# Probing Diagram Representations in Large Vision-Language Models

**Anonymous ACL submission**


## Abstract

Large vision-language models (LVLMs) demonstrate strong performance on diagram understanding benchmarks, yet they still struggle with understanding *relationships* between elements, particularly those represented by nodes and directed edges (e.g., arrows). To investigate the causes of this limitation, we probe the internal representation of LVLMs using a synthetic diagram dataset based on directed graphs. Our linear probing experiments reveal that signals related to diagram representations, such as nodes and edges, can be extracted from the internal states of LVLMs, including those of the vision encoder and the language model. In particular, even for diagram elements that LVLMs struggle to answer correctly when queried in visual question answering tasks, such as edge directions and connectivity between non-adjacent nodes, these signals remain extractable. This reveals a gap between the model's internal signals and its outputs. We further conducted causal intervention experiments on the outputs of the vision encoder. The results show that, for some aspects, the information identified through probing is not merely a detectable signal, but a representation that causally affects the final outputs of the model.


## 1 Introduction

Diagrams convey complex information through visual layouts that people can grasp quickly, making them an important communication medium across various domains, including academic research (Hsu et al., 2021; Kohn and Aladjem, 2006) and business (Jošt et al., 2017; Kernbach and Eppler, 2010). To support clearer communication with diagrams, many systems for diagram understanding have been studied (Kembhavi et al., 2016; Li et al., 2025).

Large vision-language models (LVLMs) (Bai et al., 2025a,b; Liu et al., 2024; Team et al., 2025) have shown strong performance on benchmarks related to diagram understanding (Zhang et al., 2025; Lu et al., 2024). However, prior studies report that LVLMs still struggle with understanding certain visual elements in diagrams, such as relational information indicated by arrows and lines (Hou et al., 2025; Zhu et al., 2025; Pan et al., 2024). Understanding the causes of these limitations requires analyzing how LVLMs internally represent visual elements in diagrams, yet no prior work has comprehensively examined this for basic elements such as nodes, edges, and global structure.

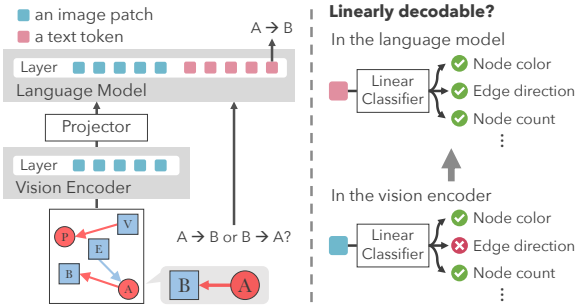


Figure 1: Overview of this study. We analyze internal representations in LVLMs using linear probing on a synthetic diagram dataset. We find that signals related to diagram information can be extracted from the hidden states of LVLMs. The locations from which these signals can be extracted (e.g., the vision encoder or language model, and image-input or text-input positions) differ depending on the diagram aspect. We also find that, even for aspects on which LVLMs often fail when asked to answer as visual question answering tasks, signals corresponding to those aspects can still be extracted from the hidden states.

To fill this gap, we comprehensively investigate how diagram-specific visual elements are internally distinguished in LVLMs. We focus on whether information about nodes, edges, and global structure is represented internally in the vision encoder and the language model. In contrast to prior analyses that focus on natural images (Gandelsman et al., 2024; Tao et al., 2024), our work targets diagrams, which rely on symbolic representations to express

| Category | Aspect $a$ | Label set $\mathcal{Y}$ |
|---|---|---|
| Single | `Node Color` | {red, green, yellow, blue, brown, orange, pink, purple} |
| | `Node Shape` | {circle, square, pentagon, hexagon, septagon} |
| | `In-deg.` | {0, 1, 2, 3, 4} |
| | `Out-deg.` | {0, 1, 2, 3, 4} |
| Multiple | `Edge Color` | {red, green, yellow, blue, brown, orange, pink, purple} |
| | `Edge Style` | {solid, dashed} |
| | `Edge Exist.` | {exist, not exist} |
| | `Edge Direct.` | {forward, backward} |
| | `Multi-hop Path` | {exist, not exist} |
| Global | `Node Count` | {1, 2, 3, 4, 5} |
| | `Edge Count` | {1, 2, 3, 4, 5} |

Table 1: The 3 categories, 11 aspects, and their corresponding label sets $\mathcal{Y}$.

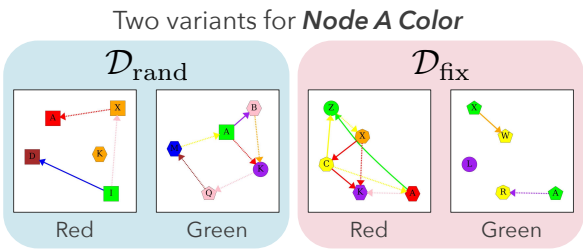


Figure 2: Examples of synthetic diagrams. Each diagram contains five nodes, and we control evaluation aspects such as node color, shape, and edge connectivity. We provide two variants: $\mathcal{D}_{\text{rand}}$, which uses random node layouts (left part), and $\mathcal{D}_{\text{fix}}$, which uses fixed layouts (right part).

structured relationships. To enable controlled analysis, we construct a synthetic diagram dataset as a testbed and apply probing (Alain and Bengio, 2017) and causal intervention (Zhang and Nanda, 2024) methods to examine the linear separability of node, edge, and global information within the vision encoder and the language model.

Our contributions are threefold. First, we introduce a synthetic diagram dataset with controllable visual elements, enabling controlled analysis while reducing the influence of domain-specific knowledge. Second, our probing analysis shows that information about nodes and edges can be extracted from the hidden states of either the vision encoder or the language model, revealing that such information exists within the model as linearly decodable signals. At the same time, we observed a gap: even for aspects that the model cannot solve in the VQA setting, the corresponding signals are still present. Third, through causal interventions on the vision encoder outputs, we confirmed that, for several aspects, the detected signals serve as internal representations of diagram visual information; that is, these signals causally affect the model's outputs. These findings provide insight into how LVLMs achieve diagram understanding.[1]

## 2 Task and Model Setup

As a testbed for our analysis, we evaluate the VQA task on synthetically created simple diagram images (graphs). This VQA task can be solved as a classification task. As shown in Figure 1, the LVLM takes as input a synthetically created simple diagram image $x$ and a question $q$, and outputs the answer $y \in \mathcal{Y}$ (where $\mathcal{Y}$ is the set of answer labels) to the question as text.

In our experiments, we primarily focus on Qwen3-VL-8B-Instruct (Bai et al., 2025a) as the main target of analysis. In addition, we conduct the same experiments on Qwen2.5-VL 7B (Bai et al., 2025b), LLaVA1.5 7B (Liu et al., 2024), and Gemma3-4B-IT (Team et al., 2025). These results are reported in Appendix D and E.

## 3 Dataset Construction

To investigate how LVLMs perceive basic visual elements in diagrams, we construct a synthetic dataset of directed-graph diagrams. The diagrams consist of nodes and edges and serve as a simplified approximation of real-world diagrams, retaining only their core visual elements. This setup allows precise control over evaluation factors such as node color and edge direction. Moreover, synthetic data helps reduce biases and shortcuts inherited from natural-image and natural-language datasets (Kafle and Kanan, 2017; Gururangan et al., 2018; Dancette et al., 2021; Healey et al., 2024), enabling fine-grained analysis and evaluation.

### 3.1 Diagram Specification

Following prior work on diagram understanding (Zhu et al., 2025; Sartori et al., 2025), each diagram satisfies the following properties: (1) Nodes and edges are colored using one of eight colors: red, green, yellow, blue, brown, orange, pink, or purple. (2) Each diagram contains 5 nodes. (3) Node shapes are one of circle, square, pentagon, hexagon, or septagon. (4) Each node is labeled with a single alphabetic character as an identifier.

[1]The code and data will be released upon acceptance.

(5) Edge styles are either solid or dashed. Further details are provided in Appendix B.1.

### 3.2 Evaluation Aspects

We define 11 aspects as basic visual information in diagrams, as shown in Table 1. These aspects are based on visual information emphasized in prior works (Silva et al., 2011; Pan et al., 2024; Sterzik et al., 2024; Tseng et al., 2025; Hou et al., 2025; Zhu et al., 2025).

Diagram understanding fundamentally requires composing multiple visual components, primarily through binding visual features with objects and aggregating information across components. These operations apply at different spatial scopes, based on which we classify the 11 aspects into the following three categories:

- **Single**: Visual information localized around a single node.
- **Multiple**: Combinations of multiple localized pieces of visual information. Since edges are defined between two nodes, their recognition requires a multi-step process: first identifying the endpoint nodes, then recognizing the edge attributes between them.
- **Global**: Visual information that cannot be determined without considering all nodes and edges in the diagram.

In analyzing each aspect, we focus on attributes of the node with identifier A (node A) and the edge connecting nodes A and B (edge AB) as representatives[2]. For example, in the `Node Color` aspect, we evaluate whether the color of node A is represented (i.e., linearly decodable) in the model.

### 3.3 Dataset Definition

For each aspect $a$, we define a dataset $\mathcal{D}$ to verify whether it can be recognized as follows:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}. \quad (1)$$

Here, $x_i$ is the $i$-th diagram image and $y_i$ is its label. Specifically, this is a dataset containing labels $y_i$ for each aspect as shown in Table 1.

In this study, we train probes to analyze the internal representations of an LVLM. To train and evaluate these probes, for each aspect, we construct two variants with different node layouts:

- $\mathcal{D}_{\text{rand}}$: For each diagram in the dataset, the node layout is determined independently at random.
- $\mathcal{D}_{\text{fix}}$: For all diagrams in the dataset, nodes with each identifier are placed at the same spatial position.

Figure 2 shows examples of each variant. For each aspect, both $\mathcal{D}_{\text{rand}}$ and $\mathcal{D}_{\text{fix}}$ consist of 100 data points per classification class.

**Training dataset for probing.** We construct the diagram dataset with random node layouts, $\mathcal{D}_{\text{train}}$, as the base for the probe training dataset. In pilot experiments, we observed that the probe achieved near-perfect accuracy on certain patches by exploiting shortcuts, such as predicting based on colors present in the input patch without identifying the target node or edge (see Appendix D.1 for details). To prevent this, we prepare an additional dataset $\mathcal{D}_{\text{rand}}^{\perp}$, which consists of diagrams in which the target node or edge does not exist, with the gold label $y = \perp$ (e.g., the question asks about node A, but node A does not exist in the image). This forces the probe to verify target existence before predicting its attributes, which resolved the shortcut issue in our experiments. $\mathcal{D}_{\text{rand}}^{\perp}$ contains the same number of samples as $\mathcal{D}_{\text{rand}}$. Using $\mathcal{D}_{\text{rand}}^{\perp}$, we define the probe training dataset as $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{rand}} \cup \mathcal{D}_{\text{rand}}^{\perp}$.

**Evaluation dataset for probing.** We prepare the evaluation dataset by constructing $M$ variants of $\mathcal{D}_{\text{fix}}$. Formally, denoting the $j$-th evaluation subset as $\mathcal{D}_{\text{fix}}^{j}$, the entire evaluation dataset is expressed as $\mathcal{D}_{\text{test}} = \{\mathcal{D}_{\text{fix}}^{0}, \mathcal{D}_{\text{fix}}^{1}, \cdots, \mathcal{D}_{\text{fix}}^{M-1}\}$. The fixed node layout, which is not seen during probe training, allows us to analyze the model' s internal representations in a controlled manner, and evaluating across $M$ variants ensures that the observed behaviors are not artifacts of a specific spatial configuration.

## 4 Preliminary Experiment: VQA

As a preliminary experiment, we evaluate the VQA performance of LVLMs on synthetic diagrams to confirm above-chance-level performance. Specifically, we assess whether the models can correctly predict the gold label $y$ given a diagram $x$ and question $q$ in a simple VQA setting. The question $q$ includes the label set $\mathcal{Y}$ as answer choices. For example, for the `Node Color` aspect, the question is "What color is node A?" with choices $\mathcal{Y} = \{\text{red}, \text{green}, \ldots\}$. The same input format is

[2]The choice of identifiers is arbitrary, and the same analysis can be applied to others such as node C.

| | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Chance level | 12.5 | 20.0 | 20.0 | 20.0 | 12.5 | 50.0 | 50.0 | 50.0 | 50.0 | 20.0 | 20.0 |
| Qwen3-VL 8B | **95.5** | **78.4** | **53.7** | **40.6** | **80.6** | **85.7** | **81.2** | **53.9** | **55.9** | **78.7** | **83.8** |

Table 2: $\mathrm{Acc}_{\mathrm{VQA}}$ for Qwen3-VL 8B. Accuracy exceeded chance level for all aspects, but `Edge Direction` and `Multi-hop Path` remain near chance, while other aspects achieve substantially higher accuracy.

used in the probing and causal intervention experiments. The full list of questions for each aspect is provided in Table 4.

### 4.1 Evaluation Metric

We define the final score as the mean accuracy across all evaluation data subsets. Formally, it is defined as:

$$\mathrm{Acc}_{\mathrm{VQA}}(q, \mathcal{D}_{\mathrm{test}}) = \underset{d_{\mathrm{fix}} \in \mathcal{D}_{\mathrm{test}}}{\mathrm{mean}}\ \mathrm{acc}_{\mathrm{VQA}}(q, d_{\mathrm{fix}}). \quad (2)$$

Here, $\mathrm{acc}_{\mathrm{VQA}}$ is the accuracy of the LVLM on the VQA task for dataset $d_{\mathrm{fix}} \in \mathcal{D}_{\mathrm{test}}$, and $\mathrm{mean}_{s \in \mathcal{S}}(\cdot)$ denotes the element-wise arithmetic mean of the values associated with each $s \in \mathcal{S}$. We judge a prediction as correct if the LVLM's output text contains the label $y$ as a substring.

### 4.2 Experimental Results

Table 2 summarizes the $\mathrm{Acc}_{\mathrm{VQA}}$ of Qwen3-VL 8B. The model achieves high accuracy on aspects such as `Node Color` and `Edge Style`, while accuracy on `Edge Direction` and `Multi-hop Path` remains near chance level. This suggests that Qwen3-VL recognizes basic visual attributes but struggles with relational properties that require identifying edge directions or tracing multi-step paths. See appendix A for additional discussion.

## 5 Probing

We investigate whether visual information in diagrams is linearly encoded within LVLMs. To this end, we perform linear probing on the hidden states of each layer and patch in both the vision encoder and the language model, and examine whether there are hidden states from which such signals can be captured. Linear probing is an effective method for identifying which parts of a model contain signals corresponding to specific information (Alain and Bengio, 2017).

### 5.1 Probe Definition

Let $\boldsymbol{h}_{l,t}$ denote the hidden state at layer $l$ and position $t$ of the vision encoder or language model

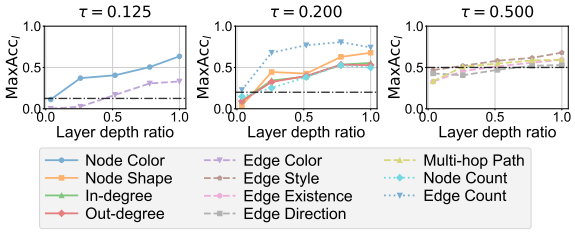


Figure 3: Layer-wise maximum accuracy in the vision encoder of Qwen3-VL 8B. The x-axis denotes the relative layer position (0 is the input layer and 1 is the final layer), and the y-axis denotes accuracy. Aspects sharing the same threshold are drawn with the same line style, which also matches the style of the corresponding black threshold lines. Accuracy for Single and Global aspects rises with depth, whereas Multiple aspects stay near the threshold throughout.

given diagram $x$ and question $q$. The probe $f$ that predicts the VQA task answer or $\perp$ from the hidden state is defined as a single-layer linear classifier:

$$\hat{y} = f(\boldsymbol{h}_{l,t}) = \underset{\mathcal{Y} \cup \{\perp\}}{\arg\max}\ \boldsymbol{W}\boldsymbol{h}_{l,t} + \boldsymbol{b}, \quad (3)$$

where $\hat{y} \in \mathcal{Y} \cup \{\perp\}$ denotes the predicted answer label, $\boldsymbol{W} \in \mathbb{R}^{(|\mathcal{Y}|+1) \times |\boldsymbol{h}_{l,t}|}$ is the weight matrix and $\boldsymbol{b} \in \mathbb{R}^{(|\mathcal{Y}|+1)}$ is the bias term.

We train probes for three components using the training dataset $\mathcal{D}_{\mathrm{train}}$ described in § 3.3: (i) the vision encoder and (ii) the image-input part of the language model, for which we train a per-layer probe $f_l$ from image-patch hidden states $\boldsymbol{h}_{l,t}$ at all patch positions $t$; and (iii) the text-input part of the language model, for which we train a separate probe $f_{l,t}$ for each layer $l$ and each token position $t$ in question $q$. Training details, including hyperparameters and how we address shortcut learning of probes, are provided in Appendix D.

### 5.2 Evaluation Metric

Let $\mathrm{acc}_{l,t}(d_{\mathrm{fix}})$ denote the accuracy of probe $f$ (the match rate between $\hat{y}$ and $y$) when using the hidden state at layer $l$ and position $t$ on evaluation data subset $d_{\mathrm{fix}} \in \mathcal{D}_{\mathrm{test}}$. The final evaluation score is

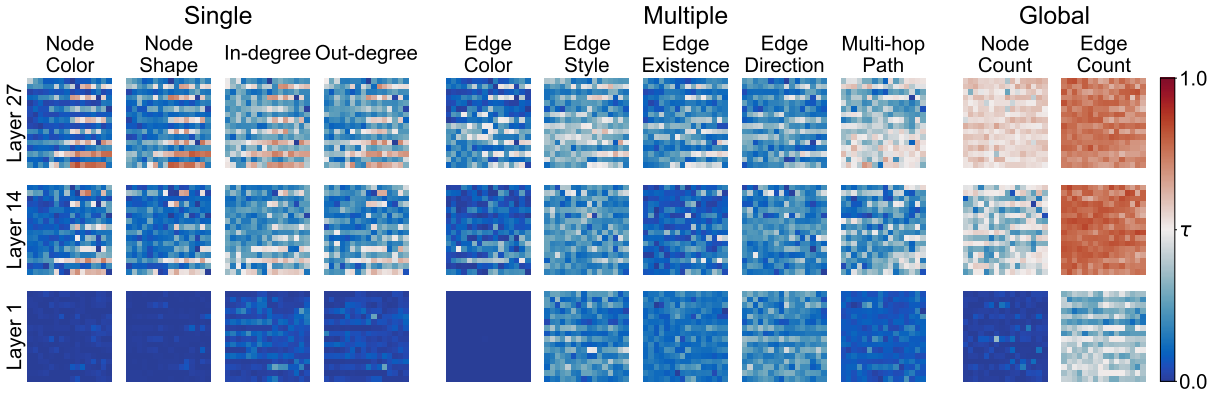


Figure 4: Position-wise accuracy in the vision encoder of Qwen3-VL 8B. Each heatmap shows accuracy by patch position for a specific layer and aspect. The node layout of the evaluation diagrams is the same as that of the diagrams in the right part of Figure 2. Single aspects localize near node positions (bottom-right), Global aspects spread across background patches, and Multiple aspects show few high-accuracy patches throughout.

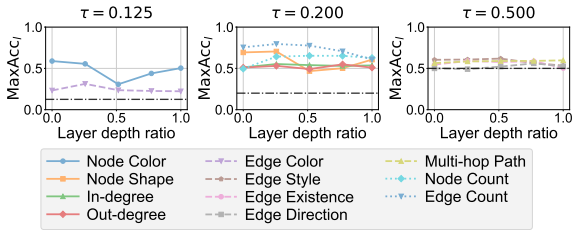


Figure 5: $\mathrm{MaxAcc}_l$ per layer in the image-input part of the language model in Qwen3-VL 8B The x-axis denotes the layer position (0 is the input layer and 1 is the final layer), and the y-axis denotes accuracy. Aspects sharing the same threshold are drawn with the same line style, which also matches the style of the corresponding black threshold lines. Compared with the final vision encoder layer, accuracy shows little change overall.

the mean accuracy across all subsets:

$$\mathrm{Acc}_{l,t}(\mathcal{D}_{\mathrm{test}}) = \underset{d_{\mathrm{fix}} \in \mathcal{D}_{\mathrm{test}}}{\mathrm{mean}} \ \mathrm{acc}_{l,t}(d_{\mathrm{fix}}). \quad (4)$$

To obtain a per-layer score of how linearly encoded the target information is, we define the maximum probing accuracy $\mathrm{MaxAcc}_l$ at layer $l$:

$$\mathrm{MaxAcc}_l(\mathcal{D}_{\mathrm{test}}) = \underset{d_{\mathrm{fix}} \in \mathcal{D}_{\mathrm{test}}}{\mathrm{mean}} \left(\max_t \mathrm{acc}_{l,t}(d_{\mathrm{fix}})\right). \quad (5)$$

That is, $\mathrm{MaxAcc}_l$ takes the maximum over positions and averages across subsets. We introduce a threshold $\tau = 1/|\mathcal{Y}|$ (chance-level accuracy excluding $\perp$), and consider probing successful when $\mathrm{Acc}_{l,t} > \tau$.

