraints.
- **Cartoon Style**: Introduces dynamic and emotive visual semantics, such as cartoonish or anthropomorphic traits, highlighting the model's ability in affective style transfer.

To increase both the discriminability and difficulty of the Style transfer task, each sample is manually assigned a target style that is maximally divergent from the original SVG style. For instance, icon-like SVGs with well-defined geometric outlines are more often mapped to Pixel or Cartoon styles, while graphics involving shading or gradients are preferentially assigned to the Line art style. This strategy maximizes the stylistic shift in each task instance, thus enhancing the challenge for models and the effectiveness of evaluation.

Each task instance is composed of three elements: (1) an original SVG graphic, (2) a target style label (chosen from the four defined categories), and (3) a natural language instruction (e.g., please convert this graphic to pixel art style). This formulation not only provides a consistent prompt format, but also enforces the requirement that the output preserves semantic structure while completing the stylistic transformation, facilitating clearly aligned downstream evaluation.

Evaluation Metrics. We develop a two-tier automated assessment framework leveraging LLMs to quantify transfer quality from global and local perspectives.

Global Ranking Evaluation. Based on the results of the multi-dimensional evaluation, as shown in Table 12 we identify six high-performing models for subsequent holistic assessment: the proprietary models Claude-3.7-Sonnet [1], Gemini-2.0-Flash [43], and GPT-4o [16], along with the open-source models DeepSeek-R1 [12], QwQ-32B [59], and Qwen3-32B [59].

Table 12: Comparison of Overall Rank Evaluation

Models	Winrate(%)
Claude-3.7-Sonnet [1]	61.54
DeepSeek-R1(Reference Model) [12]	50.55
Gemini-2.0-Flash [43]	49.28
QwQ-32B [59]	47.06
GPT-4o [16]	45.45
Qwen3-32B [59]	40.26

The final evaluation focuses on three criteria: semantic preservation, stylistic fidelity, and visual quality. We employ the AlpacaEval framework using DeepSeek-R1 [12] as the reference model. The outputs of each competing model are compared against the reference model's results across all samples. As shown in Table 12, the final rankings are derived from aggregated win-loss statistics, sorted by win rate relative to the baseline.

Local Automated Multi-dimensional Evaluation. The stylized SVG outputs are first rendered into image format and subsequently scored on a scale of 1–5 by an evaluation model (with a score of 0 assigned to invalid or erroneous generations). We design five metric including Content Preservation (CP), Detail Fidelity (DF), Style Consistency (SC), Color Harmony (CH) and Composition Balance (CB). More details are shown in Table 13. For each dimension of evaluation, we define qualitative descriptors that correspond to each level of score. We adopt GPT-4o-mini [16] as the evaluation model, which references the original image, the stylized image, and the descriptor texts to first generate detailed feedback and then assign a final score. To ensure consistency of evaluation, the model operates with a deterministic temperature setting of 0.05.

Prompt for Automatic Evaluation of Style Transfer

You are a fair judge assistant tasked with providing clear, objective feedback based on specific criteria, ensuring each assessment reflects the absolute standards set for performance.

Task Description: An instruction (might include an Input inside it), a response to evaluate, and a score rubric representing a evaluation criteria are given.

1. Make a brief description of the style transfer that how it modifies the image 1 to image 2.
2. Write a detailed feedback that assess the quality of the response strictly based on the given score rubric, not evaluating in general.
3. After writing a feedback, write a score that is an integer between 1 and 5. You should refer to the score rubric.
4. The output format should look as follows: "Feedback: (write a feedback for criteria) [RESULT] (an integer number between 1 and 5)"
5. Please do not generate any other opening, closing, and explanations.

The instruction to evaluate:
transfer the provided Image 1 to {style}

Response to evaluate:
the given Image 2

Score Rubrics:
{rubric}

Feedback:

Prompt for Style Transfer

You are a professional SVG designer with extensive experience in vector graphics creation. Your task is to perform a style transfer - recreate the provided reference SVG. Maintain the basic structure and given semantic description of the SVG, but adjust it to the desired style. Provide your answer strictly in the following format: Answer:{SVG Code}, without adding any explanations, comments, or other content.

Reference SVG to transform:
{reference_svg}

Description of the reference SVG:
{description}

the style to transfer:
{style}

F Understanding

Construction Details. To systematically evaluate the cognitive capabilities of large vision-language models (VLMs) on SVG-based graphics, we propose an understanding task comprising two subtasks: Perceptual Understanding and Semantic Understanding. Given the current limitations of multimodal models in directly parsing SVG code, we adopt an indirect approach. Specifically, raw SVGs are rendered into standard bitmap images using CairoSVG, which are then fed into the Qwen2.5-VL-72B-Instruct model for automatic generation of multi-choice question-answer (QA) samples.

