Graphic Design with Large Multimodal Model

Yutao Cheng[1*], Zhao Zhang[1*], Maoke Yang[1*], Hui Nie[1 2], Chunyuan Li[1], Xinglong Wu[1], Jie Shao[1]

[1] ByteDance Inc.
[2] Institute of Computing Technology, Chinese Academy of Sciences
{yutao.135,yangmaoke,shaojie.mail}@bytedance.com
zzhang@mail.nankai.edu.cn

Abstract

In the field of graphic design, automating the integration of design elements into a cohesive multi-layered artwork not only boosts productivity but also paves the way for the democratization of graphic design. One existing practice is Graphic Layout Generation (GLG), which aims to layout sequential design elements. It has been constrained by the necessity for a predefined correct sequence of layers, thus limiting creative potential and increasing user workload. In this paper, we present Hierarchical Layout Generation (HLG) as a more flexible and pragmatic setup, which creates graphic composition from ***any-ordered*** sets of design elements. To tackle the HLG task, we introduce CreatiGraphist, the first layout generation model based on large multimodal models. CreatiGraphist efficiently reframes the HLG as a sequence generation problem, utilizing RGB-A images as input, outputs a JSON draft protocol, indicating the coordinates, size, and order of each element. We develop multiple evaluation metrics for HLG. CreatiGraphist outperforms prior arts and establishes a strong baseline for this field.

Introduction

Graphic design (Jobling and Crowley 1996) fundamentally serves as a form of visual communication. It involves the creation and combination of symbols, images, and text to express certain ideas or messages. This field requires significant expertise and time investment to produce aesthetically pleasing graphic compositions. Recently, there is a significant paradigm shift in leveraging AI for automating the layout of given design elements into cohesive graphic compositions heralds (Zhong, Tang, and Yepes 2019; Yin, Mei, and Chen 2013; Ganin et al. 2021). This could potentially reduce the workload of professional designers and provide an avenue for beginners to create their design pieces, making graphic design more democratize and effective.

A preliminary attempt to automate this process is observed in the task known as **G**raphic **L**ayout **G**eneration (GLG) (Yamaguchi 2021). GLG attempts to intelligently arrange the **size** and **position** of each provided elements into attractive compositions under the assumption of a *predefined order of layers*. However, establishing the appropriate ordering of these layers is a design cornerstone that, if misorganized, can fracture the visual hierarchy, leading to disarray in the intended message delivery. Requiring users to prescribe an accurate layer sequence prior to layout not only burdens them with foresight and planning but also stifles layout algorithms, restricting their capacity to transcend such confines in the pursuit of innovative and aesthetically superior outcomes.

In practical graphic design applications, the **order**, **size**, and **position** of elements are fundamental components that form the entire layout, ultimately determining the visual impact, attraction, and communicability of the design work. Therefore, it is crucial to consider them simultaneously. As illustrated in the right panel of Figure 1, spatial fault (such as incorrect size and position) as well as order fault can significantly compromise the aesthetics of a graphic design.

To advance the layout generation task towards a more end-to-end approach, this paper introduces a new task called **H**ierarchical **L**ayout **G**eneration (HLG). HLG aims to create a visually appealing graphic composition from a collection of **any-ordered** elements by meticulously considering both their spatial arrangement and the ordering of layers. The term ***hierarchical*** emphasizes the significance of element ordering, setting HLG apart from conventional layout generation practices.

For the HLG task, we introduce CreatiGraphist, the first layout generation model built upon Large Multimodal Model (LMM) (Liu et al. 2023; Chen et al. 2023a). The layout generation task is challenging that it digests diverse input elements such as RGB-A materials, RGB images, and texts, and the desired outcomes must precisely reflect the intricate relationships among these multimodal input elements. LMMs are well-suited for this task, as they can unify different modalities, like images, text, coordinates (Peng et al. 2023; Chen et al. 2023a) into tokens. This allows for flexible configuration of various tasks, such as HLG, GLG, and more variants. Furthermore, LMMs demonstrate significant potential for scaling (Kaplan et al. 2020; Team et al. 2023), enabling the pursuit of enhanced performance through the use of larger models and more extensive datasets. For these reasons, LMMs were a natural choice for our foundational architecture in developing CreatiGraphist. In our specific approach for HLG, we train CreatiGraphist with graphic com-

[*]These authors contributed equally.

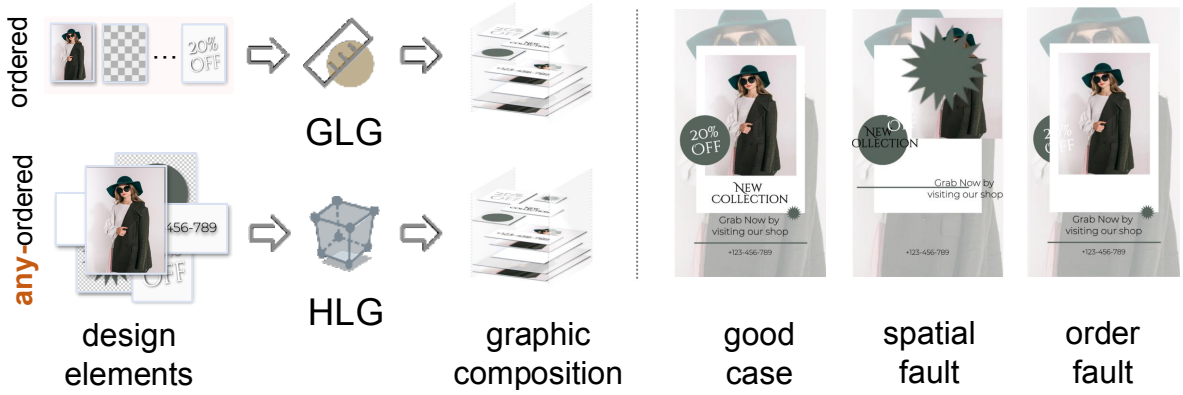

Figure 1: **Schematic diagram of hierarchical layout generation.** (Left) A comparison between the traditional *GLG* task and the newly proposed *HLG* task, with the major difference in that HLG relaxes the constraint of GLG, so that arbitrary-ordered multimodal input elements can be processed. (Right) Spatial fault (wrong size or position) as well as order fault can significantly compromise the aesthetics of a graphic design.

