A. CI Implementations of CompCap

This section elaborates the implementation of each CI type. We show the detailed pipeline implementation of collage in A.1; image-text in A.2; chart in A.3; diagram in A.4; Code inA.5; and table in A.6.

A.1. Collage Implementation

We summarize the workflow in Figure 4. In this section, we first show how the image-captions used for composing collage are retrieved, then we elaborate on the design of collage caption and LLM prompting.

A.1.1. Data Sources, Layout, and Retrieval Engines

We retrieve image-caption pairs from existing image-caption datasets. And we maintain a curated entity list of public figures, artworks, landmarks, and brands sourced from web data.

Pre-processing. We first process the datasets for better retrieval quality and efficiency:

1. *Construct entity-sample lookup table.* For each image, we identify entities in the original caption that match entries in our maintained entity list and create an entity-image lookup table specifically for entity-based retrieval.
2. *Pre-compute embeddings.* For each sample in both datasets, we pre-compute Dino-v2 image embeddings and CLIP caption embeddings.

Layout. We define two collage layouts: grid collage and auto collage. In the grid collage layout, images are arranged in an $n \times m$ grid, where $n, m \in \{1, 2, 3, 4\}$. To increase layout diversity, cells within the grid can merge to form larger cells. Since rows and columns in the grid layout are aligned, the layout will specify the width/height ratio for each image within a cell, posing constraints for the retrieval process. To further enhance diversity, we introduce the auto collage layout, where only rows or columns are aligned. This enable composing images of arbitrary width/height ratio into a collage image. We demonstrate some examples of the layout in Figure 7.

Similarity-based retrieval. We start by uniformly sampling an image-caption pair as the anchor data $x_\alpha = (I_\alpha, T_\alpha)$ and then retrieve the top 20 most similar image-caption pairs from the database $\mathcal{D}$. Let I_α^{dino} and T_α^{clip} represent the Dino-v2 image embedding and CLIP text embedding for the anchor data, while I^{dino} and T^{clip} are the embeddings of an data $x = (I, T) \in \mathcal{D}$. The similarity score between x_α and x is computed as follow:

$$\text{sim}(x_\alpha, x) = \cos(I_\alpha^{\text{dino}}, I^{\text{dino}}) + \cos(T_\alpha^{\text{clip}}, T^{\text{clip}}).$$

From the top 20 candidates, we randomly select samples to construct the collage. Where width/height ratios are specified for candidate images, a filter is applied to the database prior to calculating similarity.

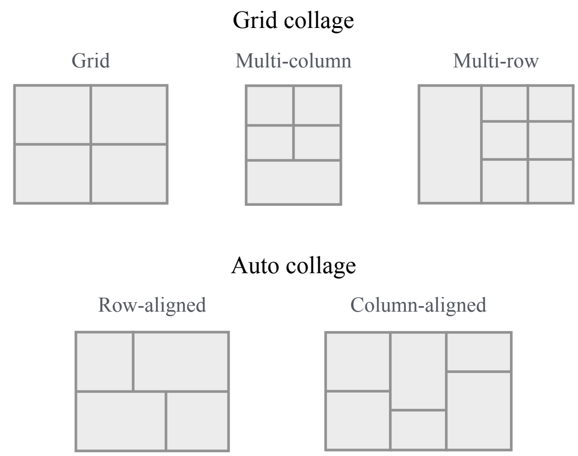


Figure 7. Examples of collage layout.

Entity-based retrieval. To optimize retrieval, we narrow down the entity list to include only entities that appear more than twice in the dataset. We randomly sample a keyword from this list and apply rule-based matching in the database to select related data. Since such data are sparse, we only use the auto layout to compile collages to avoid the width/height restrictions. In post-processing, we further de-duplicate collages to ensure variety.

In both similarity-based and entity-based retrieval, there may be cases where retrieved images are only loosely related to the anchor image. For instance, when retrieving images based on an anchor image of a cricket game, some results might instead depict baseball due to their visual similarities. However, as long as the corresponding caption accurately describes the image, the final generated caption for the collage will maintain accuracy. This introduces counterfactual samples into the CompCap-118K dataset, contributing to model debiasing by providing varied contexts.

A.1.2. Caption and Prompt Design

We design the caption such that it provides a detailed walk-through on the images in the collage. Particularly, it either goes over the images by rows or by columns. For each row or column, we specify the demonstration order to be from left to right or from top to bottom, which comes first depends on the generated layout. For instance, for grid collage that comes with a multi-column cell, the caption is designed to go over images by row. And for auto collage whose columns are aligned, the caption is designed to go over images by column.

In order for LLMs to generate desired caption, different layout uses prompt that are slightly different in terms of the coordinate system notation and the in-context examples.

: Generate a complete caption for a collage of photographs. Each photograph is labeled with its position, and its caption is also provided.

The position of the photograph follows the coordinate notations below:
1. **(idx, idx)**:
   - This notation specifies a particular cell within the grid.
   - The first `idx` represents the row number, and the second `idx` represents the column number.
   - Example: `(2,1)` refers to the cell located at the second row and first column.

2. **(idx, :)**:
   - This notation is used to indicate an entire row within the grid.
   - The `idx` specifies the row number, and the colon `:` indicates that all columns in that row are included.
   - Example: `(1,:)` refers to all the cells in the first row.

