A Construction Pipeline of the CoDeLayout Dataset

In this section, we present the complete data collection and annotation pipeline used to construct the CoDeLayout dataset.

A.1 Compositional Elements Collection

To collect relational element pairs for compositional layout understanding, we employ two independent and complementary strategies: a *rule-based* method and a *model-based* method. The rule-based method relies on explicit structural and geometric constraints to extract interpretable and well-defined compositional pairs. In contrast, the model-based method leverages language models to infer latent design relationships not easily captured by deterministic rules. We provide visual illustrations of both mining pipelines in Figure 1 and Figure 2, respectively.

Although our dataset primarily consists of samples mined using the rule-based method, the model-based approach offers promising potential to expand the scale and diversity of compositional pairs. Both strategies are detailed in the following subsections.

Rule-based Compositional Pairs Mining To collect high quality compositional pairs for training and evaluation, we implement a rule-based mining pipeline that extracts interacting element pairs (e^q, e^{comp}) from multi-layer layout instances $\mathcal{L} = (\mathcal{I}, \mathcal{M})$. For each element e_i its metadata $\mathcal{M}_{e_i}$ includes attributes such as `position`, `zIndex`, `opacity`, and `type`. Based on these attributes, we identify four primary composition types commonly found in graphic design: *Clipping*, *Blending*, *Overlaying*, and *Morphing*.

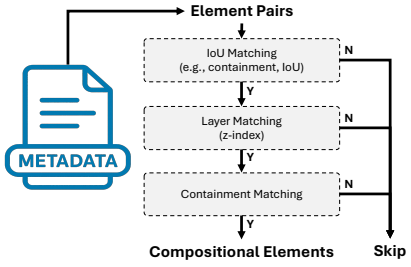

Fig. 1: Rule-based compositional pairs mining pipeline. Candidate element pairs are filtered based on geometric (IoU), layering (z-index), and containment criteria.

Clipping extracts a region from a base image as a separate layer, enabling insertion of new elements between the base and the cutout. This composition can be detected when one element is spatially contained within another with a gap in stacking order. Blending integrates different crops of the same image into stylized shapes, often requiring low spatial overlap with shape or orientation differences. Overlaying combines an element with its outline or shadow to emphasize visual hierarchy, typically represented by moderate overlap and adjacent `zIndex` together with the similar bounding box size. Morphing replaces text with graphics to produce expressive visual effects, where a small visual element appears embedded within or aligned to a text element.

To extract such pairs, we apply rule-based filters on geometry, stacking, and size. These include spatial constraints such as IoU thresholds and containment

Table 1: Rule-based criteria for extracting compositional element pairs. ✓: satisfied, ✗: excluded.

Composition Type	Z-index Rule	IoU Range	Containment	BBox Size Ratio
Clipping	$\lvert z_1 - z_2 \rvert \geq 2$	N/A	✓	N/A
Blending	N/A	≤ 0.1	✗	≤ 1.1
Overlaying	$\lvert z_1 - z_2 \rvert = 1$	[0.3, 0.8]	✗	≤ 1.1
Morphing	$z_{\text{graphic}} > z_{\text{text}}$	N/A	✓	$\approx$ Char size

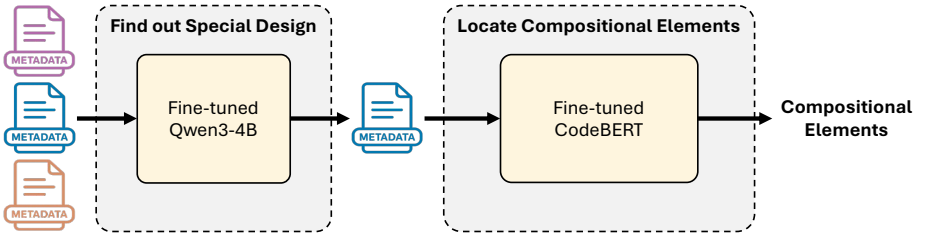

Fig. 2: Model-based compositional pairs mining pipeline. Compositional layouts are first detected by a fine-tuned Qwen3-4B model, followed by element-level compositional pair identification via a fine-tuned CodeBERT.

logic, relative layering via `zIndex`, and element size difference via bounding box size ratios. The specific thresholds and rule logic for each composition type are summarized in Table 1. We retain only elements with `opacity` equal to 1 and an bounding box size between 1% and 99% of the canvas.

This method provides an efficient and interpretable way to collect large scale, diverse compositional element pairs.

