Generation and Evaluation of Editable Graphical Abstracts for Academic Papers

TAKURO KAWADA[1,a)] SHUNSUKE KITADA[1,b)] HITOSHI IYATOMI[1,c)]

Abstract

Graphical abstracts (GAs) are visual summaries that convey the key ideas, methods, and findings of academic papers at a glance. However, existing GA generation methods typically produce raster graphics that are difficult to post-edit and risk hallucinating or fabricating scientific data through image generation. We propose a framework for generating data-grounded GAs directly as editable vector graphics, enabling element-level editing in common drawing tools. We also introduce the Structural Independence Coefficient (SIC) to quantify editing simplicity. Experiments and a user study show that our method improves editability while preserving visual quality, accelerating reliable scientific communication within AI for Science.

1. Introduction

Graphical Abstracts (GAs) are visual summaries that concisely present the key findings of research papers, often appearing as teasers or as *Figure 1* at the beginning of papers. GAs enable readers to quickly grasp the overall contribution of a study and enhance its visibility and impact [1], [7], [10]. Designing a compelling GA requires careful prioritization of information and thoughtful visual composition [8], [11], posing a challenge for many researchers.

Amid the momentum of AI for Science, automatic generation of GAs and related scientific visualizations has attracted increasing attention. However, most conventional approaches [6], [14], [16] generate raster graphics, which lack structural information and are therefore difficult to edit. Furthermore, these methods do not structurally integrate user-provided data and instead generate all visual elements. As a result, challenges remain in producing reliable scientific figures grounded in real data.

To address these limitations, we introduce a new task termed *Editable GA Generation*. Instead of generating GAs as static raster graphics, this task aims to produce editable vector graphics that support human post-editing. In particular, we emphasize two key requirements: the editability of the generated results and the reliable integration of researcher-provided data. To realize this task, we propose two core components. First, we introduce the *Structural Independence Coefficient (SIC)*, a metric that quantifies editing simplicity based on the structural independence of visual elements. It evaluates editing locality, ensuring that local edits remain local without global side effects. A high SIC reflects a highly editable figure where local edits remain local without global side effects. Second, we propose *GenGA* that directly generates editable vector GAs in SVG format. By explicitly representing structural information, GenGA enables element-level editing while preserving user-provided data. The generated outputs can be readily imported into common drawing tools such as PowerPoint, Adobe Illustrator, Figma, and draw.io, allowing seamless manual refinement within existing workflows.

[1] Graduate School of Science and Engineering, Hosei University
[a)]takuro.kawada.3g@stu.hosei.ac.jp
[b)]info@shunk031.me
[c)]iyatomi@hosei.ac.jp

2. Proposed Tasks, Metric, and Framework

In this paper, we consider the practical workflow of GA creation, where editable and reliable figures grounded in real data are required, and make three contributions: (1) We formulate Editable GA Generation as a structural generation problem that produces editable vector graphics; (2) We introduce the Structural Independence Coefficient (SIC), a metric for evaluating the editing simplicity of generated GAs; and (3) We introduce GenGA, a framework for solving Editable GA Generation.

2.1 Task Formulation

We define Editable GA Generation as the task of generating a GA in a highly editable representation. We characterize *editability* by two aspects: (1) *Editing availability*, i.e., whether a target edit can be performed in a drawing tool, depending on the representation format, i.e., vector or raster; and (2) *Editing simplicity*, i.e., how easily an available edit can be performed, depending on design and structural complexity. Based on these considerations, we formulate the task as follows:

$$\Phi : (T_{\text{full-text}}, A) \rightarrow I_{\text{vector}}, \tag{1}$$

where $T_{\text{full-text}}$ denotes the full text of a target paper, and $A = \{(a^{(m)}_{\text{raster}}, a^{(m)}_{\text{caption}}) \mid m \in \{1, 2, \ldots, N_{\text{a}}\}\}$ is a set of user-provided visual assets, each consisting of a raster graphic $a^{(m)}_{\text{raster}}$ (e.g., an input-output example or a device photograph) and its corresponding caption $a^{(m)}_{\text{caption}}$. The output I_{vector} denotes a vector-based GA in which each $a^{(m)}_{\text{caption}}$ are incorporated as visual components, enabling element-level editing in standard drawing tools.