### 5.3 Experimental Results

**Single (node) and Global (counting) aspects become linearly decodable in vision encoder.** Figure 3 reports $\mathrm{MaxAcc}_l$ at each layer of the vision encoder. For aspects in the Single and Global categories, accuracy increases with depth, suggesting that signals for these aspects become more clearly linearly decodable in deeper layers. In contrast, accuracies for aspects in the Multiple category remain relatively low compared to other categories even in the final layers, suggesting that these aspects are more difficult to decode linearly from any single hidden state of the vision encoder.

**Single (node) aspects are encoded near node positions, while Global (counting) aspects are encoded broadly across background regions.** Figure 4 visualizes $\mathrm{acc}_{l,t}(d_{\mathrm{fix}})$ as heatmaps over layers and positions of the vision encoder. In this figure, we report results using the same subset $d_{\mathrm{fix}}$ as the examples shown in the right part of Figure 2 (i.e., diagrams sharing the same node layout). For aspects in the Single category, high-accuracy patches are localized around the bottom-right area where the target node A is located, indicating that signals related to node information are primarily present in that region and nearby positions. For aspects in the Global category, accuracy rises above the threshold across almost all positions in later layers, suggesting that global information is broadly distributed across hidden states, including background regions. This high accuracy on background patches suggests that the vision encoder uses hidden states corresponding to background positions as storage or aggregation points. While prior work on register tokens (Darcet et al., 2024) in Vision Transformer (Dosovitskiy et al., 2021) reports that global information is concentrated in a small subset (around 2%) of background tokens, our results sug-

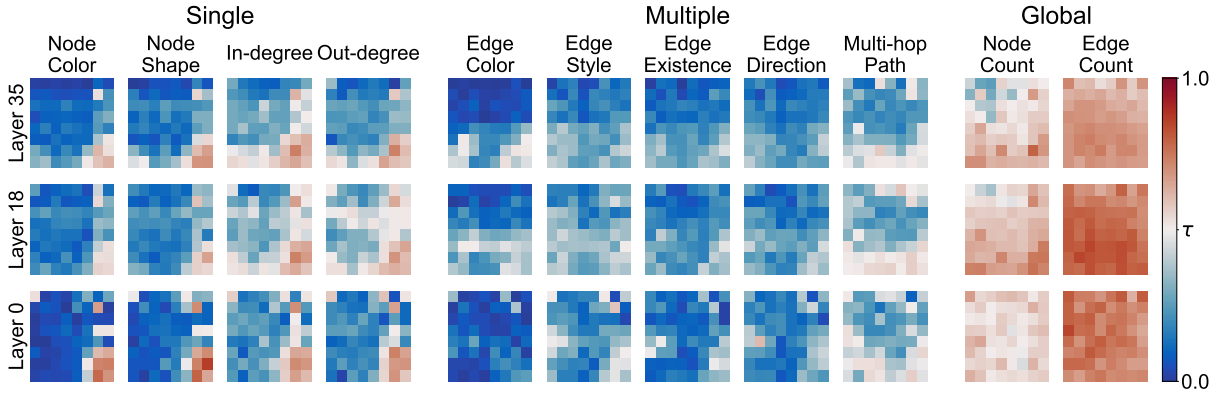


Figure 6: $\mathrm{acc}_{l,t}(d_{\mathrm{fix}})$ in the language model of Qwen3-VL 8B for a representative subset $d_{\mathrm{fix}}$. The node layout of the evaluation diagrams is the same as that of the diagrams in the right part of Figure 2. The spatial distribution of high-accuracy patches remains largely unchanged across language model layers.

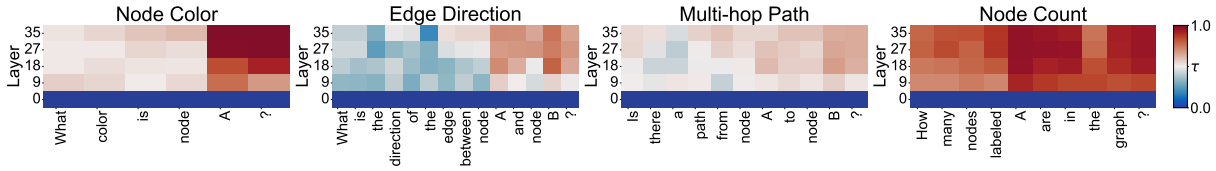


Figure 7: Probing results for text tokens in the language model of Qwen3-VL 8B. The x-axis shows token position in the question, and the y-axis shows layer position. Accuracy rises sharply at tokens specifying the target node or edge.

gest that global information is distributed across nearly all background patches in the diagram domain.

**Visual information is largely preserved in the image-input part of the language model.** Figure 5 shows $\mathrm{MaxAcc}_l$ across language model layers. $\mathrm{MaxAcc}l$ at layer 0 remains almost unchanged from that in the last layer of the vision encoder, suggesting that signals of visual information are largely preserved through the adapter. In addition, throughout the language-model processing of the image-input part, $\mathrm{MaxAcc}l$ changed only slightly across all aspects. These results indicate that linearly decodable signals are also preserved during this process. The position-wise analysis (Figure 6) also shows little change across layers in the number or spatial distribution of high-accuracy positions.

**Text conditioning constructs linearly decodable signals.** Figure 7 shows $\mathrm{acc}_{l,t}$ across layers and token positions in the text input part of the language model. Results for aspects other than `Node Color`, `Edge Direction`, `Multi-hop Path`, and `Node Count` are provided in appendix D.3. For most aspects, accuracy increased sharply at tokens that specify the target nodes or edges. This suggests that the language model selectively aggregates information from image-position hidden states into text-position hidden states, conditioned on the text input. In contrast, aspects in the Global category showed relatively high accuracy from the first text token. This suggests that global information is aggregated even without explicit conditioning. For `Multi-hop Path`, although accuracy was relatively low compared with other aspects, it slightly exceeded the threshold across a wide range of layers in the text-input part, suggesting that the model contains some linearly decodable signal for this aspect.

In summary, these results show that signals related to node and edge information are present in either the vision encoder or the language model. We also confirm that where and when these signals are observed differ depending on the aspects. In particular, for `Edge Direction` and `Multi-hop Path`, although VQA accuracy is low, signals corresponding to these types of information are still present within the model, revealing a gap between the model's internal states and its outputs. We further conduct experiments on a more realistic dataset while maintaining as much experimental control as possible. In this setting, we also find linearly decodable signals of visual information in both the vision encoder and the language model.

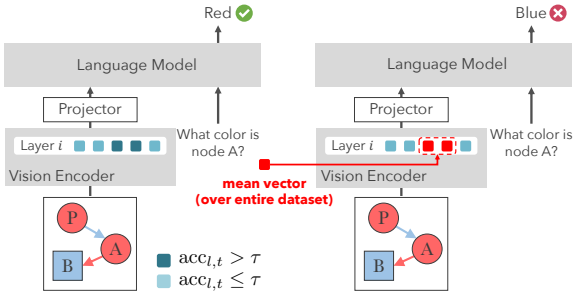


Figure 8: Overview of the causal intervention. Hidden states at patch positions whose probing accuracy exceeds the threshold are replaced with the uninformative hidden state computed over all images and positions in the evaluation dataset.

See Appendix F for more details.

## 6 Causal Intervention

In § 5, we found signals of diagram visual information in the hidden states of the vision encoder and the language model using linear probing. However, it remains unclear whether the model actually uses these signals for inference, that is, whether the signals serve as diagram representations. In this section, we focus on signals in the vision encoder and selectively corrupt hidden states with high probing accuracy. We then measure the resulting change in VQA performance to evaluate their causal contribution.

### 6.1 Method

As shown in Figure 8, our procedure is to: (1) identify the target layers and positions, (2) replace the selected hidden states with an uninformative hidden state averaged over the evaluation dataset, and (3) compare VQA accuracy (§ 4.1) between interventions on high-probing-accuracy positions and random positions.

**Target layers.** We apply interventions at early, middle, and late layers of both the vision encoder (layers 1, 14, and 27) and the language model (layers 0, 18, and 35)[3].

**Target positions.** We intervene at positions where the probing accuracy exceeds the threshold $\tau$. We replace all such target positions simultaneously in a single intervention. For dataset $d_{\text{fix}} \in \mathcal{D}_{\text{test}}$ and layer $l$, the set of target positions for intervention is defined as:

$$\mathcal{S}^l_{d_{\text{fix}}} = \{t \in \{1, \ldots, T\} \mid \mathrm{acc}_{l,t}(d_{\text{fix}}) > \tau\}, \quad (6)$$

where $T$ is the total number of positions.

**Intervention operation.** Following prior work (Neo et al., 2025), we use the dataset-mean hidden state as an uninformative reference. For each $d_{\text{fix}} \in \mathcal{D}_{\text{test}}$, we compute $\boldsymbol{\mu}_{l,d_{\text{fix}}}$ at layer $l$ as:

$$\boldsymbol{\mu}_{l,d_{\text{fix}}} = \underset{x_i \in d_{\text{fix}},\, t \in \{1,\ldots,T\}}{\mathrm{mean}} \boldsymbol{h}_{i,l,t}. \quad (7)$$

By averaging over all images and positions, the mean hidden state retains the general activation statistics of the layer but removes information specific to any particular image or position. For each input $x_i$, we replace the hidden states at intervention target positions $t \in \mathcal{S}^l_{d_{\text{fix}}}$ with this vector:

$$\tilde{\boldsymbol{h}}_{i,l,t} = \begin{cases} \boldsymbol{\mu}_{l,d_{\text{fix}}} & (t \in \mathcal{S}^l_{d_{\text{fix}}}), \\ \boldsymbol{h}_{i,l,t} & (t \notin \mathcal{S}^l_{d_{\text{fix}}}) \end{cases}. \quad (8)$$

We compute $\mathrm{Acc}_{\mathrm{VQA}}$ under this intervention setting. We refer to this setting as *accuracy-based*.

**Control experiment.** To verify that the intervention effect truly comes from targeting high-probing-accuracy positions, we conduct a control experiment that intervenes at randomly selected positions instead. Specifically, we sample a set $\mathcal{R}^l_{d_{\text{fix}}} \subset \{1, \ldots, T\}$ with $|\mathcal{R}^l_{d_{\text{fix}}}| = |\mathcal{S}^l_{d_{\text{fix}}}|$ and perform replacement as:

$$\tilde{\boldsymbol{h}}_{i,l,t} = \begin{cases} \boldsymbol{\mu}_{l,d_{\text{fix}}} & (t \in \mathcal{R}^l_{d_{\text{fix}}}), \\ \boldsymbol{h}_{i,l,t} & (t \notin \mathcal{R}^l_{d_{\text{fix}}}) \end{cases}. \quad (9)$$

We compute $\mathrm{Acc}_{\mathrm{VQA}}$ under this intervention setting. We refer to this setting as *controlled*.

**Evaluation criteria** We compute the accuracy difference between the *accuracy-based* setting and the *controlled* setting (*accuracy-based* − *controlled*). Because both settings use the same mean-vector replacement procedure, generic effects of replacement cancel out in the difference, isolating the contribution of the targeted positions. Therefore, a negative value indicates that the information at these positions causally contributes to the model's inference.

### 6.2 Results

Table 3 reports the accuracy difference between the *accuracy-based* and *controlled* settings at each

[3]We exclude vision encoder layer 0, which contains only patch embeddings, and language model layer 36, whose hidden states, except at the final position, do not affect the output.

| | Layer | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vision encoder | 1 | **−6.0** | ±0.0 | ±0.0 | ±0.0 | ±0.0 | **−1.1** | ±0.0 | ±0.0 | ±0.0 | **−5.5** | **−6.0** |
| | 14 | **−77.4** | **−46.7** | **−24.0** | **−3.6** | **−4.8** | **−0.8** | **−0.8** | +0.2 | **−0.8** | **−24.4** | ±0.0 |
| | 27 | **−78.2** | **−55.7** | **−17.5** | **−11.1** | **−32.9** | **−8.3** | +0.1 | **−0.6** | +0.6 | **−5.8** | ±0.0 |
| Language model | 0 | **−74.5** | **−52.0** | **−19.7** | **−8.5** | **−28.8** | **−9.1** | **−2.5** | **−0.5** | **−0.3** | **−3.7** | ±0.0 |
| | 18 | **−16.5** | **−10.4** | **−3.9** | +0.1 | **−30.7** | **−4.5** | **−0.5** | +1.8 | **−1.8** | **−0.8** | ±0.0 |
| | 35 | **−0.1** | +0.0 | ±0.0 | +0.1 | +0.0 | ±0.0 | ±0.0 | **−0.2** | +0.1 | +0.1 | ±0.0 |

Table 3: Accuracy difference between *accuracy-based* setting and *controlled* setting (*accuracy-based* − *controlled*). Negative values (**bold**) indicate that a decrease in accuracy was observed due to the intervention. For `Edge Count`, probing accuracy exceeds $\tau$ at all positions in most layers, making the *accuracy-based* and *controlled* settings equivalent (±0.0), precluding causal assessment for this aspect.

layer of Qwen3-VL 8B. For the Single setting and the `Edge Color`, `Edge Style`, and `Node Count` aspects, we observed a relatively large drop in accuracy compared with the *controlled* setting. This indicates that, for these aspects, the outputs of the vision encoder contain internal representations that contribute to the model's predictions. For `Edge Direction` and `Multi-hop Path`, the model's VQA accuracy is already close to chance level (§ 4.2), and thus, little change is expected regardless of the intervention. Exceptionally, for `Edge Count`, probing accuracy exceeds $\tau$ at all positions in most layers, so all patches are replaced with $\boldsymbol{\mu}_{l,d_{\text{fix}}}$ in both settings, making the two conditions equivalent (±0.0); thus, we cannot determine its causal contribution under our intervention design.

## 7 Related Work

**Diagram understanding.** Prior work has developed diagram understanding systems for tasks such as VQA (Lu et al., 2024; Wang et al., 2024) and image captioning (Hsu et al., 2021; Huang et al., 2023). While classical approaches relied on rule-based methods such as pattern matching and predefined schemas (Blostein, 1996; Song et al., 2017), methods using LVLMs are now mainstream. Understanding diagrams requires recognizing visual elements such as symbols and shapes, and capturing their relationships. Studies (Pan et al., 2024; Zhu et al., 2025; Hou et al., 2025) demonstrate that LVLMs recognize these elements adequately, while they struggle to capture relationships. The underlying cause of this limitation is unclear. Our work clarifies a contributing factor to this problem by analyzing how LVLMs internally represent diagram elements and their relationships.

**Internal analysis of LVLMs.** To understand how LVLMs process visual information, Gandelsman et al. (2024) and Tao et al. (2024) analyzed internal representations. For example, Gandelsman et al. (2024) analyzed how natural images are represented within CLIP (Radford et al., 2021), a common vision encoder for LVLMs. They revealed that multiple attention heads capture specific properties such as color, shape, and number. They also found that visual information localizes in patches corresponding to specific image regions. Tao et al. (2024) analyzed global and local information in the language model of LVLMs for natural images. They revealed that middle layers encode global information, while later layers encode local information. In contrast, our work analyzes the internal representations of LVLMs for diagrams, which, unlike natural images, represent information symbolically through text and shapes and structurally through relationships such as edge connectivity.

## 8 Conclusion

We constructed a synthetic diagram dataset and conducted probing and causal intervention analyses to clarify how LVLMs represent visual information in diagrams. Our probing analysis shows that information about nodes and edges is encoded as linearly decodable signals in the hidden states of either the vision encoder or the language model. Interestingly, we found that these structural signals remain present even when the model fails to answer the corresponding VQA queries, suggesting a gap between the model's internal signals and its outputs. Causal intervention experiments further confirm that, for several aspects, the detected signals in the vision encoder outputs causally affect the model's outputs and thus serve as diagram representations. These findings contribute to a better understanding of how LVLMs perform diagram comprehension.

## Limitations

**Coverage of real-world diagrams.** We use synthetic diagrams to minimize the influence of dataset biases and shortcut learning that can arise from natural-language and natural-image data. Although using a synthetic dataset enables well-controlled and fine-grained analysis, our setup does not necessarily capture all complex situations found in real-world diagrams. In addition, we focus on basic visual information, such as node color and shape, and edge direction. Although complex information may be represented by combining these cues, how the model represents composite visual patterns and higher-order structures remains unexamined.

**Probing methods.** Probing-based approaches for interpreting the internal mechanisms of language models have been widely studied in our field (Conneau et al., 2018; Tenney et al., 2019; Campbell et al., 2023; Li et al., 2023). However, there are some criticisms regarding the validity of probing approaches (Liu et al., 2023; Burns et al., 2023). In particular, our analysis relies on linear probes, which measure only linear decodability. The absence of linear decodability does not imply that the corresponding information is absent; it may instead be encoded nonlinearly or distributed across multiple positions. We also consider it a viable option to adopt a broader range of probing methods (Gurnee et al., 2023; Bricken et al., 2023).

## Ethics statement

We do not anticipate any significant ethical concerns because (i) we conducted no human-subject experiments, and (ii) our tasks involve only simple, synthetically generated diagrams and do not address ethically sensitive topics.

## References


Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA. Association for Computing Machinery.

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025a. Qwen3-VL technical report.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025b. Qwen2.5-VL technical report.

B Berlin and P Kay. 1999. *Basic Color Terms: Their Universality and Evolution*. The David Hume Series. Cambridge University Press, Cambridge, England.

Dorothea Blostein. 1996. General diagram-recognition methodologies. In *Graphics Recognition Methods and Applications*, pages 106–122, Berlin, Heidelberg. Springer Berlin Heidelberg.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, and 6 others. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2023. Discovering latent knowledge in language models without supervision. In *The Eleventh International Conference on Learning Representations*.

James Campbell, Phillip Guo, and Richard Ren. 2023. Localizing lying in llama: Understanding instructed dishonesty on true-false questions through prompting, probing, and patching. In *Socially Responsible Language Modelling Research*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Corentin Dancette, Rémi Cadène, Damien Teney, and Matthieu Cord. 2021. Beyond question-based biases: Assessing multimodal shortcut learning in visual question answering. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1554–1563.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. 2024. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Yossi Gandelsman, Alexei A Efros, and Jacob Steinhardt. 2024. Interpreting CLIP's image representation via text-based decomposition. In *The Twelfth International Conference on Learning Representations*.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Jennifer Healey, Laurie Byrum, Md Nadeem Akhtar, and Moumita Sinha. 2024. Evaluating nuanced bias in large language model free response answers. In *Natural Language Processing and Information Systems*, pages 378–391, Cham. Springer Nature Switzerland.

Yifan Hou, Buse Giledereli, Yilei Tu, and Mrinmaya Sachan. 2025. Do vision-language models really understand visual language? In *Forty-second International Conference on Machine Learning*.

Ting-Yao Hsu, C Lee Giles, and Ting-Hao Huang. 2021. SciCap: Generating captions for scientific figures. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3258–3264, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Chieh-Yang Huang, Ting-Yao Hsu, Ryan Rossi, Ani Nenkova, Sungchul Kim, Gromit Yeuk-Yin Chan, Eunyee Koh, Clyde Lee Giles, and Ting-Hao 'kenneth' Huang. 2023. Summaries as captions: Generating figure captions for scientific documents with automated text summarization. In *Proceedings of the 16th International Natural Language Generation Conference*, pages 80–92, Prague, Czechia. Association for Computational Linguistics.

Gregor Jošt, Jernej Huber, Marjan Heričko, and Gregor Polančič. 2017. Improving cognitive effectiveness of business process diagrams with opacity-driven graphical highlights. *Decision Support Systems*, 103:58–69.

Kushal Kafle and Christopher Kanan. 2017. An analysis of visual question answering algorithms. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1983–1991.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. 2016. A diagram is worth a dozen images. In *Computer Vision – ECCV 2016*, pages 235–251, Cham. Springer International Publishing.

Sebastian Kernbach and Martin J Eppler. 2010. The use of visualization in the context of business strategies: An experimental evaluation. In *2010 14th International Conference Information Visualisation*, pages 349–354.

Kurt W Kohn and Mirit I Aladjem. 2006. Circuit diagrams for biological networks. *Molecular Systems Biology*, 2(1):MSB4100044.

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *The Eleventh International Conference on Learning Representations*.

Xue Li, Yiyou Sun, Wei Cheng, Yinglun Zhu, and Haifeng Chen. 2025. Chain-of-region: Visual language models need details for diagram analysis. In *The Thirteenth International Conference on Learning Representations*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26286–26296.