Perceptual QA focuses on low-level visual features of the graphic, including shape recognition, color identification, and relative spatial positioning. Representative questions include: "Which of the following basic shapes is present in the image?", "What is the dominant color of the figure?", or "In which direction is the ellipse located relative to the center?"

Semantic QA targets higher-level semantic interpretation, such as the meaning, functionality, or plausible real-world use of the graphic. Example questions include: "Which of the following scenarios is the image most likely to represent?" or "Which software interface might use this icon?"

To ensure the quality of the QA samples, all generated questions and answer options undergo manual validation. This includes filtering out ambiguous phrasing and reconstructing distractors to guarantee that each question has a single correct answer and well-separated alternatives.

Evaluation Metric. We adopt accuracy as the sole evaluation metric for the Understanding Task. For each multiple-choice question, the model is required to select exactly one correct answer. Accuracy is computed as the ratio of correctly answered questions to the total number of questions within each subtask. This metric offers a straightforward and interpretable means of assessing model performance on both perceptual and semantic dimensions of visual understanding.

Table 13: Details of Score Rubrics used in Automatic Evaluation

criteria description	Content Preservation: To what extent does the SVG conversion preserve the basic content and main elements of the original image?
score1 description	Essential elements are missing or severely distorted; core content is unrecognizable.
score2 description	Key elements are present but with noticeable omissions or alterations.
score3 description	Most main elements are preserved, though some secondary features may be simplified.
score4 description	All critical content is accurately retained with minor detail loss.
score5 description	Perfect preservation of all primary and secondary elements without compromise.
criteria description	Detail Fidelity: To what extent does the SVG conversion preserve the details of the original image?
score1 description	Fine details are completely lost; output appears overly simplified or blurred.
score2 description	Basic details are visible but intricate features (e.g., textures, small text) are poorly rendered.
score3 description	Moderate detail retention; some high-frequency elements may lack precision.
score4 description	High-fidelity details with only subtle imperfections in complex areas.
score5 description	Pixel-perfect detail reproduction matching the original's complexity.
criteria description	Style Consistency: Compared to the reference style, how well does the conversion match the target style?
score1 description	Style is inconsistent or contradictory to reference (e.g., line art vs. painterly).
score2 description	Partial style adherence with noticeable deviations in techniques/effects.
score3 description	Broadly matches reference style but lacks nuanced execution.
score4 description	Close stylistic alignment with minor variations in rendering methods.
score5 description	Seamless style replication that could pass as original artwork.
criteria description	Color Harmony: How harmonious is the color combination in the conversion result?
score1 description	Clashing or jarring colors; poor contrast/balance distracts from content.
score2 description	Some discordant color pairs but maintains basic readability.
score3 description	Generally pleasing palette though certain hues may feel slightly off.
score4 description	Well-balanced colors with intentional aesthetic cohesion.
score5 description	Masterful color theory application enhancing visual appeal.
criteria description	Composition Balance: How balanced is the visual composition of the conversion result?
score1 description	Unbalanced weighting creates visual tension or emptiness.
score2 description	Some elements feel misaligned or disproportionately emphasized.
score3 description	Adequate balance though certain areas could benefit from adjustment.
score4 description	Strong compositional flow with deliberate focal points.
score5 description	Expert-level layout adhering to design principles (e.g., rule of thirds, negative space).

Prompt for Generating Questions in Perceptual and Semantic QA

Please analyze the provided icon image and its caption, then generate 2 multiple choice questions (one for each category below). Each question should have four options (A, B, C, D) with exactly one correct answer.

Generate these two types of questions:

1. Perceptual Question: Focus on the visual features of the icon such as shapes, number of elements, or spatial arrangement.
2. Semantic Question: Explore the meaning, function, or use-case of the icon. Focus on what the icon represents or where it might typically appear.

Format requirements: - Question: [question text] Options: A) [option A]; B) [option B]; C) [option C]; D) [option D]

Answer: [correct option letter]

Guidelines:

- All questions must be directly grounded in the icon image and its caption
- All alternative options should be plausible but clearly distinguishable from the correct answer
- Ensure questions have varying difficulty levels appropriate to their category
- The correct answer should be objectively verifiable based on the image and caption

Prompt for Perceptual and Semantic QA

You are a svg analysis expert. Follow these steps carefully to answer the given multiple choice question.