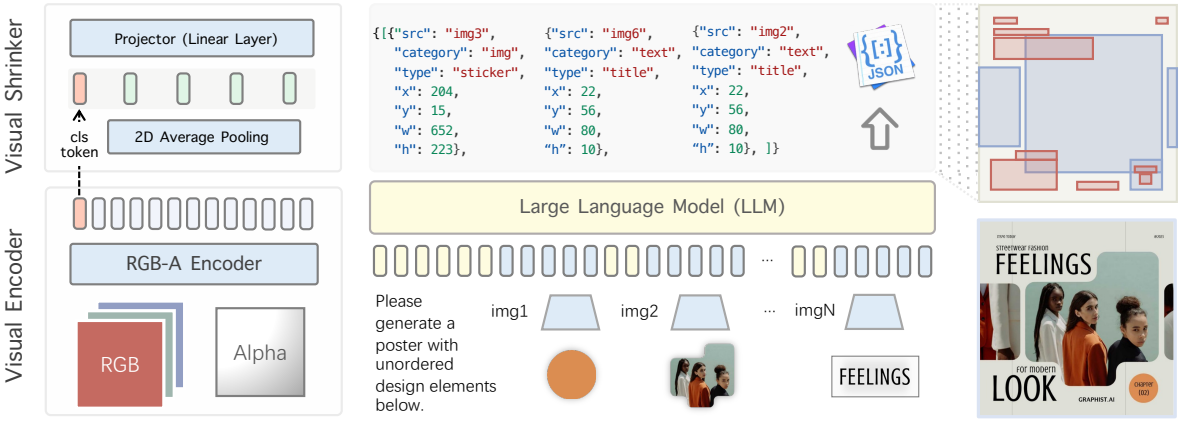

Figure 2: **Graphist Pipeline.** Graphist comprises three components: RGB-A Encoder, Visual Shrinker, and a LLM. It accepts a variety of design elements and generates a graphic composition in JSON format end-to-end.

position data, in which, as shown in Figure 2, each design element is represented as an RGB-A image as input, the model then generates a JSON draft protocol, which specifies the coordinates, size, order and other attributes of each design elements in end-to-end manner.

We introduce two new metrics to evaluate model performance on the HLG task: Inverse Order Pair Ratio (IOPR) and GPT-4V Eval. The former assesses the accuracy of the layer order in the graphic composition, while the GPT-4V Eval, leveraging the capabilities of GPT-4V (OpenAI 2023), quantifies the overall aesthetic quality. Additionally, we have incorporated a human rating score to align with the subjective perceptions of real individuals. After quantitative and qualitative analysis, it is demonstrated that CreatiGraphist is a state-of-the-art solution that not only performs well on traditional GLG task but also achieves remarkable results on the HLG task.

We summarize our contributions as follows:

- We introduce the Hierarchical Layout Generation (HLG) task, which creates graphic compositions from any-ordered design elements. HLG overcomes the limitations of GLG, which typically requires pre-determined layer ordering, enabling more flexible and practical AI-assisted graphic design.
- We present CreatiGraphist, the first LMM-parameterized layout generation model that can be trained end-to-end. CreatiGraphist processes multimodal design elements and generates graphic compositions in JSON format, which can be automatically rendered into the canvas.
- We develop evaluation metrics for HLG, including IOPR and GPT-4V Eval. CreatiGraphist shows superior performance in these metrics, setting a robust benchmark for the field.

Related Work

Graphic Layout Generation

Graphic design is a form of visual art that combines multimodal elements (*e.g.*, images, texts, and symbols) to create aesthetically pleasing compositions which can effec-

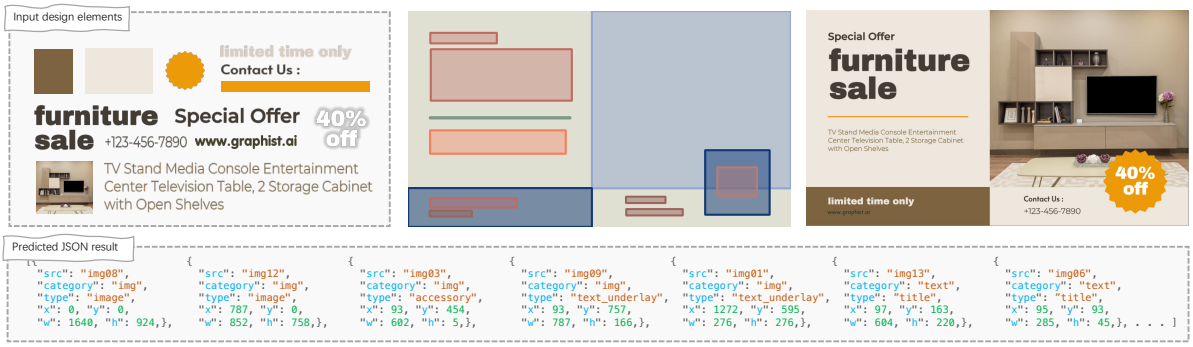

Figure 3: **A user-generated case via graphist web demo**. The top-left figure represents the input design elements to Graphist. Below it, we present the corresponding output JSON code generated by Graphist. The final two images in the top row illustrate the visualized results: first is the layout visualization, and the second is the graphic composition by putting these elements according to the JSON protocol. Additional examples are available in Figure 5.

tively convey information to the audience. As one of the core technologies in automated graphic design, layout generation methods has been widely used in various scenarios, such as document (Zhong, Tang, and Yepes 2019), UI (Deka et al. 2017), social media snippets (Yin, Mei, and Chen 2013), banners (Lee et al. 2020), poster and advertisement (Hsu et al. 2023), icon/logo (Carlier et al. 2020), CAD Sketches (Para et al. 2021), slides (Fu et al. 2022).