3. **(:, idx)**:
   - This notation indicates all the rows in a specific column of the grid.
   - The colon `:` represents all rows, and `idx` specifies the column number.
   - Example: `(:,2)` refers to all the cells in the second column.

In-context example selection (multi-column)

Below is an example:
[Layout, caption, and instruction]:
- number of images: 3
* (1,1): A close-up shot of a tie knot featuring a silver or light gray tie intricately tied in an Eldredge knot.
* (2,1): A very big cute white polar bear is laying on his back in the snow.
* (:,2): A woman driving a car with a dog in the backseat down a street.

[Recaption]:
The image is a collage consisting of three distinct sections arranged in two rows:
1. **Left Column**:
  - **Top**: A close-up shot of a tie knot. The tie is silver or light gray, and the knot is intricately tied, likely in an Eldredge knot.
  - **Bottom**: A polar bear laying on his back half on snow.
2. **Right Column**: A person drives in a car as a dog sits in the back.
This collage showcases different settings and activities, from formal attire to nature and casual outdoor scenery, highlighting versatility in environments and attire.

Now, given the following layout and captions, perform the same task:

 # [Layout, caption, and instruction]:
- number of images: 3
* (1,1): A refreshing salad with mixed greens, tender pieces of salmon, edamame, green beans, and a sprinkle of lemon zest.
* (2,1): Grilled steak slices served atop a bed of spinach and arugula, complemented by fresh apple slices and blueberries.
* (:,2): A savory salad of thinly sliced beef topped with a generous handful of microgreens.

Create a new caption strictly following the instructions below:
1. Use the position labels in the caption to determine the demonstration order, but don't include the position label.
2. For each caption of the image, rephrase it to make it fluent, but don't add additional information that can not be inferred from the original captions.
3. Your response should start with "The image".
4. Your response should strictly follow the format provided by the example above.

 # [Recaption]:

: The image is a collage consisting of three distinct sections arranged in two columns:
1. **Left Column**:
   - **Top**: A vibrant salad featuring mixed greens, tender salmon pieces, edamame, green beans, and a hint of lemon zest.
  - **Bottom**: Grilled steak slices atop a bed of spinach and arugula, complemented by fresh apple slices and blueberries.
2. **Right Column**: Grilled steak slices served over a bed of spinach and arugula, complemented by fresh apple slices and blueberries.
This collage showcases a variety of fresh, flavorful salads, each offering a unique combination of ingredients for a balanced, appetizing presentation.

Figure 8. **Prompt design and response example for grid layout.**

: Generate a complete caption for a collage of photographs. Each photograph is labeled with its position, and its caption is also provided.

Below is an example:
[Layout, caption, and instruction]:
-number of images: 5
Row 0
* Image 0: A luxurious overwater suite overlooking Bora Bora, French Polynesia.
* Image 1: The beautiful beach in Bora Bora Resort and Bungalows.
* Image 2: A view of the Bora Bora Resort from above.
Row 1
* Image 0: The water bungalows at Intercontinental Club Borabu Atoll, Bora Bora Island.
* Image 1: The beaches of Bora Bora Island, Vanuatu.

[Recaption]:
The image is a collage of five beautiful tropical beach scenes, showcasing different luxurious and scenic views. Here are the descriptions of each part of the collage, from top to bottom:
1. **Top Row:**
 - **First Image:** A luxurious overwater suite overlooks the crystal-clear waters of Bora Bora.
 - **Second Image:** The beautiful beach at Bora Bora Resort and Bungalows.
 - **Third Image:** Aerial view of Bora Bora Resort's overwater bungalows and turquoise waters.
2. **Bottom Row:**
 - **First Image:** The iconic water bungalows at Intercontinental Club Borabu Atoll on Bora Bora Island.
 - **Second Image:** Serene beaches of Bora Bora Island, Vanuatu, with golden sands and gentle waves.
The collage showcases stunning tropical paradises, featuring luxurious overwater bungalows, pristine beaches, and breathtaking views from Bora Bora in French Polynesia. Each image highlights the beauty and tranquility of these exotic locations.

Now, given the following layout and captions, perform the same task:

 # [Layout, caption, and instruction]:
-number of images: 4
Row 0
* Image 0: Sunset over the moai on easter island, one of easter island's many unesco sites. A row of moai statues on a grassy field, overlooking the ocean. The sun is setting, casting a warm glow on the statues and the cliff in the background.
* Image 1: The moai on easter island, one of the world's most visited sites. A row of moai statues on a grassy hill.
Row 1
* Image 0: The head of rapio in easter island with grass and shrubs growing on the hillside. A rocky hillside with a few green plants.
* Image 1: Five moai statues on the sea wall at sunrise, easter island, chile. Sunset over the ocean with silhouettes of moai statues on Easter Island.

Create a new caption following the instruction below:
1. Use the position labels in the caption to determine the demonstration order, but don't include the position label.
2. For each caption of the image, rephrase it to make it fluent.
3. Your response should start with "The image".
4. When there is only one row, only need to indicate column position.
5. When there is only one column, only need to indicate row position.

