The 28th Meeting on Image Recognition and Understanding

VASCAR: Content-Aware Layout Generation via Visual-Aware Self-Correction

Jiahao ZHANG[1,a)] Ryota YOSHIHASHI[2,b)] Shunsuke KITADA[2]
Atsuki OSANAI[2] Yuta NAKASHIMA[1]

Abstract

Large language models (LLMs) have proven effective for layout generation due to their ability to produce structured description, such as HTML. In this paper, we argue that their limitation in visual understanding leads to insufficient performance in tasks requiring visual content, e.g., content-aware layout generation. Therefore, we explore whether large vision-language models (LVLMs) can be applied to content-aware layout generation and propose the training-free ***V**isual-**A**ware **S**elf-**C**orrection L**A**yout Gene**R**ation* (**VASCAR**), taking inspiration from the iterative revision of designers. VASCAR enables LVLMs (e.g., GPT-4o and Gemini) iteratively refine their outputs with reference to layout rendered layout images. Extensive experiments and user study demonstrate VASCAR's effectiveness and versatility, achieving state-of-the-art (SOTA) layout generation quality.

1. Introduction

Content-aware layout generation involves creating layout designs (e.g., posters [4], [7] and magazines [9], [31]) based on given visual content [16], [25]. While various approaches [2], [6], [7], [11], [35] have been proposed to automate this process, it still remains difficult due to the scarcity of high-quality annotated datasets. Also, they often lack human-like iterative refinement and typically require manual adjustments to meet design standards [8], [15].

Recent advances in large language models (LLMs) have demonstrated strong capabilities in layout generation [24], [28]. Proprietary LLMs can generate layouts via in-context learning (ICL)[1], achieving high performance with limited annotated samples and without fine-tuning[14]. However, challenges remain: (1) effectively incorporating visual content to enhance generation and (2) ensuring aesthetic quality in both subjective and objective evaluations. Inspired by large vision-language models (LVLMs) that can assess design quality [5], and by the iterative revision and heuristic-guided processes used by human designers [13], [18], [19], the proposed framework, VASCAR (Visual-Aware Self-Correction Layout Generation), aims to mimic such workflows in a multi-modal and training-free manner.

VASCAR introduces an iterative, self-refining layout generation process where rendered layouts act as visual prompts [30], [32]. Using a canvas as a query and a few multi-modal ICL examples, LVLMs generate HTML-formatted layout candidates. These are evaluated through a layout scorer and refined based on suggestions derived from design heuristics [29], enabling autonomous improvement through multiple iterations [33]. Experiments on two datasets [7], [35] show that VASCAR surpasses both generative and LLM-based methods in content-based metrics and achieves competitive layout FID [10], validating its effectiveness in content-aware layout generation.

2. Method

Figure 1 shows an overview of VASCAR. To achieve iterative self-corrective layout generation with an LVLM, our pipeline is based of the novel *visual-aware self correction*, which consists our key contribution. In each step, LVLM-based layout generation and refinement are performed. VASCAR consists of *layout generator* to generate the initial layouts, *prompt optimizer* to evaluate the layouts based on automatic criteria and construct the next-step prompts, and *ICL retriever* to identify multiple exemplars that serve as potential design constraints.

Task Definition Let $\boldsymbol{x}_\mathrm{q}$ denote the input canvas on which we generate a layout, and $\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}$ the dataset of pairs of a canvas $\boldsymbol{x}$ and corresponding annotated *layout* $\boldsymbol{y}$, where $|\mathcal{D}| = N$. The canvas $\boldsymbol{x}$ may be an image of a product but without any visual layout elements. A layout is a set of annotations where to put visual elements in a canvas, i.e., $\boldsymbol{y} = \{\boldsymbol{e}\}$ where $|\boldsymbol{y}| = K$ and each visual element $\boldsymbol{e} = (c, l, t, w, h)$ is represented by its *element type* c(e.g., logo, text, and underlay) and the bounding box. VASCAR generates a layout $\boldsymbol{y}_\mathrm{q}$ for $\boldsymbol{x}_\mathrm{q}$ with referring to a subset of $\mathcal{D}$ identified by the ICL retriever.