Kevin Liu, Stephen Casper, Dylan Hadfield-Menell, and Jacob Andreas. 2023. Cognitive dissonance: Why do language model outputs disagree with internal representations of truthfulness? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4797, Singapore. Association for Computational Linguistics.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. In *The Twelfth International Conference on Learning Representations*.

Clement Neo, Luke Ong, Philip Torr, Mor Geva, David Krueger, and Fazl Barez. 2025. Towards interpreting visual information processing in vision-language models. In *The Thirteenth International Conference on Learning Representations*.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, and 108 others. 2025. gpt-oss-120b & gpt-oss-20b model card.

Huitong Pan, Qi Zhang, Cornelia Caragea, Eduard C Dragut, and Longin Jan Latecki. 2024. FlowLearn: Evaluating large vision-language models on flowchart understanding. In *ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain - Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, pages 73–80. IOS Press.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Camilo Chacón Sartori, Christian Blum, and Filippo Bistaffa. 2025. VisGraphVar: A benchmark generator for assessing variability in graph analysis using large vision-language models. *IEEE Access*, 13:21788–21810.

Samuel Silva, Beatriz Sousa Santos, and Joaquim Madeira. 2011. Using color in visualization: A survey. *Computers & Graphics*, 35(2):320–333.

Shubhankar Singh, Purvi Chaurasia, Yerram Varun, Pranshu Pandya, Vatsal Gupta, Vivek Gupta, and Dan Roth. 2024. FlowVQA: Mapping multimodal logic in visual question answering with flowcharts. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1330–1350, Bangkok, Thailand. Association for Computational Linguistics.

Dan Song, Dongming Wang, and Xiaoyu Chen. 2017. Retrieving geometric information from images: the case of hand-drawn diagrams. *Data Mining and Knowledge Discovery*, 31(4):934–971.

Anna Sterzik, Nils Lichtenberg, Jana Wilms, Michael Krone, Douglas W Cunningham, and Kai Lawonn. 2024. Perception of line attributes for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1041–1051.

Mingxu Tao, Quzhe Huang, Kun Xu, Liwei Chen, Yansong Feng, and Dongyan Zhao. 2024. Probing multimodal large language models for global and local semantic representations. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13050–13056, Torino, Italia. ELRA and ICCL.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean-Bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Chin Tseng, Arran Zeyu Wang, Ghulam Jilani Quadri, and Danielle Albers Szafir. 2025. Shape it up: An empirically grounded approach for designing shape palettes. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):349–359.

Shaowei Wang, Lingling Zhang, Longji Zhu, Tao Qin, Kim-Hui Yap, Xinyu Zhang, and Jun Liu. 2024. CoG-DQA: Chain-of-guiding learning with large language models for diagram question answering. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13969–13979.

Fred Zhang and Neel Nanda. 2024. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, Peng Gao, and Hongsheng Li. 2025. MATHVERSE: Does your multi-modal LLM truly see the diagrams in visual math problems? In *Computer Vision – ECCV 2024*, pages 169–186, Cham. Springer Nature Switzerland.

Yingjie Zhu, Xuefeng Bai, Kehai Chen, Yang Xiang, Jun Yu, and Min Zhang. 2025. Benchmarking and improving large vision-language models for fundamental visual graph understanding and reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30678–30701, Vienna, Austria. Association for Computational Linguistics.

# A Details of VQA Experiment

**Evaluation setup.** Table 4 shows the questions used to evaluate each aspect in our experiments.

**Additional discussion.** Table 2 shows that accuracy varies substantially within the Multiple category: `Edge Color`, `Edge Style`, and `Edge Existence` are answered with high accuracy, whereas `Edge Direction` and `Multi-hop Path` remain near chance. This trend suggests a gradation in the degree and type of integration required. `Edge Color` and `Edge Style` can be determined by inspecting any local part of the edge, and `Edge Existence` can often be determined by detecting whether a visible connection between the queried nodes is present. In contrast, `Edge Direction` requires identifying the two queried endpoint nodes and then locating and interpreting the arrowhead between them. `Multi-hop Path` further requires composing directional relations across multiple edges. These latter aspects impose stronger demands on compositional integration, which likely accounts for the observed performance gap.

# B Dataset

## B.1 Detailed Dataset Specification

The color candidates for nodes and edges were selected from the Basic Color Terms (11 colors) defined by Berlin and Kay (1999), excluding white, black, and gray, resulting in eight colors. The shape candidates for nodes were chosen as basic geometric figures that do not strongly evoke specific meanings, including circles and regular polygons. We excluded equilateral triangles because their shape resembles edge arrowheads, which could cause confusion between nodes and edges.

# C Preliminary: VQA

Table 5 reports the VQA accuracy of all models evaluated in this study. Across all models, `Node Color` and `Edge Style` achieved relatively high accuracy, while `In-degree` and `Out-degree` remained close to chance level. Among the Multiple category, `Edge Color` and `Edge Existence` were answered above chance by most models, whereas `Edge Direction` stayed near chance for all models and `Multi-hop Path` for most models. These trends are consistent with the results for Qwen3-VL reported in § 4.2 and suggest that recognizing edge directions and multi-step relational structure is difficult across models.

# D Probing

## D.1 Shortcut learning of probe

In the probing experiments, we observed shortcuts across all aspects where the probe predicted answers based solely on superficial features of the input patches. These shortcuts resulted in 100% accuracy on patches in certain regions. For example, for `Node Color`, the probe achieved 100% accuracy on patches containing the target node by simply predicting the color of the input patch. Beyond `Node Color`, we also confirmed shortcuts that exploited information such as patch shape and position.

These shortcuts do not reflect correct understanding of the diagram's visual information. To correctly classify `Node Color`, the probe must first identify node A and then predict its color. However, shortcut-based predictions skip the identification of node A. This phenomenon occurs because the probe determines that predicting superficial features of input patches maximizes overall accuracy based on the dataset distribution.

To address this problem, as described in § 3.3, we add diagrams without node A as dummy data to the training dataset. This approach ensures that when the input does not contain the information necessary for prediction, the probe outputs the dummy class (N/A), thereby avoiding falsely high accuracy through shortcuts.

## D.2 Probe Training Hyperparameters

We used linear classifiers for probe training, training separately for each aspect. The hyperparameters differed only in batch size between image-part probing and text-part probing. Table 6 shows the hyperparameters used for each type of probing.

For hyperparameter search, we used Optuna (Akiba et al., 2019) with TPESampler to perform three hyperparameter search trials. The search targets were batch size and learning rate, where batch size was selected categorically from the candidates shown in Table 6, and learning rate was searched on a log scale ranging from 0.0001 to 0.01. For the evaluation metric, we split 10% of the training data as validation data and used accuracy on the validation data. We did not use early stopping during training; instead, we evaluated performance on the validation data at each epoch and saved the model that achieved the highest accuracy.

| Aspect | Question |
|---|---|
| `Node Color` | What color is node A? |
| `Node Shape` | What shape is node A? |
| `In-degree` | What is the in-degree of node A? |
| `Out-degree` | What is the out-degree of node A? |
| `Edge Color` | What color is the edge between node A and node B? |
| `Edge Style` | What style is the edge between node A and node B? |
| `Edge Existence` | Does an edge exist between node A and node B? |
| `Edge Direction` | What is the direction of the edge between node A and node B? |
| `Multi-hop Path` | Is there a path from node A to node B? |
| `Node Count` | How many nodes labeled A are in the graph? |
| `Edge Count` | How many edges are in the graph? |

Table 4: Questions used to evaluate each aspect.

| | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Chance level | 12.5 | 20.0 | 20.0 | 20.0 | 12.5 | 50.0 | 50.0 | 50.0 | 50.0 | 20.0 | 20.0 |
| Qwen3-VL 8B | 95.5 | 78.4 | 53.7 | 40.6 | 80.6 | 85.7 | 81.2 | 53.9 | 55.9 | 78.7 | 83.8 |
| Qwen2.5-VL 7B | 86.6 | 72.6 | 40.4 | 33.8 | 61.0 | 73.3 | 65.5 | 49.7 | 53.3 | 93.8 | 38.7 |
| LLaVA1.5 7B | 40.1 | 31.3 | 22.1 | 24.9 | 33.2 | 47.3 | 54.4 | 50.0 | 50.0 | 20.0 | 26.1 |
| Gemma3-4B-IT | 68.7 | 51.8 | 33.9 | 35.9 | 53.2 | 67.0 | 61.8 | 49.5 | 50.0 | 30.9 | 20.1 |

Table 5: VQA accuracy of all models evaluated in this study. Across all models, `Edge Direction` remains near chance and `Multi-hop Path` is difficult for most models, consistent with the pattern observed for Qwen3-VL in Table 2.

### D.3 Additional results

In this section, we present additional analysis results beyond the Qwen3-VL 8B results reported in the main text. Specifically, we report detailed probing results for Qwen3-VL 8B and results with different node layouts, as well as probing results for additional models (Qwen2.5-VL 7B, LLaVA1.5 7B, Gemma3-4B-IT).

**Qwen3-VL 8B (detailed probing results).** Figure 9 and Figure 10 present image probing results with two additional intermediate layers that were omitted in Figure 4 and Figure 6. As described in § 5.3, Single category information is encoded near node positions, while Global category information is distributed across background regions. Figure 11 presents text probing results for aspects omitted from Figure 7. As described in § 5.3, accuracy sharply increased at token positions that specify the target nodes or edges.

**Qwen3-VL 8B (different node layouts).** Figure 13 and Figure 14 show results with different node layouts (see Figure 12). In the main text, we placed the target node (node A) at the bottom-right, but here we show results with node A placed on the left side. The distribution of patches with high accuracy for node information changed according to the position of node A. This result demonstrates that the probe appropriately captures positional information of the target node.

**Qwen3-VL 8B (larger nodes and edges).** Figure 16 and Figure 17 show probing results when the sizes of nodes and edges are increased (see Figure 15 for example diagrams). The results exhibit the same trends as the default-size setting reported in § 5.3. This indicates that node and edge sizes do not fundamentally affect the encoding patterns, and that the findings generalize beyond a specific choice of visual element sizes.

**Qwen3-VL 8B (7-node graphs).** We report probing results on diagrams containing 7 nodes instead of 5 (see Figure 18 for example diagrams). Figures 19 and 20 show the layer-wise results, Figures 21 and 22 show the patch-wise results, and Figure 23 shows the text-token results. The overall accuracy is slightly lower and fewer patches exceed the threshold than in the 5-node setting, which we attribute to the increased task difficulty from the larger number of nodes. Nonetheless, the results exhibit the same trends as the default 5-node setting reported in § 5.3, indicating that the observed encoding patterns are robust to moderate increases in graph size.

| Parameter | Image Part | Text Part |
|---|---|---|
| Epochs | 1000 | 1000 |
| Batch size* | {1000, 2000, 4000, 8000, 16000} | {10, 20, 40, 80, 160} |
| Learning rate* | [0.0001, 0.01] (log scale) | [0.0001, 0.01] (log scale) |
| Optimizer | AdamW | AdamW |
| betas | [0.9, 0.999] | [0.9, 0.999] |
| eps | 1e-8 | 1e-8 |
| weight decay | 0.01 | 0.01 |

Table 6: Hyperparameters used for probe training. Batch size and learning rate were subject to hyperparameter search using Optuna. Optimizer parameters use PyTorch's AdamW default values. *Subject to hyperparameter search

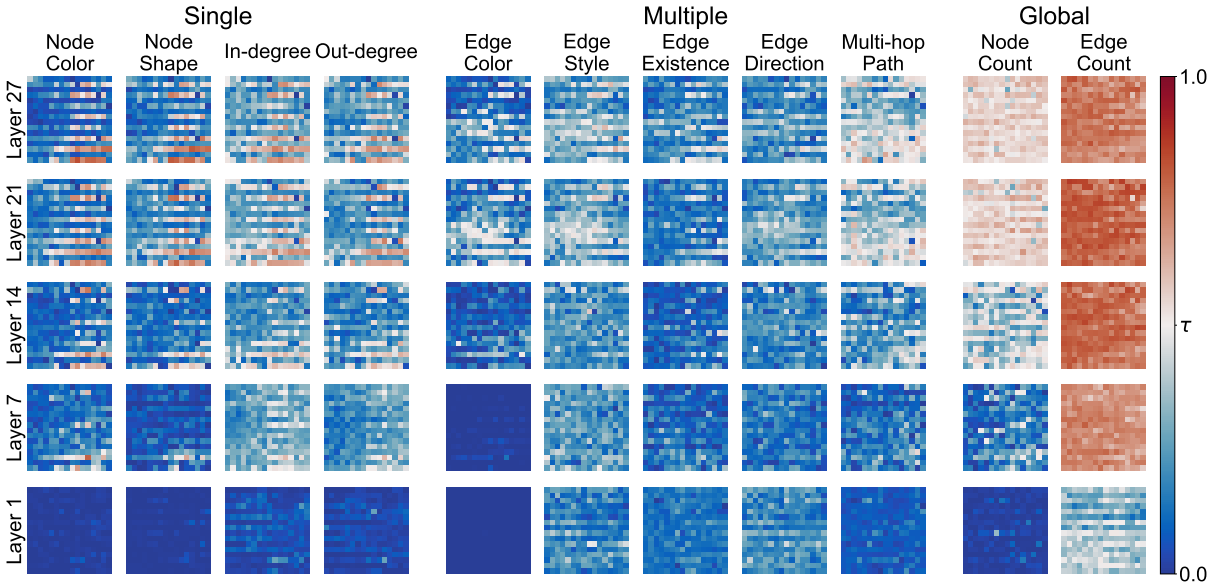


Figure 9: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B vision encoder. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are consistent with Figure 4.

**Qwen2.5-VL 7B.** Figure 24 and 26 show the layer-wise probing results, and Figure 25 and 27 show the patch-wise results for Qwen2.5-VL 7B. The overall trends are similar to Qwen3-VL, with node information showing high accuracy while edge relationships remained near chance level. In the patch-wise analysis, patches corresponding to node positions showed higher accuracy for Single category aspects, while Global category aspects showed high accuracy across a wide range of positions, including background regions.

Figure 28 shows the probing results for text tokens. Similar to Qwen3-VL, we observed a trend where accuracy sharply increased at token positions that specify nodes or edges.

**LLaVA1.5 7B.** Figure 29 and 31 show the layer-wise probing results, and Figure 30 and 32 show the patch-wise results for LLaVA1.5 7B. The overall trends are similar to Qwen3-VL and Qwen2.5-VL, but a distinctive difference was observed in the patch-wise analysis. In LLaVA1.5, background patches showed higher accuracy than patches corresponding to node positions. This result indicates that LLaVA1.5 aggregates visual information into background patches.

Figure 33 shows the probing results for text tokens. Similar to other models, we observed information aggregation at token positions that specify target nodes or edges.

**Gemma3-4B-IT.** Figure 34 and 36 show the layer-wise probing results, and Figure 35 and 37 show the patch-wise results for Gemma3-4B-IT. The overall trends are similar to Qwen3-VL and Qwen2.5-VL, but accuracy was lower for some node information and edge relationships compared to other models.

Figure 38 shows the probing results for text tokens. As with the other models, we observe an

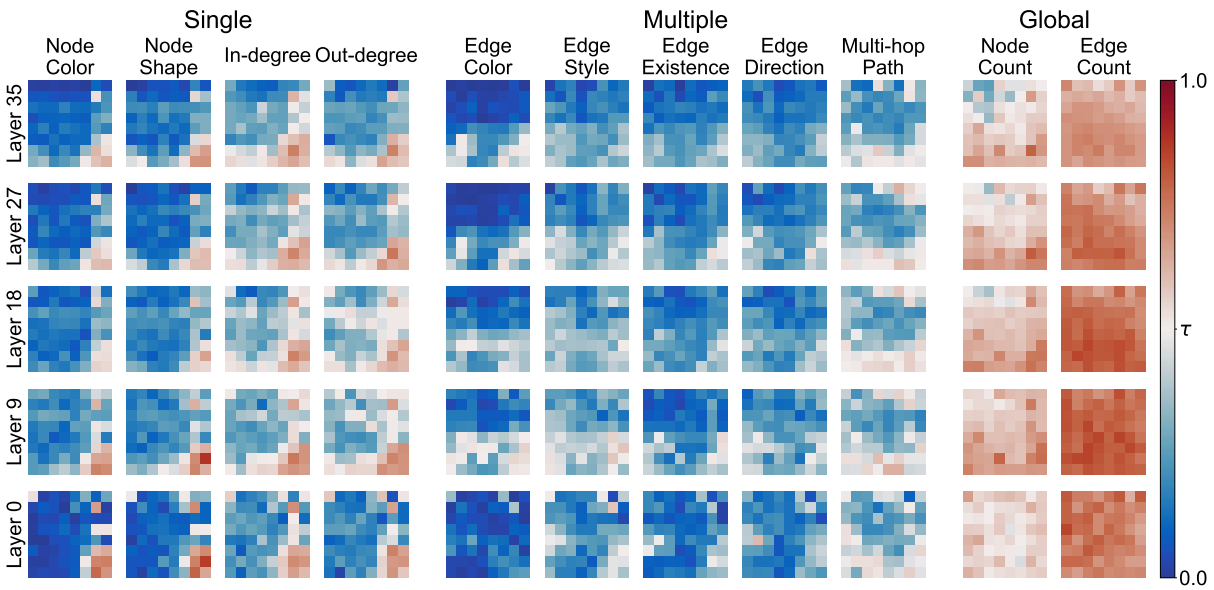


Figure 10: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B language model. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are consistent with Figure 6.

aggregation of information at the token positions that specify the target nodes or edges; however, the effect is weaker than in the other models.

## E Additional results for Causal intervention

In § 6, we reported the causal intervention results as the accuracy difference between the *accuracy-based* and *controlled* settings for Qwen3-VL 8B. In this section, we first present the individual accuracy for the *accuracy-based*, and *controlled* settings for Qwen3-VL 8B, and then report results for additional models (Qwen2.5-VL 7B, LLaVA1.5 7B, and Gemma3-4B-IT). In addition, we report *clean*, which denotes the accuracy obtained without any intervention.

**Qwen3-VL 8B.** Figure 39 shows the *clean*, *accuracy-based*, and *controlled* accuracy at each intervention layer for Qwen3-VL 8B. The results are discussed in detail in § 6.

**Qwen2.5-VL 7B.** Table 7 and Figure 40 report the results for Qwen2.5-VL 7B. Similar to Qwen3-VL, the intervention caused a large drop in accuracy for aspects such as `Node Color`, `Node Shape`, `Edge Color`, and `Node Count`, confirming that these aspects are linearly encoded in both the vision encoder and the language model and causally contribute to inference.

**LLaVA1.5 7B.** Table 8 and Figure 41 report the results for LLaVA1.5 7B. Although LLaVA1.5 has lower baseline VQA performance than the other models, we still observe a performance drop after the intervention for aspects such as `Node Color`, `Node Shape`, and `Edge Color`.

**Gemma3-4B-IT.** Table 9 and Figure 42 report the results for Gemma3-4B-IT. We observe a drop in performance after the intervention for multiple aspects, including `Node Color`, `Node Shape`, and `Edge Color`, indicating that linearly decodable information in both the vision encoder and the language model contributes to VQA inference.

Across all four models, replacing linearly decodable patches with the dataset-wide mean reduced accuracy for `Node Color`, `Node Shape`, and `Edge Color` in both the vision encoder and the language model. Qwen3-VL and Qwen2.5-VL additionally showed drops for `Node Count`. These results confirm that the linear representations identified by probing are not merely present but causally contribute to VQA inference. In contrast, aspects that are poorly captured by linear probes, such as `Edge Direction`, `Edge Existence`, and `Multi-hop Path`, showed little to no change after intervention, suggesting that the model relies on nonlinearly encoded information or on other mechanisms beyond the probed linear features for these aspects.