 # [Recaption]:

: The image is a collage of four remarkable scenes from Easter Island, capturing the island's iconic moai statues against stunning natural backdrops. Here are the descriptions of each part of the collage, arranged from left to right and top to bottom:
1. **Top Row:**
 - **First Image:** A breathtaking sunset over the moai statues on Easter Island, one of its many UNESCO sites. The statues stand tall on a grassy field, overlooking the ocean as the sun sets, casting a warm glow on both the moai and the cliff in the background.
 - **Second Image:** A serene scene featuring a row of moai statues standing on a grassy hill. These ancient statues, situated on one of the world's most visited sites, are set against a beautiful sky, embodying the island's mystique.
2. **Bottom Row:**
 - **First Image:** A close-up of the moai head of Rapio amidst a rocky hillside. The photo captures the rugged landscape with patches of green grass and shrubs adding a touch of vitality to the scene.
 - **Second Image:** A dramatic view of three moai statues silhouetted against the sky at sunrise. The statues stand along a sea wall, with the sun's rays beginning to pierce the horizon, creating a captivating blend of light and shadow.
This collage showcases the enigmatic beauty and historical significance of Easter Island, emphasizing the awe-inspiring presence of its moai statues set against the island's stunning natural scenery.

Figure 9. **Prompt design and response example for auto layout.**

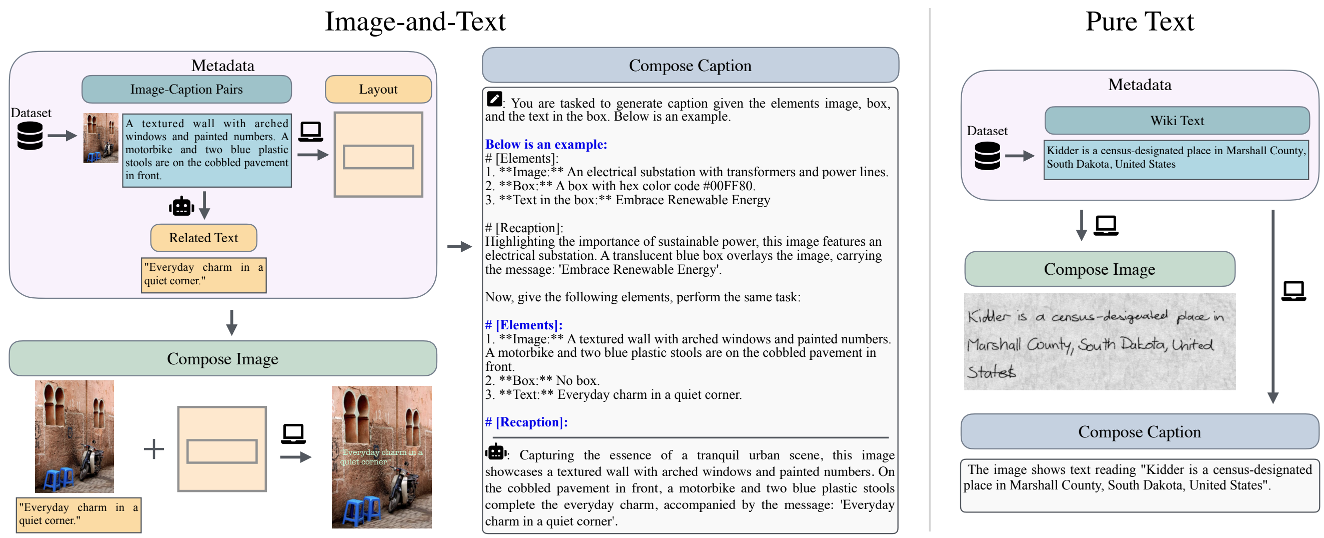


Figure 10. **The Image-Text implementation.**

We demonstrate the designed prompts and the example outputs from LLMs in Figures 8 and 9. We use Llama-3.1-70B [17] for all caption generation.

Caption post-processing. We find that some of the responses from LLMs do not completely follow the given instructions. For example, the response may start with "#Recaption:"or "Here is the generated caption:" before the actual image caption, or contain a follow-up question such as "Let me know if you have further instruction" after the caption. To address this, we perform a manual check on a batch of generated responses and summarize all possible patterns, and implement a post-processing pipeline to replace and delete undesired text automatically.

A.2. Image-Text Implementation

The curation pipeline for the Image-Text CI class is illustrated in Figure 10. This class is designed to assess MLLMs' capabilities in extracting text from images and understanding the relationship between text and visuals. We divide this into two subcategories: image-and-text and pure-text. The image-and-text category tests the model's ability to infer the relationship between text and the image, expecting MLLMs to interpret how text interacts with visual elements. The pure-text category focuses solely on text extraction. We present text in various styles against different backgrounds to strengthen the MLLMs' robustness in text recognition.

A.2.1. Image-and-text Pipeline

Similar to collage, we sample image-caption pair as the background. We only consider random retrieval as we only retrieve one image at a time. We then prompt an LLM to generate relevant text content to the image based on the

: Given an image, you are tasked to inference in what topic or scenario will the image be shown. Generate a topic that may related to the images, and then text that might comes along with the topic. The text may be used for social media purpose.

 # [Caption]:
A helmeted snowboarder gliding down the snowy slope of a mountain, surfing along its side.

Following the instructions below when generating:
1. Generate one topic and one accompanying sentence.
2. The accompanying sentence should not describe the content of the image. Do not include tags in the sentence.
3. Do not rephrase the provided caption.
4. Be imaginary.
5. follow the format below:

topic:
sentence:

:
topic: Winter Adventure Escapes
sentence: Unleash the thrill of the mountains and discover your next snowy getaway.