Model-based Compositional Pairs Mining To complement rule-based mining and discover compositional patterns beyond heuristic definitions, we introduce a two-stage model-based pipeline leveraging pretrained language models.

Stage 1: Compositional Layout Detection. We first determine whether a design instance contains compositional structure using a causal language model. Each layout $\mathcal{L} = (\mathcal{I}, \mathcal{M})$ is serialized into a sequence of element metadata tokens. We fine-tune the Qwen3-4B model to perform next token prediction over these sequences:

$$\mathcal{L}_{\text{mine}} = -\sum_{t} \log P_\theta(y_t \mid y_{<t}) \tag{1}$$

This objective encourages the model to learn high-order layout dependencies such as alignment, symmetry, and functional grouping. Since compositional layouts are generally more structurally complex and harder to predict sequentially, we treat layouts with higher sequence perplexity and loss as candidates for compositional element identification.

Stage 2: Compositional Element Identification. Given a detected compositional layout, we identify likely compositional elements using a probing strategy based on masked reconstruction difficulty. For each element, we randomly

Fig. 3: An example of compositional layout discovered exclusively by the model-based pipeline. The two side elements are symmetric crops derived from the same original bottle image, mirrored and stitched to create a visually complete object.

mask a subset of its attributes (e.g., position, size, zIndex), and prompt the model to reconstruct the masked values from the unmasked context. Elements with high reconstruction loss are considered structurally ambiguous or compositionally entangled, and thus likely candidates for participating in compositional relationships.

Together, this two-stage model-based pipeline offers a scalable and flexible alternative to rule-based mining. It enables the extension of the CoDeLayout dataset with structurally rich and diverse compositional annotations beyond deterministic heuristics.

Fig. 3 shows a compositional layout identified solely by the model-based pipeline. The design involves cropping a complete bottle image, duplicating it, and mirroring the crop across a vertical axis to form a new, visually perfect symmetrical bottle. This operation exhibits sophisticated design intent through spatial symmetry and perceptual completion.

Such structure cannot be captured by standard rule-based heuristics, as it does not satisfy spatial containment, overlap, or transparency conditions. This example illustrates the strength of the model-based approach in recognizing implicit compositional logic, such as mirroring and replication, thereby enabling the discovery of non-trivial layout structures beyond handcrafted rules.

A.2 Annotation Pipeline

To generate high-quality annotation for compositional understanding, we construct an automated annotation pipeline that combines structural parsing, visual grounding, and natural language QA synthesis. Each annotated instance corresponds to a compositional element pair (e^q, e^{comp}) and is processed in three consecutive stages.

Text-based Structural Analysis. In the first stage, layout metadata $\mathcal{M}$ is parsed to generate a structured factual description $T(e^q, e^{comp})$, summarizing spatial alignment and layering order:

$$T(e^q, e^{comp}) = f_{\text{text}}(\mathcal{M}, e^q, e^{comp}) \tag{2}$$

This is accomplished by prompting a large language model (GPT-4) with the element pair and completed layout metadata. The prompt includes requests for bounding box comparison, z-index difference, alignment cues, and type consistency.

Visual Verification. The second stage verifies whether the structural description is consistent with visual evidence. A large vision language model (GPT-4o) first receives a triplet of images:

$$\mathcal{I}_{\text{triplet}} = \{\mathcal{I}, \mathcal{I}_{e^q}, \mathcal{I}_{e^{comp}}\}$$

where $\mathcal{I}$ is the full design rendering, and $\mathcal{I}_{e^q}, \mathcal{I}_{e^{comp}}$ are isolated element crops. All images are encoded using base64, and those exceeding 20MB are automatically compressed and resized to meet processing constraints. The model then evaluates whether $T(e^q, e^{comp})$ is consistent with the pixel-level appearance:

$$V(e^q, e^{comp}) = f_{\text{vis}}(T(e^q, e^{comp}), \mathcal{I}, \mathcal{I}_{e^q}, \mathcal{I}_{e^{comp}}) \tag{3}$$

Question and Answer Generation. In the final stage, given the outputs from the Text-based Structural Analysis and Visual Verification stages, a question–answer pair is generated through prompt-driven instruction using GPT-4. The question $Q(e^q)$ inquires which element interacts compositionally with the reference element e^q and ask for the explanation. The answer $A(e^{comp})$ provides an explanation of the design relationship and intent using terminology grounded in the design domain:

$$(Q(e^q), A(e^{comp})) = f_{\text{qa}}(V(e^q, e^{comp}), T(e^q, e^{comp})) \tag{4}$$

A.3 Data Verification Protocol

To ensure the quality and reliability of our evaluation benchmark, we implemented a human verification protocol. The process involved three graduate annotators with expertise in computer vision and computer graphics (CV/CG), working under the guidance of professional graphic designers. To support efficient review, we developed a custom interface that simultaneously displays the rendered design, structured element-level metadata, isolated element crops, and the associated QA pairs.