2.2 Structural Independence Coefficient (SIC)

We next introduce the Structural Independence Coefficient (SIC), a metric for evaluating the editing simplicity of a figure that can be applied to both vector and raster graphics. SIC mod-

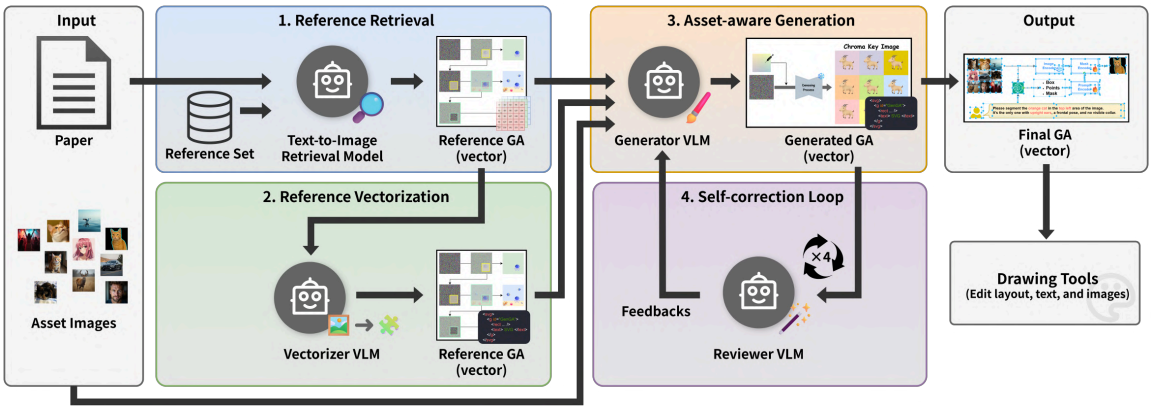


Fig. 1: Overview of our GenGA framework. Given a paper and optional asset images, GenGA generates an editable GA in SVG format that enables element-level editing and integration with existing tools. It retrieves and vectorizes reference GAs to capture visual and structural patterns, then refines the output through a self-correction loop to improve semantic alignment, layout, and readability.

els a figure as a dependency graph over its elements and quantifies the extent to which a local modification to one element propagates to others, yielding a score in $[0, 1]$. A higher SIC indicates that the effects of edits remain localized, allowing elements to be modified more independently.

Definition. We model a figure I as a weighted complete undirected graph $G = (V, E)$ that represents dependencies among visual elements. Here, $V = \{v_j \mid j \in \{1, 2, \ldots, N_\mathrm{v}\}\}$ denotes the set of nodes, where each v_j corresponds to a visual component in the figure (e.g., text, shapes, arrows, or images). $E = \{(v_j, v_k, w_{jk}) \mid j < k\}$ denotes weighted edges. Each weight $w_{jk} \in [0, 1]$ is the probability that a local edit to v_j propagates to v_k, with larger values indicating stronger dependency. By construction, we have $w_{jk} = w_{kj}$. We further assume that edit effects propagate transitively along dependencies between visual elements. To model this stochastic propagation process, we perform independent Bernoulli trials on each edge (v_j, v_k), activating the edge with probability w_{jk}. This yields a random edge set $\tilde{E} \subseteq E$ and the corresponding random graph $\tilde{G} = (V, \tilde{E})$. The number of elements affected by editing v_j is given by the size of the connected component containing v_j in $\tilde{G}$, denoted as n_j. Using this, we define the Structural Independence Coefficient (SIC) of a figure I as:

$$\mathrm{SIC}(I) = 1 - \mathbb{E}_{\tilde{G}} \left[\frac{1}{N_\mathrm{v}} \sum_{v_j \in V} \frac{n_j - 1}{N_\mathrm{v}} \right]. \tag{2}$$

Here, $(n_j - 1)/N_\mathrm{v}$ is the proportion of elements affected when editing v_j. Therefore, SIC is the complement of expected edit propagation over the figure, taking values in $[0, 1]$, where higher values indicate more localized and independent edits.

