Reverse-engineering information presentations: recovering hierarchical grouping from layouts of visual elements

Danqing Shi¹, Weiwei Cui²*, Danqing Huang², Haidong Zhang² and Nan Cao¹

Abstract
Visual elements in an information presentation are often spatially and semantically grouped hierarchically for effective message delivery. Studying the hierarchical grouping information can help researchers and designers better explore layout structures and understand design demographics. However, recovering hierarchical grouping is challenging due to a large number of possibilities for compositing visual elements into a single-page design. This paper introduces an automatic approach that takes the layout of visual elements as input and returns the hierarchical grouping as output. To understand information presentations, we first contribute a dataset of 23,072 information presentations with diverse layouts to the community. Next, we propose our technique with a Transformer-based model to predict relatedness between visual elements and a bottom-up algorithm to produce the hierarchical grouping. Finally, we evaluate our technique through a technical experiment and a user study with 30 designers. The results show that the proposed technique is promising.

Keywords: Layout design, Datasets, Deep learning, Information presentations, Infographics

1 Introduction
Information presentations (e.g., presentation slides, business reports, and advertising posters) are widely used in daily life as an essential medium of visual communication. In an information presentation, the layout refers to the 2D spatial structure which is determined by the sizes and positions of the visual elements [1]. To ensure effective message delivery, the layout is usually arranged by a set of design guidelines or principles. For example, Williams Robin provided practical design guidelines (e.g., proximity, alignment, repetition, and contrast) to suggest the layout when creating an information presentation [2]. More theoretically, the Gestalt laws provide the fundamental theory to explain the mechanisms of how human beings perceive and understand visual information [3]. In a word, a well-designed layout structure enables the quick and easy perception of the information [4, 5].

Hierarchical grouping is a popular approach to explaining the perceptual structure of the layout, which is commonly used in professional graphic design tools (see Fig. 1 for an example). Here, a group is defined as a set of related visual elements that are associated with each other, and the hierarchical grouping is the collection of groups that are organized in a hierarchical structure. A proper hierarchical grouping can benefit many downstream use cases in visual design. A primary and direct use case is assisting users in designing information presentations easier: if the visual elements of an information presentation are grouped into a meaningful hierarchical structure, users can conveniently move/resize/rotate/transform/delete a group of related elements. It can also help to apply animation effects to the elements [6]. Furthermore, other potential use cases that rely on understanding the grouping structure include design assessment [7], visual search [8], information extrac-
tion [9], information seeking [10], and design space exploration [11].

Although useful for downstream applications, most information presentations in the wild do not explicitly annotate the hierarchical grouping. To address this problem, we aim to automatically recover hierarchical grouping from layouts of visual elements in information presentations. It is a challenging task for two major reasons. First, unlike the spatial relationships between elements in the layout such as direction and distance, which can be computed by using pre-defined rules [12], the grouping structure is the implicit knowledge hidden in the layout. Second, there are many factors that may affect the grouping structure, which is hard for researchers to enumerate all the rules and principles. Several existing works have already paid attention to recovering grouping from arbitrary images [5] and visualizations [13], but not from layouts of visual elements in information presentations. One of the most related works [14] implements four Gestalt laws to produce grouping in web page design which performs well in their test cases. However, extensive human efforts are still required for researchers to determine better design rules to improve accuracy. Therefore, our emphasis in this paper is to incorporate a learning-based approach to a more intelligent grouping technique.

The large collection of information presentations available online serves as a great resource for our research. Therefore, we first constructed a large-scale dataset of information presentations. Specifically, as PowerPoint (denoted as PPT) is one of the most commonly used formats for information presentation in daily life, we have collected approximately 1000 PPT files from online public websites, which finally gave us 23,072 well-designed information presentations. Each information presentation contains approximately 27 visual elements on average. By parsing visual elements from these slides, we extracted typical types of elements, such as shape, text, picture, and chart. To assist in understanding the layout, we also parsed all spatial features of each visual element.

Next, we proposed an automatic approach to produce a hierarchical grouping of visual elements from information presentations. To address the problem by not enumerating all possible factors, we first incorporated a learning-based approach to model the relatedness between visual elements in information presentations. Specifically, we introduced a Transformer-based model, which can capture the contextual layout information to predict relatedness between visual elements by learning from human-labeled information presentations. In addition, we improved the original Transformer-based model by integrating the spatial-aware attention mechanism to involve the relative spatial information among elements. Second, our technique hierarchically groups all the elements from bottom to top according to the predicted relatedness. In detail, we defined a graph model of the information presentation layout, in which nodes are visual elements and edges are the adjacent connections between elements. Then, we built a graph-based algorithm to group the visual elements hierarchically.

To evaluate the technique, we conducted a technical evaluation to compare the accuracy results of our models with a baseline method. We found that our technique can correctly predict pairwise relatedness with high accuracy. Furthermore, we conducted a user study with thirty designers to evaluate the final hierarchical grouping results. The results show that our technique has comparable quality with the slides manually grouped by designers.

To summarize, we make the following three key contributions in this paper:

1) **Dataset.** We contribute a large-scale design dataset to the community. It includes 23,072 information presentations with 620,878 visual elements collected from the Internet. The dataset is freely available for academic purposes.

2) **Technique.** We present an automatic approach for recovering hierarchical grouping in information presentations. It consists of two major modules: a Transformer-based model for predicting relatedness between visual elements and a bottom-up algorithm for grouping the visual elements hierarchically.

3) **Evaluation.** We conduct a technical evaluation and a user study to evaluate our proposed technique. The results show that our technique can produce accurate hierarchical grouping results.

2 Background and related work

This section provides three categories of related work: graphic design layout, grouping of information, and data construction for graphic design.

2.1 Graphic design layout

To create effective and aesthetic information presentations, designers need to put significant effort into design-
ing the layout [4]. Therefore, an extensive range of techniques has been proposed to assist layout design. Some studies in the community developed automatic layout algorithms that can place graphical and textual elements in proper positions based on certain design rules [1]. For example, DesignScape [15] is an interactive system that aids the design process by interactively generating layout suggestions, such as position, size, and alignment. SketchPlore [16] uses a real-time layout optimizer that automatically infers the designer’s task to search for both local design improvements and global layout alternatives. As the design of information presentation also commonly uses grid layouts to define spatial structures using grid lines [17], a more recent work GRIDS [18] uses a mixed-integer linear programming model for grid layout generation that optimizes the packing, alignment, grouping, and preferential positioning of elements. However, these rule-based methods typically aim to optimize certain objectives, which may ignore other aspects of the design. Recent advances in deep learning enable advanced techniques to generate graphic layouts by learning existing works. Zheng et al. [19] proposed a deep generative model to synthesize layouts based on the semantic meaning of visual elements. Vinci [20] uses a sequence-to-sequence generative model to match the visual elements and layouts for generating an aesthetic advertising poster. LayoutGAN [21] synthesizes layouts from different types of 2D visual elements by using a generative adversarial network (GAN) architecture. While these techniques are powerful in generating information presentation, they usually ignore the stage of understanding the layouts exactly.