Figure 11. Prompt for generating related text from given image.

caption. Note that we specifically instruct the LLM not to rephrase the caption. The prompt used for text content generation is shown in Figure 11. We first ask Llama-3.1-70B to infer the topic related to the image, then generate a sentence within the topic.

To enhance visual diversity, we control two primary configurations:

Box layout. We position the text within a bounding box, arranging the box alongside or overlaying the image. The

Augmentation	Applied Effect
Ink Phase	
InkBleed	Applies random noise along text edges them to mimic fuzzy, bleeding ink when blurred.
BleedThrough	Combines ink bleed with Gaussian blur to recreate an effect where ink seeps through the paper.
Paper Phase	
PaperFactory	Replaces the background with a randomly chosen texture, resized or tiled to cover the entire image.
Tessellation	Applies a repeating geometric pattern that interlocks seamlessly, giving a structured, mosaic-like texture.
NoiseTexturize	Adds a random noise pattern in varying scales to create a realistic paper texture.
BrightnessTexturize	Introduces random brightness variations to mimic subtle textural differences in paper.
Combined Phase	
DirtyDrum	Simulates a dirty drum effect by adding vertical and horizontal noise lines across the image.
DirtyRoller	Recreates the streaking effect of old or dirty document rollers in scanners.
SubtleNoise	Adds slight, uneven noise to replicate minor lighting inconsistencies seen in scans of solid colors.
BadPhotoCopy	Adds a grainy, noisy overlay to mimic the quality of a worn-out photocopier.
ShadowCast	Casts shadows on the paper to simulate natural shadows from scanning or photocopying.
ReflectedLight	Draws bright ellipses on the paper to recreate the effect of light reflecting on the surface.

Table 4. **Augraphy augmentations for pure text.** Effects are applied on the image sequentially according to the row order.

box's size, color, and opacity are randomized to increase variety.

Text style. We customize the text's appearance by adjusting its size, color, font, and spacing, ensuring it contrasts well with the background for clear visibility.

A.2.2. Pure Text Pipeline

This pipeline synthesizes both digital and handwritten text images. The text corpus is sourced from Wikipedia, with digital text generated as in the image-and-text pipeline. Additional details for handwritten text and background generation are as follows:

Handwritten text. We generate handwritten text images in SVG format following Graves [20], offering 12 distinct writing styles. Similar to digital text, color, stroke width, line spacing, and alignment are customized to increase diversity.

Background. We choose two types of background image:

1. *Natural image.* We sample images from COCO dataset, with a blurring effect applied to highlight the foreground text.
2. *Synthetic paper image.* We use Augraphy [21] to render realistic document effect, which sequentially modifies the ink style and the background paper style to create an authentic appearance. We summarize the used pipelines in Table 4.

A.3. 📊 Chart Implementation

For chart visualizations, we consider bar charts, line plots, pie charts, and choropleth maps. In this section, we first explain how the bar, line and pie charts and their captions are generated generated as they share similar data sources. Then we illustrate the map visualization and caption design. Finally, we provide a comparison of our curate dataset against previous synthetic chart-text datasets. For all chart visualizations, we use Plotly [23]. And for all caption generations, we use Llama-3.1-405B.

A.3.1. Bar, Line, and Pie Charts

Data source. The tabular data for visualization come from DVQA [25] and UniChart [45]. DVQA provides canonical tabular data suitable for bar and pie chart visualizations, while UniChart contains time-series tabular data for line charts.

Bar chart. The bar chart generation pipeline supports three bar types: single, grouped, and stacked bar charts. Single bar charts visualize one row of data, whereas grouped and stacked bar charts incorporate multiple rows. To enhance variety, each bar type includes the following customizations:

1. *Bar pattern.* Adjustments include bar texture, color, border, width, spacing, and the presence or absence of text on the bars.
2. *Orientation.* Bars can be arranged horizontally or vertically.
3. *Axes.* Customizations cover the range and tick intervals on both x-axis and y-axis.
4. *Annotations and layouts.* Variations include font styles, colors, and layout adjustments for titles, axis labels, and legends.

Line chart. For line charts, we use both single-row and multiple-row time-series data, where each line corresponds to one line in a chart. Many customizations from bar charts apply here, including axes, annotations, and layouts. The primary distinction for line charts is in line pattern customization, such as line style, color, and marker pattern.

Pie chart. Pie charts use single-row data for visualiza-

	FigureQA [26]	DVQA [25]	PlotQA [49]	MapQA [9]	CompCap-Chart
Image type	Scatter/Bar/Line/Pie	Bar	Scatter/Bar/Line	Map	Bar/Line/Pie/Map
Text type	Yes/No QA	Open-ended QA	Open-ended QA	Open-ended QA	Detailed Caption
Text generation	Template-based	Template-based	Template-based	Template-based	LLM-generated

Table 5. **Comparison of existing synthetic chart datasets.**

tion. Customizations include pie appearance adjustments, such as color, size, and display text placement. Text can be displayed inside or outside the pie; when segments are too small for text, pointers are used to indicate the designated region. Other customizations align with those used in bar and line charts.

Prompt and caption design. For generating captions, the input to LLMs includes both data details and visualization parameters. Specifically, it incorporates axes details (type, range, and label) and additional elements like background patterns, titles, and style specifications. We instruct LLMs to create captions that summarize the chart's data, identify trends, and compare groups. Figures 12, 13, 14 illustrate the designed prompts and sample outputs.