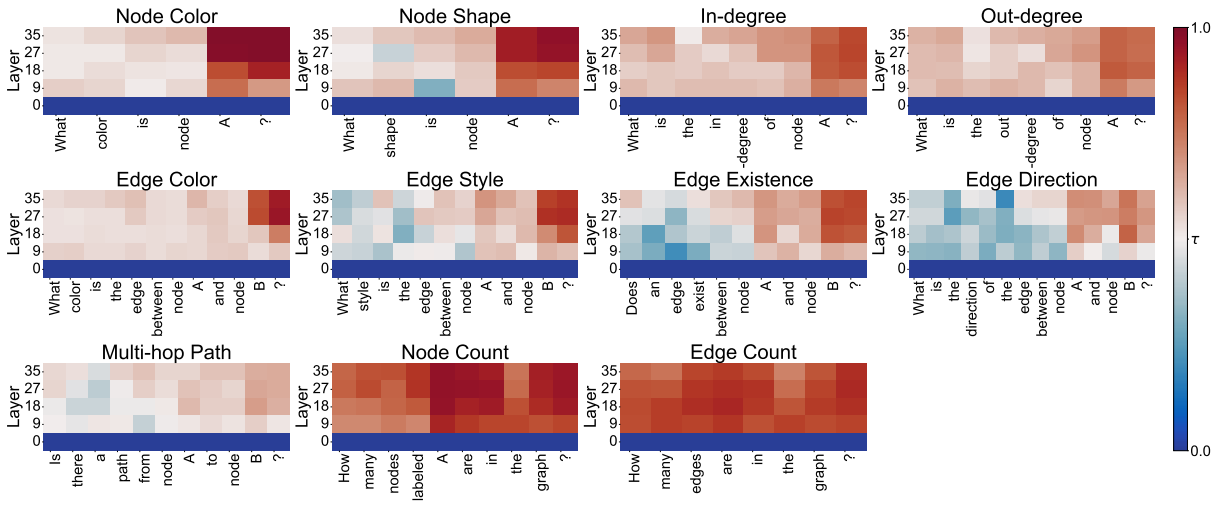


Figure 11: Probing results for text tokens in Qwen3-VL 8B. The x-axis shows token position in the question, and the y-axis shows layer position. Trends are consistent with Figure 7.

| | Layer | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vision encoder | 1 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | **−0.9** |
| | 16 | **−48.8** | **−34.9** | **−7.1** | **−5.3** | ±0.0 | **−0.1** | ±0.0 | ±0.0 | +0.3 | **−29.6** | ±0.0 |
| | 32 | **−48.4** | **−38.0** | **−9.6** | **−7.2** | **−14.3** | **−1.3** | **−1.2** | **−0.1** | +0.4 | **−3.2** | ±0.0 |
| Language model | 0 | **−60.0** | **−48.2** | **−13.2** | **−6.9** | **−25.9** | **−6.7** | **−2.3** | +0.3 | +1.1 | **−3.1** | ±0.0 |
| | 14 | **−44.8** | **−36.8** | **−9.5** | **−3.9** | **−27.4** | **−1.9** | **−1.3** | +0.2 | +1.4 | **−0.7** | ±0.0 |
| | 27 | ±0.0 | **−0.1** | ±0.0 | +0.1 | **−0.2** | **−0.1** | **−0.2** | +0.1 | **−0.3** | **−0.1** | ±0.0 |

Table 7: Accuracy difference between *accuracy-based* and *controlled* settings (*accuracy-based* − *controlled*) for Qwen2.5-VL 7B. Negative values (**bold**) indicate that a decrease in accuracy was observed due to the intervention.

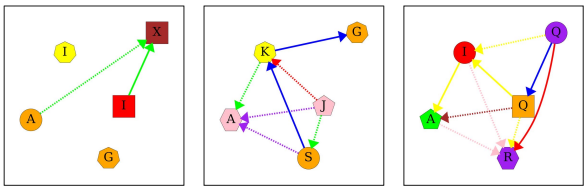


Figure 12: Example diagrams with layouts distinct from those in Figure 2.

# F Experiments on Data Closer to Real-World Diagrams

To examine whether the trends generalize to a more realistic dataset closer to real-world diagrams, we conduct VQA and probing experiments on flowchart images derived from FlowVQA (Singh et al., 2024), using Qwen3-VL 8B as the target model. The dataset retains a synthetic nature due to the controlled editing and augmentation applied to node and edge attributes, but it differs from the synthetic graphs in the main text by containing visual features specific to natural flowcharts, such as text labels and canonical layouts.

## F.1 Evaluation Aspects and Class Configuration

We analyze 10 of the 11 aspects defined in § 3.2 of the main text, excluding `Node Shape`. `Node Shape` is excluded because computing patch-wise accuracy in the same setting as the synthetic data requires the node layout to be unified across the entire evaluation set, but introducing shape variation would alter the layout.

The class configuration for each aspect is shown in Table 10. While the configuration largely matches that of the synthetic data, it differs in the following aspects:

- `In-degree` / `Out-degree`: 5 classes (0–4) in synthetic → 4 classes (1–4) in this setting. Node degrees in flowcharts closer to real-world tend to be smaller than in the synthetic data, so we adjusted the range accordingly.
- `Node Count` / `Edge Count`: 5 classes (1–5) in synthetic → 4 classes (7–10) in this setting. We adjusted the range to preserve the distribution of the original FlowVQA data while

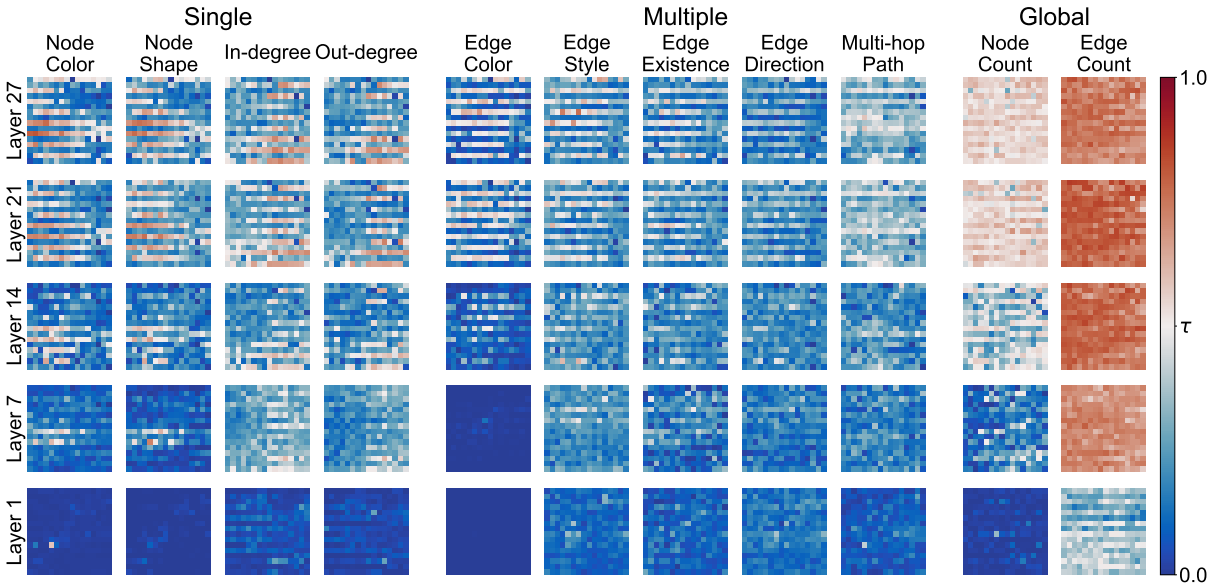


Figure 13: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B vision encoder (node A placed on the left side). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. High-accuracy patches for Single aspects localize on the left side, following node A's position.

| | Layer | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vision encoder | 1 | **−2.4** | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | +0.9 |
| | 12 | **−15.1** | **−5.5** | **−1.4** | **−0.7** | **−0.2** | **−0.1** | **−0.2** | ±0.0 | ±0.0 | ±0.0 | ±0.0 |
| | 23 | **−8.8** | **−1.6** | ±0.0 | **−0.6** | **−4.2** | +0.4 | +0.4 | ±0.0 | ±0.0 | ±0.0 | ±0.0 |
| Language model | 0 | **−12.8** | **−5.7** | ±0.0 | +0.1 | **−3.9** | **−0.5** | +0.3 | ±0.0 | ±0.0 | ±0.0 | ±0.0 |
| | 16 | **−7.9** | **−2.3** | +0.0 | +0.4 | **−3.6** | **−0.1** | **−0.8** | ±0.0 | ±0.0 | ±0.0 | ±0.0 |
| | 31 | **−0.1** | +0.1 | **−0.2** | +0.4 | +0.1 | **−0.3** | ±0.0 | ±0.0 | ±0.0 | ±0.0 | +0.0 |

Table 8: Accuracy difference between *accuracy-based* and *controlled* settings (*accuracy-based* − *controlled*) for LLaVA1.5 7B. Negative values (**bold**) indicate that a decrease in accuracy was observed due to the intervention.

matching the synthetic data in complexity.

### F.2 Dataset Construction

We construct the dataset in three steps: (1) sampling base flowcharts from FlowVQA, (2) editing and data augmentation, and (3) Mermaid rendering.

**Step 1: Sampling base flowcharts.** To match the complexity of the diagrams for comparison with the synthetic-data results in § 3, we restrict the source to FlowVQA samples with at most 10 nodes. From this sample pool, for each aspect we independently sample 10 base flowcharts for training and 1 for evaluation.

For aspects that analyze a specific node or node pair (Single and Multiple aspects), we sample flowcharts that all share a common target node (or node pair). The Single aspects target the node with the label "Return True", while the Multiple aspects target the node pair ("Return True", "End"). We choose these as targets because they are the labels that most frequently appear across FlowVQA flowcharts, allowing us to fix a consistent target across aspects while securing enough base samples.

In contrast, for the Global aspects (`Node Count`, `Edge Count`), which analyze the diagram as a whole, the classification target is the total number of nodes (or edges) in the flowchart. We therefore sample 10 training and 1 evaluation flowcharts for each class, where each class corresponds to a specific node count (or edge count). For example, for `Node Count`, we sample flowcharts with 7, 8, 9, and 10 nodes for each respective class.

The reason for restricting evaluation to a single base sample (and one per class for Global) is that, similar to the analysis on the synthetic data, computing patch-wise accuracy requires the node layout to be unified across the evaluation set, so flowcharts with differing layouts cannot be mixed.

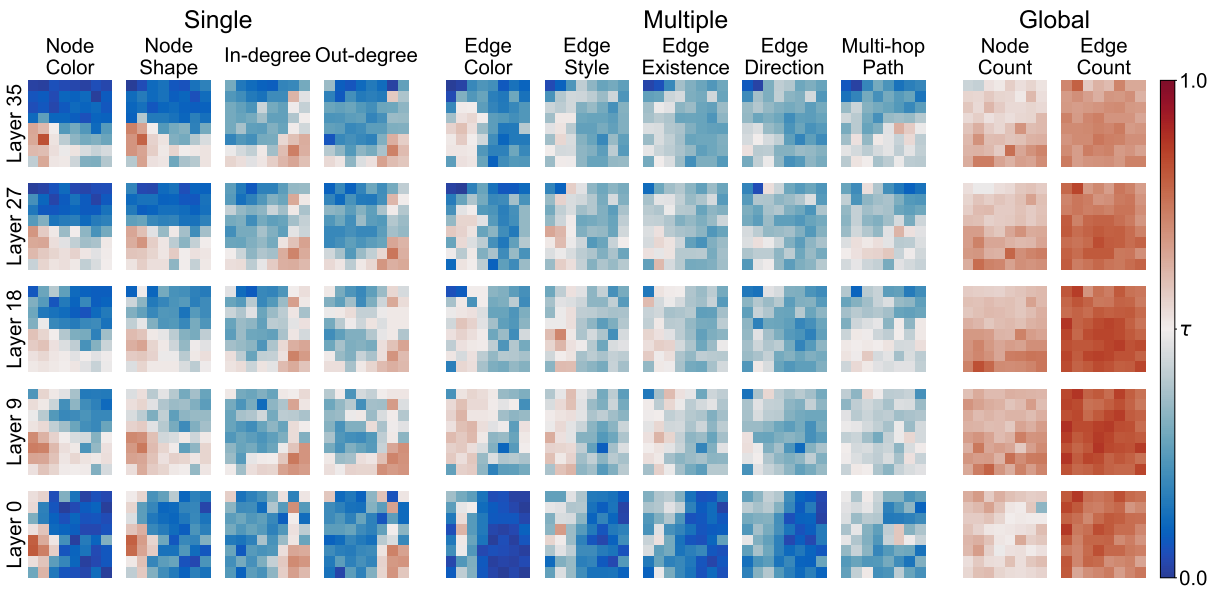


Figure 14: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B language model (node A placed on the left side). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. High-accuracy patches for Single aspects localize on the left side, following node A's position.

| | Layer | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vision encoder | 1 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | +0.1 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 |
| | 14 | **−39.6** | **−18.6** | **−4.3** | **−6.7** | **−3.1** | **−0.9** | +0.2 | +0.3 | ±0.0 | +0.3 | **−3.3** |
| | 27 | **−30.2** | **−15.6** | **−1.8** | **−4.3** | +0.5 | ±0.8 | **−0.1** | +0.1 | ±0.0 | **−5.3** | **−0.1** |
| Language model | 0 | **−34.9** | **−25.1** | **−1.0** | **−3.0** | **−28.8** | **−5.7** | **−2.3** | +0.3 | **−0.8** | +0.1 | **−0.9** |
| | 17 | **−17.9** | **−3.9** | **−0.6** | **−0.3** | **−3.0** | ±0.0 | ±0.0 | **−0.1** | ±0.0 | ±0.0 | ±0.0 |
| | 33 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | ±0.0 | +0.1 | ±0.0 | ±0.0 | ±0.0 | ±0.0 |

Table 9: Accuracy difference between *accuracy-based* and *controlled* settings (*accuracy-based* − *controlled*) for Gemma3-4B-IT. Negative values (**bold**) indicate that a decrease in accuracy was observed due to the intervention.

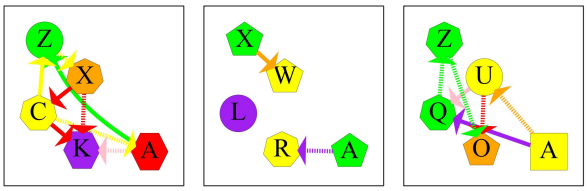


Figure 15: Example diagrams with larger nodes and edges compared to those in Figure 2.

| Category | Aspect $a$ | Label set $\mathcal{Y}$ |
|---|---|---|
| Single | `Node Color` | {red, green, yellow, blue, brown, orange, pink, purple} |
| | `In-deg.` | {1, 2, 3, 4} |
| | `Out-deg.` | {1, 2, 3, 4} |
| Multiple | `Edge Color` | {red, green, yellow, blue, brown, orange, pink, purple} |
| | `Edge Style` | {solid, dotted} |
| | `Edge Direct.` | {forward, backward} |
| | `Edge Exist.` | {exist, not exist} |
| | `Multi-hop Path` | {exist, not exist} |
| Global | `Node Count` | {7, 8, 9, 10} |
| | `Edge Count` | {7, 8, 9, 10} |

Table 10: Class configuration of the 10 aspects analyzed in this setting.

Using flowcharts with random node layouts for training and a fixed layout for evaluation also aligns with the synthetic-data setting.

**Step 2: Editing and data augmentation.** For each base flowchart, we generate samples for each class by editing the attributes of the target node (or pair); see Table 11 for the list of edit operations. In addition, we apply the following augmentation to expand the data to 100 samples per class:

- Common: randomize the colors of non-target nodes, edge colors, and edge styles.
- Training-only additional: shuffle the labels of non-target nodes and randomize their shapes.

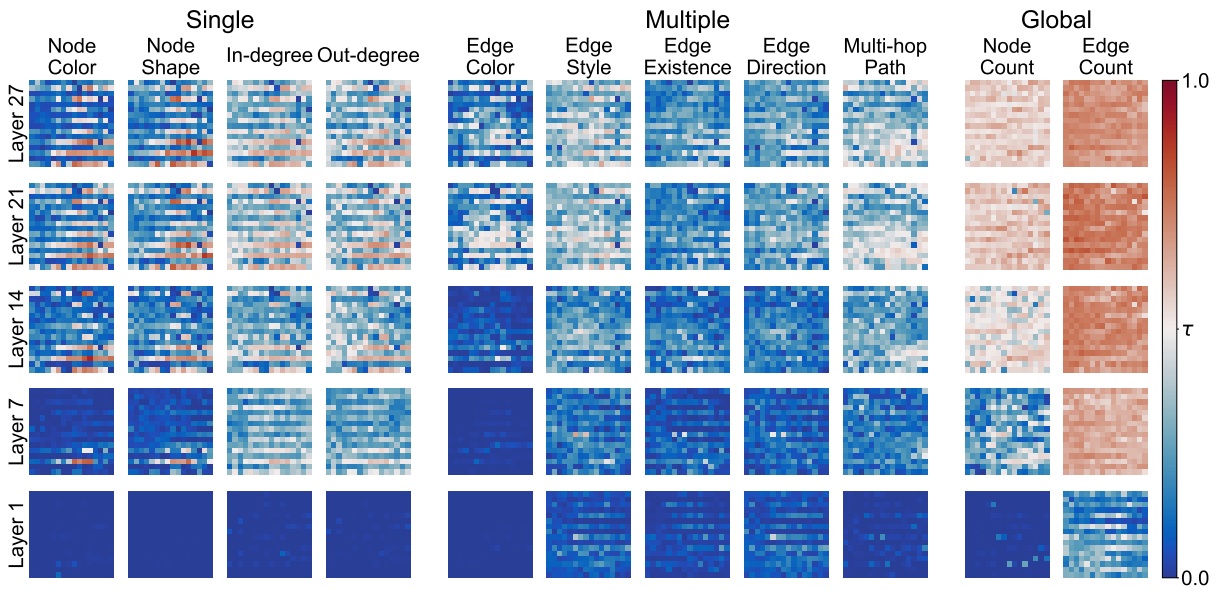


Figure 16: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B vision encoder (larger nodes and edges). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are consistent with Figure 4.

We apply label shuffling and shape randomization only to the training data to prevent the probe from exploiting non-target node text labels or shapes as shortcuts.

**Step 3: Mermaid rendering.** We render the edited Mermaid code into PNG using `mermaid-cli` and feed the resulting images directly into Qwen3-VL without resizing or other preprocessing.

### F.3 Preliminary Experiment: VQA

**Setup.** We design our own multiple-choice question prompts for this analysis, rather than using the questions provided in FlowVQA. The prompts follow a format parallel to § 4 of the main text, explicitly specifying the target node (or pair) for each aspect. The list of prompts for each aspect is given in Table 12. The evaluation metric is the same as the analysis on the synthetic data in the main text: we use accuracy.

**Results.** The VQA accuracy for each aspect is shown in Table 13. The results were largely consistent with those on the synthetic data: `Edge Direction` and `Multi-hop Path` remained near chance level, while the other aspects exceeded chance. However, `Edge Existence` and `Out-degree` also remained near chance level in this setting, in contrast to the synthetic-data results where their accuracies exceeded chance.

### F.4 Probing

**Setup.** We use the same single-layer linear classifier as in § 5 of the main text, applied to three probe variants: hidden states in the vision encoder, the image-input part of the language model, and the text-input part of the language model. The probe architecture and training procedure are identical to those used on the synthetic data. The only modifications, both for the vision encoder probe, are listed in Table 14: we set the maximum number of epochs to 200, which we found sufficient for convergence, and enlarge the batch-size search range to accommodate the increased number of patches caused by the higher input resolution. The evaluation method follows the synthetic-data analysis: we perform a layer-wise analysis tracking accuracy across layers and a position-wise analysis examining the distribution of accuracy across patch positions (for the vision encoder and the image-input part) or token positions (for the text-input part).

**All aspects, including Multiple, become linearly decodable in the vision encoder.** Figure 43 shows $\mathrm{MaxAcc}_l$ at each layer of the vision encoder, and Figure 44 shows the patch-wise accuracy distribution.