Other works in the community focus on understanding the layout in the graphic design. For instance, Bylinskii et al. [22] presented a machine learning model that predicts the relative importance of different visual elements in graphic designs. Because inferring spatial relationships among elements plays a vital role in layout suggestion, Xu et al. [23] designed an interactive user interface for visualizing two types of spatial relationships: edge alignment and equal spacing. O’Donovan et al. [24] proposed an energy-based method to learn three hidden variables in single-page layouts, including alignment, the importance of each element, and grid-based segmentation. These works are relevant to our work, which also extract the layout-related information from the design of information presentations. However, unlike these previous works, we aim to focus on hierarchical grouping, which is the implicit knowledge hidden in the layout of visual elements. In this work, we take advantage of a deep-learning-based approach to reverse-engineer information presentations, especially recovering the hierarchical grouping from layouts.

2.2 Grouping of visual information

Although layouts might be vary specific to different document types, such as presentation slides, web pages, and advertising posters, the grouping of information is general to organize the content of information presentations. A proper grouping can improve the readability and highlight relatedness between the information [25]. In practice, the proximity guideline proposed by [2] states “Group related items together”. Designers can move spatially closed visual elements to make them a cohesive group rather than unrelated items. In other words, spatial closeness usually implies a high relatedness between the elements. In addition, the Gestalt principle states that human beings generally organize information using a top-down approach [25, 26]. This principle suggests that a designer should organize visual information in a clear hierarchical grouping structure.

Automatic grouping methods are extensively utilized in image processing. Ren et al. [27] used the features derived from the classical Gestalt cues to group pixels into regions. Wattenberg et al. [13] utilized machine vision techniques for analyzing an image at multiple resolutions to describe the hierarchical segmentation of an image. Several bottom-up approaches were proposed for hierarchically grouping pixels in images or videos [28–30]. Recently, advanced deep learning methods have also been proposed to provide a range of end-to-end solutions for pixel-level grouping [31–33]. Nevertheless, these grouping techniques in image processing are all based on pixels. In contrast, grouping in information presentation focuses on the visual elements in a 2D view, which have more diverse spatial features.

There are several relevant grouping techniques for graphic design. Rosenholtz et al. [5] utilized Gestalt laws to create a simple model for perceptual grouping for arbitrary images. Zhen et al. [34] established a model to merge web page content into semantic blocks by simulating human perception. Yu et al. [35] proposed a process to analyze the structure of a web page and organize the web elements into a two-level hierarchy. However, these web-oriented grouping techniques commonly require a formal HTML structure. Recently in infographic design, InfoVIF [11] analyzed the presentation flow of static infographics and extracted the visual groups using Gestalt principles of proximity and similarity. InfoMotion [6] extends InfoVIF to reconstruct visual groups using a clustering algorithm. However, these two recent techniques use carefully designed rules to pick elements into a visual group, which may ignore other design aspects. In addition, they both focus on the flow structure extraction but not the hierarchy in the grouping of information. One similar work introduced a computational method to group DOM elements in web pages [14]. Nevertheless, they manually defined four objective functions that are hard to extend and improve. Another similar work
by O’Donovan et al. [24] proposed a region-based hierarchical segmentation. The algorithm takes graphic layouts as input and then cuts the region into two sub-regions iteratively. This approach requires the design to follow a grid layout, and there should be non-overlapping visual elements. Compared to these works, our approach does not have the constraints for manually defined rules or strict requirements for the layout.

2.3 Data collection for layout design

To support research on layout design in information presentations, a range of works construct datasets by collecting designs from public online websites. In the data visualization community, Chen et al. [36] built a dataset containing 360 images of multiple visualizations collected from academic publications and made annotations of layouts. InfoVIF [11] provides a large dataset of 13,245 infographics in image format created to explore the visual information structure in infographics. Deka et al. [37] presented Rico, the largest repository of mobile app designs by crowdsourcing, created to support data-driven applications such as user interface (UI) layout generation. Webzeitgeist [38] is another design repository of over 100,000 web pages with the statistical description of layouts. To support layout generation, Zheng et al. [19] collected a corpus of 3919 online magazine pages with rich annotations, including categories, semantic layouts, and textual summaries. Compared with these studies, we aim to construct a large-scale layout dataset that contains rich spatial summaries. Therefore, we collected 23,072 well-designed information presentations from public websites. We parsed all the related spatial features of each visual element.

3 Expert interviews

To help us understand the hierarchical grouping used in the design, we conducted a preliminary interview with ten graphic designers (eight females, aged 23-34). These designers have 6.3 years of design experience on average (from 2 to 16 years) and have professional training in design schools. They are all familiar with designing information presentations. During the interview, we guided them to explain their design experience with questions under two primary topics: benefits from hierarchical grouping and criteria for hierarchical grouping. We summarize their feedback as follows:

1) Benefits from hierarchical grouping

As most professional graphic design software (i.e., Adobe Illustrator, Sketch) supports the hierarchical grouping, all the interviewees are familiar with this function. According to our interviewees, there are two reasons why they use hierarchical grouping in their information presentation files. The first major reason (mentioned by eight interviewees) is efficiently supporting layout design. For example, if a designer wants to align two groups of items, he or she can easily select one group at the proper level of hierarchy and then align it to another group. Otherwise, he or she has to operate each item in the group, which can significantly reduce the efficiency. As another example, one expert in the interview said: “I often need to create web UI designs with various sizes to accommodate to different screen sizes, such as mobile, tablet, and laptop. A proper hierarchical grouping can save me lots of time to revise the layout.” Second, four interviewees stated that hierarchical grouping can clearly segment a graphic design, assisting collaboration with other designers on the same document. On the other hand, all the designers agreed that a design without a proper hierarchical grouping may cause trouble. Specifically, one designer commented that “Without proper grouping, it is more possible to make mistakes when adjusting the layout and users have to undo the operation.” The designers also stated that when reusing the design files in the wild, they have to manually adjust the hierarchical grouping of the visual elements before editing, which may burden their design process.