As people deepen their understanding of layout generation tasks, the modeling of layout generation problems is becoming increasingly complex. Early methods for layout generation typically relied on aesthetic rules (Yang et al. 2016) or constraints (Hurst, Li, and Marriott 2009). However, the diversity and aesthetics of generated compositions are primarily restricted by the formalized human prior knowledge, and are hard to scale up for production. Later on, some data-driven methods (Li et al. 2019) emerged, utilizing learning-based generation models for layout generation. These approaches are dedicated to fitting the data distribution of the topological relationship between elements, and focus on the alignment of elements, but neglecting the content of the elements. Recent methods (Yu et al. 2022; Li et al. 2023b) have recognized the importance of element content, and have explored ways to layout text or decorations on a given base graphic without obstructing important content. Drawing from this observation, attempts (Inoue et al. 2023b; Zheng et al. 2019) have also been made to incorporate multimodal inputs in order to better understand the content of each layer. These methods often assume that the layering order of the elements has been predefined (Yamaguchi 2021; Inoue et al. 2023b), which is difficult to achieve in the practical application. Therefore, we argue that the layer ordering needs to be fully considered in the layout generation problem.

The advancement of layout generation techniques is closely intertwined with the progress of fundamental algorithms. During the early exploration (Yang et al. 2016), researchers attempted to define beauty using formulas and constraints. Subsequently, generative methods based on GANs (Li et al. 2019, 2020), VAEs (Arroyo, Postels, and Tombari 2021), and diffusions (Zhang et al. 2023) have been widely attempted. Besides, LayoutDETR (Yu et al. 2022) combining DETR (Carion et al. 2020) and generative models has been attempted to deal with the problem of element arrangement on a given image. In the era of Transformer (Vaswani et al. 2017), several methods (Inoue et al. 2023b; Lin et al. 2023a) have adopted the powerful learning capabilities of Transformers to address layout and related control problems (Lin et al. 2023a). FlexDM (Inoue et al. 2023b) models graphic design using a BERT-like (Devlin et al. 2018) approach. The rise of large language models (Brown et al. 2020; Touvron et al. 2023a,b) has also captured the attention of researchers in the field of graphic design, with LayoutPrompter (Lin et al. 2023b) and LayoutNUWA (Tang et al. 2023b) introducing language models into layout generation in a zero-shot manner. COLE (Jia et al. 2023) fuses large language models and diffusion models to build a hierarchical graphic design framework. Our approach is also inspired by large language models (Touvron et al. 2023b; Zhang et al. 2024b) and multimodal models (Liu et al. 2023; Chen et al. 2023a). However, COLE (Jia et al. 2023) focuses on utilizing the generating power of diffusion models to directly produce multi-layer graphic designs based on user intent, while our approach prioritizes the use of existing design elements for layout generation.

Large Multimodal Models

As a bridge between language and vision, LMM (Li et al. 2023a; Yin et al. 2023; Zhang et al. 2024a) has received widespread attention recently. Related topic including autonomous driving (Cui et al. 2024), video understanding (Tang et al. 2023a), multimodal agent (Yang et al. 2023), image generation (Betker et al. 2023), embodied intelligence (Driess et al. 2023) and so on. To the best of our knowledge, Graphist is the first work to address the task of layout generation using LMM in an end-to-end manner. While COLE (Jia et al. 2023) likewise employs multimodal models in their pipeline, their main emphasis lies in generating text content and styles on predetermined base image.

Common architectures of LMM typically encompass a

pre-trained visual encoder (Radford et al. 2021) for extracting visual features, a pre-trained LLM(Touvron et al. 2023a; Team 2023) for interpreting user commands and generating responses, as well as a vision-language cross-modal connector(Liu et al. 2023; Gao et al. 2023) for aligning the visual encoder outputs to the language model. Given that the task of hierarchical layout generation involves organizing design elements on a blank canvas, the compatibility of coordinates is of paramount importance. Pix2Seq (Chen et al. 2021) was the first to explore the use of a discretization and serialization approach in object detection problems, in order to convert coordinates into token sequences that can ultimately be used in sequence generation mode. Some work, like OFA (Wang et al. 2022), VisionLLM (Wang et al. 2023), Kosmos-2 (Huang et al. 2023), build upon this form by introducing special coordinate tokens in their vocabulary. Alternatively, PerceptionGPT (Pi et al. 2023) implements an additional vision encoder and decoder specifically for processing and predicting coordinates. Most notable among these approaches is Shikra (Chen et al. 2023a), which represents spatial positions using numerical values in natural language. It has been proven succinct and effective, as echoed by related research (Bai et al. 2023b). Inspired by Shikra, our work adopts the numerical representation for coordinates in the natural language sequence.

Task Formulation

Graphic Layout Generation

For a unified representation, we consider text boxes as RGB-A images, obtained through rendering. With this framework, GLG seeks the best arrangement for a set of RGB-A elements $\mathcal{M} = \{M_i \in \mathbb{R}^{h_i \times w_i \times 4}\}_{i=1}^{n}$, with the goal of producing a harmonious graphic composition $\mathcal{S} = f(\mathcal{M})$. The function $f(\cdot)$ represents our layout generation methods, which applies transformations to each element M_i, resulting in $\mathcal{S} = \{s_i\}_{i=1}^{n}$. Here, each s_i is a quadruple (x_i, y_i, w_i, h_i), specifying the position of the element's upper-left corner and its respective sizing within the overall layout.