A.3.2. Choropleth Maps

Choropleth maps are created for four regions: European countries, global countries, the United States, and Chinese provinces.

Data source. For European and global countries, data is visualized at the country level, with each country assigned a data value. Data is sourced from Eurostat [18] and Gapminder [7], or generated randomly. For Chinese provinces, we use randomly generated data, while for the United States, randomly generated state data from MapQA [9] is used.

Visualization. Depending on the data type, choropleth maps can represent values using either a color bar for numerical data or a discrete color legend for categorical data. Each region is colored based on its value in the legend or color bar. Various visualization customizations include:

1. *Color pattern.* Varying color schemes for regions, titles, and legends.
2. *Projection.* Different projection methods for map rendering.
3. *Value dropout.* Randomly omitting values for certain regions and marking these with a distinct color.
4. *Layouts.* Randomized layout of titles, map entries, and legends.
5. *Region annotation.* Optional display of country/province names or acronyms within the map.

Prompt and caption design. We focus captions on regions that are clearly visible on the map to ensure clarity. Along with listing data values for key regions, we analyze the overall data distribution, such as trends by cardinal direction or differences between coastal and inland areas. We also include additional details like title and legend interpretation.

Figure 15 shows the prompt used to generate captions. When preparing the data table in the prompt, we include each region's data value, color name, and some geographic details:

1. *Location.* Compass direction (e.g., north, southeast).
2. *Type.* Whether the region is coastal or inland.
3. *Area.* The size of the region.

A.3.3. Post-processing

We filter and modify the generated responses from LLMs such that they mostly resembles caption of an image. We observe that the generated response describes the details without first identifying the type of the chart since it is provided in the context of the prompt. However, such information is not granted in real conversation. We rephrase the first sentence such that it always start with identifying the chart type presented in the image before actual captioning. For instance, "The line chart titled "xxxx" visually represents..." is rephrased into "The image shows a line chart titled "xxxx", which visually represents...". We implement it by rule-based matching and replacement. Apart from first sentence rephrasing, we also reuse the processing strategies stated in § A.1.2 to enhance caption quality.

A.3.4. Comparison with Existing Synthetic Datasets

Table 5 compares existing synthetic chart-text datasets [9, 25, 26, 49] with our chart-caption dataset. Unlike previous methods that generate templated question-answer pairs for charts, our pipeline emphasizes detailed captions.

Previous methods revoke a system to learn three abilities: structure understanding, data retrieval, and reasoning, often through carefully designed templates targeting a single ability. By providing precise instructions to the LLM, we enable it to generate captions that naturally integrate all three abilities. This approach not only eliminates the need for rigid templates but also encourage diversity in the generated captions.

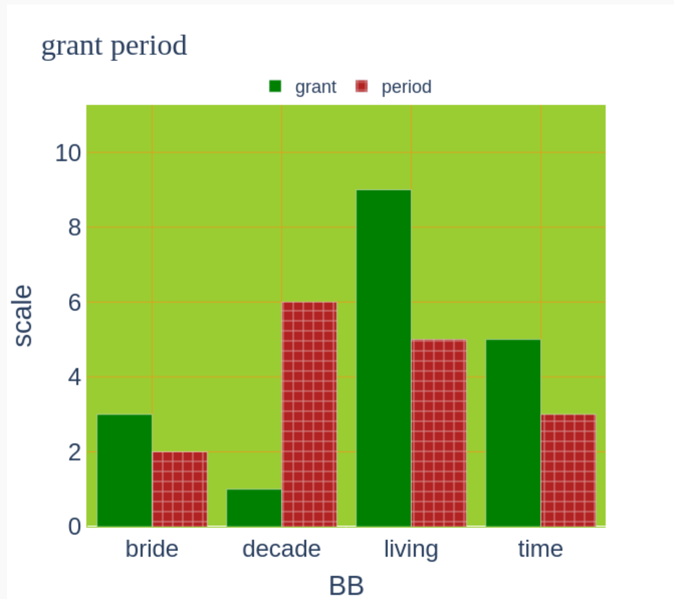
A.4. Diagram Implementation

We employ Mermaid [29], a JavaScript-based diagramming tool, to convert markdown text into diagram images. This tool allows us to transform text into visual representations seamlessly. Additionally, we prompt LLMs to analyze the markdown text, generating captions that not only describe each element in the diagram but also clarify relationships

: Generate a detailed caption describing the contents of a stacked bar chart based on the following metadata. The name of bar is shown in the first level in the Data, and the second level is the category within each bar. The caption should focus on the categories compared, the values of each category, and how they are visually represented. Mention the bar colors, axis labels, title, and the chart orientation, ensuring the description is clear and easy to understand.

 # [Metadata]:

1. **Bars**:
   - Data:
     - Category "bride":
       - Group "grant": 3.00
       - Group "period": 2.00
     - Category "decade":
       - Group "grant": 1.00
       - Group "period": 6.00
     - Category "living":
       - Group "grant": 9.00
       - Group "period": 5.00
     - Category "time":
       - Group "grant": 5.00
       - Group "period": 3.00
   - Color:
     - grant: green
     - period: firebrick
2. **Axes**:
   - x-axis:
     - type: enumeration
     - range: ['bride', 'decade', 'living', 'time']
     - label: "BB"
   - y-axis:
     - type: interval
     - range: [0, 11]
     - label: "scale"
3. **Misc**:
   - chart orientation: vertical
   - background color: yellowgreen
   - title: "grant period"
   - has bar text: False



Following the instructions below when generating

1. If the plot doesn't have bar text, you should give a rough number instead of copying the data value with full precision.
2. Ensure the caption captures all relevant details and trends observed in the chart.