2.1 Visual-Aware Self-Correction

Visual-aware self-correction refers to our overall pipeline that first generates layouts on the query canvas, and refines the layouts iteratively with visual feedback. First, we introduce a unified notation for prompt-based generation and refinement with an LLM or LVLM by $Y_\mathrm{q} = G(\boldsymbol{x}_\mathrm{q}, Y; p)$, where $G(\cdot)$ represents the generator, $\boldsymbol{x}_\mathrm{q}$ denotes the input

[1] Osaka University
[2] LY Corporation
[a)] {jiahao@is./n-yuta@}ids.osaka-u.ac.jp
[b)] {ryoshiha/s.kitada/atsuki.osanai}@lycorp.co.jp

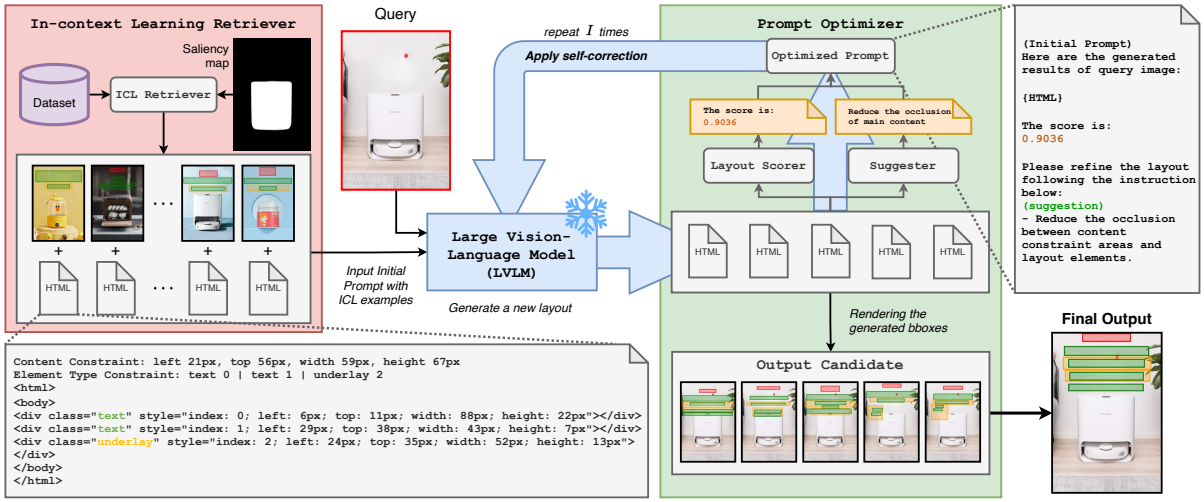

Fig. 1 An overview of VASCAR, an LVLM-based self-correction pipeline.

canvas, Y denotes the current state of layouts under refinement if provided, and p denotes a prompt text that instructs the model to generate or refine layouts. The output Y_{q} is a set of layouts each of whose elements is denoted by $\boldsymbol{y}_{\mathrm{q}}$, since a generator may output multiple candidates of layouts per call. In the initial layout generation, p_0, the default prompt instructs the model to generate layouts from scratch. To iteratively refine the layouts with self-feedback, the same LLM or LVLM used for generation can be employed by simply switching the prompt to a refinement mode as

$$Y_{\mathrm{q}}^0 = G(\boldsymbol{x}_{\mathrm{q}}, \emptyset; p_0) \quad (1)$$

$$Y_{\mathrm{q}}^i = G(\boldsymbol{x}_{\mathrm{q}}, Y_{\mathrm{q}}^{i-1}; p(Y_{\mathrm{q}}^{i-1})) \qquad (i = 1, 2, 3, ..., I), \quad (2)$$

where Y_{q}^i denotes layouts generated in the i-th iteration. The prompt $p(Y_{\mathrm{q}}^{i-1})$ is a refinement prompt that is conditioned by the previous state Y_{q}^{i-1}. To indicate the absence of the previous-step layouts, we explicitly use an empty set $\emptyset$. We construct refinement prompts by modifying the default prompt with additional instructions and the previous states of the generated layouts.