For all aspects, accuracy exceeded the threshold within the vision encoder, indicating that the information for every aspect was linearly decodable from a single hidden state already within the vision

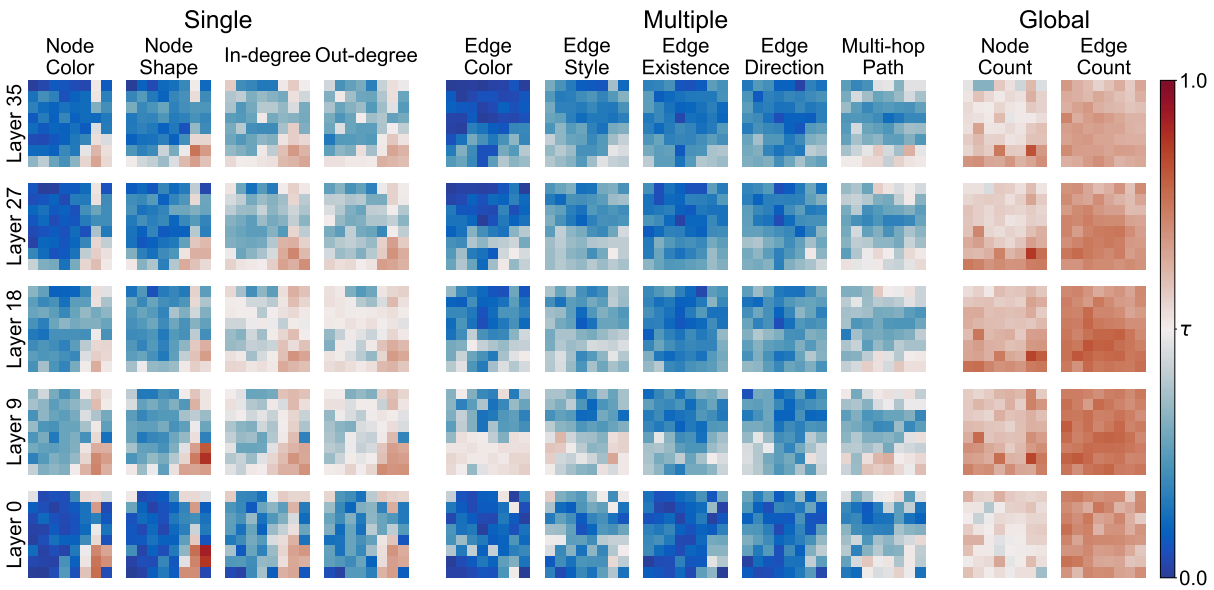


Figure 17: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B language model (larger nodes and edges). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are consistent with Figure 6.

| Aspect | Edit operation |
|---|---|
| `Node Color` | Randomly select the color of the target node from 8 colors. |
| `In-deg.` | Add or delete in-edges to set the in-degree of the target node to one of 1–4. |
| `Out-deg.` | Add or delete out-edges to set the out-degree of the target node to one of 1–4. |
| `Edge Color` | Randomly select the color of the edge between the target pair from 8 colors. |
| `Edge Style` | Change the edge between the target pair to solid or dotted. |
| `Edge Direct.` | Reverse the direction of the edge between the target pair. |
| `Edge Exist.` | Add or delete the edge between the target pair. |
| `Multi-hop Path` | Edit intermediate edges to change the reachability between the target pair. |
| `Node Count` | No edits (the class is determined at sampling time). |
| `Edge Count` | No edits (the class is determined at sampling time). |

Table 11: Edit operations for each aspect.

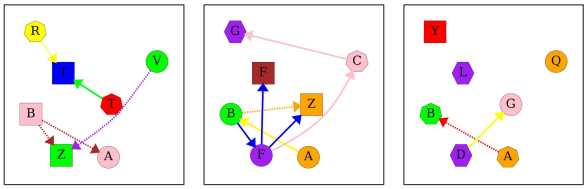

Figure 18: Example diagrams with 7 nodes compared to those in Figure 2.

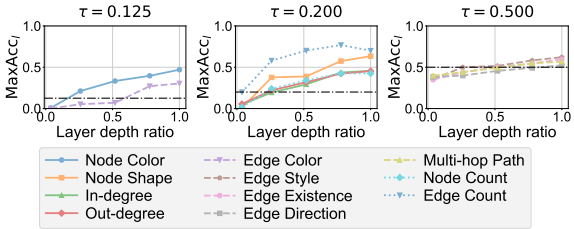


Figure 19: Layer-wise maximum accuracy in the vision encoder of Qwen3-VL 8B (7-node graphs). Trends are similar to the 5-node setting (Figure 3), although the overall accuracy is slightly lower.

encoder.

In the patch-wise distribution, Single aspects showed high-accuracy patches concentrated around the target node, while Global aspects showed high accuracy across a broad range of patches, including background patches. These spatial patterns for the two categories are consistent with the synthetic-data results (§ 5.3). In contrast, the Multiple aspects showed a different trend from the synthetic data: while most Multiple aspects did not exceed the threshold within the vision encoder in the synthetic setting, in this setting all Multiple aspects exceeded the threshold.

**Linearly decodable signals are preserved in the image-input part of the language model.** Figure 45 shows $\mathrm{MaxAcc}_l$ across language model lay-

| Aspect | Prompt |
|---|---|
| `Node Color` | What color is the ‘Return True’ node? |
| `In-deg.` | What is the in-degree of the ‘Return True’ node? |
| `Out-deg.` | What is the out-degree of the ‘Return True’ node? |
| `Edge Color` | What color is the edge between the ‘Return True’ node and the ‘End’ node? |
| `Edge Style` | What style is the edge between the ‘Return True’ node and the ‘End’ node? |
| `Edge Direct.` | What is the direction of the edge between the ‘Return True’ node and the ‘End’ node? |
| `Edge Exist.` | Does an edge exist between the ‘Return True’ node and the ‘End’ node? |
| `Multi-hop Path` | Is there a path from the ‘Return True’ node to the ‘End’ node? |
| `Node Count` | How many nodes are in the graph? |
| `Edge Count` | How many edges are in the graph? |

Table 12: VQA prompts for each aspect. “Answer with only one word from the choices.” and “ (Choices: <comma-separated-choices>)” are appended to every prompt.

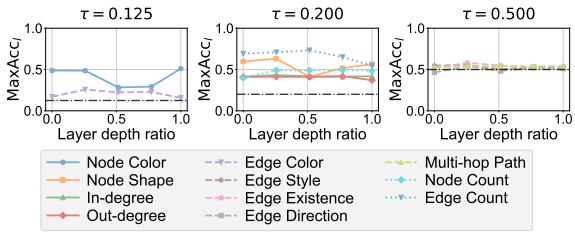


Figure 20: $\mathrm{MaxAcc}_l$ per layer in the language model of Qwen3-VL 8B (7-node graphs). Trends are similar to the 5-node setting (Figure 5), although the overall accuracy is slightly lower.

| Category | Aspect | Acc. | Chance |
|---|---|---|---|
| Single | `Node Color` | 0.953 | 0.125 |
| | `In-deg.` | 0.468 | 0.250 |
| | `Out-deg.` | 0.250 | 0.250 |
| Multiple | `Edge Color` | 0.786 | 0.125 |
| | `Edge Style` | 0.985 | 0.500 |
| | `Edge Direct.` | 0.500 | 0.500 |
| | `Edge Exist.` | 0.530 | 0.500 |
| | `Multi-hop Path` | 0.500 | 0.500 |
| Global | `Node Count` | 0.458 | 0.250 |
| | `Edge Count` | 0.455 | 0.250 |

Table 13: VQA accuracy of Qwen3-VL 8B in this setting. Chance indicates the random-prediction accuracy based on the number of classes.

ers, and Figure 46 shows the patch-wise accuracy distribution.

All aspects maintained accuracy above the threshold throughout the image-input part, and the patch-wise distribution showed little change from the vision encoder result. These results indicate that the linearly decodable signals observed in the vision encoder were preserved through the image-input part, consistent with the synthetic-data results (§ 5.3).

**`Edge Direction` and `Edge Existence` signals are confined to intermediate layers in the text-input part.** Figure 47 shows $\mathrm{acc}_{l,t}$ across layers and token positions in the text-input part of the language model.

Similar to the synthetic-data results (§ 5.3), accuracy was particularly high at token positions that specify the target node or pair. However, for `Edge Direction` and `Edge Existence`, high accuracy appeared only in intermediate layers and did not propagate to the final layer. This suggests that, although signals corresponding to these aspects existed within the model, they were not propagated to the final layer; this differs from the synthetic-data results, where these aspects maintained high accuracy through the final layer.

| Parameter | Vision encoder probe |
|---|---|
| Epochs | 200 |
| Batch size* | {4000, 8000, 16000, 32000, 64000} |

Table 14: Hyperparameters of probe training in this setting that differ from the synthetic-data setting (Table 6). All other hyperparameters are identical to those of the synthetic-data setting. *Subject to hyperparameter search.

**Possible causes of the gap from the synthetic results.** The results in this setting differ from those on the synthetic data in two respects: (1) Multiple aspects became linearly decodable already within the vision encoder, and (2) in the text-input part of the language model, the signals for `Edge Direction` and `Edge Existence` remained in intermediate layers and did not propagate to the final layer.

Several factors may explain this gap, including the complexity of text labels in FlowVQA flowcharts, differences in input image resolution, and the diversity of node and edge layouts. Identifying which factor predominantly produces the gap

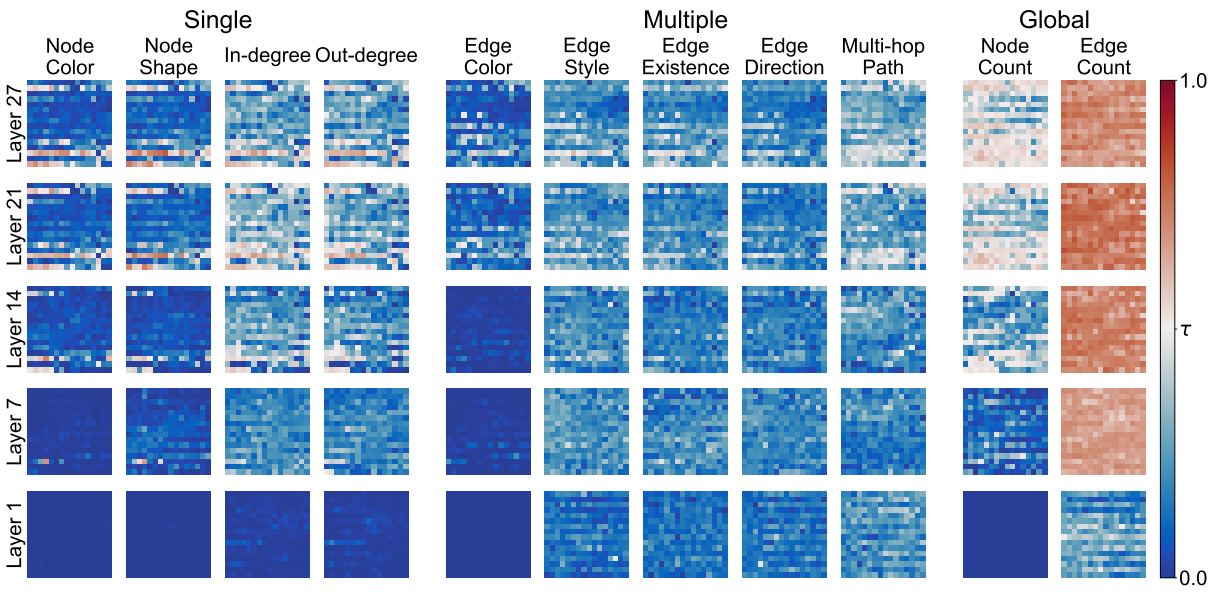


Figure 21: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B vision encoder (7-node graphs). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are consistent with the 5-node setting (Figure 4), although fewer patches exceed the threshold.

is an interesting question we leave for future work.

## G Analysis of LLaVA Training Data

Prior studies report that LVLMs struggle with understanding relational information indicated by arrows and lines (Hou et al., 2025; Zhu et al., 2025; Pan et al., 2024). Consistent with these findings, our VQA experiments show that all models achieved near-chance accuracy on `Edge Direction` (§ 4.2, Appendix C). One possible explanation is that images containing arrows and text describing arrow directions are underrepresented in the training dataset. To investigate this hypothesis, we analyze the pre-training dataset and instruction tuning dataset of LLaVA1.5, whose training dataset is publicly available.

### G.1 Setup

We quantify how many samples mention arrow directions in the pre-training dataset (558,128 samples) and the instruction tuning dataset (665,298 samples) of LLaVA1.5. The analysis consists of two steps. In the first step, we filter samples whose text contains the substring "arrow." Since the texts are lowercased during preprocessing, we perform case-insensitive matching. In the second step, we classify whether each filtered sample mentions the direction of an arrow (e.g., pointing left, pointing upward) using gpt-oss-120b (OpenAI et al., 2025) as the judge. Note that we analyze only the text portion of each sample, without referring to the associated images.

### G.2 Results

Out of 558,128 samples in the pre-training dataset, 1,108 (0.20%) contain the substring "arrow." Out of 665,298 samples in the instruction tuning dataset, 4,408 (0.66%) contain the substring "arrow." Of these, 161 (0.03% of all samples) and 646 (0.10%) were judged as mentioning arrow direction in the pre-training and instruction tuning dataset, respectively. These results suggest that the scarcity of arrow direction descriptions in the training dataset may be one factor contributing to the low `Edge Direction` accuracy. However, since our analysis targets only the text portion and does not account for visual information in the images, further investigation is needed to draw stronger conclusions. Additionally, this analysis covers only LLaVA1.5; whether similar trends hold for the training datasets of other models remains an open question.

## H Fine-tuned Model Analysis

To investigate whether training on diagram data changes how LVLMs represent visual information, we fine-tune Qwen3-VL 8B on our synthetic diagram dataset and analyze the resulting changes in VQA accuracy and internal representations.

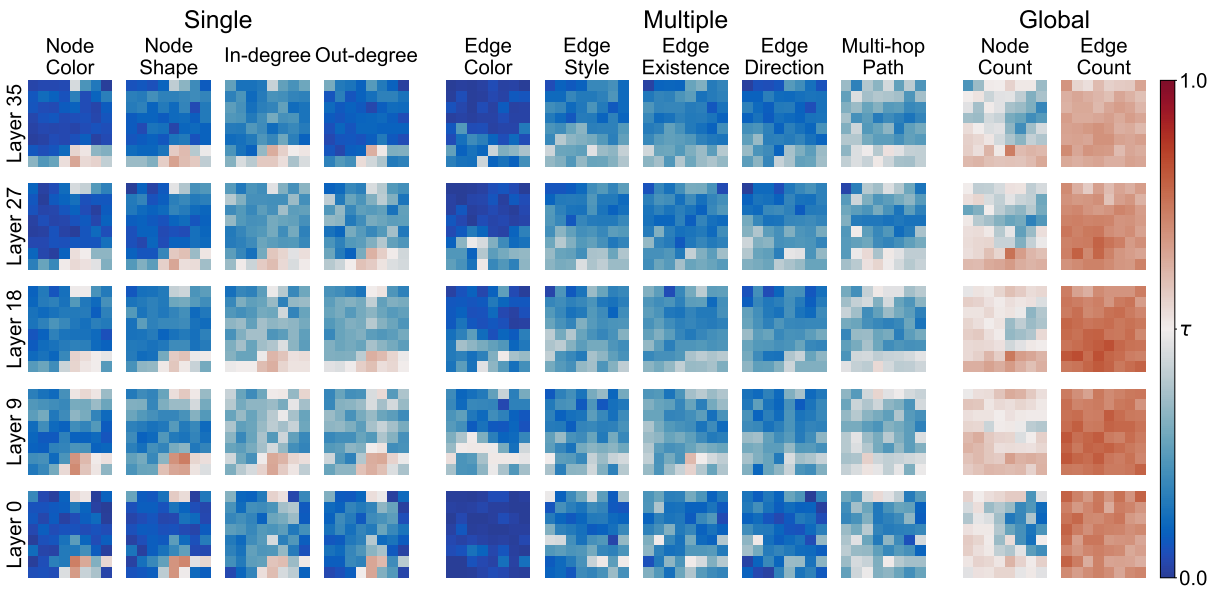


Figure 22: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B language model (7-node graphs). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are consistent with the 5-node setting (Figure 6), although fewer patches exceed the threshold.

| Parameter | FT w/L | FT w/VL |
|---|---|---|
| Epochs | 10 | 10 |
| Batch size | 8 | 8 |
| Learning rate | 1e-4 | 1e-4 |
| Optimizer | AdamW | AdamW |
| betas | [0.9, 0.999] | [0.9, 0.999] |
| eps | 1e-8 | 1e-8 |
| weight decay | 0.01 | 0.01 |
| LoRA rank | 4 | 4 |
| LoRA alpha | 16 | 16 |
| Target modules | q_proj, v_proj | qkv, q_proj, v_proj |

Table 15: Hyperparameters for LoRA fine-tuning.

### H.1 Training Setup

We fine-tune Qwen3-VL 8B using LoRA (Hu et al., 2022) in two configurations: (1) fine-tuning only the language model, and (2) fine-tuning both the vision encoder and the language model. The input format follows the VQA setting described in § 2: each training sample consists of a diagram image and a question, and the model is trained to output the corresponding label as text.

The training dataset is constructed by combining $\mathcal{D}_{\text{rand}}$ across all 11 aspects into a single dataset, resulting in 4,900 samples in total.

The LoRA and training hyperparameters are summarized in Table 15.

### H.2 VQA Accuracy after Fine-tuning

We evaluate VQA accuracy of the fine-tuned models using the same evaluation protocol as in § 4. Table 16 reports the results. Fine-tuning improved accuracy across all aspects regardless of the configuration.

Comparing the two configurations, aspects in the Single and Global categories achieved higher accuracy when only the language model was fine-tuned (FT w/L), with the exception of `Out-degree`. In contrast, aspects in the Multiple category consistently achieved higher accuracy when both the vision encoder and language model were fine-tuned (FT w/VL).

This difference may relate to how each category of information is represented before fine-tuning. As shown in § 5.3, Single and Global information is already linearly encoded in the vision encoder before fine-tuning. Fine-tuning the vision encoder may disrupt these pre-existing representations, leading to lower accuracy for these aspects under the FT w/VL setting. Conversely, Multiple category information is less linearly decodable from individual hidden states in the vision encoder before fine-tuning. Fine-tuning the vision encoder may enable the formation of better representations for these aspects, resulting in higher accuracy under the FT w/VL setting.

### H.3 Changes in Internal Representations

We apply the same probing analysis as in § 5 to the fine-tuned models and compare the results with those of the pre-trained model reported in § 5.3.

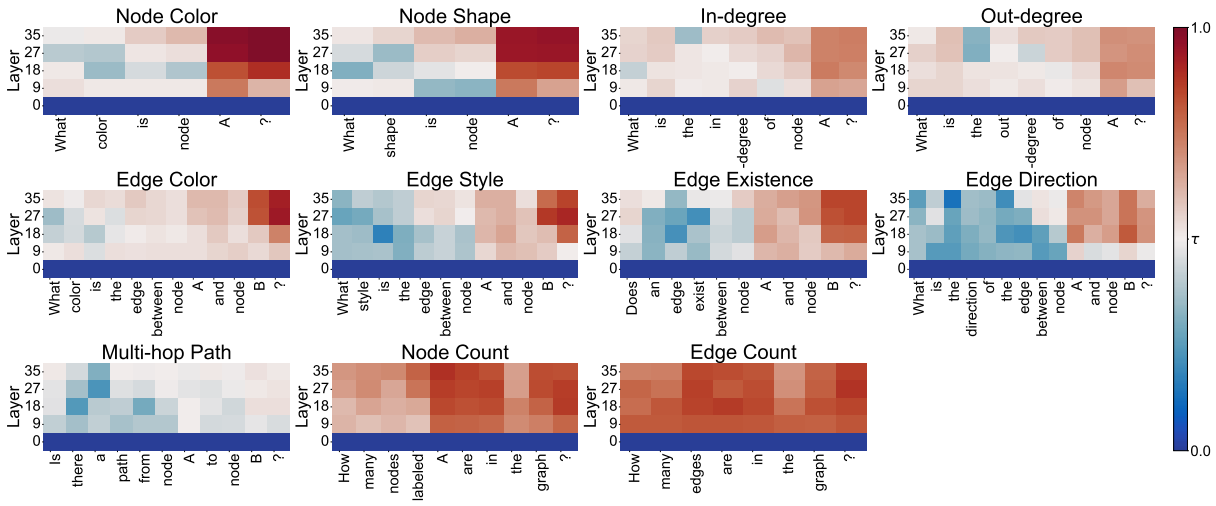


Figure 23: Probing results for text tokens in the language model of Qwen3-VL 8B (7-node graphs). The x-axis shows token position in the question, and the y-axis shows layer position. Trends are consistent with the 5-node setting (Figure 7).

| | Node Color | Node Shape | In-deg. | Out-deg. | Edge Color | Edge Style | Edge Exist. | Edge Direct. | Multi-hop Path | Node Count | Edge Count |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Baseline | 95.5 | 78.5 | 53.6 | 40.8 | 80.7 | 85.7 | 81.2 | 53.9 | 55.9 | 78.6 | 83.9 |
| FT w/L | **99.0** | **86.8** | **71.1** | 66.0 | 90.6 | 91.1 | 86.1 | 70.8 | 59.1 | **79.9** | **88.3** |
| FT w/VL | 93.5 | 85.0 | 63.8 | **68.3** | **91.6** | **91.5** | **87.8** | **72.7** | **68.0** | 77.3 | 87.8 |
| Chance level | 12.5 | 20.0 | 20.0 | 20.0 | 12.5 | 50.0 | 50.0 | 50.0 | 50.0 | 20.0 | 20.0 |

Table 16: VQA accuracy of Qwen3-VL 8B before and after fine-tuning. Baseline denotes the pre-trained model without fine-tuning. FT w/L denotes fine-tuning only the language model, and FT w/VL denotes fine-tuning both the vision encoder and the language model. Fine-tuning improved accuracy across all aspects; Single and Global aspects benefit more from FT w/L, while Multiple aspects benefit more from FT w/VL.