Annotators evaluated each sample based on two criteria: (1) **Correctness**: whether the identified element pair exhibits a clear and structurally valid compositional interaction in the layout; (2) **Plausibility**: whether the textual description accurately reflects the underlying design intent.

Only samples with unanimous agreement on correctness among all three annotators were retained in the final CoDeLayout test set.

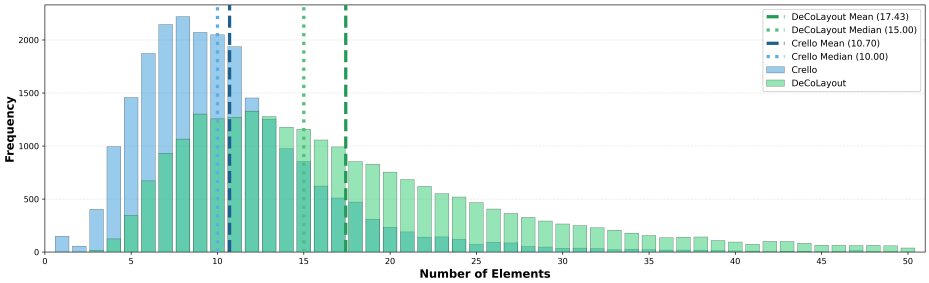

Fig. 4: Comparison of layer count distributions between CoDeLayout and Crello. CoDeLayout exhibits higher mean (17.43) and median (15.00) layer counts than Crello (mean 10.70, median 10.00), reflecting greater structural complexity and design richness.

Fig. 5: Examples from *Overlaying*.

Fig. 6: Examples from *Clipping*.

Fig. 7: Examples from *Blending*.

Fig. 8: Examples from *Morphing*.

B Additional Comparison with Existing Datasets

As shown in Fig. 4, CoDeLayout demonstrates a clear advantage in structural complexity compared to Crello [1], a design dataset of similar scale. Specifically, the average number of layers per instance in CoDeLayout is 17.43 with a median of 15.00, whereas Crello exhibits a lower mean of 10.70 and a median of 10.00.

In addition to having more layers per design, CoDeLayout also shows a broader and more uniform distribution over layer depths. This indicates that the dataset not only includes deeper hierarchies but also captures a wider range of design complexity. Such structural richness is essential for compositional layout understanding, where multi-element interactions and layer semantics play a central role.

C Additional Sample Visualization Across Categories

To better illustrate the compositional layout structures present in our dataset, we present additional representative examples from each of the four defined categories: *Overlaying*, *Clipping*, *Blending*, and *Morphing*. These examples, shown

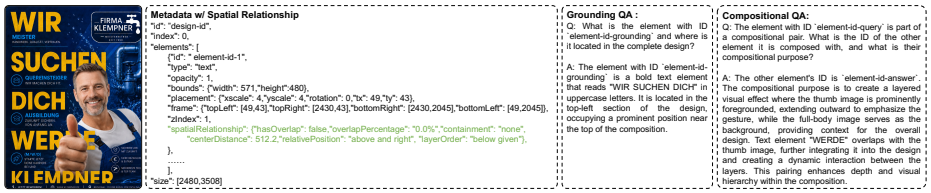

Fig. 9: A complete training instance from the CoDeLayout dataset. Each sample contains four components: (1) a high-fidelity rendered design; (2) structured metadata with layer-aware spatial attributes; (3) a Grounding QA pair aligning textual metadata of an element with its visual location; and (4) a Compositional QA pair modeling structural dependencies and design intent between entangled elements. This joint supervision enables MASON to mitigate semantic drift and structural ambiguity.

in Fig. 5 through Fig. 8, are selected to reflect complex design scenarios that require strong multi-modal and structural understanding.

Each figure highlights multiple annotated instances, where the compositional relationship between element pairs is non-trivial and often involves spatial alignment, layering logic, or semantic transformation. These samples serve to complement the main dataset visualization by illustrating the range of structural challenges covered under each category.

D Complete Training Example

To illustrate the CoDeLayout dataset and the MASON post-training paradigm, Figure 9 presents a complete training instance. The example includes four components: (1) a high-fidelity rendered design image; (2) structured element-level metadata augmented with layer-aware spatial attributes to reduce structural ambiguity; (3) a Grounding QA pair that trains the model to align textual metadata (e.g., "WIR SUCHEN DICH") with its precise visual location and semantics; and (4) a Compositional QA pair that requires the model to identify element pairs and reason about their joint design intent, such as creating depth through foreground–background overlap.