Task instruction:

1. Answer the given multiple choice question below according to the svg code.
2. Output the answer in the format 'Answer: X' in the last line, where X is one of A, B, C, or D.

SVG Code:

{svg_image}

Question:

{question}

Options:

{options_str}

Important Notes:

- You should answer exactly the given multiple-choice question, DO NOT propose a new question.
- Your output in the last line must be strictly in the format 'Answer: X', where X is one of A, B, C, or D.

Now, answer the given question:

G Editing

G.1 Bug Fixing

Construction Details. The bug fixing task is designed to systematically evaluate a model’s ability to identify and repair syntactic anomalies in SVG. This task simulates common error scenarios that arise in real-world design workflows, focusing on three major categories of typical faults:

- **Tag Errors:** Structural issues in the XML hierarchy due to incorrect or malformed element tags.
- **Path Command Errors:** Logical inconsistencies in shape rendering caused by invalid or misused path commands.
- **Attribute Errors:** Disruptions in visual appearance resulting from incorrect attribute names or values.

Each task instance consists of the following components:

- **Corrupted SVG:** Automatically generated using our custom tool `SVGErrorGenerator`, which injects faults with clearly defined types and diverse manifestations, ensuring that the rendered output is visibly degraded.
- **Ground-Truth SVG:** The original, error-free SVG file, used to evaluate the correctness of the model’s output.

The error generation process supports fine-grained control over fault injection parameters, including error type, frequency, and random seed, allowing for balanced sample diversity and task difficulty. Supported error types include tag misspellings or omissions, illegal path command substitutions, and incorrect attribute keys or values.

Evaluation Metrics. To comprehensively assess the performance of models in this task, we adopt the accuracy of bug fixing as the core evaluation metric:

- **Accuracy (ACC):** Measures the proportion of SVG outputs that are fully restored to a structurally and semantically correct state. Evaluation is based on a strict equivalence check against the ground-truth SVG to determine successful repairs.

This metric characterizes a model’s end-to-end capability in the detect-locate-repair loop. Importantly, we emphasize that successful repairs must not only restore structural validity, but also recover the intended design semantics and rendering fidelity.

Prompt for Bug Fixing Task

You are a professional SVG repair engineer with expertise in SVG standards and common error types.

Your task is to analyze submitted SVG code, precisely locate errors, and fix problems using the principle of minimal modification.

Please only modify the parts causing errors while keeping the rest of the code unchanged.

After fixing, return the complete corrected code in the following strict format: Answer:{SVG code}, providing the complete fixed code only in the SVG code position, without adding any explanations, comments, or other content.

BUG SVG:
{bug_svg}

G.2 Code Optimization

Construction Details. The code optimization task is designed to evaluate the model’s ability to perform structural compression and syntactic refactoring of SVG source code while preserving its rendered appearance. Inspired by the optimization strategies employed by the widely adopted open-source tool `SVGGO` [10], we construct an automated dataset tailored for structure-aware SVG code optimization.

Each sample in the dataset comprises the following three components:

- **Optimization Instruction:** Expressed in natural language, explicitly prompting referencing the `SVGGO` [10] principle the model to compress, simplify, and remove redundancies in the SVG code according to professional standards.
- **Original SVG:** Our dataset, featuring diverse structures and frequently containing optimizable redundancies.
- **Groud-Truth SVG:** Generated automatically via a custom Python interface to the `SVGGO` [10] library, ensuring structural validity and visual fidelity with respect to the original SVG.

The dataset spans a wide range of graphical complexities, providing the model with realistic optimization scenarios that reflect practical code refinement tasks.

Evaluation Metrics. To comprehensively assess the performance of the model in two key dimensions: compression effectiveness and visual fidelity. We propose the following dual-metric evaluation scheme:

- **Compression Code Ratio (CCR):** This metric quantifies the reduction in byte size of the optimized SVG relative to the original SVG. It serves as a proxy for the model’s ability to compactly restructure the code without altering its visual rendering. The compression ratio is defined as:

$$CCR = (1 - \frac{\text{Optimized Size}}{\text{Original Size}}) \times 100\% \quad (2)$$

- **MSE:** To ensure visual fidelity, we compute the pixel-wise Mean Squared Error (MSE) between rasterized images of the original SVG and the model-optimized SVG. This metric captures rendering discrepancies and evaluates whether the visual output remains perceptually consistent after optimization.