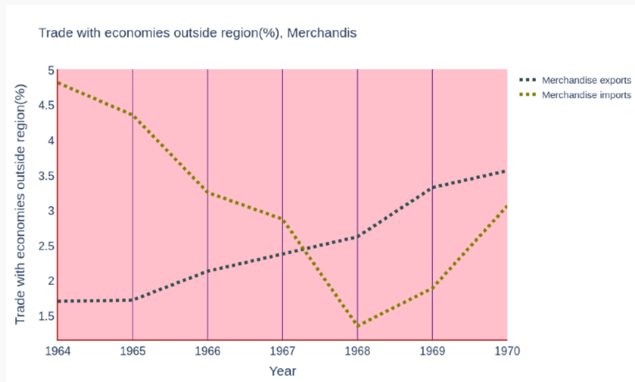
: The grouped bar chart titled "grant period" visually compares the values of two groups, "grant" and "period," across four categories: "bride," "decade," "living," and "time." The chart is oriented vertically and features a yellowgreen background. Each category on the x-axis, labeled "BB," is represented by two bars: one for the "grant" group and one for the "period" group. The y-axis, labeled "scale," ranges from 0 to 11, indicating the values of each bar.
- In the "bride" category, the green "grant" bar reaches approximately 3 units, while the firebrick "period" bar is slightly lower at around 2 units.
- For the "decade" category, the green "grant" bar is much shorter, at about 1 unit, whereas the firebrick "period" bar is significantly taller, reaching around 6 units.
- In the "living" category, the green "grant" bar is the tallest among all categories, standing at roughly 9 units, while the firebrick "period" bar is about 5 units high.
- Lastly, in the "time" category, the green "grant" bar is approximately 5 units tall, and the firebrick "period" bar is slightly shorter, at around 3 units.
This chart effectively highlights the differences in values between the "grant" and "period" groups across the specified categories, with the green and firebrick colors providing a clear visual distinction between the two groups.ata is easy to interpret. Overall, the chart effectively illustrates the contrasting trends in exports and imports over the given period.

Figure 12. **Prompt design and response example for bar chart.**

: Generate a detailed caption describing the contents of a line chart based on the following metadata. The x and ys (there are multiple lines) are shown in the Data. The caption should focus on the lines compared, and how they are visually represented. Mention the title, axis labels, line style, and legend, ensuring the description is clear and easy to understand.

 # [Metadata]:

1. **Lines**:
   - Data:
     - x: [1964, 1965, 1966, 1967, 1968, 1969, 1970]
     - y:
       - Merchandise exports: [1.7, 1.7, 2.1, 2.4, 2.6, 3.3, 3.6]
       - Merchandise imports: [4.8, 4.4, 3.3, 2.9, 1.4, 1.9, 3.1]
2. **Axes**:
   - x-axis:
     - range: [1964, 1970]
     - label: "Year"
   - y-axis:
     - range (approximate): [1, 5]
     - label: "Trade with economies outside region(%)"
3. **Misc**:
   - background color: pink
   - title: "Trade with economies outside region(%), Merchandis”
   - linestyle: dot



Following the instructions below when generating

1. You should give a rough number instead of copying the data value with full precision.
2. Ensure the caption captures all relevant details and trends observed in the chart.

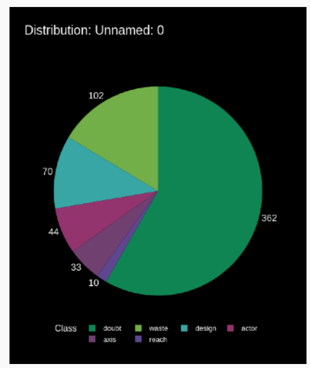
: The line chart titled "Trade with economies outside region(%), Merchandise” visually represents the trends in merchandise exports and imports from 1964 to 1970. The x-axis is labeled "Year" and spans from 1964 to 1970, while the y-axis, labeled "Trade with economies outside region(%)", ranges approximately from 1% to 5%.

Two lines are plotted on the chart, each using a dotted line style for visual distinction. The first line represents "Merchandise exports" and shows a general upward trend over the years. Starting at around 1.7% in 1964, it remains steady until 1965, then gradually increases to approximately 2.1% in 1966, 2.4% in 1967, 2.6% in 1968, 3.3% in 1969, and peaks at about 3.6% in 1970. The second line represents "Merchandise imports" and exhibits a more fluctuating pattern. It begins at roughly 4.8% in 1964, decreases to about 4.4% in 1965, and continues to decline to around 3.3% in 1966 and 2.9% in 1967. The lowest point is observed in 1968 at approximately 1.4%, followed by a slight increase to 1.9% in 1969, and then a more significant rise to about 3.1% in 1970.

The chart's background is colored pink, enhancing the visual appeal and making the dotted lines stand out. A legend is included to clearly differentiate between the "Merchandise exports" and "Merchandise imports" lines, ensuring the data is easy to interpret. Overall, the chart effectively illustrates the contrasting trends in exports and imports over the given period.