In constructing the function G with an LVLM, we need to leverage its ability to access layout aesthetics by directly feeding a rough proxy of the resulting design with visual elements being given as their bounding boxes [5]. We propose to feed these proxies, referred to as *rendered images*, to the LVLM to provide richer cues for better layouts. A rendered image is generated by placing bounding boxes of each element type with different colors, denoted by $\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{y})$. Most LVLMs accept a text prompt combined with multiple images, and we can concretize the function G using an LVLM with a capacity for visual feedback by

$$G(\boldsymbol{x}_{\mathrm{q}}, Y; p) = \mathrm{LVLM}(\{\boldsymbol{z}(\boldsymbol{x}_{\mathrm{q}}, \boldsymbol{y}_{\mathrm{q}}) | \boldsymbol{y}_{\mathrm{q}} \in Y\}, p), \quad (3)$$

where $\mathrm{LVLM}(V, t)$ denotes an LVLM viewed as a function that maps a pair of image set V and prompt text t on the output layout candidates.

Positive reinforcement in candidate-set feedback has been shown to be beneficial [29] by keeping high-score examples for better results. Therefore, we select the top-5 layout from candidate pool using a scoring function $v(\boldsymbol{y}_{\mathrm{q}})$ from Y_{q} and discard the others to create the reference samples for the multi-modal optimized prompt. The initial prompt is also added into the multi-modal optimized prompt, as the LVLM needs to generate a parsable HTML layout. We perform self-correction for I iterations by modifying the samples in the multi-modal optimized prompt, mimicking a designer ' s iterative process, and select the layout with the highest score among Y_{q}^I as the final output.

2.2 Prompt Strategy for Layout Generator

We construct our layout generator by a frozen LVLM enhanced with ICL [17], [23], [26], [34], incorporating a small subset of the training set into the prompts. We select samples with saliency maps [20], [21] similar to that of the query following [14]. Given an input $\boldsymbol{x}_{\mathrm{q}}$, we extract a small subset from the training set, denoted as $\mathcal{S}(\boldsymbol{x}_{\mathrm{q}})$ whose size is M. These examples serve as ICL exemplars, allowing the layout generator to generate a layout $\boldsymbol{y}_{\mathrm{q}}$ w.r.t. $\boldsymbol{x}_{\mathrm{q}}$ and $\mathcal{S}(\boldsymbol{x}_{\mathrm{q}})$. The prompt p_0 consists of (i) instructions for generating a layout, (ii) the input image $\boldsymbol{x}_{\mathrm{q}}$, (iii) its saliency map $\boldsymbol{s}_{\mathrm{q}}$, (iv) rendered image $\boldsymbol{z}$ of each ICL example in $\mathcal{S}$, (v) saliency map $\boldsymbol{s}$ of each ICL example, and (vi) the corresponding layout $\boldsymbol{y}$. Following [14], the saliency map is represented as a bounding box. Taking $\boldsymbol{s}_{\mathrm{q}}$ as an example, it can be represented by a serialized bounding box $(l_{\mathrm{q}}, t_{\mathrm{q}}, w_{\mathrm{q}}, h_{\mathrm{q}})$, and is encoded into a textual format through the transformation E_{s}, i.e., $E_{\mathrm{s}}(s_{\mathrm{q}}) =$ "`left` l_{q} `px, top` t_{q} `px, width` w_{q} `px, height` h_{q} `px.`". With this default initial prompt p_0, the LVLM generates multiple layouts Y_{q}^0, where different layouts can be generated by properly setting the temperature of the LVLM. Additionally, we reuse the rendered images defined in eq. (3) for ICL samples to exploit LVLMs' multi-modal ability. All rendered images from $\mathcal{S}(\boldsymbol{x}_{\mathrm{q}})$ and the input image $\boldsymbol{x}_{\mathrm{q}}$ are fed into the LVLM alongside the rendered images for visual feedback. In self-correction, refinement prompts $p(Y_{\mathrm{q}})$ follow the same structure as the initial prompt but differ in that they include the previous layout candidates, combined with textual *refinement suggestion* as described in section 2.3, in addition to

the ICL examples. The examples of real prompts can be found in Appendix.

2.3 Multi-modal Prompt Optimizer

LVLMs' capability to access the suitability and aesthetics of generated layout is strong but still limited compared to fine-tuned models [24]. To complement this, we propose a multi-modal prompt optimizer inspired by [29], and designers who iteratively create appropriate layouts for specific contexts and refine them based on guidelines [3], [13]. Our optimizer consists of a *layout scorer* to evaluate the generated layout and a *suggester* to guide the LVLM in making effective adjustments.