Hierarchical Layout Generation

Given a set of RGB-A elements $\mathcal{M} = \{M_i \in \mathbb{R}^{h_i \times w_i \times 4}\}_{i=1}^{n}$, HLG seeks to arrange them into a well-constructed graphic composition $\mathcal{S} = \{s_i\}_{i=1}^{n} = f(\mathcal{M})$. In this context, $f(\cdot)$ denotes the layout action implemented by our method, which we refer to as CreatiGraphist. Each element s_i in the layout prediction encompasses five numerical values $(x_i, y_i, w_i, h_i, l_i)$, associated with M_i top/left coordinates, width, height and hierarchy, respectively. As shown in Figure 1, compared to the conventional GLG task (Yamaguchi 2021), HLG framework emphasizes the importance of element stratification by imposing a hierarchy l_i that dictates not only the placement but also the hierarchical order of each element, thus ensuring a compositionally harmonious layering.

Proposed Method

CreatiGraphist Architecture

CreatiGraphist is constructed using a LMM, which receives the input of multimodal design elements and predicts a JSON fragment formatted like Figure 2. CreatiGraphist comprises three components: (i) *RGBA-Encoder*. We utilize a ViT-L/14 with 224×224 four-channel input as RGBA-Encoder, initialized with visual tower parameters of CLIP (Radford et al. 2021), excluding the alpha channel. (ii) *LLM*. Our LLM foundation incorporates Qwen1.5-0.5B/7B (Bai et al. 2023a). (iii) *Visual Shrinker*. To manage the extensive elements processing requirements of CreatiGraphist, the Visual Shrinker compresses ViT's $16 \times 16 + 1$ (cls-token) output grid feature tokens into only 5 tokens, thereby saving computational costs. Specifically, it compress the 2D output of ViT's 16×16 into 2×2 tokens via 2D average pooling and concatenates it with the cls-token to form $\mathbf{V} \in \mathbb{R}^{5 \times D}$. Consequently, it uses one MLP layer to map the $\mathbf{V}$ to $\mathbf{V}' \in \mathbb{R}^{5 \times D'}$ for modal alignment and dimension matching of LLM, where D' is the word embedding dimension defined by LLM. Visual embedding can be inserted into anywhere of input sequence. Inspired by Shikra (Chen et al. 2023a), we employ the digital tokens to represent coordinates, devoid of any vocab, specialty encoders, or any pre-/post-detectors for encoding position information. Model variants include CreatiGraphist-Tiny utilizing Qwen1.5-0.5B and CreatiGraphist-Base engaging Qwen1.5-7B.

Training Strategy

The proposed CreatiGraphist is trained in three stages. During Stage-1, CreatiGraphist focuses on the patch embedding and projector layers and trains using an efficient method with a large batch size and reduced sequence length. This phase primarily aims to calibrate the visual encoder to interpret alpha channels in RGB-A imagery and align projector layers with both visual and linguistic features.

Progressing to Stage-2, the training expands to encompass both the projector layer and the full LLM. Training tasks from the initial stage are retained; however, the HLG task is administered with greater frequency to highlight the model's fundamental understanding of graphic layout.

Finally, Stage-3 maintains the focus on the projector layer and the complete LLM, but aiming to adapt the model to a broader range of graphic design task types. Traditional GLG task is also regarded as a specific task type in this stage. We list primary configurations during training in Table 1.

To train the model to arrange inputs layer order and spatial coordinates, we randomly shuffle the input elements with a 0.75 probability for all of the Graphic Design datasets used in the training process. In these three stages, the first stage trains 10k steps, while the second and third stages both train 20k steps.

Stage	BS	Length	Tune Part	Task	Training datasets	
					Graphist	Graphist*
1	128	1536	PE Projector	Cap. HLG	ShareGPT4v(Chen et al. 2023b) Flickr30k(Plummer et al. 2015) Crello(Yamaguchi 2021)	+ IH-RGBA
2	64	2048	Projector LLM	Cap. HLG	ShareGPT4v(Chen et al. 2023b) Flickr30k(Plummer et al. 2015) Crello(Yamaguchi 2021)	+ IH-RGBA + IH-Design
3	64	3584	Projector LLM	Cap. HLG GLG	ShareGPT4v(Chen et al. 2023b) Flickr30k(Plummer et al. 2015) Crello(Yamaguchi 2021)	+ IH-RGBA + IH-Design

Table 1: **Implementation details** including training stages, datasets, and essential parameters: "BS" for batch size, "Length" for total sequence length. "Graphist" is trained on academic data; "Graphist*" on proprietary data. "PE." denotes the Patch Embedding Layer. "Cap." signifies image captioning task. "IH-RGBA" is in-house image-text dataset, where images with alpha channel. "IH-Design" is 80k in-house Graphic Design Dataset like Crello (Yamaguchi 2021).

Experiment

Datasets

Crello dataset [1] furnishes an array of graphic compositions derived from a web-based design utility, namely Crello (Now change the name to VistaCreate[2]). It covers an extensive range of graphic compositions suited for various applications such as social media infographics, digital banner ads, blog headers, and printed poster templates. Inside the dataset, each graphic composition includes detailed information about the layering order, spatial positioning, and categorical details of the design elements. The dataset is advantageous for tasks like GLG and has been the foundation for several methods (Yamaguchi 2021; Inoue et al. 2023b). Additionally, it is a good playground for HLG task. In Flex-DM (Inoue et al. 2023b), the dataset is partitioned into 19,095 training, 1,951 validation, and 2,375 testing examples. However, they used the Crello v2, but since the current version released by Crello is v4, we used the intersection of all parts in the two version test sets, a total of 242 graphic compositions as the test set in experiments.

Evaluation Metrics

For evaluation metrics, we defined the IOPR to assess layer ordering, and leveraged GPT-4V to evaluate the poster quality. We deliberately refrained from utilizing conventional evaluation metrics like LayoutFID (Inoue et al. 2023a) and MaxIoU (Kikuchi et al. 2021) among others. This decision stems from the recognition that the aesthetics of a poster are inherently diverse and encompass a broad spectrum of potential outcomes. Given identical materials, numerous aesthetically pleasing results are indeed possible. Therefore, we consider it inappropriate to assess the quality of a layout outcome purely on the basis of a numerical absolute value.