2) Criteria for hierarchical grouping

When discussing the best practice of grouping, designers mentioned several criteria. First, the items in a cohesive unit should have a high probability of being grouped together. For the example of Fig. 1(a), the item in the list, which consists of an icon, a title, and a description to convey a single message, should be grouped together because these elements form a highly cohesive unit. Second, designers agreed among themselves on the significance of spatial distance when grouping elements. Elements relating to each other should be grouped close together. Designers also admitted that they only considered grouping elements in the neighborhood in most situations. Finally, the grouping was suggested to follow a visual hierarchy, which plays a key role in the planning of the layout to help viewers navigate through the information presentation more easily.

Design requirements. Our goal is to propose an automatic approach to recovering hierarchical grouping from information presentations. To achieve the goal, we summarize two design requirements based on the feedback gathered from the interviews with designers.

R1 Capturing relatedness. The first requirement is to capture relatedness between visual elements, which can be affected by many factors, including the types and spatial features. The technique should be intelligent enough to predict relatedness by considering as much information as possible.

R2 Constructing a hierarchical structure. The grouping result should be a hierarchical tree structure according to the visual hierarchy. A parent group can contain multiple sub-groups. However, a sub-group cannot be across multiple parent groups. In this way, the exact group can be easily accessed from its parent group.
To fulfill these requirements, the design of our technique consists of two major modules (1) a deep learning model for predicting pairwise relatedness (Sect. 5) and (2) a graph-based algorithm for grouping (Sect. 6). In the first module, the deep learning model can understand the relatedness between visual elements by learning from labeled training data \( (R1) \). In the second module, we model the layout as a graph model to represent the adjacent connection. Then, we propose a graph-based algorithm to group the visual elements from bottom to top hierarchically \( (R2) \).

4 Collection of information presentations

To understand information presentations, we construct a large-scale dataset of information presentations with diverse layouts from the Internet. First, we describe the procedure to collect the data. Then, we analyze the statistical results and extract the spatial features.

4.1 Collection procedure

We chose PowerPoint files as the target file format to construct our dataset because they are the most commonly used information presentations. Each slide in the PPT file is an information presentation that consists of a set of graphical and textual elements. Furthermore, PPT files contain the actual spatial information for each visual element, which can be parsed by an open-source tool.\(^1\)

Therefore, we can obtain accurate positions of visual elements from information presentations without using any computer-vision methods such as InfoVIF \(^{11}\).

To ensure the design quality and diversity, we collected PPT files through two key steps: (1) searching a large number of candidates in the wild and (2) selecting proper slides to include in our dataset.

In the first step, we searched for public websites such as SlideShare\(^2\) that are used for sharing slides. We also searched related keywords such as “slide template”, “marketing slides” and “presentation design” in popular search engines to collect open-source PPT files. We included as many slides as possible in this step. At the end of this step, we downloaded approximately 1000 PPTs in total, including more than 40,000 information presentations.

In the second step, we selected infographics-related information presentations from these candidates to construct our dataset. To ensure that the layouts of information presentations are diverse and qualified, we define the following four criteria to filter the slides.

- **C1** To ensure that visual elements of an information presentation are worth grouping, we ask that slides should have more than three visual elements. As the counterexample is shown in Fig. 2(a), it only includes an icon shape and two textual objects. Too few elements are not valuable to be grouped.

- **C2** To ensure the diversity of layouts, we ask that slides should contain at least two visual groups. As we can see in the counterexample in Fig. 2(b), which is a title page, the four elements (a rectangle, an icon shape, and two textual objects) form a coherent group to show a single message.

- **C3** To ensure an information presentation is visually attractive and informative, we ask that slides should include both graphical and textual elements. The counterexample in Fig. 2(c) presents a slide with only text information.

- **C4** The elements in the slide should be arranged in a well-organized structure. It also means the elements should form an intuitively regular pattern \(^{11}\). In Fig. 2(d), a counterexample shows a layout whose elements are disordered to some extent.

We wrote a simple program to filter out invalid slides (C1 and C3). As C2 and C4 are related to subjective judgment, human effort is involved in assessing whether a slide is qualified. Two authors of this paper manually filtered these slides. Both of them are experts in designing information presentations. One has studied in a design school for three years, and the other has worked on information presentation for over five years. Each slide takes one author approximately 5 seconds to decide whether to keep it or remove it. They kept the cases they were unsure about and met every three days to discuss these unsure cases and achieve consensus. The whole process took seven days (about six hours per day) to look through all the slides. Figure 3(e) and Fig. 3(f) are two typical information presentations in-
They have diverse visual elements and clear layout structures.

4.2 Results

We finally collected a total of 23,072 slides with 620,878 visual elements. The average number in each slide is 26.9. Figure 3(a) presents the distribution of the number of elements in our data collection. The majority of the slides (83.7%) contain less than 40 elements. More than half of the elements in our data collection. Figure 3(b) presents the distribution of the number of visual elements. The average number in each slide is 26.9. We finally collected a total of 23,072 slides with 620,878 visually related information, such as text and images. The final dataset only keeps the features of each visual element mentioned before, which is enough for the study of layout structure. We open-sourced the dataset. More discussion about the usage of the dataset is provided in Sect. 8.2.

5 Prediction of pairwise relatedness

Before we group visual elements in an information presentation, we first consider the relatedness between elements, which also implies the probability of grouping two elements together (R1). This section introduces a deep learning model to this end. First, the basic model architecture and the working mechanism is presented. Next, we introduce an improvement to enhance the performance. Finally, the training process and model implementation is briefly described.

5.1 Model architecture

To predict the relatedness between any two visual elements in an information presentation, we propose a Transformer-based [39] model for pairwise prediction.
There are two main reasons for doing so. First, as an information presentation could be formalized as a design sequence [20], it is reasonable to adopt a sequence model in our task. Compared with traditional sequence models such as recurrent neural networks, the self-attention mechanism [39] in Transformer models can capture long-range contextual information in the whole layout design. Specifically, understanding the spatial information of all visual elements in the design as global context is helpful when predicting the relatedness between two visual elements. Second, the language model on the Transformer architecture has been widely adopted. It would be easy to adapt the pre-training to have a promising improvement with the growth of data collection.