Instantiating nodes V and edges E. To compute SIC in practice, we instantiate the abstract graph $G = (V, E)$ from a given figure I by defining its editing units V and edge weights $\{w_{jk}\}$ according to the representation of I, i.e., whether the figure is represented as vector graphics or raster graphics. (i) In vector graphics, visual elements such as text, shapes, and arrows are explicitly represented as structured objects. Each element is treated as an editing unit and mapped to a node in V. Edge weights are defined based on explicit structural relationships. Specifically, if two elements (v_j, v_k) share structural constraints (e.g., grouping, common transformations, or clipping), then $w_{jk} = 1$; otherwise, $w_{jk} = 0$. Raster graphics embedded in vector graphics are normally treated as atomic `<image>` nodes. As an exception, when such an image dominates the figure, we regard the entire figure as a raster-dominant representation and apply the raster formulation described below. (ii) In raster graphics, explicit structural elements are unavailable. We therefore approximate editing units as visually homogeneous regions obtained by Felzenszwalb segmentation [2], treating each region as a node in V. This is motivated by the observation that, in common painting tools, operations such as recoloring or background modification can be performed at the region level. Edge weights are defined as:

$$w_{jk} = \exp(-b_{jk}), \tag{3}$$

where b_{jk} denotes the boundary strength between adjacent regions. The boundary strength is computed as:

$$b_{jk} = \frac{1}{|B_{jk}|} \sum_{p \in B_{jk}} \frac{||\nabla I(p)||}{g_{\max}}, \tag{4}$$

where B_{jk} is the set of boundary pixels, $\nabla I(p)$ is the image gradient at pixel p computed using the Sobel operator, and $g_{\max} = 4\sqrt{2}$ is the theoretical maximum gradient magnitude. This formulation ensures that strong boundaries result in weak dependencies, thereby limiting edit propagation across regions.

2.3 GenGA Framework

We introduce GenGA, a framework for solving Editable GA Generation. GenGA takes as input a paper $T_{\text{full-text}}$ and optionally assets A (e.g., an input-output example), and generates an editable GA in SVG format I_{vector}, preserving editing availability that conventional methods lack. As illustrated in Figure 1, the framework consists of the following four phases: (1) Reference Retrieval: Given the input paper $T_{\text{full-text}}$, we retrieve GAs from existing papers that are semantically relevant and use them as visual references. (2) Reference Vectorization: The retrieved raster-format references are converted into structured vector representations, capturing layout and structural information while

abstracting away non-structural content. (3) Asset-aware Generation: Given $T_{\text{full-text}}$, assets A, and the references, we generate a GA in SVG format. (4) Self-correction Loop: The generated result is iteratively reviewed and refined to improve its quality.

Reference Retrieval. This phase retrieves a semantically relevant GA as a reference. Using a text-to-image retrieval model $f_{\text{retrieve}}(\cdot)$, we retrieve a reference GA as:

$$I_{\text{ref}}^{\text{raster}} = f_{\text{retrieve}}(T_{\text{full-text}}, R_{\text{raster}}), \tag{5}$$

where $T_{\text{full-text}}$ is the input paper and $R_{\text{raster}} = \{r_\ell \mid \ell \in \{1, 2, \ldots, N_r\}\}$ denote a set of candidate raster-format GAs. In the subsequent asset-aware generation phase, the retrieved reference provides domain-specific visual conventions.