Inverse order pair ratio (IOPR). The appropriate order of layers is crucial for the results of HLG. We develop IOPR, which is a ratio representing the fraction of overlap element pairs that are in inverse order according to the model's predictions out of all possible overlapping element pairs. It is calculated as

$$\text{IOPR} = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\mathbf{1}\left(\mathcal{O}_j < \mathcal{O}_i \wedge \texttt{overlap}\left(i,j\right)\right)}{\sum_{i=0}^{n-1}\sum_{j=i+1}^{n}\mathbf{1}} \tag{1}$$

where n is the number of layers in the hierarchical structure. $\mathbf{1}$ is an indicator function that returns 1 if the argument condition is true and 0 otherwise. $\mathcal{O}$ denotes the output order or predicted order of the layers as determined by the model. $\mathcal{O}_i$ and $\mathcal{O}_j$ correspond to the predicted order positions of the i^{th} and j^{th} layers, respectively. $\texttt{overlap}(i,j)$ is a predicate function that determines whether the i^{th} and j^{th} layers overlap. A high IOPR would suggest that the model is quite accurate in predicting the correct order of layers. Contrastingly, a low IOPR would suggest the model often predicts the wrong sequence, indicating lower prediction accuracy. In our experiments, we employed IOPR_{min} and IOPR_{avg} to evaluate the performance of the model, representing respectively the average score and the minimum score in the test dataset.

GPT-4V Eval. In addition to the layers order, the overall aesthetic quality and harmony of the elements in the graphic composition are vitally important. We utilize GPT-4V to evaluate our approach and compare it with other alternatives. Following COLE (Jia et al. 2023), we use four scores named GPT-4V rating including S_{DL}, S_{GI}, S_{IO} and S_{TV} use GPT-4V to evaluate the quality of the graphic composition we generate.

- S_{DL} means the graphic design should present a clean, balanced, and consistent layout. The organization of elements should enhance the message, with clear paths for the eye to follow.
- S_{GI} reflects that any graphics or images used should enhance the design rather than distract from it. They should

[1]https://huggingface.co/datasets/cyberagent/crello

[2]https://create.vista.com/

Method	Task	S_{DL}	S_{GI}	S_{IO}	S_{TV}	IOPR
Flex-DM	*GLG*	5.43	6.13	4.69	4.60	-
GPT-4V	*GLG*	5.10	5.83	4.54	3.84	-
Gemini	*GLG*	4.87	5.62	4.39	4.09	-
Graphist	*GLG*	5.60	6.64	4.84	4.86	-
Graphist∗	*GLG*	**5.72**	**6.70**	**5.03**	**5.37**	-
GPT-4V	*HLG*	4.19	4.66	3.71	3.25	0.45
Gemini	*HLG*	5.06	5.94	4.33	4.21	0.70
Graphist	*HLG*	5.66	6.60	5.02	4.93	0.96
Graphist∗	*HLG*	**5.85**	**6.90**	**5.10**	**5.24**	**0.97**

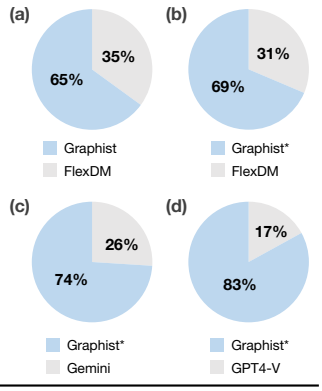

Table 2: **GPT-4V Eval on Crello**. Performance of different methods on the Crello dataset. Graphist is Graphist-Base built upon Qwen1.5-7B. The scores on the left are GPT-4V rating and IOPR_{avg}, while the chart on the right showcases a comparative evaluation using GPT-4V voting against the FlexDM (Inoue et al. 2023b), Gemini-1.5-Pro(Reid et al. 2024) (Gemini) and GPT-4V(OpenAI 2023). Flex-DM is compared based on the GLG task (a, b), while Gemini-1.5-Pro and GPT-4V are based on the HLG task (c, d).

be high quality, relevant, and harmonious with other elements.

- S_{IO} evaluates the innovation level of the design.
- S_{TV} represents text readability. A lower score would be assigned if the readability of the text is poor due to the color of the text being similar to the background color or overlapping of the text.

As a supplement to GPT-4V rating (Jia et al. 2023), we propose GPT-4V voting. Here, GPT-4V also partakes in a comparative analysis. It selects the most proficient graphic composition when confronted with two competing outputs. This preference distribution acts as a testament to the discernibility of GPT-4V in recognizing and preferring one method over the other across a range of comparative samples. GPT-4V rating (Jia et al. 2023) and GPT-4V voting together form the GPT-4V Eval for evaluating layout ability.

Comparison with SoTA

We trained our model as outlined in Section and compared it to other state-of-the-art methods. In the following comparison, we represent the model trained only on public datasets as "**CreatiGraphist**", while "**CreatiGraphist***" denotes the model trained on in-house data in addition to public datasets. Our models are compared with previous SoTA layout generation method Flex-DM and SoTA LMM models (OpenAI 2023; Reid et al. 2024). In both GLG and HLG tasks, CreatiGraphist* achieved the highest scores in all the benchmarks. CreatiGraphist* demonstrates a clear advantage in terms of text visibility, layout balance, and visual appeal, especially in the HLG task. The HLG task, being a more challenging but more real-world-related usage strategy, suggests through our experimental results that CreatiGraphist* has managed to learn some of the basic principles of graphic design. Results in Table 2 indicate that our approach achieves the best results in the metrics S_{DL}, S_{GI}, and S_{IO} when dealing with an any-ordered input. The flexibility introduced by allowing randomness in input ordering contributes to the superior performance observed in S_{DL} and S_{IO} metrics versus a structured sequence. Nevertheless, this any-ordered approach increases the complexity of text layer placement, reflected by a reduced score in S_{TV}.