The model architecture is demonstrated in Fig. 4. It relies on supervised training based on human-labeled information presentations (Fig. 4(a)). The input of the model is a series of visual elements parsed from the information presentation (Fig. 4(b)), and the output is an association matrix of predicted relatedness between elements. The core model is based on the Transformer backbone, which is similar to the latest work for learning layout representation [40]. It consists of three components: (1) an embedding layer to represent visual elements with heterogeneous features (Fig. 4(c)), (2) a Transformer encoder for contextualized representation (Fig. 4(d)), and (3) a prediction layer to predict pairwise relatedness (Fig. 4(e)). The model is trained to minimize the cross-entropy loss between the output and the ground truth (Fig. 4(f)).

5.2 Working mechanism

Pre-processing. Consider the example layout design in Fig. 4(a). The pre-processing module (Fig. 4(b)) first parses the design file to extract all visual elements in the design and then orders it into a sequence of visual elements according to the z-index. We represent the design sequence as: \( [x_1, x_2, \ldots, x_n] \), where \( x_i \) indicates the \( i \)-th element in the sequence. According to the statistics in Sect. 4.2, we set the maximum sequence length to 128, as most of the slides in our dataset contain 128 elements or fewer. For efficient batch training, we fill the design sequence to this maximum length with special padding tokens.

Embedding. To abstract visual elements into a data format that is understandable to the Transformer model, we convert each visual element \( x_i \) into a set of feature embedding vectors. For the numerical features, including the z-index, position, size, and rotation, we embed them using the positional encoding method [39]. For the remaining two categorical features type and alignment, we embed them using a learnable embedding table. The final representation of a visual element is a single embedding vector \( e_i \) by concatenating all the feature embedding vectors as follows:

\[
e_i = \Theta(e_i^t \oplus e_i^s \oplus e_i^r \oplus e_i^a \oplus e_i^a)
\]

where \( e_i^t \) to \( e_i^a \) denote the feature embedding vectors for type, z-index, position, size, rotation, and alignment, respectively. \( \Theta \) is a non-linear transformation function. In this way, the output of the embedding layer is a sequence of visual element embeddings: \( [e_1, e_2, \ldots, e_n] \), which will serve as the input to the Transformer encoder.

Encoding. The Transformer encoder (Fig. 4(d)) updates the element embeddings into contextualized vectors \( [h_1, h_2, \ldots, h_n] \). According to the query-key-value attention mechanism in Transformer, the model will involve three learnable weight matrices: query weight matrix \( W_Q \), key weight matrix \( W_K \), and value weight matrix \( W_V \). The self-attention mechanism computes the attention score between the query of element embedding \( e_j \) and the key of element embedding \( e_i \) as follows:

\[
\alpha_{ij} = \frac{1}{\sqrt{d_k}} (e_i W_Q) (e_j W_K)^T,
\]

where \( \sqrt{d_k} \) is the scaling factor to stabilize gradients during training. The output of the Transformer encoder is the weighted average of the value vectors of each element embedding \( e_j \) with normalized attention scores:

\[
h_i = \sum_j \frac{\exp(\alpha_{ij})}{\sum_k \exp(\alpha_{ik})} e_j W_V.
\]

Predicting. In the prediction layer (Fig. 4(e)), we predict the pairwise relatedness between every two elements. We use an association matrix \( R \) with size \( (n \times n) \) to denote the prediction result. In the matrix, each item \( R(x_i, x_j) \) represents the relatedness between element \( x_i \) and \( x_j \) ranging from 0 to 1, where 1 means the model is confident that the two elements should be assigned in the same group and 0 means there is no possibility to group them. The predicted
relatedness $R(x_i, x_j)$ is the average value of two related attention scores $r_{ij}$ and $r_{ji}$:

$$r_{ij} = \frac{\exp W^q_{r} h_i \cdot W^r_{r} h_j}{\sum_j \exp W^q_{r} h_i \cdot W^r_{r} h_j}$$  \hspace{1cm} (4)

where $W^q_{r}$ and $W^r_{r}$ are two learnable matrices for query and memory according to the query-memory attention mechanism [41]. The $r_{ij}$ denotes the attention score between the query of $h_i$ and the memory of $h_j$; vice versa for $r_{ji}$.

5.3 Improvement
The spatial relations between elements, including the relative z-index and relative position in both the x-axis and y-axis, are informative for measuring the relatedness between elements. However, the basic attention mechanism in the current Transformer encoder only considers the absolute position of visual elements, while the relative information is not explicitly modeled.

To better utilize this information, we improve the model by incorporating the spatial-aware attention mechanism [42, 43]. Specifically, we compute a new attention score $\alpha_{ij}$ by adding three learnable biases (relative position biases $b^x$ and $b^y$, relative z-index bias $b^z$) to the original score as follows:

$$\alpha_{ij} = \alpha_{ij} + b^x + b^y + b^z.$$  \hspace{1cm} (5)

We have empirically verified that adding this spatial-aware attention mechanism can increase the accuracy of the model, which our technique will use by default. The evaluation of the improvement is presented in Sect. 7.1.

5.4 Training
To train the prediction model, we first label the relatedness between elements as ground truth. The labeling process is simple: PowerPoint is used to group the related items (Fig. 4(a)). Each pair of elements is labeled as a binary value (one if they are in the same group or zero if not). It is worth noting that this labeling process only needs a single grouping hierarchy, so that general users without a design background can easily label the slides based on their perception and cognition. In addition, we expand the size of labeled training data by crowdsourcing as the task is easy to perform. The training objective is to minimize the cross-entropy loss between the association matrix from the model and the ground truth matrix (Fig. 4(f)).

5.5 Implementation details
The model was implemented in PyTorch [44]. All the trainable parameters are randomly initialized and updated via the Adam optimizer [45], with a learning rate of 0.0001. The training process enables a 0.3 dropout rate to avoid over-fitting. The maximum length of the input design sequence is 128 elements. The batch size is set to 16. The model was trained on an Nvidia Tesla-V100 (16 GB) GPU.

6 Grouping of visual elements
This section introduces a bottom-up approach to recover the hierarchical grouping from layouts of visual elements. We first define a graph model of the layout based on the connections of adjacent elements ($R2$) and then propose a graph-based algorithm to group the visual elements hierarchically ($R3$).

6.1 Graph modeling
According to the expert interviews, designers generally follow the principle of proximity to organize the layout and place related elements adjacently. Therefore, we only consider grouping elements in the neighborhood in our implementation. Thus, we can explicitly form a graph model to represent the layout using the spatial features of the visual elements (Fig. 5). Formally, we define the graph model $G$ of the information presentation, where nodes are visual elements $[x_1, x_2, \ldots, x_n]$ and edges are connections between the neighborhood elements, where we denote the edge that connects elements $x_i$ and $x_j$ as $(x_i, x_j)$. The edge includes a weight of proximity that can be computed by the spatial distance between two elements (denoted as $d(x_i, x_j)$). We use the largest distance among three perspectives: Euclidean distance, horizontal distance, and vertical distance [11].