The layout scorer uses multiple layout evaluation metrics (detailed in section 3.1). Each metric is normalized to [0, 1], where the larger value means better. Let $\mathcal{M}$ be the set of the criteria, and $f_m(\boldsymbol{x}_\mathrm{q})$ gives the normalized score of $\boldsymbol{y}_\mathrm{q}$ for criterion $m \in \mathcal{M}$. The fused score is given by:

$$v(\boldsymbol{y}_\mathrm{q}) = \sum_{m \in \mathcal{M}} \lambda_m \cdot f_m(\boldsymbol{y}_\mathrm{q}), \tag{4}$$

where $\lambda_m \in \lambda$ is an empirically determined weight.

The fused score $v(\boldsymbol{y}_\mathrm{q})$ informs the layout generator of the quality $\boldsymbol{y}_\mathrm{q}$, but it does not tell which aspects are good and which are not. Therefore, the suggester indicates this w.r.t. the individual metrics f_m. If it falls below a preset threshold, an additional instruction text is added, indicating the adjustment direction (e.g., "*Reduce the overlap.*"). The threshold θ_m is set to the average of each evaluation metric across the ICL examples in $\mathcal{S}(s_\mathrm{q})$, i.e.,

$$\theta_m = \frac{1}{|\mathcal{S}(\boldsymbol{x}_\mathrm{q})|} \sum_{\boldsymbol{y} \in \mathcal{S}(\boldsymbol{x}_\mathrm{q})} f_m(\boldsymbol{y}). \tag{5}$$

The additional prompts from the layout scorer and suggester are then appended to the instruction in the refinement prompt $p(Y_\mathrm{q}^i)$.

3. Experiments

3.1 Experimental Setup

Datasets. We use two e-commerce poster layout datasets: PKU [7] and CGL [35], which contain visual elements like *logo*, *text*, *underlay*, and *embellishment* (CGL only). PKU has 9,974 annotated and 905 unannotated posters, while CGL contains 60,548 annotated and 1,000 unannotated posters. Following prior work [6], [7], [35], we use an inpainting model [27] to create image-layout pairs and adopt RALF ' s split [6] for fair evaluation. ICL examples are retrieved from training data only. We compare VASCAR with task-specific models: CGL-GAN [35], RALF [6]; LLM-based methods: LayoutPrompter [14], PosterLlama [24]; and Real Data (ground-truth). LayoutPrompter and PosterLlama were reproduced under our setup for fair comparison.

Evaluation Metrics. Following prior works [6], [7], we adopt metrics from two perspectives: Content metrics evaluate the harmony between the generated layout $\boldsymbol{y}$ and image $\boldsymbol{x}$: Occlusion (Occ ↓): overlap between main visual ($\boldsymbol{s}_\mathrm{q}$) and layout elements. Unreadability (Rea ↓): spatial gradient within text regions. Graphic metrics assess layout structure regardless of visual content: Overlay (Ove ↓): average IoU across elements (excluding underlay). Non-alignment (Align ↓): spatial misalignment [12]. Underlay Effectiveness (Und ↑): ratio of valid underlay coverage. FID (Fréchet Inception Distance ↓): layout distribution similarity [10].

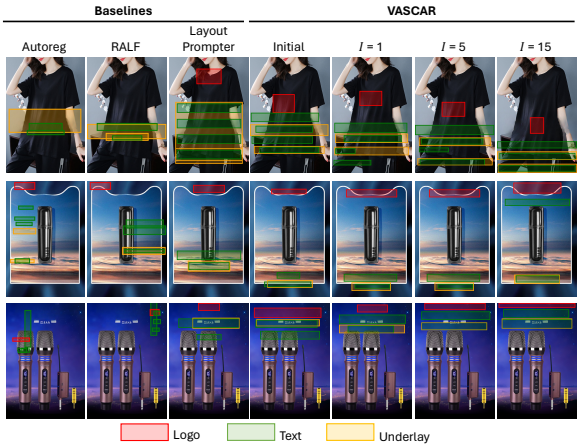

Fig. 2 Visual comparison of baselines and VASCAR with different values of I.

Implementation Details. LVLMs include `gpt-4o` and `gemini-1.5-flash`. We use 10 ICL examples, generate 5 layout candidates, and apply self-correction for 15 (Gemini) and 5 (GPT-4o) iterations. We set the evaluation criteria for the prompt optimizer as $\mathcal{M} = \{\mathrm{Occ}, \mathrm{Rea}, \mathrm{Ove}, \mathrm{Align}, \mathrm{Und}\}$, and the weights $\lambda = \{0.4, 0.4, 0.1, 0, 0.1\}$.