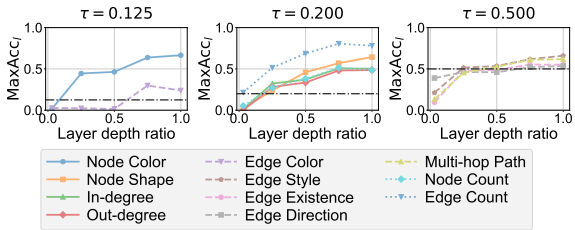


Figure 24: Layer-wise maximum accuracy in the vision encoder of Qwen2.5-VL 7B. The x-axis shows relative layer position (0 is the input layer, 1 is the final layer), and the y-axis shows accuracy. The black dotted line indicates the threshold. Trends are similar to Qwen3-VL (Figure 3): Single and Global aspects rise with depth, while Multiple aspects stay near the threshold.

Since FT w/L does not update the vision encoder, its vision encoder representations are identical to those of the pre-trained model; we therefore omit the corresponding figures and report only the language model results for this setting. In both settings, we did not observe clear changes compared to the pre-trained model, either in the layer-wise trends (Figure 48, Figure 51–52) or in the spatial distribution of high-accuracy patches (Figure 49–50, Figure 53–55). This suggests that, despite the improvement in VQA accuracy (appendix H.2), the linear representations captured by probing remain largely unchanged after fine-tuning with LoRA.

Several factors may explain this gap between the improved VQA accuracy and the unchanged linear representations. First, LoRA updates may primarily adjust nonlinear transformations or later decoding stages rather than the linear subspace captured by probing. Second, the low rank of LoRA (rank 4) may be insufficient to alter the global structure of the representation space, while still enabling local adjustments that improve task performance. Disentangling these factors requires further investigation, which we leave as future work.

# I Computational resources

We used NVIDIA RTX 6000 Ada, RTX PRO 6000 Blackwell Max-Q, and H100 GPUs in this study.

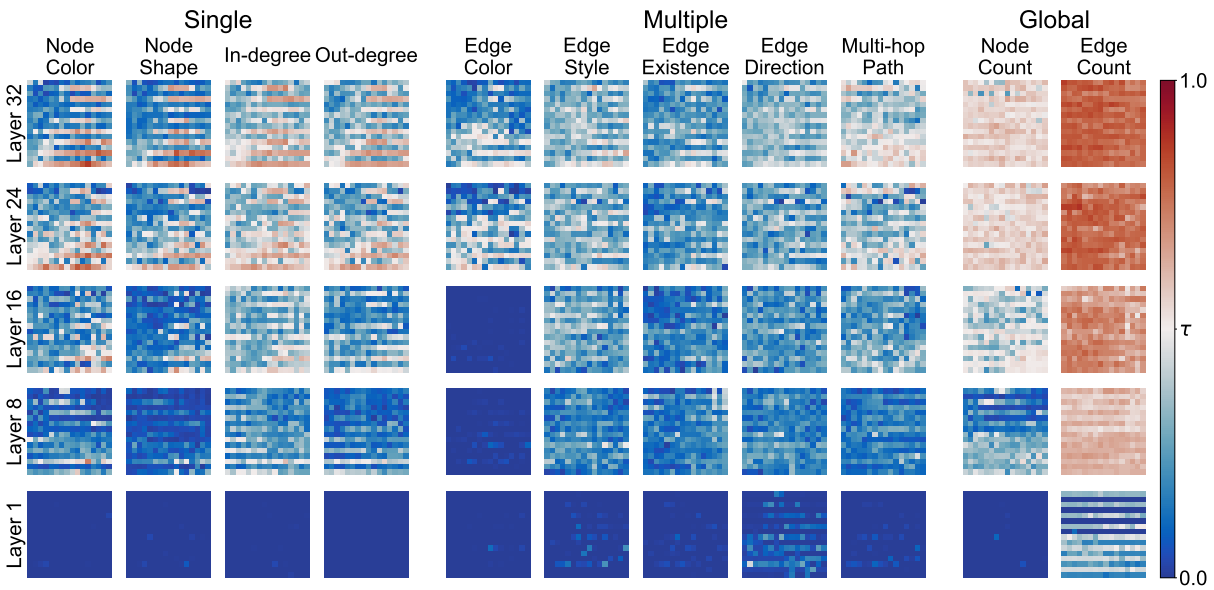


Figure 25: Accuracy for each hidden state in the image input part of the Qwen2.5-VL 7B vision encoder. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are similar to Qwen3-VL (Figure 4).

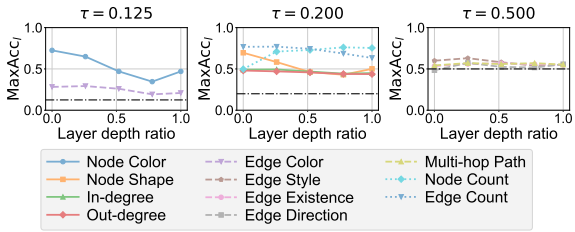


Figure 26: Layer-wise maximum accuracy in the language model of Qwen2.5-VL 7B. The x-axis shows relative layer position (0 is the input layer, 1 is the final layer), and the y-axis shows accuracy. The black dotted line indicates the threshold. Trends are similar to Qwen3-VL (Figure 5).

## J Usage of AI assistants

For writing this paper and developing the experimental code, we used AI assistants (e.g., ChatGPT and GitHub Copilot). Their use was limited to tasks such as code completion, translation, text editing, and table formatting, and all technical content is based solely on the authors' ideas.

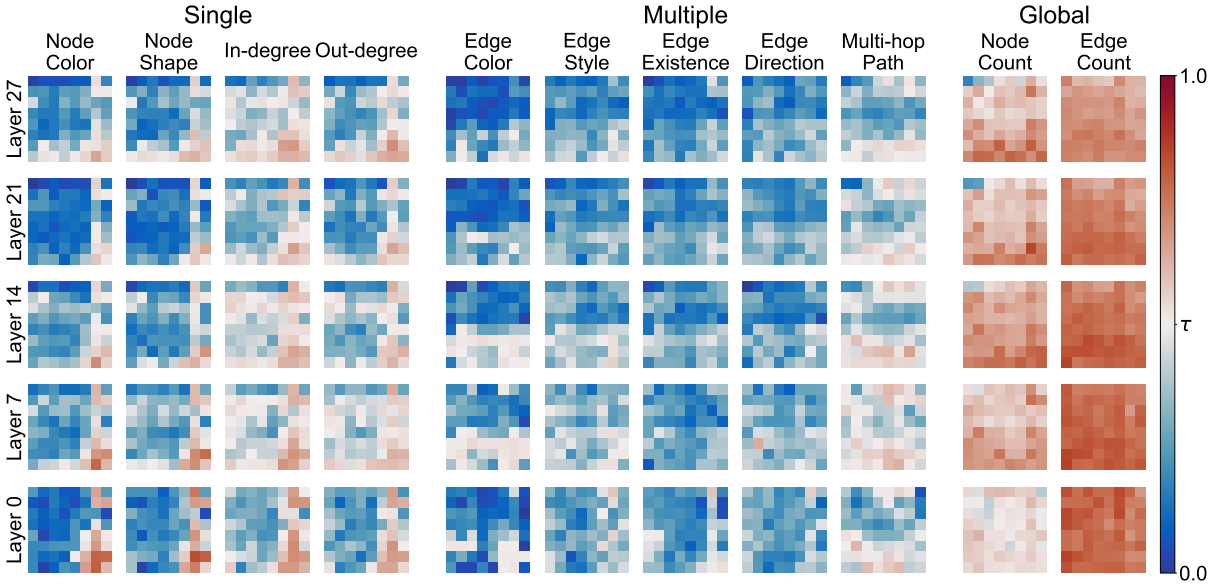


Figure 27: Accuracy for each hidden state in the image input part of the Qwen2.5-VL 7B language model. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are similar to Qwen3-VL (Figure 6).

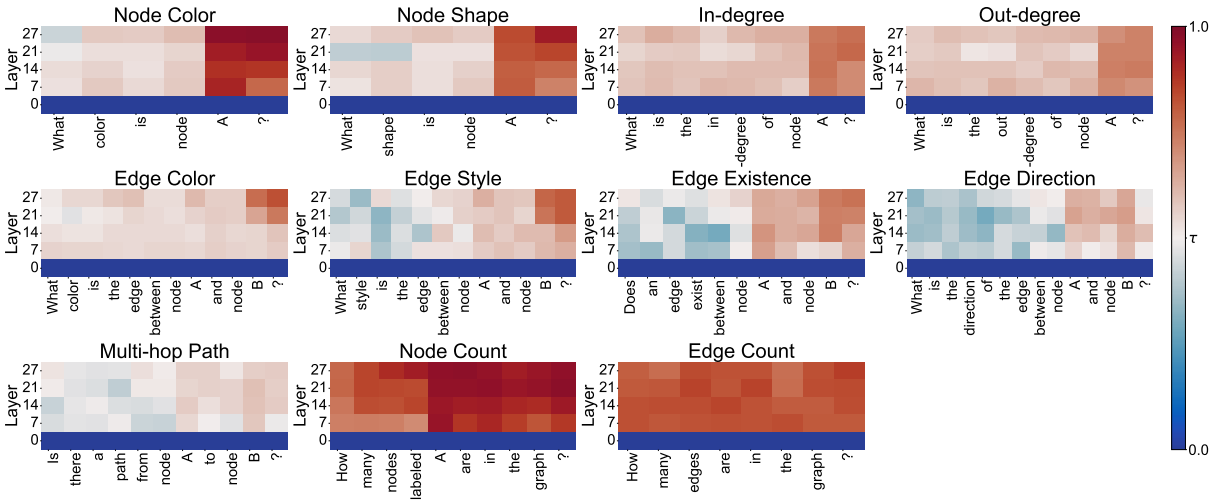


Figure 28: Probing results for text tokens in Qwen2.5-VL 7B. The x-axis shows token position in the question, and the y-axis shows layer position. Trends are similar to Qwen3-VL (Figure 7).

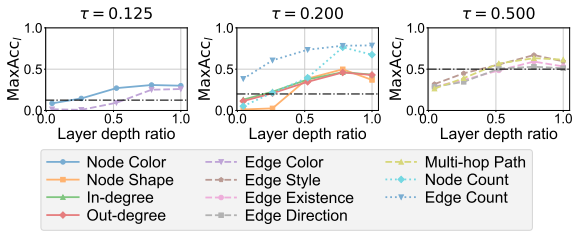


Figure 29: Layer-wise maximum accuracy in the vision encoder of LLaVA1.5 7B. The x-axis shows relative layer position (0 is the input layer, 1 is the final layer), and the y-axis shows accuracy. The black dotted line indicates the threshold. Trends are similar to Qwen3-VL (Figure 3).

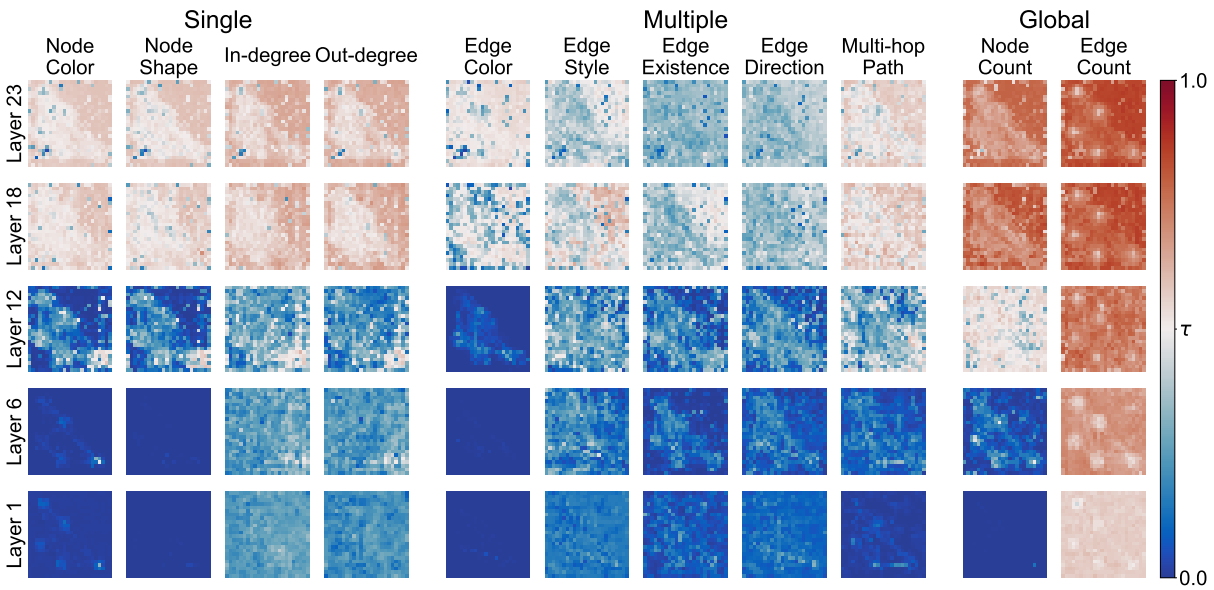


Figure 30: Accuracy for each hidden state in the image input part of the LLaVA1.5 7B vision encoder. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Unlike Qwen3-VL (Figure 4), background patches show higher accuracy than patches corresponding to node positions.

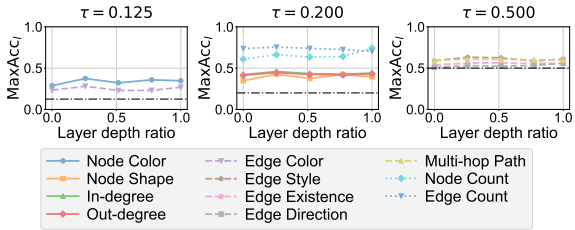


Figure 31: Layer-wise maximum accuracy in the language model of LLaVA1.5 7B. The x-axis shows relative layer position (0 is the input layer, 1 is the final layer), and the y-axis shows accuracy. The black dotted line indicates the threshold. Trends are similar to Qwen3-VL (Figure 5).

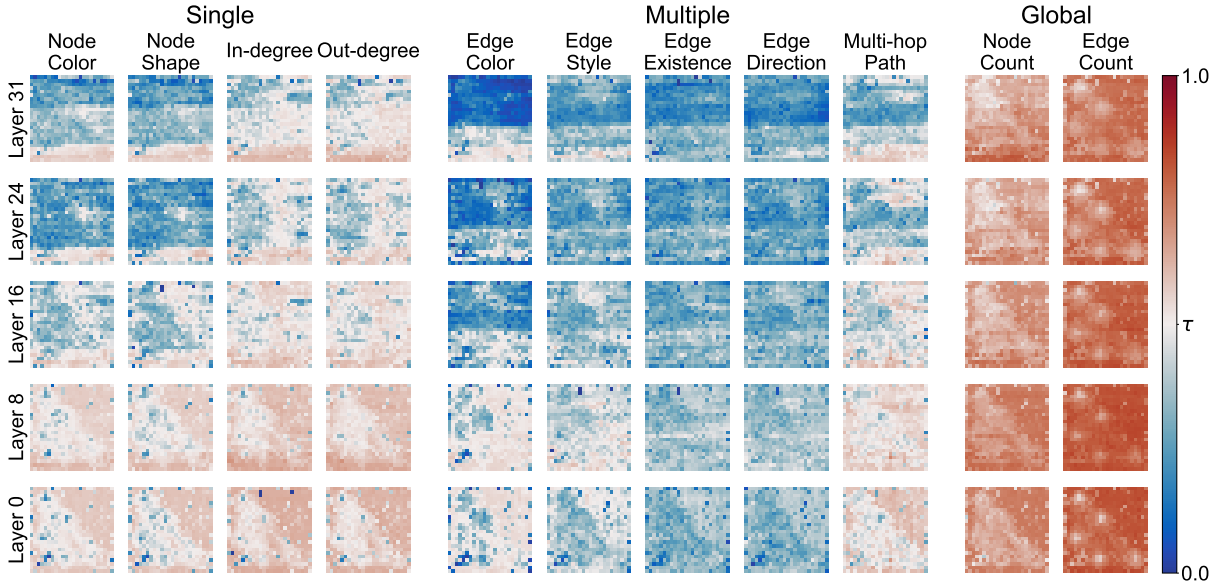


Figure 32: Accuracy for each hidden state in the image input part of the LLaVA1.5 7B language model. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Unlike Qwen3-VL (Figure 6), background patches show higher accuracy than patches corresponding to node positions.

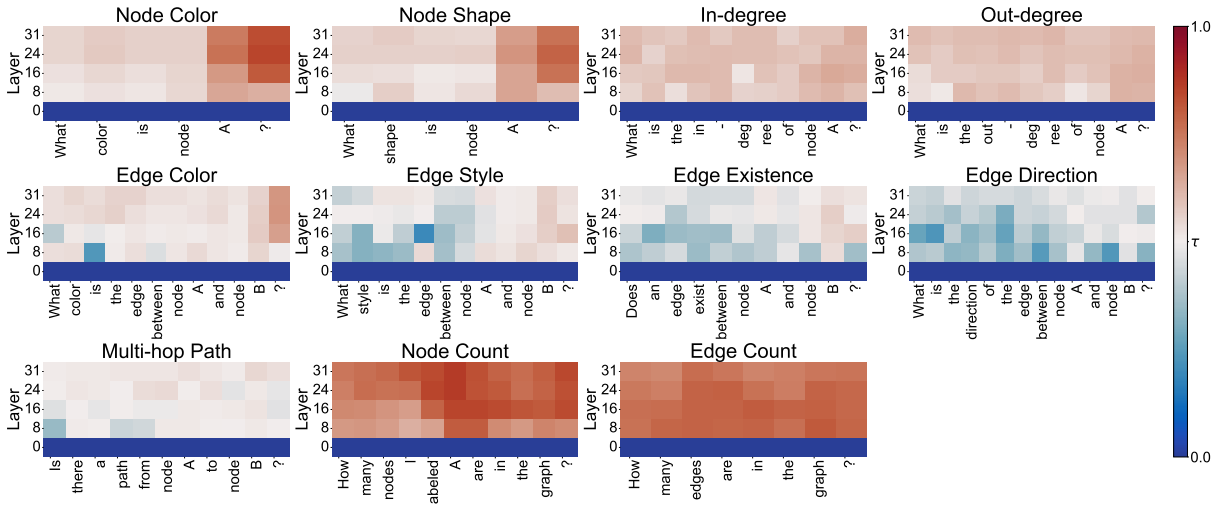


Figure 33: Probing results for text tokens in LLaVA1.5 7B. The x-axis shows token position in the question, and the y-axis shows layer position. Trends are broadly similar to Qwen3-VL (Figure 7), though less pronounced.

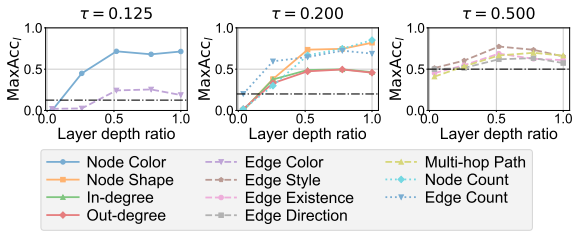


Figure 34: Layer-wise maximum accuracy in the vision encoder of Gemma3-4B-IT. The x-axis shows relative layer position (0 is the input layer, 1 is the final layer), and the y-axis shows accuracy. The black dotted line indicates the threshold. Trends are similar to Qwen3-VL (Figure 3), though accuracy is lower for some aspects.

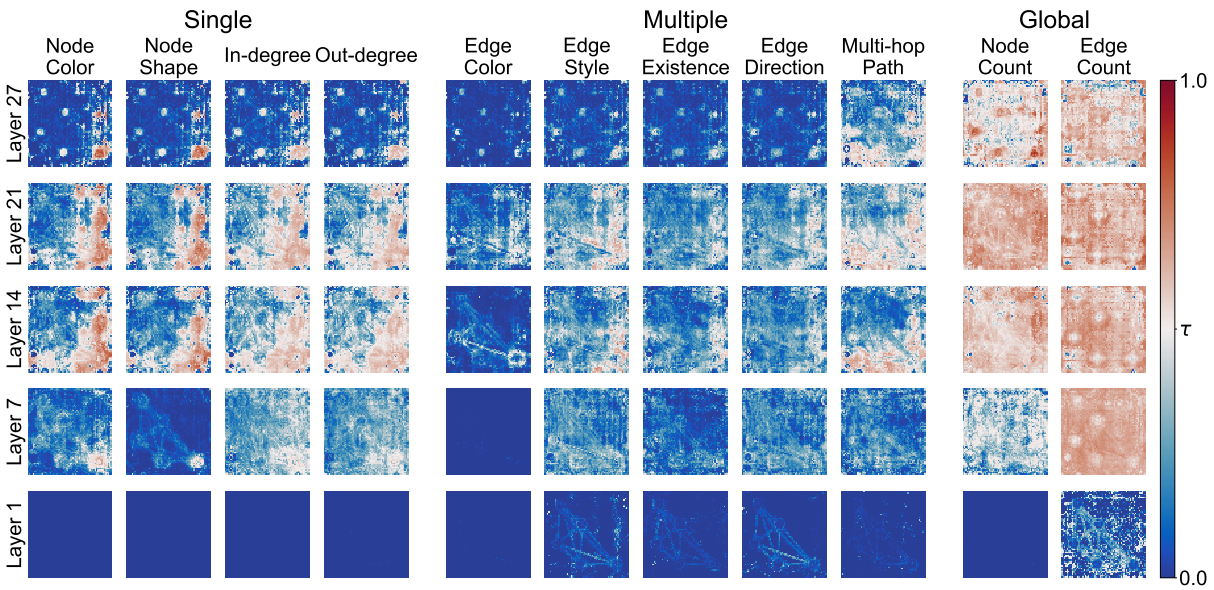


Figure 35: Accuracy for each hidden state in the image input part of the Gemma3-4B-IT vision encoder. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are similar to Qwen3-VL (Figure 4), though accuracy is lower for some aspects.