Reference Vectorization. This phase converts the retrieved raster-format references $I_{\text{ref}}^{\text{raster}}$ into a structured vector representation $I_{\text{ref}}^{\text{vector}}$. This conversion is performed using a vision-language model (VLM) $f_{\text{vectorize}}(\cdot)$ as follows:

$$I_{\text{ref}}^{\text{vector}} = f_{\text{vectorize}}\left(I_{\text{ref}}^{\text{raster}}\right). \tag{6}$$

The vectorization model extracts structural components such as regions, arrows, and text blocks, and represents them as elements in SVG format. Importantly, non-structural elements such as photographs, icons, plots, and example inputs are replaced with simple placeholder elements (i.e., rectangular regions) that preserve only position and size without encoding visual content. This design avoids introducing unreliable geometric approximations of non-structural content, e.g., approximating photographic content using geometric primitives such as lines and shapes. In the subsequent asset-aware generation phase, the vectorized reference provides structural guidance for generating GAs in SVG format.

Asset-aware Generation. This phase generates a GA as a structured SVG while explicitly incorporating user-provided assets as non-generative constraints. An initial vector representation is generated using a VLM $f_{\text{generate}}(\cdot)$:

$$I_{\text{vector}}^{(0)} = f_{\text{generate}}\left(T_{\text{full-text}}, A, I_{\text{ref}}^{\text{raster}}, I_{\text{ref}}^{\text{vector}}\right). \tag{7}$$

Crucially, user-provided assets A are not treated as generative targets but as fixed structural constraints, and are represented as placeholder elements. Unlike prior approaches that generate all visual content end-to-end, GenGA treats scientific data as immutable inputs rather than generative targets, ensuring that experimentally derived content is neither hallucinated nor altered. As a result, the generated GA remains faithful to the original data while maintaining structural consistency and editability, enabling reliable data-grounded scientific visualization.

Self-correction Loop. This phase iteratively improves the generated GA through feedback-driven refinement. Starting from the initial result $I_{\text{vector}}^{(0)}$, the model performs iterative review and refinement. Let $I_{\text{vector}}^{(t)}$ denote the vector representation at iteration t. Since SVG is a code-based representation, visual evaluation is performed on its rasterized form:

$$I_{\text{raster}}^{(t)} = f_{\text{rasterize}}\left(I_{\text{vector}}^{(t)}\right). \tag{8}$$

Feedback is obtained via a VLM $f_{\text{review}}(\cdot)$:

$$T_{\text{feedback}}^{(t)} = f_{\text{review}}\left(T_{\text{full-text}}, I_{\text{raster}}^{(t)}\right). \tag{9}$$

The reviewer evaluates aspects such as semantic consistency, layout conflicts, and readability. The generator $f_{\text{generate}}(\cdot)$ then regenerates the output using both $I_{\text{vector}}^{(t)}$ and $T_{\text{feedback}}^{(t)}$:

$$I_{\text{vector}}^{(t+1)} = f_{\text{generate}}\left(T_{\text{full-text}}, A, I_{\text{vector}}^{(t)}, T_{\text{feedback}}^{(t)}\right). \tag{10}$$

This process is repeated for $t = 0, \ldots, N_t - 1$.

3. Experimental Setup

We conduct experiments on generating GAs from the abstract and introduction of papers in markdown format. We use the large-scale SciGA-145k dataset [9], which contains papers paired with GAs, and focus on the Computer Science domain, which has the largest number of samples. For evaluation, we use a test set of 2,053 papers. In the reference retrieval phase, we use an additional set of 16,416 GAs, disjoint from the evaluation set, as retrieval candidates. Assets are semi-automatically extracted from the original GAs using the layer decomposition model LayerD [15], captioned with Gemini-3.1-Pro-Preview [3], and manually verified by the authors.

Implementation Details. We use GPT-5.2 [12] and Gemini-3.1-Pro-Preview as backbone models for the generator and reviewer, respectively, and compare their performance. We employ Long-CLIP-4-Inter-GA-Rec [9] for reference retrieval and use Gemini-3.1-Pro-Preview for reference vectorization. In the self-correction loop, we perform four refinement iterations.