6.2 Hierarchical grouping algorithm
Next, we use the bottom-up approach to group visual elements hierarchically. Our hierarchical grouping algorithm is inspired by the previous graph-based algorithm for image segmentation [46, 47]. Their objective is to group pixels or regions that have a similar visual appearance. Instead, our objective is to group visual elements relatively and spatially. The basic idea of the algorithm is merging small groups into a larger group iteratively by considering relatedness and spatial distances between visual elements in different groups. The algorithm works as follows (Algorithm 1).
Algorithm 1: Hierarchical Grouping

**Input**: graph model $G$, association matrix $R$

**Output**: hierarchies of grouping $\mathcal{H} = [H_0, H_1, \ldots ]$

1. $H_0 \leftarrow \{x_1, x_2, \ldots , x_n\} \in G; \mathcal{H} \leftarrow [H_0];$
2. edges $\leftarrow \text{sortByWeight}(\text{edges} \in G);$ 
3. $\tau \leftarrow \tau_{\text{initial}}; \ T \leftarrow T_{\text{initial}};$
4. while $|H_q| > 1$ do
5. \hspace{1em} $q \leftarrow q + 1; \ H_q \leftarrow H_{q-1};$
6. \hspace{1em} for $(x_i, x_j) \in \text{edges}$ do
7. \hspace{2em} if $x_i$ and $x_j$ in two different groups $G_a$ and $G_b$ then
8. \hspace{3em} $C_1 \leftarrow R(x_i, x_j) > T$
9. \hspace{3em} $C_2 \leftarrow d(x_i, x_j) \leq \text{MID}(G_a, G_b, \tau)$;
10. \hspace{3em} if $C_1 \land C_2$ then
11. \hspace{4em} $H_q \leftarrow \text{merge}(G_a, G_b)$
12. \hspace{end}
13. \hspace{end}
14. \hspace{if} $|H_q| < |H_{q-1}|$ then
15. \hspace{1em} $\mathcal{H} \leftarrow H_q;$
16. \hspace{end}
17. \hspace{1em} $T \leftarrow x \times T; \ \tau \leftarrow \beta \times \tau;$
18. \hspace{end}
19. return $\mathcal{H};$

The input of the algorithm is the graph model with visual elements $\{x_1, x_2, \ldots , x_n\}$ as nodes and all connections as edges $\{(x_1, x_2), \ldots , (x_i, x_j)\}$. The outputs are levels of hierarchy $\mathcal{H} = [H_1, H_2, \ldots , H_l]$ from bottom to top, where $H_l$ contains the groups at the $l$th level of hierarchy. We start with the initial grouping $H_0$, where each visual element is assigned to an individual group (line 1). Next, we sort edges by non-decreasing spatial distance (line 2) and initialize two parameters $\tau$ and $T$. $\tau$ is the threshold parameter that controls the degree to which the distance between two groups must be greater than their internal distances in order for there to be evidence of a boundary between them [47]. $T$ is the threshold parameter for the relatedness (line 3). These two parameters influence the granularity of the grouping. A larger $\tau$ and a smaller $T$ lead to a higher level of hierarchy with larger groups. After the initialization, the algorithm begins to merge small groups into a larger one iteratively until all the elements are in the one group (line 4-18). In each iteration, we enumerate all edges by order (line 6). If two elements in an edge (e.g., $x_i$ and $x_j$) belong to two different groups (line 7), we will consider whether to merge these two groups based on following two conditions:

**Relatedness**. The first condition is whether the predicted relatedness $R(x_i, x_j)$ between $x_i$ and $x_j$ is larger than the threshold $T$ (line 9). $R(x_i, x_j)$ can be accessed from the association matrix $R$ from our model.

**Spatial distance**. The second condition is whether the spatial distance between $x_i$ and $x_j$ is smaller than the minimal internal distance of both groups (line 8). The minimal internal distance MID is defined as:

$$\text{MID} = \min \left( I(G_a) + \frac{\tau}{A(G_b)}, I(G_b) + \frac{\tau}{A(G_a)} \right).$$  

(6)

where $A(G)$ computes the area size of the group, and $I(G)$ computes the largest distance in the minimum spanning tree MST of the group $G$:

$$I(G) = \max_{(x_i, x_j) \in \text{MST}(G)} d(x_i, x_j).$$  

(7)

If both conditions are true (line 10), the two groups will be merged into a larger group (line 11). After each iteration, if the number of groups decreases, it means the algorithm constructs a higher-level grouping $H_q$ given the previous one $H_{q-1}$ (lines 15 - 17). We will update the constant parameters by decreasing $T$ and increasing $\tau$ before starting the next iteration (line 18). The constant parameters $\alpha$ and $\beta$ can be manually tuned to adapt to different types of information presentations. In our implementation in PowerPoint files, we set $\alpha = 0.9$ and $\beta = 1.1$. Finally, the algorithm outputs all levels of hierarchy $\mathcal{H}$ (line 19) that allow selection of the desired grouping at any desired granularity (R3).

7 Evaluation

In this section, we evaluate our technique, applied to hierarchical grouping of information presentations, through a technical experiment and a user study.

7.1 Technical experiment

We first present quantitative assessments of the prediction model. In the experiment, we manually labeled 1008 slides from our data collection. Although it only includes a small portion of the data collection, it can create enough pairs of visual elements for training. The labeled slides include 27,151 visual elements, which comprise 519,415 element pairs in total. We divide the labeled data as 80/20 for training and testing.

Given a pair of visual elements in an information presentation, our approach can predict the relatedness. If the relatedness score is higher than a threshold $t$ ($t = 0.5$), our approach will predict that they are probably in the same group. We use accuracy as the evaluation metric because we have the human-labeled group truth. Formally, accuracy is equal to $M/N$, where $M$ is the number of the correct predicted pairs, and $N$ is the total number of pairs in the testing data.

We compared the performances of the following three machine learning models on the testing dataset. Note that we did not involve other layout-based machine learning
models here because most of the works focus on layout generation [21, 48], which is not applicable in the grouping task.

1) **Neural Network Baseline.** We implemented a three-layer neural network as the baseline, which uses the same input feature embedding vectors of the visual elements and outputs the results of pairwise classification.

2) **Our Transformer Model without Spatial-aware Attention.** We implemented a basic version of our model without the spatial-aware attention mechanism. It shows the power of the original Transformer architecture.