Figure 13. **Prompt design and response example for line plot.**

: Generate a detailed caption describing the contents of a pie chart based on the following metadata. The name, color and value of each slice in the pie are shown in “pie data”, and miscellaneous information including title, data type (value or percentage) are also provided. Ensure the description is clear and easy to understand.



 # [Metadata]:
1. **Pie data**:
   - Class "waste" (olivedrab): 102
   - Class "reach" (darkslateblue): 10
   - Class "design" (lightseagreen): 70
   - Class "axis" (dimgray): 33
   - Class "doubt" (seagreen): 362
   - Class "actor" (purple): 44
2. **Misc**:
   - Title: "Distribution: Unnamed: 0"
   - Data type: value

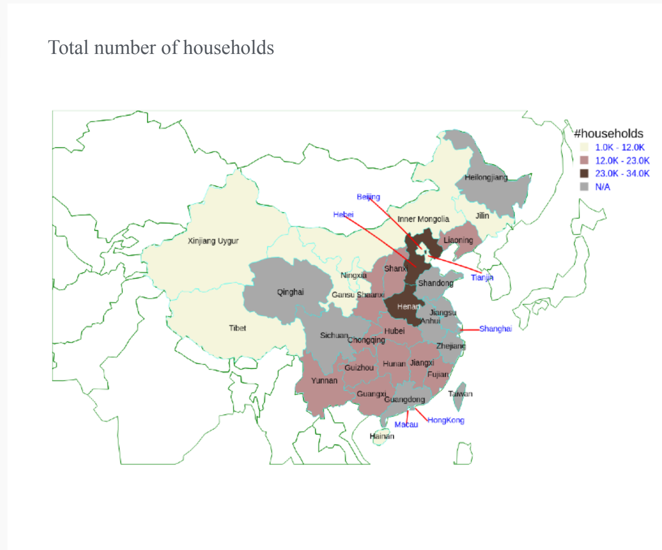
Ensure the caption captures all relevant details observed in the chart.

: The pie chart titled "Distribution: Unnamed: 0” visually represents the distribution of six different classes based on their respective values. Each slice of the pie is color-coded and labeled with its corresponding class and value. The largest segment, colored in seagreen, represents the "doubt" class with a value of 362. Following this, the "waste" class, depicted in olivedrab, has a value of 102. The "design" class, shown in lightseagreen, holds a value of 70. The "actor" class, marked in purple, has a value of 44. The "axis" class, colored in dimgray, has a value of 33. Lastly, the smallest segment, represented in darkslateblue, corresponds to the "reach" class with a value of 10. This chart provides a concise visualization of the distribution of values across different classes.

Figure 14. **Prompt design and response example for pie chart.**

: Generate a detailed caption describing the contents of a choropleth map based on the following metadata. The metadata provides information of the regions in the map and the legend of the plot. The regions' data contains region name, class, color in map, and addition region information. The legend title may contain the unit of the value.

[Metadata]:
1. **Data**:
```md
| Region | Category | Color in map | Location | Geographical type |
|---|---|---|---|---|
| Tibet | 1.0K - 12.0K | Very Light Beige | Southwest | Inland |
| Ningxia | 1.0K - 12.0K | Very Light Beige | Northwest | Inland |
...
| Chongqing | 12.0K - 23.0K | Muted Brown | Central | Inland |
| Guizhou | 12.0K - 23.0K | Muted Brown | Southwest | Inland |
...
| Hebei | 23.0K - 34.0K | Dark Brown | North | Coastal/Inland |
| Henan | 23.0K - 34.0K | Dark Brown | Central | Inland |
| Anhui | N/A | gray | East | Inland |
| Guangdong | N/A | gray | South | Coastal |
| Heilongjiang | N/A | gray | Northeast | Inland |
...
```



2. **Legend**:
   - Type: Categorical legend (4 classes)
   - Class colors:
     - Class '1.0K - 12.0K': Very Light Beige
     - Class '12.0K - 23.0K': Muted Brown
     - Class '23.0K - 34.0K': Dark Brown
     - Class 'N/A': gray
   - Title/Unit: households
3. **Misc**:
   - Scope: China
   - Title: Total number of households

Following the instructions below when generating
1. You should demonstrate key information by inspecting regions' colors based on the class color indicated in the legend.
2. Provide summarization if there is any location/geographical pattern.
3. Ensure the caption captures all relevant details and trends observed in the map plot.