Figure 4: **Results visualization of the GLG task on the Crello dataset.** The results for Flex-DM were derived from their open-source code, whereas the results for GPT-4V and Gemini-1.5-Pro are obtained in zero-shot manner.

Visualized results

In the visualization of the GLG outcomes in Figure 4, CreatiGraphist outperforms competitors including Flex-DM (Inoue et al. 2023b), GPT-4V (OpenAI 2023), and Gemini-1.5-Pro (Reid et al. 2024). The other methodologies grapple with challenges such as text overlap and image distortion.

To assess the real-world applicability of our method, we engaged a group of non-expert volunteers to submit their de-

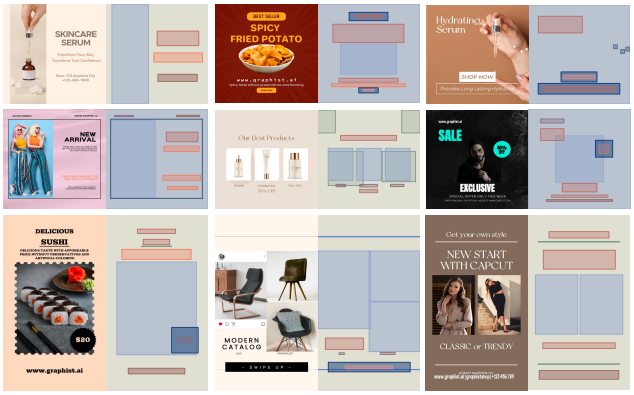

Figure 5: **More user-generated designs via graphist web demo**. To evaluate usability, we invited numerous non-expert volunteers to upload their design components to our Graphist web demo, resulting in the creation of their own design projects. The image displays a selection of these high-quality outcomes alongside their respective layouts. Within the layout map, various colors signify distinct layer attributes as recognized by the model.

sign assets via our CreatiGraphist web demo and chose a selection of representative outputs for analysis. As illustrated in Figure 5, our model successfully generates visually appealing and cohesive designs across varying canvas dimensions and design elements.

Ablation Studies

In this section, we delve into a series of ablation studies to examine the underlying factors that influence the layout quality of Graphist. The specific aspects under investigation include the number of visual tokens, and the encoding of transparency through RGB-A image channels. More experimental results can be found in supplementary materials.

Token Length	S_{DL}	S_{GI}	S_{IO}	S_{TV}	IOPR$_{\text{min}}$	IOPR$_{\text{avg}}$
5 tokens	5.44	6.35	4.94	4.60	0.641	0.963
17 tokens	5.47	6.28	4.88	4.67	0.667	0.965

Table 3: **Different visual token length**. Comparison of different sequence length performance on the Crello dataset based on Graphist-Tiny. From our observation, five tokens are sufficient for layout generation.

Influence of visual token length. In our method, we typically represent an image using a sequence of 5 visual tokens. To evaluate the effects of increased token quantity, potentially enhancing the image representation's granularity, we conducted experiments using a 17 tokens format. The composition of 17 tokens includes 1 cls token and 16 visual tokens, achieved by using a 4×4 pooling kernel size. The results presented in Table 3 suggests that lengthening the visual token sequence does not necessarily lead to performance improvements. According to the outcomes, the representation of an image with a quintet of tokens sufficiently captures the necessary information for this specific task.

Method	S_{DL}	S_{GI}	S_{IO}	S_{TV}	IOPR$_{\text{min}}$	IOPR$_{\text{avg}}$
RGB-A	5.44	6.35	4.94	4.60	0.641	0.963
RGB	5.24	6.14	4.62	4.22	0.502	0.951

Table 4: **Different input channel**. Comparison of RGB input and RGB-A input performance on the Crello dataset. The results indicate that RGB-A performs better, particularly in S_{IO} and S_{TV}. This experiment based on Graphist-Tiny model.

RGB vs RGB-A. To assess the effect of omitting the alpha channel, we have also tested the model with traditional three-channel RGB images. As illustrated in Table 4, inputs with the additional alpha channel (RGB-A) yield higher-quality outputs with more accurate layer ordering when compared to RGB inputs. The alpha channel provides detailed information that aids the model in discerning textural elements and gradients within the image layers. Importantly, when working with text layers, the alpha channel helps separate text from busy backgrounds, making it easier and more accurate to place the text clearly.

Conclusion

This paper represents a step forward from traditional graphic layout generation by introducing the hierarchical layout generation task, which enhances graphic design automation by effectively handling any-ordered design elements, thereby increasing creative potential and efficiency. To address this more challenging task, we proposed CreatiGraphist, a novel LMM that tackles HLG tasks as sequence generation challenges. CreatiGraphist takes RGB-A images as input and produces JSON draft protocols that define the layout parameters of graphic compositions. To appropriately evaluate HLG tasks, we introduced two metrics: the Inverse Order Pair Ratio and GPT-4V Eval. Evaluation results demonstrate that CreatiGraphist achieves state-of-the-art results, providing a strong baseline for generating automated graphic designs that are more creative and diverse.

It's important to emphasize that in real-world layout scenarios, the randomness of layout elements is a major challenge. A chaotic order can directly destroy the intended meaning of elements. We need not only the correct positioning but also the right sequence.

References

Arroyo, D. M.; Postels, J.; and Tombari, F. 2021. Variational transformer networks for layout generation. In *CVPR*, 13642–13652.

Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; Hui, B.; Ji, L.; Li, M.; Lin,

J.; Lin, R.; Liu, D.; Liu, G.; Lu, C.; Lu, K.; Ma, J.; Men, R.; Ren, X.; Ren, X.; Tan, C.; Tan, S.; Tu, J.; Wang, P.; Wang, S.; Wang, W.; Wu, S.; Xu, B.; Xu, J.; Yang, A.; Yang, H.; Yang, J.; Yang, S.; Yao, Y.; Yu, B.; Yuan, H.; Yuan, Z.; Zhang, J.; Zhang, X.; Zhang, Y.; Zhang, Z.; Zhou, C.; Zhou, J.; Zhou, X.; and Zhu, T. 2023a. Qwen Technical Report. *arXiv preprint arXiv:2309.16609*.

Bai, J.; Bai, S.; Yang, S.; Wang, S.; Tan, S.; Wang, P.; Lin, J.; Zhou, C.; and Zhou, J. 2023b. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*.

Betker, J.; Goh, G.; Jing, L.; Brooks, T.; Wang, J.; Li, L.; Ouyang, L.; Zhuang, J.; Lee, J.; Guo, Y.; et al. 2023. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3): 8.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, 213–229. Springer.

Carlier, A.; Danelljan, M.; Alahi, A.; and Timofte, R. 2020. Deepsvg: A hierarchical generative network for vector graphics animation. *Advances in Neural Information Processing Systems*, 33: 16351–16361.

Chen, K.; Zhang, Z.; Zeng, W.; Zhang, R.; Zhu, F.; and Zhao, R. 2023a. Shikra: Unleashing Multimodal LLM's Referential Dialogue Magic. *arXiv preprint arXiv:2306.15195*.

Chen, L.; Li, J.; Dong, X.; Zhang, P.; He, C.; Wang, J.; Zhao, F.; and Lin, D. 2023b. ShareGPT4V: Improving Large Multi-Modal Models with Better Captions. *arXiv preprint arXiv:2311.12793*.

Chen, T.; Saxena, S.; Li, L.; Fleet, D. J.; and Hinton, G. 2021. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*.

Cui, C.; Ma, Y.; Cao, X.; Ye, W.; Zhou, Y.; Liang, K.; Chen, J.; Lu, J.; Yang, Z.; Liao, K.-D.; et al. 2024. A survey on multimodal large language models for autonomous driving. In *WACV*, 958–979.

Deka, B.; Huang, Z.; Franzen, C.; Hibschman, J.; Afergan, D.; Li, Y.; Nichols, J.; and Kumar, R. 2017. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th annual ACM symposium on user interface software and technology*, 845–854.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Driess, D.; Xia, F.; Sajjadi, M. S.; Lynch, C.; Chowdhery, A.; Ichter, B.; Wahid, A.; Tompson, J.; Vuong, Q.; Yu, T.; et al. 2023. PaLM-E: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*.

Fu, T.-J.; Wang, W. Y.; McDuff, D.; and Song, Y. 2022. DOC2PPT: automatic presentation slides generation from scientific documents. In *AAAI*, 634–642.

Ganin, Y.; Bartunov, S.; Li, Y.; Keller, E.; and Saliceti, S. 2021. Computer-aided design as language. *Advances in Neural Information Processing Systems*, 34: 5885–5897.

Gao, P.; Han, J.; Zhang, R.; Lin, Z.; Geng, S.; Zhou, A.; Zhang, W.; Lu, P.; He, C.; Yue, X.; et al. 2023. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*.

Hsu, H. Y.; He, X.; Peng, Y.; Kong, H.; and Zhang, Q. 2023. PosterLayout: A New Benchmark and Approach for Content-aware Visual-Textual Presentation Layout. In *CVPR*, 6018–6026.

Huang, S.; Dong, L.; Wang, W.; Hao, Y.; Singhal, S.; Ma, S.; Lv, T.; Cui, L.; Mohammed, O. K.; Liu, Q.; et al. 2023. Language is not all you need: Aligning perception with language models. *arXiv preprint arXiv:2302.14045*.

Hurst, N.; Li, W.; and Marriott, K. 2009. Review of automatic document formatting. In *Proceedings of the 9th ACM symposium on Document engineering*, 99–108.

Inoue, N.; Kikuchi, K.; Simo-Serra, E.; Otani, M.; and Yamaguchi, K. 2023a. LayoutDM: Discrete Diffusion Model for Controllable Layout Generation. In *CVPR*, 10167–10176.

Inoue, N.; Kikuchi, K.; Simo-Serra, E.; Otani, M.; and Yamaguchi, K. 2023b. Towards Flexible Multi-modal Document Models. In *CVPR*, 14287–14296.

Jia, P.; Li, C.; Liu, Z.; Shen, Y.; Chen, X.; Yuan, Y.; Zheng, Y.; Chen, D.; Li, J.; Xie, X.; et al. 2023. COLE: A Hierarchical Generation Framework for Graphic Design. *arXiv preprint arXiv:2311.16974*.

Jobling, P.; and Crowley, D. 1996. Graphic design: reproduction and representation since 1800.

Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T. B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; and Amodei, D. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Kikuchi, K.; Simo-Serra, E.; Otani, M.; and Yamaguchi, K. 2021. Constrained graphic layout generation via latent optimization. In *Proceedings of the 29th ACM International Conference on Multimedia*, 88–96.

Lee, H.-Y.; Jiang, L.; Essa, I.; Le, P. B.; Gong, H.; Yang, M.-H.; and Yang, W. 2020. Neural design network: Graphic layout generation with constraints. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, 491–506. Springer.

Li, C.; Gan, Z.; Yang, Z.; Yang, J.; Li, L.; Wang, L.; and Gao, J. 2023a. Multimodal foundation models: From specialists to general-purpose assistants. *arXiv preprint arXiv:2309.10020*, 1(2): 2.

Li, F.; Liu, A.; Feng, W.; Zhu, H.; Li, Y.; Zhang, Z.; Lv, J.; Zhu, X.; Shen, J.; and Lin, Z. 2023b. Relation-Aware Diffusion Model for Controllable Poster Layout Generation. In *Proceedings of the 32nd ACM international conference on information & knowledge management*, 1249–1258.