3.2 Input-unconstrained Generation

We follow the experimental setup of RALF [6] to evaluate VASCAR on the input-unconstrained generation task (table 1). On PKU, VASCAR outperforms all baselines across nearly all metrics, even surpassing fine-tuned models. In CGL, it also achieves competitive results on Rea, Ove, and Und. However, we observe a degradation in FID, which we attribute to varying self-correction needs across samples —simple cases converge early, while excessive corrections in complex cases may lead to overfitting to the suggester's feedback, diverging from the true layout distribution.

Qualitative Comparison. Figure 2 shows the visual comparison of several baselines including Autoreg, RALF, and LayoutPrompter vs. our VASCAR across different self-correction I including Initial i.e.$I = 0$. We confirm that VASCAR can generate well-arranged layouts avoiding overlapping and occluding the main content. VASCAR's results improve progressively with iterations.

Trends of each metric. As shown in fig. 3, all metrics improve as the number of self-correction iterations increases, except for FID.

Human evaluation. To further demonstrate the effectiveness of VASCAR, we conducted a user study on the PKU test split. comparing it against strong baselines: RALF [6], LayoutPrompter [14], and Initial. We invited 15 participants to evaluate 30 groups, selecting one preferred choice from each group. The user study results, based on voting percentages, are shown in fig. 4. VASCAR outperforms other baselines significantly according to human evaluation.

Method	*Training-free*	PKU						CGL					
		Content		Graphic				Content		Graphic			
		Occ ↓	Rea ↓	Align ↓	Und ↑	Ove ↓	FID↓	Occ ↓	Rea ↓	Align ↓	Und ↑	Ove ↓	FID↓
Real Data	-	0.112	0.0102	0.0038	0.99	0.0009	1.58	0.125	0.0170	0.0024	0.98	0.0002	0.79
CGL-GAN [35]	✗	0.138	0.0164	0.0031	0.41	0.0740	34.51	0.157	0.0237	0.0032	0.29	0.1610	66.75
RALF [6]	✗	0.119	0.0128	0.0027	0.92	0.0080	3.45	0.125	0.0180	0.0024	0.98	0.0040	**1.32**
PosterLlama† [24]	✗	–	–	–	–	–	–	0.154	0.0135	0.0008	0.97	0.0030	2.21
LayoutPrompter† [14]	✓	0.220	0.0169	0.0006	0.91	0.0003	3.42	0.251	0.0179	**0.0004**	0.89	**0.0002**	4.59
VASCAR (GPT-4o)	✓	0.129	**0.0091**	**0.0002**	**0.99**	**0.0002**	**3.14**	0.141	**0.0102**	0.0005	**0.99**	**0.0002**	5.69
VASCAR (Gemini)	✓	**0.113**	0.0117	0.0013	0.98	0.0003	3.34	0.125	0.0122	0.0010	0.98	0.0007	6.27

Table 1 Unconstrained generation results on the PKU and CGL test split. † shows our reproduced results based on the RALF split [6].