3) **Our Transformer Model with Spatial-aware Attention.** We implemented the final version of our model by adding the spatial-aware attention mechanism, which can leverage the relative position information.

We summarize the performances of these three models in Table 1. The neural network baseline approach achieves 77.49% accuracy, while our approaches outperform it significantly by 8.23% and 8.83%, respectively. The result is expected because our models leverage the Transformer architecture to capture the contextual information from the whole layout structure. Next, let us look at the performance difference between our model with and without the spatial-aware attention mechanism. The accuracy of our final model (86.32%) is slightly higher than that without spatial-aware attention (85.72%), which indicates that using relative position information can improve the relatedness prediction in the 2D layout to some extent. The experiment shows the ability of our prediction model to capture the relatedness between visual elements in information presentations.

### 7.2 User study

We conducted a user study to evaluate the quality of the generated hierarchies. Although there are several grouping techniques for graphic design [6, 11], they focus on structure extraction but not hierarchical grouping. Other related hierarchical grouping techniques [14, 24] require the design of information presentations following specific rules, such as using a grid layout or no overlaps among visual elements, which limits the scope of the techniques used. Therefore, we directly compared the results with the baseline from human designers. The experiment consists of two major phases: (1) Ten designers manually grouped visual elements hierarchically as the baseline. (2) The other 20 designers rated the results from the baseline and our approach. We hypothesized that our technique could produce a comparable result with the human designers.

**Study Data.** To prepare the study data, we first randomly sampled 50 slides from our dataset. We invited ten designers from our preliminary interview to manually group visual elements hierarchically in the slides. Before they started, we briefly introduced the grouping task and the basic usage of PowerPoint. During the task, each designer was given a PowerPoint file with ten different slides. They were asked to hierarchically group the elements in each slide. We record the completion time for each designer. Finally, each designer completed ten hierarchical grouping results, which gave us a total number of 100 labeled slides (one slide had two labeled versions from two random designers, respectively). An interesting finding was that designers may have different grouping solutions for the same information presentation. We also used our technique to automatically recover hierarchical grouping from these slides. To compare the automatic results with the manually labeled results, we constructed 100 comparison pairs. Each pair includes one result from a designer and one result from our technique for the same slide. We ensured that the generated grouping result had the same number of hierarchies for a fair comparison by deleting excess hierarchies.

**Participants.** We recruited 20 participants (14 females, aged from 22 to 31) from a design school. All the participants were familiar with design software such as Adobe Illustrator and Microsoft PowerPoint. They all confirmed having experience in making information presentations using Microsoft PowerPoint.

**Task and Procedure.** The experiment began with a brief introduction. Each participant was asked to complete 25 comparison tasks. In each comparison task, the participants were given two hierarchical grouping results and were asked to provide an answer from three choices (“A is better than B”, “B is better than A”, or “A and B are about same”). This setting was fairer and easier for participants to evaluate the relative quality of two grouping results than scoring a single result. The order of the grouping results for...
comparison was randomized. Figure 6 illustrates a snapshot of a side-by-side comparison used in one comparison task, where A is manually grouped by a designer and B is automatically recovered from our technique. Thus, we collected 25 comparisons × 20 participants = 500 responses. It took approximately 30 minutes on average (ranging from 25 to 40) for each participant to complete all tasks. After they completed the tasks, we asked them to leave comments for the result of their preferences.

Quantitative Results We analyzed the comparison results to investigate the preferences in terms of grouping quality. In general, the preferences for both sides are balanced, as demonstrated in Fig. 7(a). Specifically, 39% of the responses (195/500) remarked that the grouping results from our technique are better, and 42.4% of the responses (212/500) indicated a preference for the results grouped by designers. The rest of the responses (18.6%, 93/500) indicated that both results are approximately the same. Then, we further investigated the preferences of each participant. The mean of preferences per participant and 95% confidence intervals indicated that the preference for the designer’s grouping results is only slightly higher than our technique’s (Fig. 7(b)). According to the Student t-test (t(19) = 0.45, p = 0.66), there was no significant difference between the preferences for both sides. On the other hand, our technique can save the extra time of manual grouping, which takes about 30 minutes (M = 30.1, SD = 6.69) for designers to group elements in a ten-page slide in our user study. As a result, our technique can significantly improve efficiency and produce hierarchical grouping results comparable to human work.

Qualitative Results During the study, participants were not informed that one in each testing pair was generated by an algorithm. After the study, we asked their opinions about the two results. All participants commented that they did not notice that the grouping was completed by machine. In participants’ opinion, most of the grouping results were about the same, but they might select one of them according to their personal preferences. For example, P4 commented: “Generally, there are no big differences between the two sides. However, when I look at a graphic design, I will think about how to refine the layout. Therefore I would love to choose the proper grouping that may make my design process more convenient.” P12 said: “Most differences are very slight for me to give a choice. I think it is reasonable because designers may have different opinions when grouping the elements”. After we told them that one side was from an automatic approach, we asked them to comment on the effectiveness and usefulness of the technique. Eighteen participants appreciated the technique that might significantly improve the design efficiency. Furthermore, some of them offered some suggestions for future works. For example, P9 commented that it would be more convenient if each group could be automatically given a proper group name. P7 said: “I hope the technique can give personalized suggestions by learning from my grouping behaviors. For example, I can demonstrate operations in one design file. Then, I want all my downloaded public design resources to be processed as I demonstrated.” Although most participants gave positive feedback, two participants worried that the automatic grouping of elements might limit the creativity of the design to some extent. One of them (P10) said that the grouping structure may affect the designer’s strategy to the downstream tasks of layout design.

7.3 Sample grouping results
The gallery in Fig. 8 demonstrates eight hierarchical grouping results recovered by the automatic approach. (a) and (b) have three hierarchies, and the others (c)-(h) have two hierarchies. Our automatic approach has the ability to group visual elements properly at each level of granularity.

8 Discussion
8.1 Failure cases and limitations
We observe some failure cases during the evaluation. We summarize two main causes of the failure. One typical failure is that the current approach cannot understand the semantic meaning of a user-defined shape. For example, Fig. 9(a) shows a user-defined shape on the left that is composed of multiple freeform shapes. Our technique favors grouping them all into a single group at each hierarchical level. However, the designers in our user studies prefer to split it into two parts as a “computer” and a “phone” at the lowest level of grouping hierarchy. Although it is easy for human beings, our current technique cannot understand the semantic meaning of these shape compositions. Another primary cause of failure is that some complex connectors may affect the grouping accuracy. Simple connectors are lines or arrows to connect visual elements and express their relations. However, the complex
Figure 8  Eight examples of hierarchical groupings recovered by our technique. (a) and (b) have three hierarchies, and the others (c)-(h) have two hierarchies.