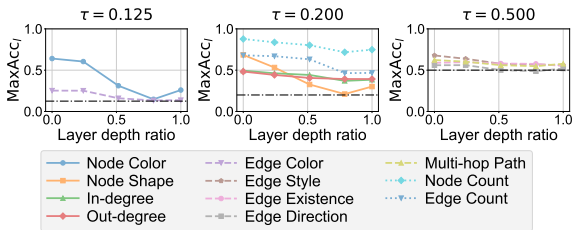


Figure 36: Layer-wise maximum accuracy in the language model of Gemma3-4B-IT. The x-axis shows relative layer position (0 is the input layer, 1 is the final layer), and the y-axis shows accuracy. The black dotted line indicates the threshold. Trends are similar to Qwen3-VL (Figure 5), though accuracy is lower for some aspects.

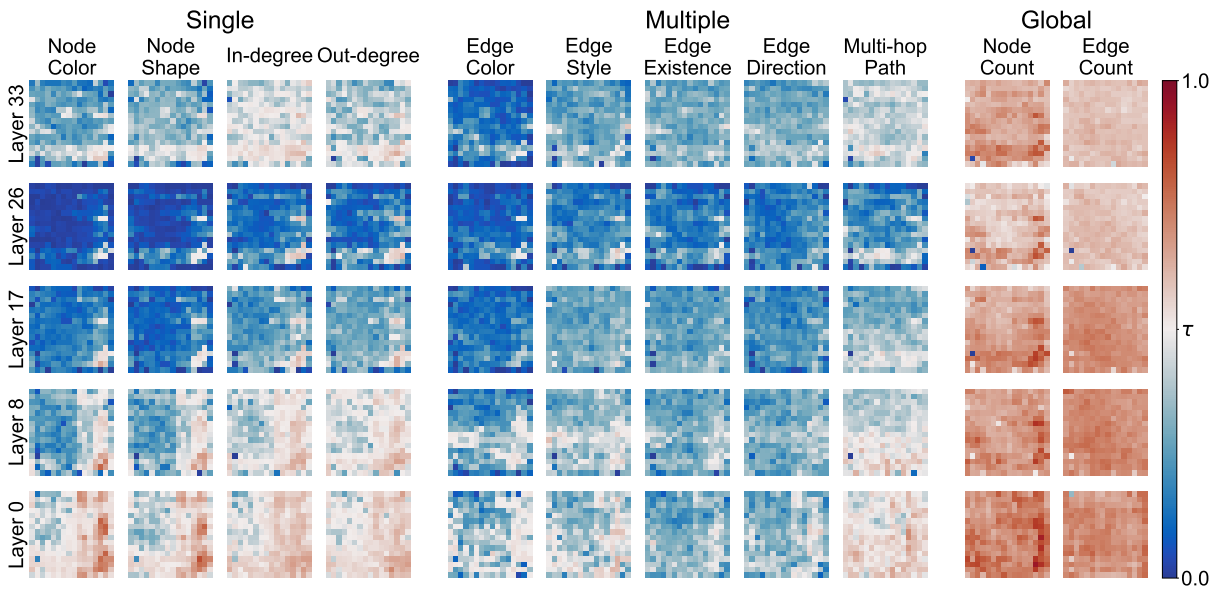


Figure 37: Accuracy for each hidden state in the image input part of the Gemma3-4B-IT language model. Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect. Spatial patterns are similar to Qwen3-VL (Figure 6), though accuracy is lower for some aspects.

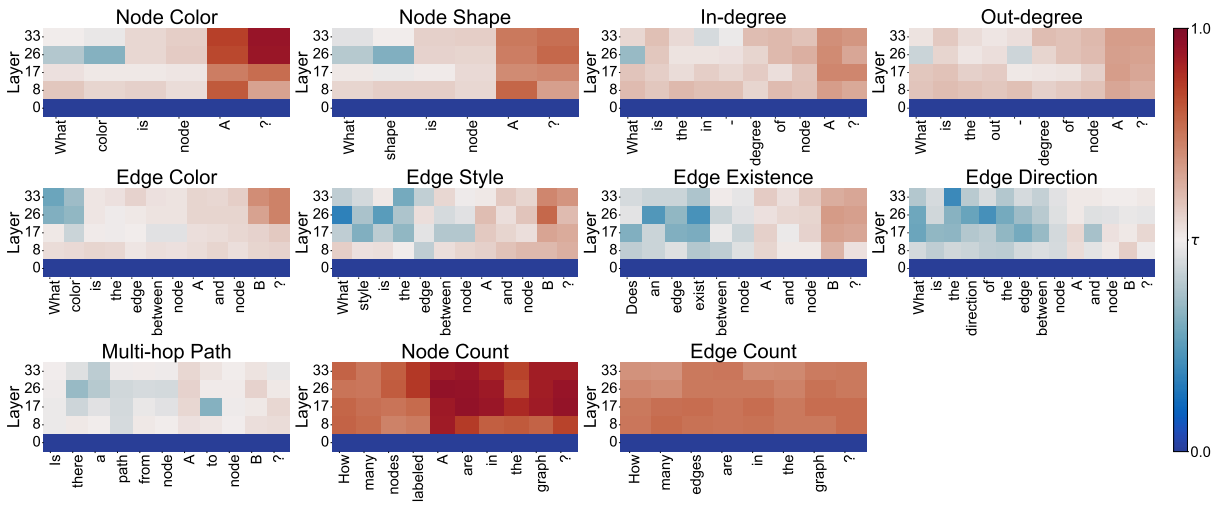


Figure 38: Probing results for text tokens in Gemma3-4B-IT. The x-axis shows token position in the question, and the y-axis shows layer position. Trends are broadly similar to Qwen3-VL (Figure 7), though less pronounced.

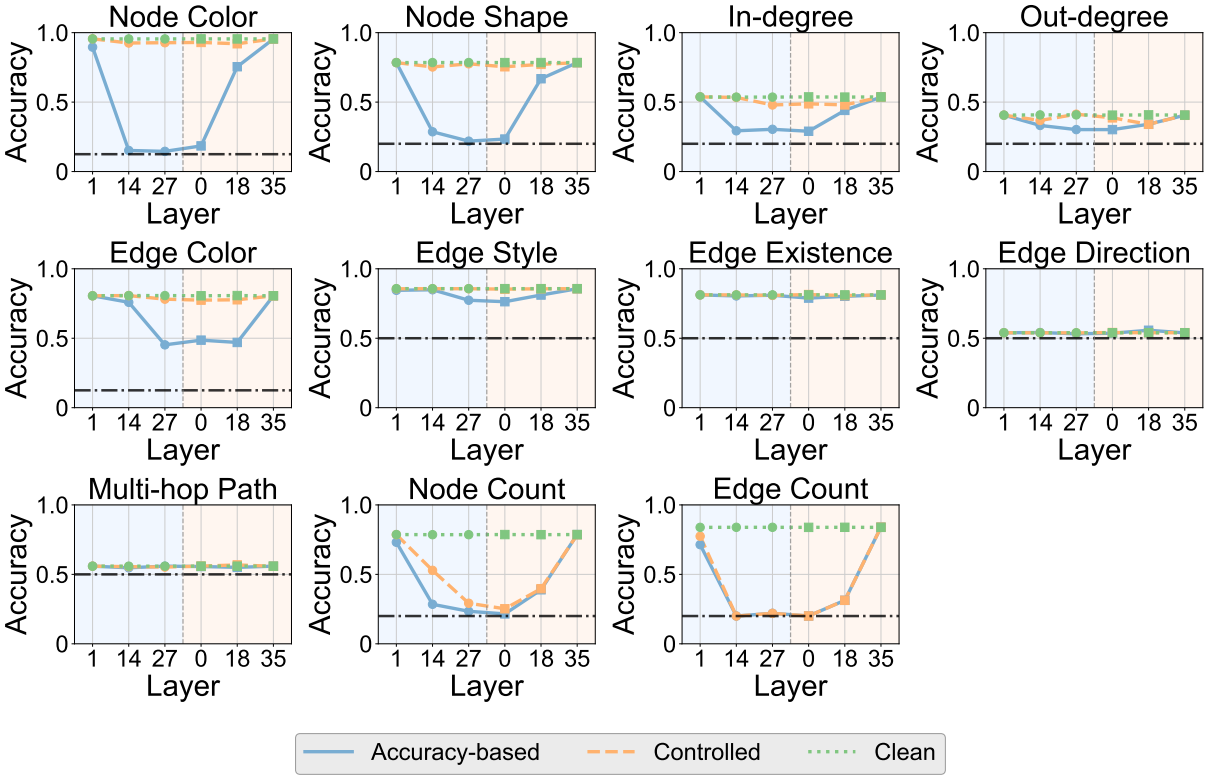


Figure 39: *Clean*, *accuracy-based*, and *controlled* accuracy at each intervention layer for Qwen3-VL 8B. The background colors of the plots indicate the results for the vision encoder and the language model, respectively.

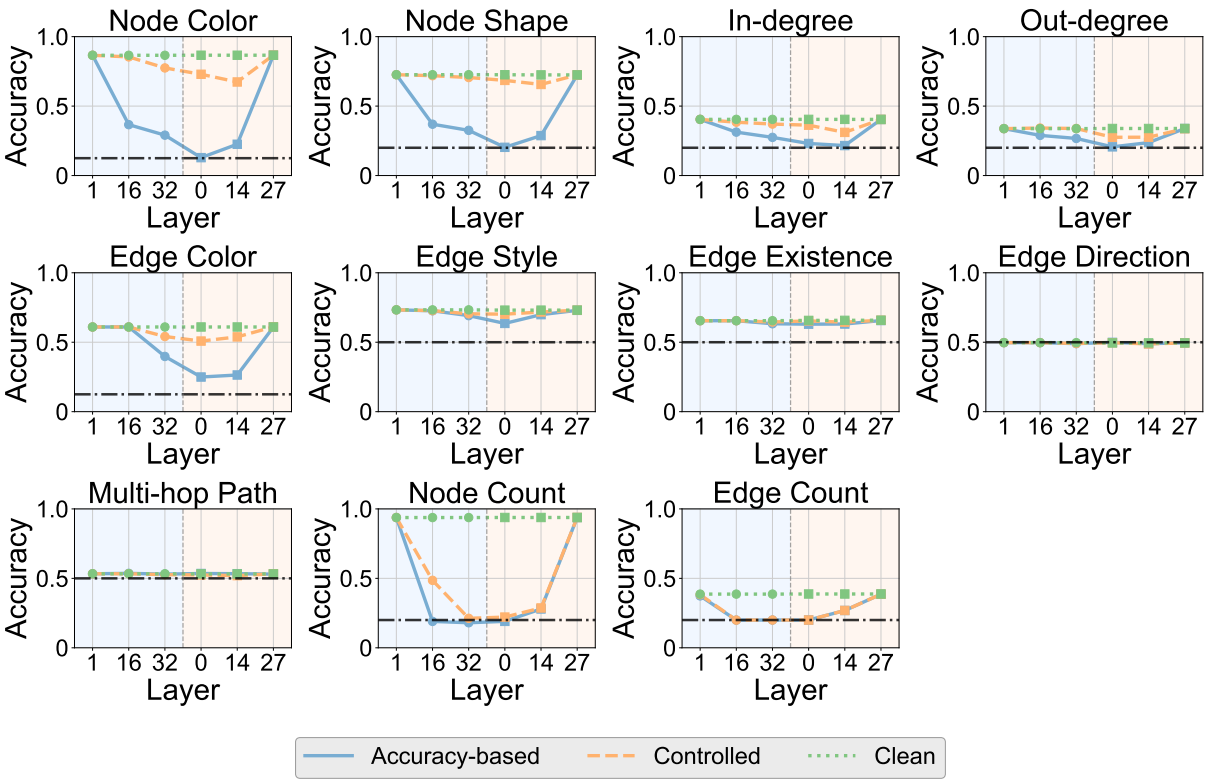


Figure 40: *Clean*, *accuracy-based*, and *controlled* accuracy at each intervention layer for Qwen2.5-VL 7B. The background colors of the plots indicate the results for the vision encoder and the language model, respectively.

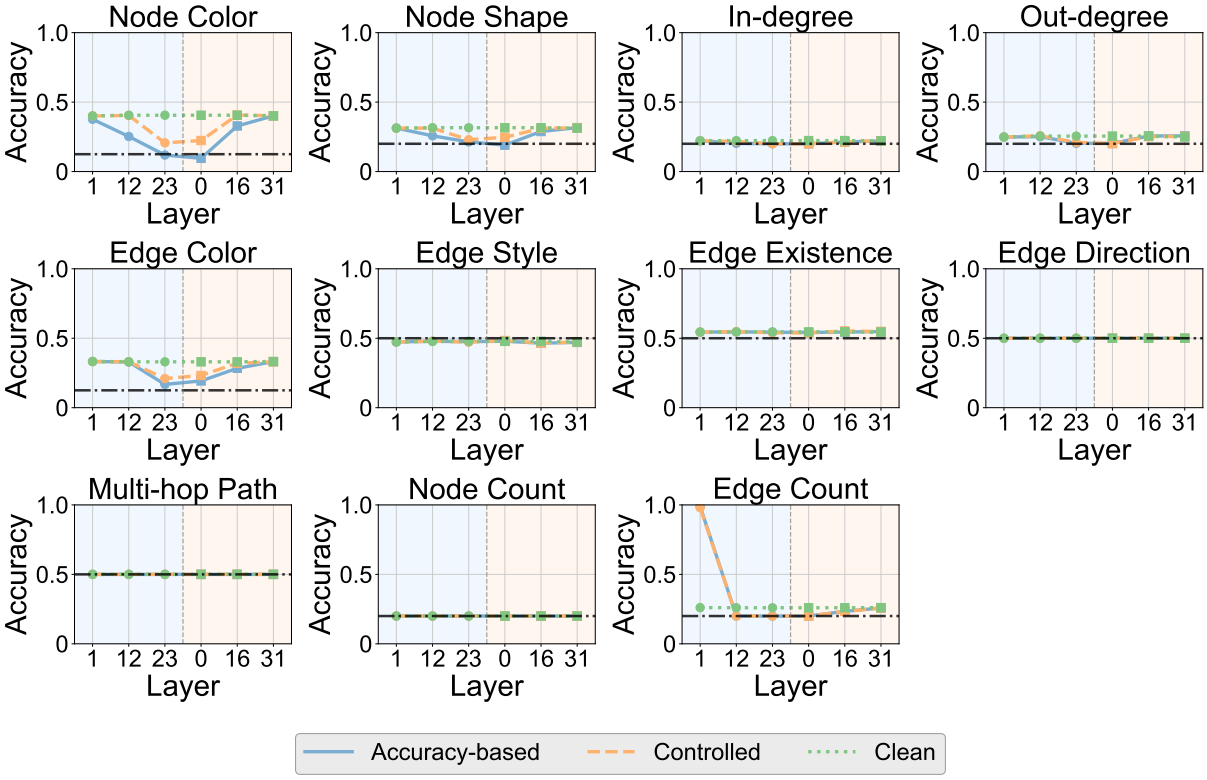


Figure 41: *Clean*, *accuracy-based*, and *controlled* accuracy at each intervention layer for LLaVA1.5 7B. The background colors of the plots indicate the results for the vision encoder and the language model, respectively.

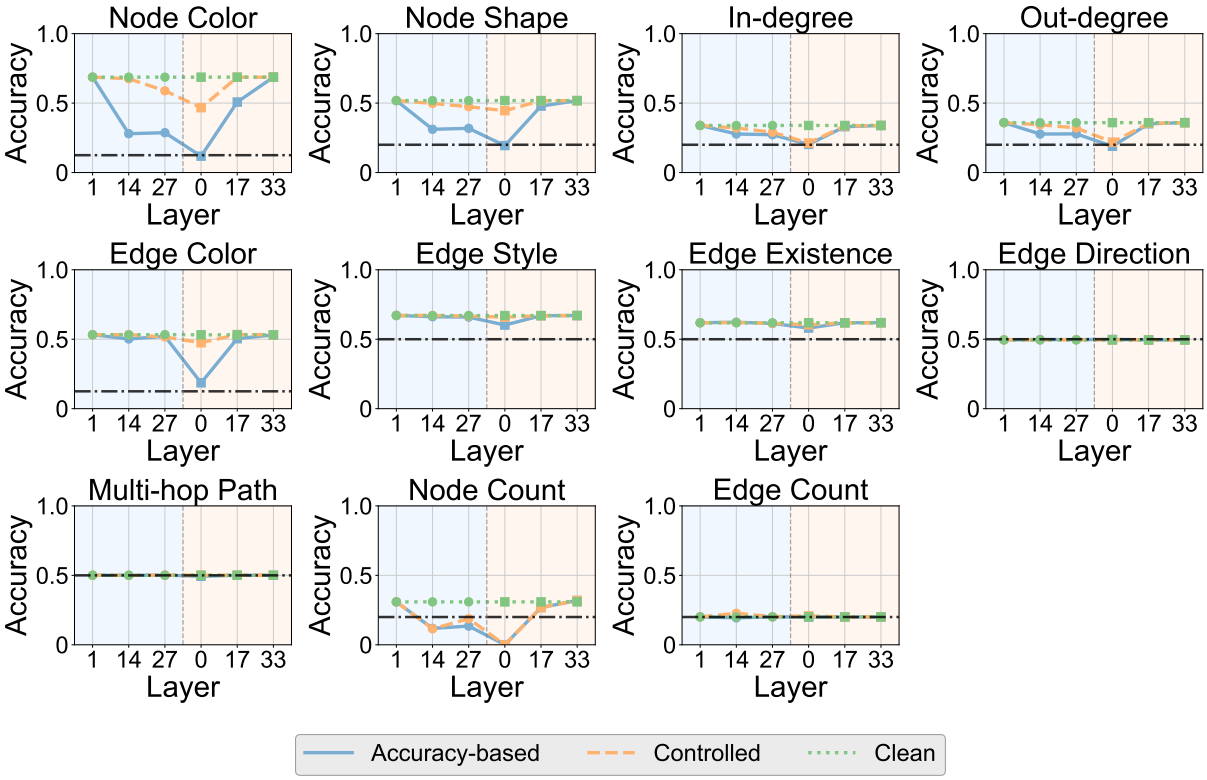


Figure 42: *Clean*, *accuracy-based*, and *controlled* accuracy at each intervention layer for Gemma3-4B-IT. The background colors of the plots indicate the results for the vision encoder and the language model, respectively.

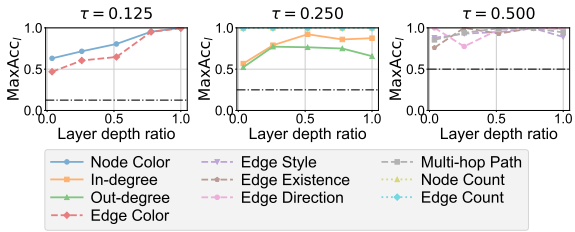


Figure 43: $\mathrm{MaxAcc}_l$ per layer in the vision encoder of Qwen3-VL 8B in this setting. The x-axis denotes the relative layer position, and the y-axis denotes accuracy. All aspects exceed the threshold within the vision encoder, including Multiple aspects whose accuracy does not reach the threshold in the synthetic-data setting.