Method	PKU					CGL				
	Content		Graphic			Content		Graphic		
	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓
Real Data	0.112	0.0102	0.99	0.0009	1.58	0.125	0.0170	0.98	0.0002	0.79
C → S + P										
RALF	0.124	0.0138	0.90	0.0100	**2.21**	0.126	0.0180	0.97	0.0060	**0.50**
VASCAR (GPT-4o)	0.117	**0.0094**	**1.00**	**0.0002**	3.01	0.139	**0.0099**	**0.99**	**0.0002**	4.86
VASCAR (Gemini)	**0.107**	0.0100	0.99	0.0003	2.82	**0.123**	0.0111	0.98	0.0005	5.51
C + S → P										
RALF	0.125	0.0138	0.87	0.0100	**0.62**	0.128	0.0185	**0.96**	0.0060	**0.21**
VASCAR (GPT-4o)	0.123	0.0117	**0.90**	**0.0009**	1.11	0.140	0.0132	0.88	**0.0003**	1.67
VASCAR (Gemini)	**0.104**	**0.0107**	0.88	0.0018	2.21	**0.122**	**0.0123**	0.85	0.0028	2.39
Completion										
RALF	0.120	0.0140	0.88	0.0120	**1.58**	0.126	0.0185	0.96	0.0050	**1.04**
VASCAR (GPT-4o)	0.120	**0.0063**	**1.00**	**0.0004**	6.19	0.137	**0.0068**	**0.99**	**0.0005**	5.57
VASCAR (Gemini)	**0.119**	0.0097	0.99	0.0006	4.74	0.135	0.0098	0.98	0.0009	5.18
Refinement										
RALF	0.113	0.0109	0.95	0.0040	**0.13**	0.126	0.0176	**0.98**	0.0020	**0.14**
VASCAR (GPT-4o)	0.108	**0.0072**	**0.99**	**0.0005**	1.18	0.125	**0.0091**	0.95	**0.0008**	0.59
VASCAR (Gemini)	**0.104**	0.0095	0.97	0.0010	0.32	**0.114**	0.0096	0.96	0.0013	0.37
Relationship										
RALF	0.122	0.0141	0.85	0.0090	2.23	**0.126**	0.0184	**0.95**	0.0060	**0.55**
VASCAR (GPT-4o)	0.151	0.0139	0.92	0.0011	**1.94**	0.153	0.0139	0.91	**0.0004**	2.61
VASCAR (Gemini)	**0.119**	**0.0117**	**0.96**	**0.0008**	2.00	0.132	**0.0123**	**0.95**	0.0011	3.72

Table 2 Quantitative result of five constrained generation tasks on the PKU and CGL test splits.

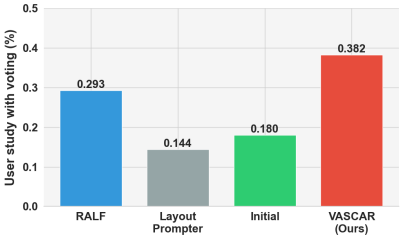

Fig. 4 User study results on PKU test split.

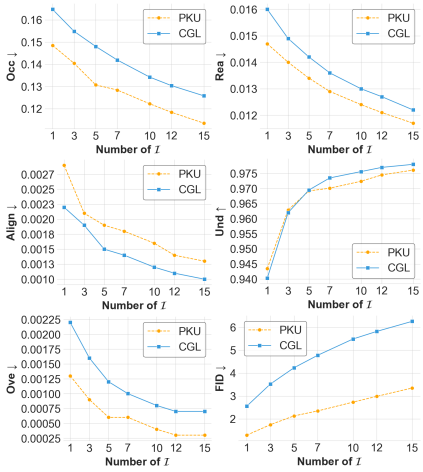

Fig. 3 The trends of each metric based on I.

Setting	Content		Graphic			
	Occ ↓	Rea ↓	Align ↓	Und ↑	Ove ↓	FID↓
Rendered Image (Ours)	**0.1304**	**0.0134**	**0.0017**	0.97	0.0012	2.13
Text-only	0.1529	0.0153	0.0024	0.97	0.0009	1.91
Saliency Map	0.1356	0.0140	0.0023	0.97	0.0009	**1.69**
Inpainting Image	0.1312	0.0137	0.0022	**0.98**	**0.0004**	2.37
Original Poster	0.1315	**0.0134**	0.0023	**0.98**	0.0006	2.34

Table 3 Multi-modal analysis for VASCAR on PKU.

3.3 Input-constrained Generation

We evaluated VASCAR on the five input-constrained generation tasks followed by [6]: **Category → Size + Position** (C → S + P) task specifies the category and the number of visual elements to place. **Category + Size → Position** (C + S→P) specifies the size of elements. **Completion** refers to a task where the model is given all information about partial element (including category, size, and position) and asked to complete the layout. **Refinement** represents the task of improving a perturbed layout, which is generated by adding Gaussian noise of mean 0 and variance 1 to the ground-truth layout [22]. **Relationship** specifies the positional and size relationships between elements. We adopt a text prompt preprocessing strategy inspired by [14], enabling VASCAR to effectively address content-aware layout generation tasks. VASCAR achieves strong performance across various input-constrained generation tasks (table 2). Although VASCAR shows slightly higher FID scores than RALF in some tasks, this is likely due to the layout scorer's weighting λ_m, which emphasizes Occ and Rea, potentially increasing the distributional distance from the ground-truth layouts. Nonetheless, VASCAR consistently outperforms others in content-related metrics, reinforcing its strength in generating appealing designs.