Figure 9  Failure cases. (a) our technique failed to segment two semantic groups of freeform shapes; (b) it was affected by connectors.

one might be composed of multiple elements. For the example in Fig. 9(b), the connectors among three head portraits are a composition of a dashed line (line), a circle (preset shape), and an icon (freeform shape). Designers can recognize these connectors and group the head portraits with the corresponding text below. However, our current technique groups all the above visual elements as a whole by mistake. We believe that feeding more training data can mitigate this problem if the model can see more semantic groups. Our technique also has some limitations in terms of our research scope. The first limitation is that our current approach does not involve the content of each visual element, covering text, picture, and shape. Understanding the semantic meaning of the content may assist in understanding the layout structure, which is a promising future work. In addition, we could involve more semantic features, such as role labels (e.g., title, subtitle, and body). The second limitation is related to user preferences. As we found in the user study, there are multiple ways to group visual elements. Even two professional designers may have different strategies to perform the element grouping task. Our technique can provide the grouping results at different granularities but fails to adjust the strategy according to the user preference. As a participant in the user study mentioned, a promising area for future research might be learning from the user’s design behaviors. Third, this paper does not attempt to build a complete perceptual model of visual grouping. Instead, we propose a learning-based method for hierarchical grouping as a design aid within the domain of information presentations. To understand how designers group the visual elements, more theoretical research on visual perception should be studied further.

8.2 Usage scenarios

1) Usage of the dataset

We see our dataset of information presentations as a useful starting point for exploring layout structures in graphic design. In addition to our grouping research, this dataset is also helpful for researchers in other tasks. For instance, it can help researchers understand the design demographics of information presentations as the collected data are created by various designers. In addition, it can enable a series of studies for design-based machine learning, such as layout clustering, layout classification, layout recommendation, and layout generation. Moreover, this dataset can also serve as a rich design resource to inspire designers when designing layouts.

2) Usage of the technique

The technique of automatic hierarchical grouping can be applied to the design resources in the wild. Although professional graphic design tools (e.g., PowerPoint and Adobe
Illustrator) provide the grouping function for users to organize the grouping structure, the given grouping information in the existing information presentations can be very noisy as designers might only care about the final presentation during the design process. Some unintentional actions or mistakes may also affect the grouping results. Our technique can fix this problem by producing hierarchical groupings for a large batch of information presentations in a short time.

9 Conclusion
In this paper, we present an automatic approach for reverse-engineering information presentations. We first collected a large-scale dataset with 23,072 well-designed slides as our research resource. Based on the dataset, we propose a technique to recover hierarchical grouping from layouts of visual elements. Our technique incorporates the advanced deep learning model to understand the relatedness between visual elements. It also includes a bottom-up graph-based algorithm to produce the hierarchical grouping by considering the relatedness and spatial distances. A technical experiment and a user study evaluate the power of the technique, which demonstrates comparable ability with human designers. We believe that future applications can benefit from our contributed dataset and technique.

Acknowledgements
Danqing Shi did this work at Microsoft Research Asia, supervised by Dr. Weewei Cui.

Funding
This work was done during Danqing Shi’s research internship at Microsoft Research Asia.

Availability of data and materials
The datasets generated during and/or analyzed during the current study are available in the reip repository: https://github.com/sdq/reip

Declarations

Competing interests
The authors declare no competing interests.

Author contributions
Material preparation, data collection and model design were performed by DS, WC and DH. The first draft of the manuscript was written by DS and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Author details
1 Tongji University, Shanghai, China. 2 Microsoft Research Asia, Beijing, China.

Received: 5 November 2022 Revised: 14 January 2023
Accepted: 3 March 2023 Published online: 15 June 2023