E GPT-3.5 Assisted Evaluation Details

To assess the quality of generated explanations beyond lexical-overlap metrics such as BLEU and ROUGE, we introduce a GPT-3.5-assisted evaluation protocol. The resulting metric, referred to as GPT-Score, measures the semantic alignment between the model prediction and the annotated design intent.

For each layout instance, the ground truth annotation and the explanation generated by the evaluated model are provided to `gpt-3.5-turbo-0125` using the prompt template shown in Table 2. `gpt-3.5-turbo-0125` then assigns a score from 0 to 10 based on a single criterion: **Semantic Alignment**. This criterion

Table 2: Prompt template for GPT-3.5-assisted evaluation of compositional design intent.

GPT-3.5 Prompt

You are an expert in graphic design and layout analysis. Your task is to evaluate how well the evaluated model explains the design intent of a compositional element pair in a layout.

You will be given the Ground Truth annotated intent and the Model Prediction. Rate the prediction on a scale from 0 to 10, where higher scores indicate better semantic alignment.

Semantic Alignment: Measures whether the generated explanation is semantically consistent with the annotated design intent. A high score should be assigned if the prediction correctly captures the visual and structural purpose of the composition.

Output the score followed by a brief justification. Base your judgment strictly on the provided Ground Truth.

[Ground Truth]
Composition Type: {Type}, Intent: {Annotated_Intent}

[Model Prediction]
{}

Output format:
Semantic Alignment: <Score from 0 to 10>
Reason:

Table 3: Performance on stronger backbones and larger open-source VLMs. MASON further improves with Qwen3-VL-8B, while larger open-source VLMs still lag behind. Morphing remains the most challenging category.

Model	Overlaying	Clipping	Blending	Morphing	W-Acc	Avg-Acc
MASON (Qwen2.5-VL-7B)	95.65	90.91	95.51	70.21	91.66	88.07
MASON (Qwen3-VL-8B)	95.65	93.94	96.15	76.60	93.23	90.59
Qwen3-VL-32B	92.17	83.33	85.26	53.19	83.07	78.49
Qwen3-VL-235B-A22B	92.17	78.79	91.03	57.45	85.16	79.86

evaluates whether the generated explanation accurately reflects the visual and structural purpose of the composition described in the annotation.

F Results on Stronger Backbones and Larger Open-Source VLMs

Table 3 evaluates MASON on a stronger backbone and compares it with larger open-source VLMs. Replacing Qwen2.5-VL-7B with Qwen3-VL-8B further improves both weighted and average accuracy. Morphing also improves from 70.21% to 76.60%, yet remains the most challenging category. Despite their substantially larger sizes, Qwen3-VL-32B and Qwen3-VL-235B-A22B still lag behind MASON, suggesting that compositional layout understanding cannot be solved by scaling alone.

Table 4: Module ablation on CoDeLayout under the full-data setting (type-imbalanced). DF, MA, and SP denote Direct Finetune, Multimodal Alignment, and Structural Perception, respectively. Results report category-wise, Weighted, and Average accuracy (↑). Weighted accuracy accounts for category proportions, while Average accuracy is the unweighted mean across categories.

Model	Overlaying	Clipping	Blending	Morphing	Weighted Acc.	Average Acc.
DF	93.04	83.33	94.87	65.96	88.80	84.30
DF+MA	93.91	84.85	94.87	57.45	88.28	82.77
DF+SP	96.52	89.39	95.51	68.09	91.40	87.38
MASON	95.65	90.91	95.51	70.21	91.66	88.07

Table 5: Held-out composition-type generalization of Direct Finetune and MASON on CoDeLayout. Each cell reports accuracy in the format (Direct Finetune \ MASON), with rows denoting the training type and columns the test type. Diagonal entries denote in-category evaluation, while off-diagonal entries measure generalization to unseen composition types. Only cross-category entries are color-coded: green indicates MASON performs better, and blue indicates Direct Finetune performs better. MASON achieves better transferability in most settings.