Baselines. We compare GenGA with author-created GAs, raster-based methods including NanoBanana-Pro [4] and PaperBanana [16], their post-hoc vectorized variants using Gemini-3.1-Pro-Preview, and AutoFigure [17], a hybrid raster-vector method that overlays editable elements on a raster background.

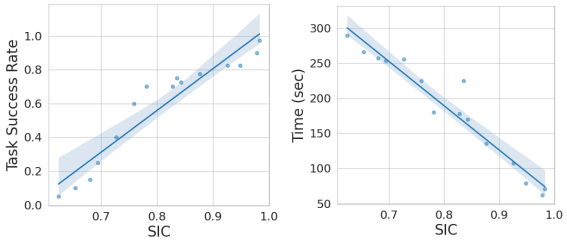
Evaluation Metrics. We evaluate Editable GA Generation using five complementary metrics: **(1) CLIP-S** [5] measures semantic alignment as the CLIPScore between the paper abstract and the generated GA. **(2) Overlap Ratio** measures layout clarity as the ratio of overlapping areas between SVG element bounding boxes to the total area, excluding parent–child overlaps. **(3) SIC** evaluates editing simplicity. **(4) Contribution QA Accuracy** evaluates whether the GA conveys the paper's main contribution: GPT-5.2 generates a ground-truth contribution statement from the full text, three hard negatives are sampled from papers with similar abstracts using Sentence-BERT [13], and GPT-5.2 and Gemini-3.1-Pro-Preview select the correct contribution statement from the four candidates given only the GA. **(5) VLM-as-a-Judge** follows prior work [16]: GPT-5.2 and Gemini-3.1-Pro-Preview compare each generated GA with the corresponding human-authored GA in faithfulness, conciseness, readability, and aesthetics, assigning 1.0, 0.5, or 0.0 for win, tie, or loss. Overall scores are determined hierarchically, prioritizing faithfulness and conciseness over readability and aesthetics.

User Study. To evaluate actual editing cost and validate SIC as its proxy, we conduct a user study with 15 professional machine learning researchers experienced in GA creation. Participants edit visually identical figures with different SIC values due to representation and structural complexity. In draw.io, they complete

Table 1: Quantitative comparison across methods. Our method achieves the highest editability (SIC) while maintaining strong semantic alignment, readability, and the ability to effectively convey key contributions. The best results for each metric are highlighted in **bold**.

Method	Output Format	CLIP-S (↑)	Overlap Ratio (↓)	SIC (↑)	Contribution QA Accuracy† (↑)	VLM-as-a-Judge†				
						Faithfulness (↑)	Conciseness (↑)	Readability (↑)	Aesthetics (↑)	Overall (↑)
Author-created GA	Raster	0.255	–	0.119	0.387 / 0.432	0.500 / 0.500	0.500 / 0.500	0.500 / 0.500	0.500 / 0.500	0.500 / 0.500
NanoBanana-Pro [4]	Raster	0.230	–	0.188	0.411 / 0.442	0.340 / 0.044	0.319 / 0.511	0.643 / 0.223	0.677 / 0.402	0.338 / 0.144
+ vectorize	Vector	0.222	0.169	0.771	0.387 / 0.426	0.307 / 0.071	0.232 / 0.464	0.307 / 0.126	0.209 / 0.142	0.170 / 0.083
PaperBanana [16]	Raster	0.250	–	0.184	0.419 / 0.462	0.375 / 0.070	0.321 / 0.537	0.660 / 0.272	**0.692 / 0.455**	0.388 / 0.194
+ vectorize	Vector	0.242	0.164	0.794	0.392 / 0.457	0.292 / 0.065	0.275 / 0.457	0.358 / 0.144	0.171 / 0.204	0.188 / 0.101
AutoFigure [17]	Vector	0.254	**0.000**	0.131	0.417 / 0.422	0.268 / 0.025	0.282 / 0.302	0.348 / 0.025	0.545 / 0.259	0.214 / 0.048
GenGA (ours)										
GPT-5.2 [12]	Vector	**0.284**	0.146	0.885	**0.427 / 0.478**	**0.531 / 0.274**	0.263 / 0.612	0.672 / 0.288	0.654 / 0.318	0.389 / 0.315
Gemini-3.1-Pro-Preview [3]	Vector	0.269	0.163	**0.943**	0.422 / 0.467	0.427 / 0.198	**0.473 / 0.690**	**0.683 / 0.392**	0.690 / 0.438	**0.473 / 0.362**

† Scores are reported using GPT-5.2 / Gemini-3.1-Pro-Preview as the judge.