References
1. Lok, S., & Feiner, S. (2001). A survey of automated layout techniques for information presentations. In The 1st International Symposium on Smart Graphics (pp. 61–88). New York: ACM.
2. Williams, R. (2015). The non-designer’s design book: design and typographic principles for the visual novice. Upper Saddle River: Pearson Education.
3. Desolneux, A., Moisan, L., & Morel, J.-M. (2004). Gestalt theory and computer vision. In Seeing, thinking and knowing (pp. 71–101). Berlin: Springer.
4. Wright, P. (1999). American Association for Artificial Intelligence Technical Report FS-99-04, 1–9. Menlo Park: AAAI Press.
5. Rosenholtz, R., Twarog, N. R., Schneck-Bielefeld, N., & Wattenberg, M. (2009). An intuitive model of perceptual grouping for HCI design. In Proceedings of the SIGCHI conference on human factors in computing systems (pp. 1331–1340). New York: ACM.
6. Wang, Y., Gao, Y., Huang, R., Cui, W., Zhang, H., & Zhang, D. (2021). Animated presentation of static infographics with infomotion. Computer Graphics Forum, 40(5), 507–518.
7. Xu, A., Huang, S.-W., & Bailey, R. (2014). Voyant: generating structured feedback on visual designs using a crowd of non-experts. In Proceedings of the 17th ACM conference on computer supported cooperative work & social computing (pp. 1433–1444). New York: ACM.
8. Hornof, A. J. (2001). Visual search and mouse-pointing in labeled versus unlabeled two-dimensional visual hierarchies. ACM Transactions on Computer-Human Interaction, 8(3), 171–197.
9. Poco, J., & Heer, J. (2017). Reverse-engineering visualizations: Recovering visual encodings from chart images. Computer Graphics Forum, 36(3), 353–363.
10. Hoque, E., & Agrawala, M. (2019). Searching the visual style and structure of d3 visualizations. IEEE Transactions on Visualization and Computer Graphics, 26(1), 1236–1245.
11. Lu, M., Wang, C., Lanir, J., Zhao, N., Pfister, H., Cohen-Or, D., & Huang, H. (2020). Exploring visual information flows in infographics. In Proceedings of the SIGCHI conference on human factors in computing systems (pp. 1–12). New York: ACM.
12. Kong, J., Zhang, K., & Zeng, X. (2006). Spatial graph grammars for graphical user interfaces. ACM Transactions on Computer-Human Interaction, 13(2), 268–307.
13. Wattenberg, M., & Fisher, D. (2004). A model of multi-scale perceptual organization in information graphics. In The 9th IEEE symposium on information visualization (InfoVis) 2003 (pp. 23–30). Los Alamitos: IEEE.
14. Koch, J., & Oulavirta, A. (2016). Computational layout perception using Gestalt laws. In Proceedings of the CHI conference extended abstracts on human factors in computing systems (pp. 1423–1429). New York: ACM.
15. O’Donovan, P., Agarwala, A., & Desigscape, A. H. (2015). Design with interactive layout suggestions. In Proceedings of the SIGCHI conference on human factors in computing systems (pp. 1221–1224). New York: ACM.
16. Todi, K., Weir, D., & Sketchplore, A. O. (2016). Sketch and explore with a layout optimiser. In Proceedings of the ACM conference on designing interactive systems (pp. 543–555). New York: ACM.
17. Lupton, E. (2004). Thinking with type: a critical guide for designers, writers, editors, & students. New York: Princeton Architectural Press.
18. Dayama, N. R., Todi, K., Saarelaiven, T., & Grids, A. O. (2020). Interactive layout design with integer programming. In Proceedings of the SIGCHI conference on human factors in computing systems (pp. 1–13). New York: ACM.
19. Zheng, X., Qiao, X., Cao, Y., & Lau, R. W. (2019). Content-aware generative modeling of graphic design layouts. ACM Transactions on Graphics, 38(4), 1–15.
20. Guo, S., Jin, Z., Sun, F., Li, J., Li, Z., Shu, Y., & Cao, N. (2021). Vinci: an intelligent graphic design system for generating advertising posters. In Proceedings of the SIGCHI conference on human factors in computing systems (pp. 1–17). New York: ACM.
21. Li, J., Yang, J., Hertzmann, A., Zhang, J., & Layoutgan, T. Xu. (2020). Synthesizing graphic layouts with vector-wireframe adversarial networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(7), 2388–2399.
22. Bylinskii, Z., Kim, N. W., O’Donovan, P., Alsheikh, S., Madan, S., Pfister, H., et al. (2017). Learning visual importance for graphic designs and data visualizations. In Proceedings of the 30th annual ACM symposium on interface software and technology. (pp. 57–69). New York: ACM.
23. Xu, P., Fu, H., Igashita, T., & Tai, C.-L. (2018). Global beautification of layouts with interactive ambiguity resolution. In Proceedings of the 27th annual ACM symposium on user interface software and technology (pp. 243–252). New York: ACM.
24. O’Donovan, P., Agarwala, A., & Hertzmann, A. (2014). Learning layouts for single-page graphic designs. IEEE Transactions on Visualization and Computer Graphics, 20(8), 1200–1213.
25. Jansen, B. J. (1998). The graphical user interface. ACM SIGCHI Bulletin, 30(2), 22–26.

26. Wickens, C. D., Hollands, J. G., Banbury, S., & Parasuraman, R. (2015). Engineering psychology and human performance. New York: Psychology Press.

27. Ren, X., & Malik, J. (2003). Learning a classification model for segmentation. In IEEE international conference on computer vision (Vol. 2, pp. 10). Los Alamitos: IEEE.

28. Arbeláez, P., Pont-Tuset, J., Baron, J. T., Marques, F., & Malik, J. (2014). Multiscale combinatorial grouping. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 328–335). Los Alamitos: IEEE.

29. Pont-Tuset, J., Arbeláez, P., Baron, J. T., Marques, F., & Malik, J. (2016). Multiscale combinatorial grouping for image segmentation and object proposal generation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(1), 128–140.

30. Xu, C., & Corso, J. J. (2016). Actor-action semantic segmentation with grouping process models. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3083–3092). Los Alamitos: IEEE.

31. Gaddde, R., Jampani, V., Kiefel, M., Kappler, D., & Gehler, P. V. (2016). Superpixel convolutional networks using bilateral inceptions. In Proceedings of the European conference on computer vision (pp. 597–613). Berlin: Springer.

32. Jampani, V., Sun, D., Liu, M.-Y., Yang, M.-H., & Kautz, J. (2018). Superpixel sampling networks. In Proceedings of the European conference on computer vision (pp. 352–368). Berlin: Springer.

33. Li, Z., Bao, W., Zheng, J., & Xu, C. (2020). Deep grouping model for unified perceptual parsing. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4053–4063). Los Alamitos: IEEE.

34. Xu, Z., & Millier, J. (2018). Identifying semantic blocks in web pages using Gestalt laws of grouping. World Wide Web, 19(5), 957–978.

35. Chen, Y., Ma, W.-Y., & Zhang, H.-J. (2003). Detecting web page structure for computing position representations. In Proceedingsofthe2018conferenceoftheNorthAmericanchapteroftheassociationforcomputationallinguistics (pp. 189–198). Stroudsburg: ACL.

36. Kamar, R., Satyanarayan, A., Torres, C., Lim, M., Ahmmed, S., Klemmer, S. R., & Talton, J. D. (2013). Webzeitgeist: design mining the web. In Proceedings of the SIGCHI conference on human factors in computing systems (pp. 3083–3092). New York: ACM.

37. Vaswani, A., Shaizer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio et al. (Eds.), Advances in neural information processing systems (Vol. 30, pp. 5998–6008). Red Hook: Curran Associates.

38. Xie, Y., Huang, D., Wang, J., & Canvesemb, C.-Y. L. (2021). Learning layout representation with large-scale pre-training for graphic design. In Proceedings of the 29th ACM international conference on multimedia. New York: ACM.

39. Wang, W., Yang, N., Wei, F., Chang, B., & Zhou, M. (2017). Gated self-matching networks for reading comprehension and question answering. In Proceedings of the 55th annual meeting of the association for computational linguistics (pp. 189–198). Stroudsburg: ACL.

40. Shaw, P., Uszkoreit, J., & Vaswani, A. (2018). Self-attention with relative position representations. In Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics (Vol. 2, pp. 464–468). Stroudsburg: ACL.

41. Xu, Y., Xu, Y., Lu Lei Cui, T., Wei, F., Wang, G., et al. (2021). Layoutlmv2: multi-modal pre-training for visually-rich document understanding. In Proceedings of the 59th annual meeting of the association for computational linguistics (ACL) 2021. Stroudsburg: ACL.

42. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer et al. (Eds.), Advances in neural information processing systems (Vol. 32, pp. 8026–8037). Red Hook: Curran Associates.

43. Kingma, D. P., & Ba Be Adam, J. (2015). A method for stochastic optimization. [Paper presentation]. The 3rd International Conference on Learning Representations (ICLR 2015), San Diego, USA.

Publisher’s Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.