Train \ Test	Overlaying	Clipping	Blending	Morphing
Overlaying	93.91 \ 94.78	77.27 \ 83.33	78.21 \ 83.33	29.79 \ 40.43
Clipping	60.00 \ 72.17	83.33 \ 96.97	64.10 \ 54.49	36.17 \ 51.06
Blending	76.52 \ 77.39	51.52 \ 34.85	92.95 \ 96.79	38.30 \ 59.57
Morphing	22.61 \ 42.61	4.55 \ 9.09	17.31 \ 43.59	72.34 \ 82.98

G Full-Data Module Ablation

As shown in Table 4, SP remains the primary contributor to performance gains, improving weighted accuracy from 88.80% to 91.40%. MA alone provides limited benefits and substantially degrades Morphing accuracy (65.96% → 57.45%), where grounding distorted text without sufficient structural context may increase ambiguity. Combining MA and SP achieves the best overall performance, reaching 91.66% weighted and 88.07% average accuracy.

H MASON Generalization

H.1 Held-Out Composition-Type Generalization

To assess generalization, we compare Direct Finetune and MASON in a held-out composition-type setting, where each model is trained on a single compositional type and evaluated on all four types. Table 5 reports accuracy in the format (Direct Finetune \ MASON). Only cross-category results are color-coded: green indicates MASON performs better, while blue indicates Direct Finetune performs better. Same-category results are shown in white. MASON consistently outperforms Direct Finetune on same-category evaluations and generalizes better in most cross-category settings, suggesting that structural perception and

Table 6: Held-out data-source generalization on 20 compositional layout samples mined from the Crello dataset using our data collection pipeline. Results report accuracy (correct/total in brackets). MASON achieves the best overall performance and better cross-source transferability.

Model	Overlaying	Clipping	Blending	Morphing	Average Acc.
GPT-o3	80.0 (4/5)	80.0 (4/5)	80.0 (4/5)	60.0 (3/5)	75.0 (15/20)
DF	100.0 (5/5)	80.0 (4/5)	100.0 (5/5)	60.0 (3/5)	85.0 (17/20)
MASON	100.0 (5/5)	80.0 (4/5)	100.0 (5/5)	80.0 (4/5)	90.0 (18/20)

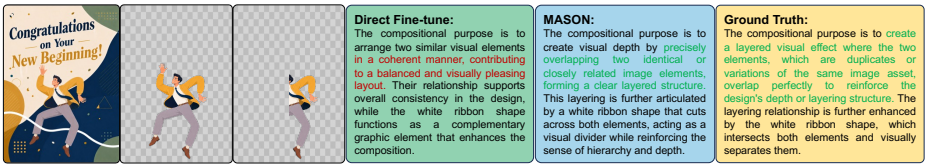

Fig. 10: Qualitative comparison of generated design intents. Given a complex compositional layout (left), the Direct Fine-tuning (DF) baseline produces a generic explanation that fails to capture precise inter-element relationships. In contrast, MASON correctly interprets the structure and identifies the white ribbon as a "visual divider," demonstrating stronger reasoning over layer-aware structural dependencies and closer alignment with the Ground Truth.

multimodal alignment help learn more transferable layout representations. One exception is generalization between Clipping and Blending, where Direct Finetune performs better, possibly because these composition types exhibit contrasting spatial patterns, with Clipping involving substantial overlap and Blending maintaining greater spatial separation.

H.2 Held-Out Data-Source Generalization

Since CoDeLayout is the first benchmark for compositional layout understanding, no directly compatible benchmark currently exists. We therefore conduct a preliminary held-out data-source evaluation on 20 compositional layout samples mined from the Crello dataset [1] using our data collection pipeline. As shown in Table 6, MASON achieves the best overall performance and demonstrates better cross-source generalization than Direct Finetune.

I Qualitative Comparison of Design Intents

To further demonstrate the effectiveness of our paradigm, we present a qualitative comparison of generated design intents. As shown in Figure 10, the Direct Fine-tuning (DF) baseline struggles to interpret complex structural dependencies and often produces generic descriptions. For example, it vaguely describes the layout as a "coherent" and "visually pleasing" arrangement, while referring to the

white ribbon merely as a "complementary graphic element" without recognizing its spatial role.

In contrast, MASON generates precise and contextually grounded interpretations that closely match the Ground Truth. It correctly identifies the compositional goal of creating visual depth by overlapping two similar image elements to form a layered structure. Moreover, MASON captures the spatial relationships between elements, noting that the white ribbon "cuts across both elements" and functions as a visual divider that reinforces hierarchy and depth. These results demonstrate that incorporating structural perception and multimodal alignment enables MASON to move beyond surface-level descriptions toward deeper compositional understanding.

References

1. Suzuki, T., Liu, K.J., Inoue, N., Yamaguchi, K.: Layerd: Decomposing raster graphic designs into layers. In: ICCV (2025)