(a) Task Success Rate (b) Editing Time

Fig. 2: Relationship between SIC and human editing performance. Higher SIC correlates with (a) a higher edit success rate and (b) shorter editing time.

predefined edits (e.g., text changes, visual adjustments, and image replacement) within 5 minutes. We record interaction logs, measure editing time with unfinished tasks right-censored at the limit, and compute edit success rates.

4. Results and Discussion

SIC and Human Editing Performance. Figure 2 summarizes the relationship between SIC and human editing performance. Higher SIC consistently leads to a higher fraction of successful edits within the time limit and lower editing time, with strong correlations observed for each metric (respectively: $r = 0.950$, $p < 10^{-7}$ and $r = -0.967$, $p < 10^{-8}$). We observe that users struggle with low-SIC figures due to strong interdependencies, which require cascading adjustments even for simple edits. This occurs in tightly coupled layouts, grouped elements, and ambiguous or overlapping regions that hinder localized modifications. For raster-based figures, participants sometimes resort to manually recreating elements from scratch and overlaying them on the original figure, where ambiguous boundaries further increase the required effort by forcing larger regions to be rebuilt. Overall, these results show that SIC effectively captures practical editing cost in real-world figure revision.

Quantitative Results. Table 1 summarizes the quantitative evaluation results of GAs generated by each method. Raster-based PaperBanana achieved the highest Aesthetics scores (GPT-5.2: 0.692, Gemini-3.1-Pro-Preview: 0.455), confirming its strong rendering capability. However, it exhibits low editability, as indicated by a low SIC score of 0.184. Applying post-hoc vectorization improves SIC to 0.794 but consistently degrades other metrics. This reflects the ill-posed nature of inferring structure (e.g., grouping and hierarchy) from pixel-based representations, leading to information loss and geometric inconsistencies that hinder both visual quality and editability. AutoFigure achieves a perfect Overlap Ratio of 0.000 because it fixes most of the figure as a single raster background and overlays limited vector elements such as text and icons. While this avoids spatial interference between elements, the underlying structure remains largely uneditable, resulting in a low SIC score of 0.131.

In contrast, GenGA directly generates all elements as a hierarchical vector structure, achieving a strong balance between visual quality and editability. In particular, GenGA with Gemini-3.1-Pro-Preview achieves a SIC score of 0.943, significantly outperforming all baselines. Although its Overlap Ratio (0.163) appears higher, this reflects intentional layout design (e.g., text placed within structured regions) rather than undesirable overlaps, and is consistent with maintaining structural coherence. GenGA also shows strong semantic alignment with the source paper. It surpasses human-authored GAs in CLIP-S (0.284 vs. 0.255), indicating effective content representation. Furthermore, it achieves the highest Contribution QA accuracy (0.478), outperforming both human-authored GAs and conventional methods. This suggests that directly grounding generation in textual reasoning enables GenGA to preserve key contributions without the information degradation introduced by intermediate image generation. In terms of overall quality, GenGA maintains competitive Aesthetics (0.438) while achieving high Conciseness (0.690) and Readability (0.683). Although it does not always outperform human-authored GAs in VLM-as-a-Judge evaluations, this gap is offset by its significantly higher editability, highlighting the importance of considering editability as a core evaluation dimension. Across backbone models, GPT-5.2 performs better in semantic alignment (CLIP-S, Faithfulness), while Gemini-3.1-Pro-Preview excels in information structuring (Conciseness, Readability). Overall, GenGA establishes a new paradigm for editable, real-data-grounded GA generation without sacrificing visual quality.