3.4 Ablation Study

The impact of multi-modal input. We argue that the visual content determines the performance of layout generation model. Therefore, we compared using only text input and different types of image input (the same image type was also used in self-correction). Table 3 shows that using only text input shows a significant gap in content metrics compared to multi-modal input. Regarding different image types, rendered images achieved the best performance in content metrics and comparable performance in graphic metrics, demonstrating that rendered images enhance LVLMs' ability to generate layouts effectively.

4. Conclusion

This paper introduces VASCAR, a novel *training-free* framework for content-aware layout generation that utilizes LVLMs through a visual-aware self-correction mechanism like a designer. The method iteratively refines layouts by incorporating feedback from both rendered visual contents and automatic metrics, improving layout quality without additional training. Extensive experiments and user study demonstrate that VASCAR excels in generating high-quality layouts and adapts well to different datasets and different LVLMs, offering a versatile solution for content-aware layout generation tasks.

References

[1] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I. and Amodei, D.: Language Models are Few-Shot Learners, *Advances in Neural Information Processing Systems*, Vol. 33, pp. 1877–1901 (2020).

[2] Chai, S., Zhuang, L., Yan, F. and Zhou, Z.: Two-stage Content-Aware Layout Generation for Poster Designs, *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 8415–8423 (2023).

[3] Duan, P., Warner, J., Li, Y. and Hartmann, B.: Generating automatic feedback on ui mockups with large language models, *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pp. 1–20 (2024).

[4] Guo, S., Jin, Z., Sun, F., Li, J., Li, Z., Shi, Y. and Cao, N.: Vinci: An Intelligent Graphic Design System for Generating Advertising Posters, *Proceedings of the 2021 CHI conference on human factors in computing systems*, pp. 1–17 (2021).

[5] Haraguchi, D., Inoue, N., Shimoda, W., Mitani, H., Uchida, S. and Yamaguchi, K.: Can GPTs Evaluate Graphic Design Based on Design Principles?, *SIGGRAPH Asia 2024 Technical Communications* (2024).

[6] Horita, D., Inoue, N., Kikuchi, K., Yamaguchi, K. and Aizawa, K.: Retrieval-Augmented Layout Transformer for Content-Aware Layout Generation, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 67–76 (2024).

[7] Hsu, H. Y., He, X., Peng, Y., Kong, H. and Zhang, Q.: Posterlayout: A new benchmark and approach for content-aware visual-textual presentation layout, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6018–6026 (2023).

[8] Iwai, S., Osanai, A., Kitada, S. and Omachi, S.: Layout-Corrector: Alleviating Layout Sticking Phenomenon in Discrete Diffusion Model, *European Conference on Computer Vision*, Springer, pp. 92–110 (2024).

[9] Jahanian, A., Liu, J., Lin, Q., Tretter, D., O'Brien-Strain, E., Lee, S. C., Lyons, N. and Allebach, J.: Recommendation system for automatic design of magazine covers, *Proceedings of the 2013 international conference on Intelligent user interfaces*, pp. 95–106 (2013).

[10] Kikuchi, K., Simo-Serra, E., Otani, M. and Yamaguchi, K.: Constrained graphic layout generation via latent optimization, *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 88–96 (2021).

[11] Li, F., Liu, A., Feng, W., Zhu, H., Li, Y., Zhang, Z., Lv, J., Zhu, X., Shen, J., Lin, Z. et al.: Relation-aware diffusion model for controllable poster layout generation, *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 1249–1258 (2023).

[12] Li, J., Yang, J., Zhang, J., Liu, C., Wang, C. and Xu, T.: Attribute-conditioned layout gan for automatic graphic design, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 27, No. 10, pp. 4039–4048 (2020).

[13] Li, T., Cheng, C.-Y., Xie, A., Li, G. and Li, Y.: Revision Matters: Generative Design Guided by Revision Edits, *arXiv preprint arXiv:2406.18559* (2024).

[14] Lin, J., Guo, J., Sun, S., Yang, Z., Lou, J.-G. and Zhang, D.: Layoutprompter: Awaken the design ability of large language models, *Advances in Neural Information Processing Systems*, Vol. 36 (2024).

[15] Lin, J., Huang, D., Zhao, T., Zhan, D. and Lin, C.-Y.: Spot the error: Non-autoregressive graphic layout generation with wireframe locator, *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38, No. 4, pp. 3413–3421 (2024).