To enhance sensitivity, we additionally report the MSE trend under varying compression levels, allowing for finer-grained analysis of whether aggressive compression leads to visual degradation. This enables detection of trade-offs where structural compactness may come at the cost of fidelity.

The core objective of this task is to encourage models to achieve structural compression without compromising visual consistency. Hence, optimized outputs with higher compression ratios and lower MSE scores are considered indicative of stronger optimization capabilities.

Prompt for Code Optimization Task

You are an advanced SVG optimization expert, skilled at maximizing compression and optimization of SVG code while maintaining visual consistency and ensuring code correctness. Please optimize the user-provided SVG code according to the following core principles:

1. Remove metadata and editor information
 - Clear metadata, comments, and unnecessary attributes generated by design software
 - Remove hidden elements and empty tags
2. Path optimization
 - Simplify path data, reduce control points
 - Lower decimal precision (1-2 places is usually sufficient)
 - Merge similar paths
3. Attribute and style processing
 - Remove redundant or default attribute values
 - Merge duplicate styles
 - Optimize color representation (e.g., #000 instead of #000000)
4. Structure optimization
 - Remove unnecessary grouping and nesting
 - Optimize IDs and class names
 - Ensure viewBox is set correctly
5. Compression and fine-tuning
 - Remove unnecessary whitespace and units
 - Use short commands instead of long format commands

After optimization, please strictly return the complete optimized code in the following format: Answer: {SVG code}. Provide the complete optimized code only in the SVG code position, without adding any explanations, comments, or other content.

Here is the original SVG to optimize:
{origin_svg}

G.3 Style Editing

Construction Details. The SVG Editing Task is designed to evaluate the capability of large language models (LLMs) to perform localized and controllable modifications on structured graphics. We construct a synthetic dataset containing diverse types of editing operations. Each sample consists of three components: a natural language editing instruction, the original SVG graphic, and the corresponding ground-truth SVG after modification.

The editing instructions span a variety of common transformation types, including but not limited to:

- Element-level color modifications (e.g., fill color, stroke color)
- Geometric transformations (e.g., rotation, translation, scaling)
- Stylistic enhancements and adjustments (e.g., blur filters, gradients, stroke width)

All samples are automatically generated via programmatic scripts to ensure semantic clarity, structural validity, and compatibility with automated evaluation pipelines. To support fine-grained manipulation of SVG elements, we develop a custom SVG editing

toolkit based on Python and lxml, enabling precise control over element-level modifications. Each task instance pairs a well-defined natural language instruction with a reproducible target SVG, ensuring consistency and objectivity across the evaluation dataset.

Evaluation Metrics. Traditional image-level metrics (e.g., Mean Squared Error, MSE) are often insufficiently sensitive for SVG editing tasks, as they may not capture fine-grained structural changes and can be misleading due to global rendering differences. To address this, we introduce three metric evaluation schemes that emphasize both structural fidelity and perceptual relevance:

- **Relative Levenshtein Distance (RLD):** Measures the minimal structural modification cost between the model-generated SVG and the ground-truth SVG, computed over the raw SVG markup. This captures the syntactic efficiency of the model's edits.
- **Relative Mean Squared Error (rMSE):** A structure-aware ratio metric that quantifies the degree of visual correction (or degradation) relative to the original input. It is defined as:

$$rMSE = \sqrt{1 - \min \left(1, \frac{MSE(gen, gt)}{MSE(gt, ori)} \right)} \quad (3)$$

where *gen* is the model's rendered output, *gt* is the rendered ground-truth SVG, and *ori* is the rendered original SVG. A larger *rMSE* indicates that the model output more closely aligns with the intended target and represents a significant structural improvement over the original.

- **Accuracy:** We employ accuracy as the primary evaluation metric to assess models' proficiency in executing specific editing operations correctly, where accuracy is computed as the proportion of successful task completions over the total number of editing attempts across all test samples.

This evaluation framework jointly captures the syntactic and semantic accuracy of model edits, offering fine-grained insight into model performance on real-world SVG editing tasks and enhancing the reliability of comparative analysis.

Prompt for Style Editing Task

You are a professional SVG editing engineer with extensive experience in SVG editing.

Your task is to receive SVG code and modification requests from users, and make precise modifications according to their instructions. Please only modify the parts specified by the user, keeping the rest of the code unchanged. After fixing, return the complete corrected code in the following strict format: Answer:{SVG code}, providing the complete fixed code only in the {SVG code} position, without adding any explanations, comments, or other content.

Here is the original SVG:
{origin_svg}

Edit command:
{edit_command}