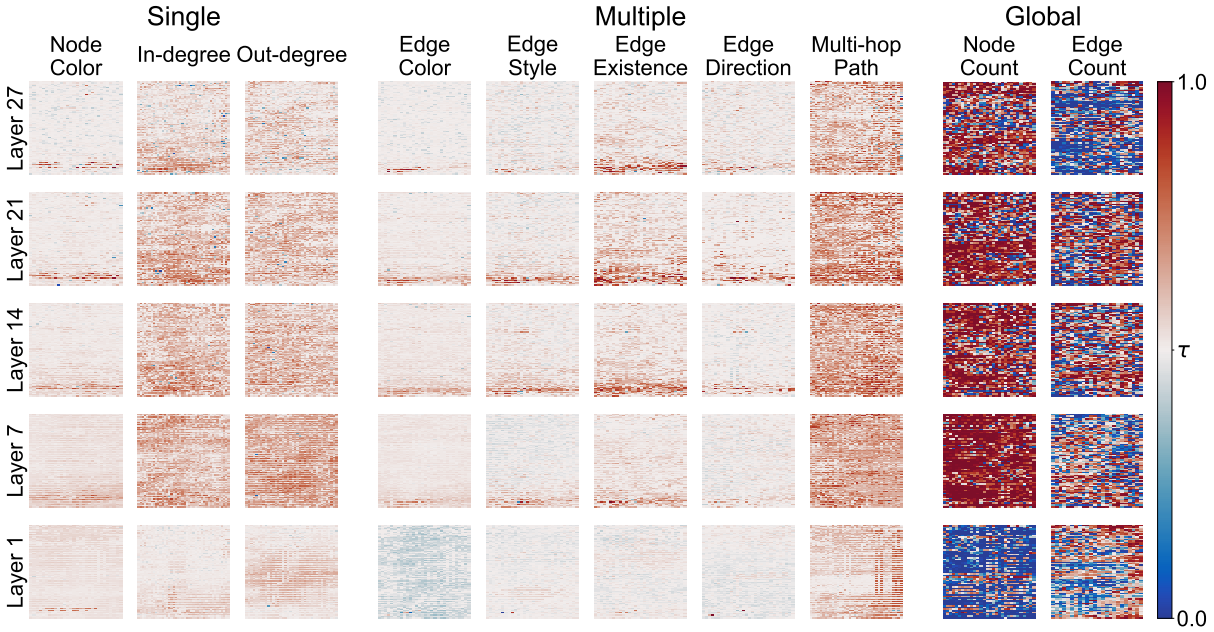


Figure 44: $\mathrm{acc}_{l,t}(d_{\mathrm{fix}})$ in the vision encoder of Qwen3-VL 8B in this setting, for a representative subset $d_{\mathrm{fix}}$. Each heatmap shows accuracy by patch position for a specific layer and aspect. Single aspects localize near the target node, Global aspects spread across background patches, and Multiple aspects also show high-accuracy patches, unlike in the synthetic-data setting.

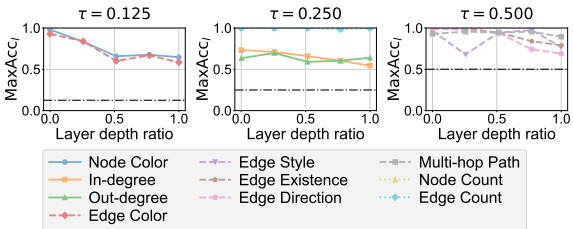


Figure 45: $\mathrm{MaxAcc}_l$ per layer in the image-input part of the language model in Qwen3-VL 8B in this setting. All aspects maintain accuracy above the threshold.

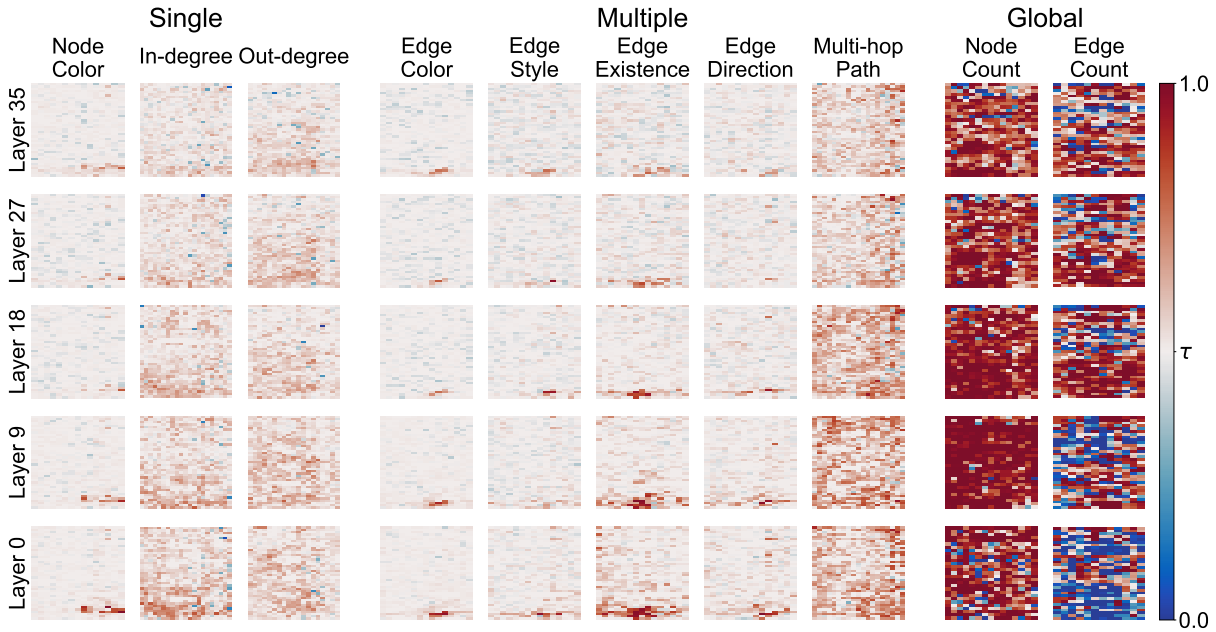


Figure 46: $\mathrm{acc}_{l,t}(d_{\mathrm{fix}})$ in the image-input part of the language model in Qwen3-VL 8B in this setting, for a representative subset $d_{\mathrm{fix}}$. The spatial distribution of high-accuracy patches is largely unchanged from the vision encoder result.

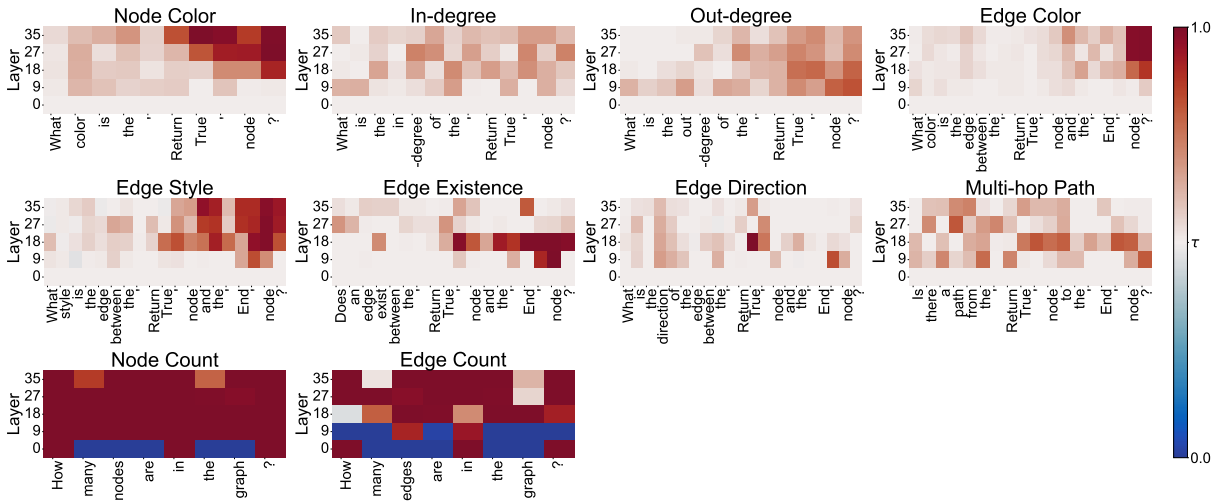


Figure 47: Probing results for text tokens in the language model of Qwen3-VL 8B in this setting. The x-axis shows token position in the question, and the y-axis shows layer position. For `Edge Direction` and `Edge Existence`, high accuracy appears only in intermediate layers and does not propagate to the final layer.

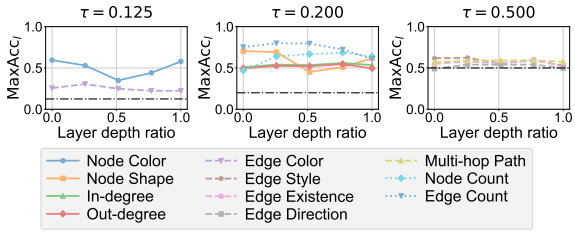


Figure 48: $\mathrm{MaxAcc}_l$ per layer in the language model of Qwen3-VL 8B (FT w/L).

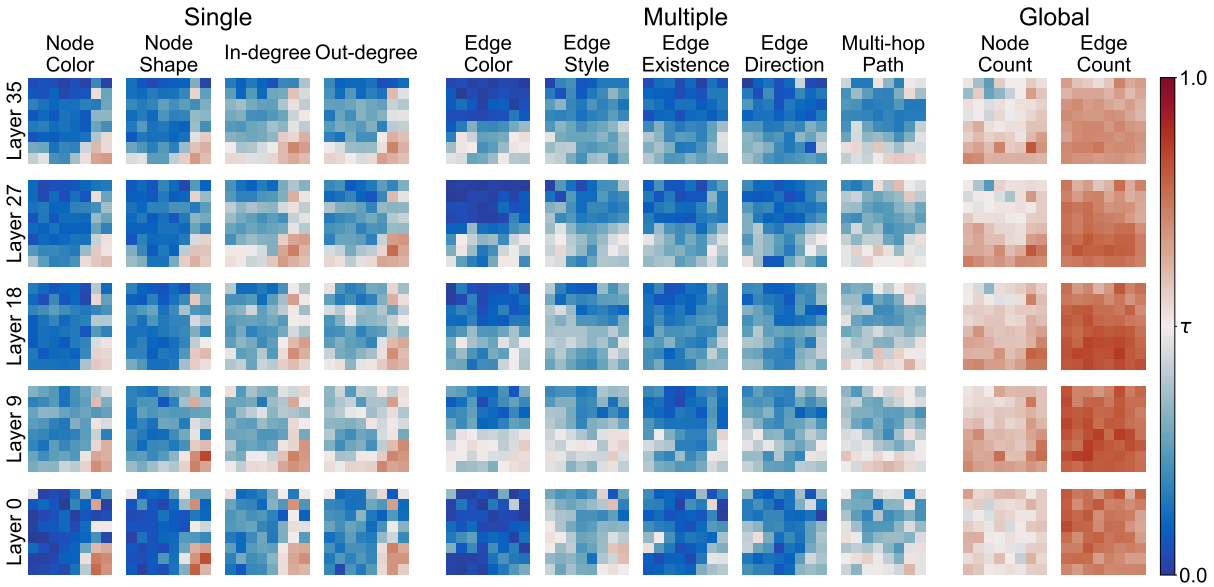


Figure 49: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B language model (FT w/L). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect.

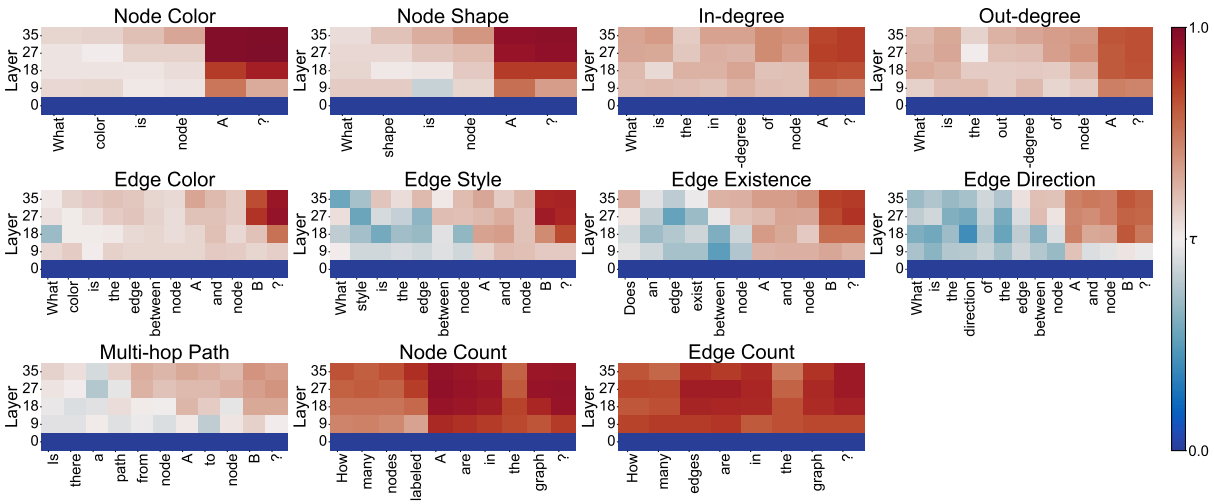


Figure 50: Probing results for text tokens in the language model of Qwen3-VL 8B (FT w/L). The x-axis shows token position in the question, and the y-axis shows layer position.

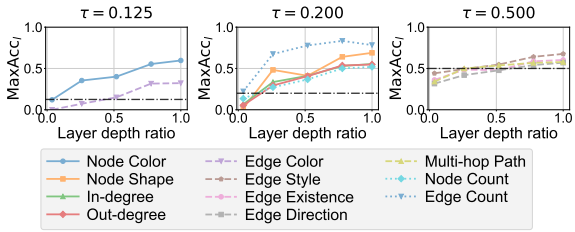


Figure 51: Layer-wise maximum accuracy in the vision encoder of Qwen3-VL 8B (FT w/VL).

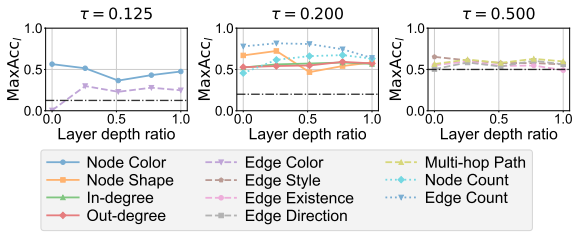


Figure 52: $\mathrm{MaxAcc}_l$ per layer in the language model of Qwen3-VL 8B (FT w/VL).

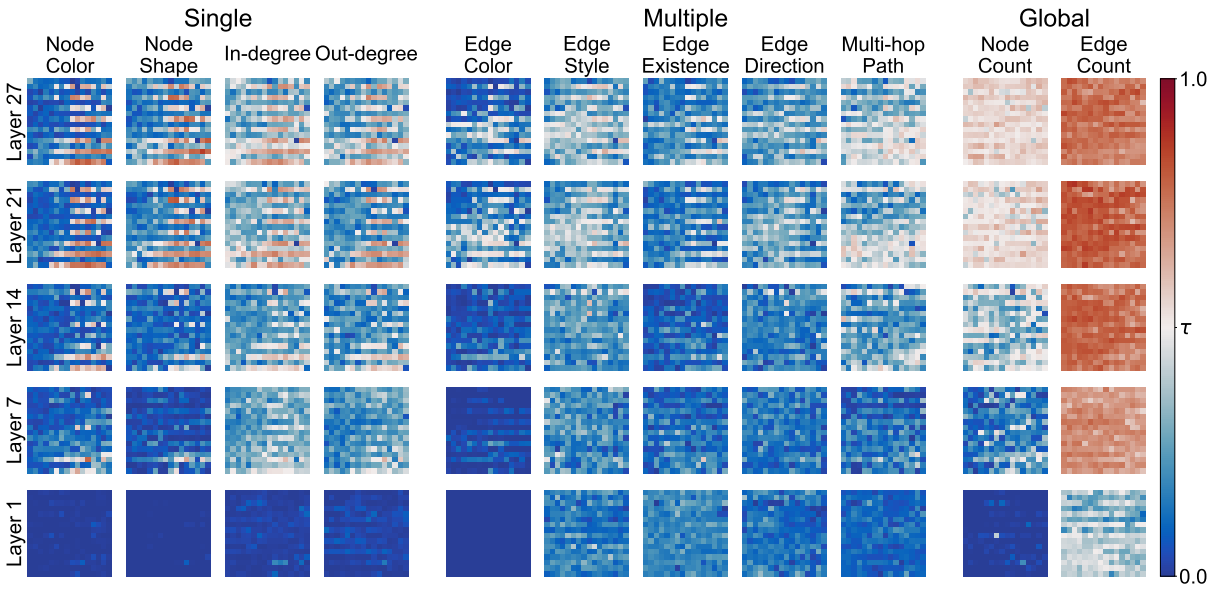


Figure 53: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B vision encoder (FT w/VL). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect.

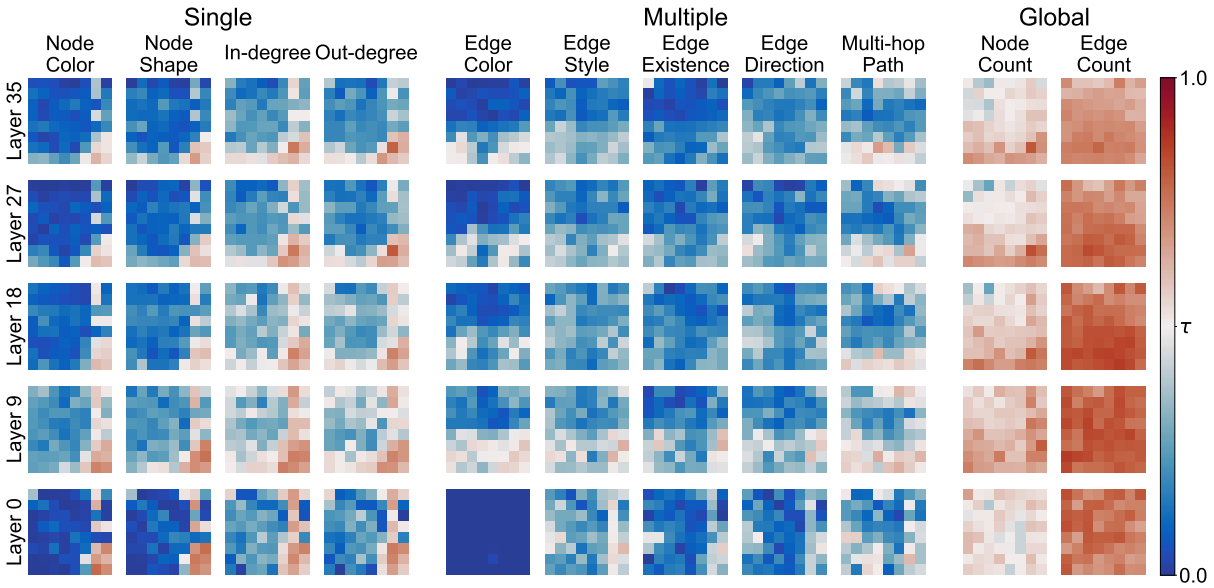


Figure 54: Accuracy for each hidden state in the image input part of the Qwen3-VL 8B language model (FT w/VL). Each heatmap shows the accuracy of individual hidden states for a specific layer and aspect.

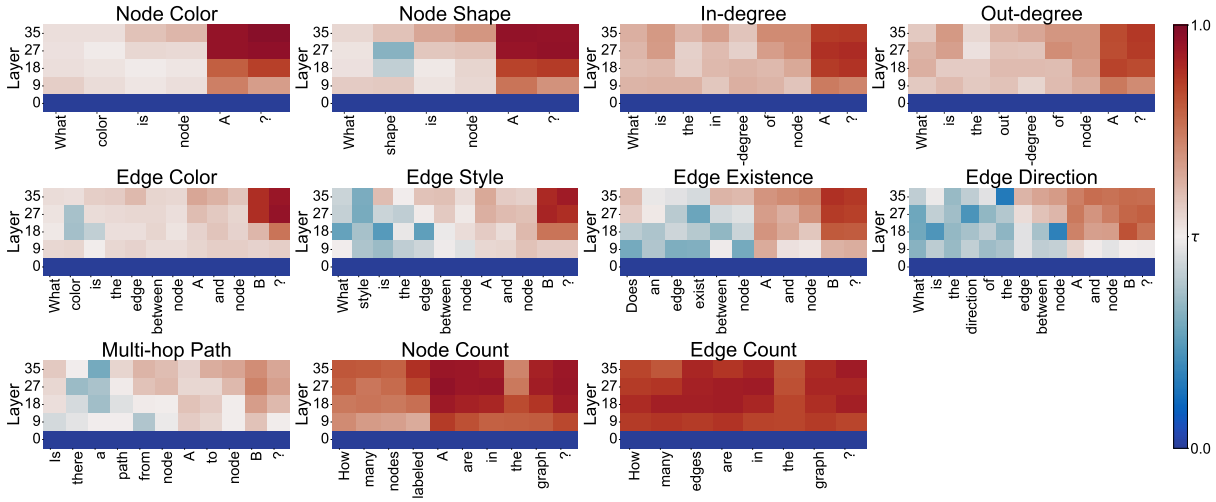


Figure 55: Probing results for text tokens in the language model of Qwen3-VL 8B (FT w/VL). The x-axis shows token position in the question, and the y-axis shows layer position.