[16] Lok, S. and Feiner, S.: A Survey of Automated Layout Techniques for Information Presentations, *Proceedings of SmartGraphics*, Vol. 2001, pp. 61–68 (2001).

[17] Lu, Y., Bartolo, M., Moore, A., Riedel, S. and Stenetorp, P.: Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity, *arXiv preprint arXiv:2104.08786* (2021).

[18] Nielsen, J.: Finding usability problems through heuristic evaluation, *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 373–380 (1992).

[19] Nielsen, J. and Molich, R.: Heuristic evaluation of user interfaces, *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 249–256 (1990).

[20] Qin, X., Dai, H., Hu, X., Fan, D.-P., Shao, L. and Van Gool, L.: Highly accurate dichotomous image segmentation, *European Conference on Computer Vision*, Springer, pp. 38–56 (2022).

[21] Qin, X., Zhang, Z., Huang, C., Gao, C., Dehghan, M. and Jagersand, M.: Basnet: Boundary-aware salient object detection, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7479–7489 (2019).

[22] Rahman, S., Sermuga Pandian, V. P. and Jarke, M.: Ruite: Refining ui layout aesthetics using transformer encoder, *Companion Proceedings of the 26th International Conference on Intelligent User Interfaces*, pp. 81–83 (2021).

[23] Razeghi, Y., Logan IV, R. L., Gardner, M. and Singh, S.: Impact of pretraining term frequencies on few-shot reasoning, *arXiv preprint arXiv:2202.07206* (2022).

[24] Seol, J., Kim, S. and Yoo, J.: PosterLlama: Bridging Design Ability of Language Model to Content-Aware Layout Generation, *European Conference on Computer Vision*, pp. 451–468 (2024).

[25] Shi, Y., Shang, M. and Qi, Z.: Intelligent layout generation based on deep generative models: A comprehensive survey, *Information Fusion*, p. 101940 (2023).

[26] Shin, S., Lee, S.-W., Ahn, H., Kim, S., Kim, H., Kim, B., Cho, K., Lee, G., Park, W., Ha, J.-W. et al.: On the effect of pretraining corpora on in-context learning by a large-scale language model, *arXiv preprint arXiv:2204.13509* (2022).

[27] Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., Kong, N., Goka, H., Park, K. and Lempitsky, V.: Resolution-robust large mask inpainting with fourier convolutions, *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2149–2159 (2022).

[28] Tang, Z., Wu, C., Li, J. and Duan, N.: LayoutNUWA: Revealing the Hidden Layout Expertise of Large Language Models, *The Twelfth International Conference on Learning Representations* (2024).

[29] Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D. and Chen, X.: Large Language Models as Optimizers, *The Twelfth International Conference on Learning Representations* (2024).

[30] Yang, J., Zhang, H., Li, F., Zou, X., Li, C. and Gao, J.: Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v, *arXiv preprint arXiv:2310.11441* (2023).

[31] Yang, X., Mei, T., Xu, Y.-Q., Rui, Y. and Li, S.: Automatic generation of visual-textual presentation layout, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, Vol. 12, No. 2, pp. 1–22 (2016).

[32] Yang, Z., Li, L., Lin, K., Wang, J., Lin, C.-C., Liu, Z. and Wang, L.: The dawn of lmms: Preliminary explorations with gpt-4v (ision), *arXiv preprint arXiv:2309.17421*, Vol. 9, No. 1, p. 1 (2023).

[33] Yang, Z., Wang, J., Li, L., Lin, K., Lin, C.-C., Liu, Z. and Wang, L.: Idea2Img: Iterative Self-refinement with GPT-4V for Automatic Image Design and Generation, *European Conference on Computer Vision*, Springer, pp. 167–184 (2024).

[34] Zhang, J., Wang, B., Li, L., Nakashima, Y. and Nagahara, H.: Instruct me more! random prompting for visual in-context learning, *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2597–2606 (2024).

[35] Zhou, M., Xu, C., Ma, Y., Ge, T., Jiang, Y. and Xu, W.: Composition-aware Graphic Layout GAN for Visual-Textual Presentation Designs, *IJCAI* (Raedt, L. D., ed.), International Joint Conferences on Artificial Intelligence Organization, pp. 4995–5001 (2022). AI and Arts.