5. Conclusion

We introduced Editable GA Generation, reframing GA generation as a structural, vector-based problem for human editing, along with SIC, a metric that captures editing simplicity and correlates with real editing cost. We further proposed GenGA, a vector-first, data-grounded framework that generates GAs with superior editability and quality. This work establishes a foundation for GA generation grounded in real research workflows and promotes more effective scientific communication.

References

[1] Bennett, H. and Slattery, F.: Graphical abstracts are associated with greater Altmetric attention scores, but not citations, in sport science, *Scientometrics*, Vol. 128, pp. 3793–3804 (2023).

[2] Felzenszwalb, P. F. and Huttenlocher, D. P.: Efficient Graph-Based Image Segmentation, *Int. J. Comput. Vis.*, Vol. 59, No. 2, pp. 167–181 (2004).

[3] Google: Gemini-3 (2025). `https://ai.google.dev/gemini-api/docs/gemini-3`.

[4] Google: NanoBanana-Pro (2025). `https://ai.google.dev/gemini-api/docs/image-generation`.

[5] Hessel, J., Holtzman, A., Forbes, M., Bras, R. L. and Choi, Y.: CLIPScore: A Reference-free Evaluation Metric for Image Captioning, *EMNLP* (2021).

[6] Huang, S., Gao, Y., Bai, J., Zhou, Y., Yin, Z., Liu, X., Chellappa, R., Lau, C. P., Nag, S., Peng, C. et al.: SciFig: Towards Automating Scientific Figure Generation, *arXiv preprint arXiv:2601.04390* (2026).

[7] Ibrahim, A. M., Lillemoe, K. D., Klingensmith, M. E. and Dimick, J. B.: Visual Abstracts to Disseminate Research on Social Media A Prospective, Case-control Crossover Study, *Annals of Surgery*, Vol. 266, No. 6, pp. 46–48 (2017).

[8] Jeyaraman, M. and Vaishya, R.: Attract readers with a graphical abstract – The latest clickbait. Journal of Orthopaedics, *Journal of Orthopaedics*, Vol. 38, No. 1, pp. 30–31 (2023).

[9] Kawada, T., Kitada, S., Nemoto, S. and Iyatomi, H.: SciGA: A Comprehensive Dataset for Designing Graphical Abstracts in Academic Papers, *Findings of CVPR* (2026).

[10] Kim, Y., Lee, J.-E., Yoo, J.-J., Jung, E.-A., Kim, S. G. and Kim, Y. S.: Seeing Is Believing: The Effect of Graphical Abstracts on Citations and Social Media Exposure in Gastroenterology & Hepatology Journals, *Journal of Korean Medical Science*, Vol. 37 (2022).

[11] Lee, J. and Yoo, J.-J.: The current state of graphical abstracts and how to create good graphical abstracts, *Science Editing*, Vol. 10, No. 1, pp. 19–26 (2023).

[12] OpenAI: GPT-5 (2025). `https://platform.openai.com/docs/models/gpt-5`.

[13] Reimers, N. and Gurevych, I.: Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, *EMNLP* (2019).

[14] Rodriguez, J. A., Vazquez, D., Laradji, I., Pedersoli, M. and Rodriguez, P.: FigGen: Text to Scientific Figure Generation, *ICLR* (2023).

[15] Suzuki, T., Liu, K.-J., Inoue, N. and Yamaguchi, K.: LayerD: Decomposing Raster Graphic Designs into Layers, *ICCV* (2025).

[16] Zhu, D., Meng, R., Song, Y., Wei, X., Li, S., Pfister, T. and Yoon, J.: PaperBanana: Automating Academic Illustration for AI Scientists, *arXiv preprint arXiv:2601.23265* (2026).

[17] Zhu, M., Lin, Z., Weng, Y., Lu, P., Xie, Q., Wei, Y., Liu, S., Sun, Q. and Zhang, Y.: AutoFigure: Generating and Refining Publication-Ready Scientific Illustrations, *ICLR* (2026).