Li, J.; Yang, J.; Hertzmann, A.; Zhang, J.; and Xu, T. 2019. Layoutgan: Generating graphic layouts with wireframe discriminators. *arXiv preprint arXiv:1901.06767*.

Li, J.; Yang, J.; Zhang, J.; Liu, C.; Wang, C.; and Xu, T. 2020. Attribute-conditioned layout gan for automatic graphic design. *IEEE Transactions on Visualization and Computer Graphics*, 27(10): 4039–4048.

Lin, J.; Guo, J.; Sun, S.; Xu, W.; Liu, T.; Lou, J.-G.; and Zhang, D. 2023a. A parse-then-place approach for generating graphic layouts from textual descriptions. In *ICCV*, 23622–23631.

Lin, J.; Guo, J.; Sun, S.; Yang, Z. J.; Lou, J.-G.; and Zhang, D. 2023b. LayoutPrompter: Awaken the Design Ability of Large Language Models. *arXiv preprint arXiv:2311.06495*.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

OpenAI. 2023. GPT-4V(ision) system card.

Para, W.; Bhat, S.; Guerrero, P.; Kelly, T.; Mitra, N.; Guibas, L. J.; and Wonka, P. 2021. Sketchgen: Generating constrained cad sketches. *Advances in Neural Information Processing Systems*, 34: 5077–5088.

Peng, Z.; Wang, W.; Dong, L.; Hao, Y.; Huang, S.; Ma, S.; and Wei, F. 2023. Kosmos-2: Grounding Multimodal Large Language Models to the World. *arXiv preprint arXiv:2306.14824*.

Pi, R.; Yao, L.; Gao, J.; Zhang, J.; and Zhang, T. 2023. PerceptionGPT: Effectively Fusing Visual Perception into LLM. *arXiv preprint arXiv:2311.06612*.

Plummer, B. A.; Wang, L.; Cervantes, C. M.; Caicedo, J. C.; Hockenmaier, J.; and Lazebnik, S. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, 2641–2649.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*, 8748–8763. PMLR.

Reid, M.; Savinov, N.; Teplyashin, D.; Lepikhin, D.; Lillicrap, T.; Alayrac, J.-b.; Soricut, R.; Lazaridou, A.; Firat, O.; Schrittwieser, J.; et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Tang, Y.; Bi, J.; Xu, S.; Song, L.; Liang, S.; Wang, T.; Zhang, D.; An, J.; Lin, J.; Zhu, R.; et al. 2023a. Video understanding with large language models: A survey. *arXiv preprint arXiv:2312.17432*.

Tang, Z.; Wu, C.; Li, J.; and Duan, N. 2023b. LayoutNUWA: Revealing the Hidden Layout Expertise of Large Language Models. *arXiv preprint arXiv:2309.09506*.

Team, G.; Anil, R.; Borgeaud, S.; Wu, Y.; Alayrac, J.-B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A. M.; Hauth, A.; et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Team, I. 2023. InternLM: A Multilingual Language Model with Progressively Enhanced Capabilities. https://github.com/InternLM/InternLM.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, P.; Yang, A.; Men, R.; Lin, J.; Bai, S.; Li, Z.; Ma, J.; Zhou, C.; Zhou, J.; and Yang, H. 2022. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *ICML*, 23318–23340. PMLR.

Wang, W.; Chen, Z.; Chen, X.; Wu, J.; Zhu, X.; Zeng, G.; Luo, P.; Lu, T.; Zhou, J.; Qiao, Y.; et al. 2023. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *arXiv preprint arXiv:2305.11175*.

Yamaguchi, K. 2021. Canvasvae: Learning to generate vector graphic documents. In *ICCV*, 5481–5489.

Yang, X.; Mei, T.; Xu, Y.-Q.; Rui, Y.; and Li, S. 2016. Automatic generation of visual-textual presentation layout. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 12(2): 1–22.

Yang, Z.; Liu, J.; Han, Y.; Chen, X.; Huang, Z.; Fu, B.; and Yu, G. 2023. Appagent: Multimodal agents as smartphone users. *arXiv preprint arXiv:2312.13771*.

Yin, S.; Fu, C.; Zhao, S.; Li, K.; Sun, X.; Xu, T.; and Chen, E. 2023. A Survey on Multimodal Large Language Models. *arXiv preprint arXiv:2306.13549*.

Yin, W.; Mei, T.; and Chen, C. W. 2013. Automatic generation of social media snippets for mobile browsing. In *Proceedings of the 21st ACM international conference on Multimedia*, 927–936.

Yu, N.; Chen, C.-C.; Chen, Z.; Meng, R.; Wu, G.; Josel, P.; Niebles, J. C.; Xiong, C.; and Xu, R. 2022. LayoutDETR: Detection Transformer Is a Good Multimodal Layout Designer. *arXiv preprint arXiv:2212.09877*.

Zhang, D.; Yu, Y.; Li, C.; Dong, J.; Su, D.; Chu, C.; and Yu, D. 2024a. Mm-llms: Recent advances in multimodal large language models. *arXiv preprint arXiv:2401.13601*.

Zhang, J.; Guo, J.; Sun, S.; Lou, J.-G.; and Zhang, D. 2023. LayoutDiffusion: Improving Graphic Layout Generation by Discrete Diffusion Probabilistic Models. In *ICCV*.

Zhang, P.; Zeng, G.; Wang, T.; and Lu, W. 2024b. TinyLLaMA: An Open-Source Small Language Model. *arXiv preprint arXiv:2401.02385*.

Zheng, X.; Qiao, X.; Cao, Y.; and Lau, R. W. 2019. Content-aware generative modeling of graphic design layouts. *ACM Transactions on Graphics (TOG)*, 38(4): 1–15.

Zhong, X.; Tang, J.; and Yepes, A. J. 2019. Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 1015–1022. IEEE.