: This choropleth map titled "Total number of households" visually represents the distribution of households across various regions in China. The map uses a categorical legend with four classes to indicate the number of households, with colors ranging from Very Light Beige to Dark Brown, and gray for regions with unavailable data.
1. Regions with the fewest households, ranging from 1.0K to 12.0K, are shaded in Very Light Beige. These regions include Tibet, Ningxia, Hainan, Tianjin, Gansu, Xinjiang Uygur, Beijing, Inner Mongolia, Jilin, and several others. These areas are predominantly located in the inland and less densely populated parts of China, such as the Northwest, North, and Southwest.
2. Regions with a moderate number of households, between 12.0K and 23.0K, are colored in Muted Brown. This category includes Chongqing, Guizhou, Shanxi, Jiangxi, Fujian, Shaanxi, Yunnan, Guangxi, Liaoning, and Hunan. These regions are more centrally located and include both inland and coastal areas, indicating a higher population density compared to Very Light Beige regions.
3. The regions with the highest number of households, ranging from 23.0K to 34.0K, are depicted in Dark Brown. These regions are Hebei and Henan, both located in the northern and central parts of China, respectively. These areas are characterized by higher population densities and significant urbanization.
4. Regions shaded in gray represent areas where data is not available. These include Anhui, Guangdong, Heilongjiang, Jiangsu, Qinghai, Shandong, Shanghai, Sichuan, Zhejiang, Taiwan, Hong Kong, and Macau. These regions are spread across various parts of China, including both coastal and inland areas.
Overall, the map reveals a clear geographical pattern where inland and less urbanized regions tend to have fewer households, while more central and urbanized regions have higher household counts. Coastal regions show a mix of household densities, reflecting their diverse urbanization levels. The use of distinct colors effectively highlights these variations, making it easy to identify regions with different household densities at a glance.

Figure 15. **Prompt design and response example for choropleth map.**

and provide potential insights. Figure 16 demonstrates the implementation of the pipeline.

A.4.1. Diagram Visualization

This section elaborates on the data source of the Mermaid codes, how they are rendered into diagram images, and customizations of the diagram style.

Data source. We acquire Mermaid code through two primary methods:

1. *GitHub Crawling.* We collect text files containing "mermaid" as a keyword from licensed repositories on GitHub.
2. *LLM-Generated Code.* We prompt LLMs to generate Mermaid code for specific diagram types such as class diagrams, ER diagrams, and flowcharts.

We filter the collected codes by running a rendering test, yielding approximately 3K valid diagram codes: 2K from GitHub and 1K generated by LLMs.

Rendering. Mermaid's advantage lies in its automatic optimization of spatial arrangements, ensuring diagrams display well in HTML. By simply declaring the required packages and placing the Mermaid code within the HTML body, the browser renders the diagram seamlessly. In our process, we generate an HTML file for each Mermaid code, open it in Chrome, and capture a screenshot of the rendered diagram. We use Selenium to automate this process of browsing and saving images.

Diagram style customizations. Mermaid also offers styling parameters to customize the theme and visual appearance of rendered diagrams. These parameters can be included in the HTML header and thus separated from the main diagram content. We prompt LLMs to generate 53 styling specifications, creating a candidate set. For each HTML file, we randomly select one styling option from this set to increase visual diversity. In cases where a styling option is incompatible with a specific diagram type, the default styling is automatically applied. We retain all successfully rendered HTML files.

A.4.2. Prompt and Caption Design

Understanding diagrams is more challenging because they contain numerous objects and emphasize the relationships among them. Specifically, object relationships in diagrams are often more complex compared to other CI types, as they frequently use arrows or nesting to indicate directions or hierarchies. Therefore, our designed captions focus on extracting elements and relationships, placing less emphasis on the diagram's appearance details.

To generate captions that provide a detailed breakdown of the diagrams, we prompt Llama-3.1-405B to read the diagram code and translate it into natural language. To ensure the generated captions are as invariant as possible to the diagram's appearance, we include only the Mermaid code in the prompt, excluding any styling-related codes. We find that minimal instruction is sufficient for the LLM to accurately analyze the code.

In post-processing the generated captions, we first modify the opening sentence to include an identification of the diagram type, similar to our approach with chart captions. Some Mermaid code retrieved from GitHub contains style arguments like hex color codes or stroke widths for text boxes. Since the LLM interprets code, these styling details sometimes appear in their responses. For example, a box labeled "Customer A" might be described in the caption as "Customer A (#a1320f, stroke width 2)". This pattern also occurs when the diagram code assigns a shorter variable name (e.g., "A") to an object like "Customer A". To enhance caption quality, we refine the LLM-generated responses by removing parentheses that contain styling arguments or variable names.

A.5. </> Code Implementation

We use Carbon [8] to create code screenshots with a customized syntax theme and font style. Specifically, it provides 29 themes and 14 font families, as well as other style parameters such as font color, presence or absence of line numbers, line space, window size, etc. We randomize those options to enhance the diversity of the generated screenshots. Next, we demonstrate what code data are used for screenshot rendering and the caption design.

Data source. While numerous code generation datasets provide a variety of sources for code snippets, we find these datasets often contain overly complex, lengthy examples with extensive comments. This complexity results in code snippets that are too long to fit within a suitably sized screenshot. Additionally, the detailed comments provide explicit explanations of functionality, whereas we aim for MLLMs to learn inference directly from the code itself, without relying on predefined explanations. Therefore, we seek to use LLMs to generate simpler code snippets in different languages.

We consider 9 programming languages: C, C++, Ruby, R, Python, Java, JavaScript, CSS, and SQL. For each language, we ask Llama-3.1-70B to generate 200 functions/topics that can be implemented by the selected language. We then prompt the LLM to generate the code given the topic and the language. We focus on relatively simple functions such as mathematical implementations, textbook algorithms and use case of data structures.

Caption design. Our goal is for MLLMs to first extract the code text from the screenshot, interpret it, and then provide an explanation of its functionality The code explanation can be obtained by LLMs. Building on the previous code generation step, we further have the Llama-3.1-70B to generate the corresponding explanation for its generated code snippet. The code snippet and its corresponding explanation are then concatenated together to compose the cap-

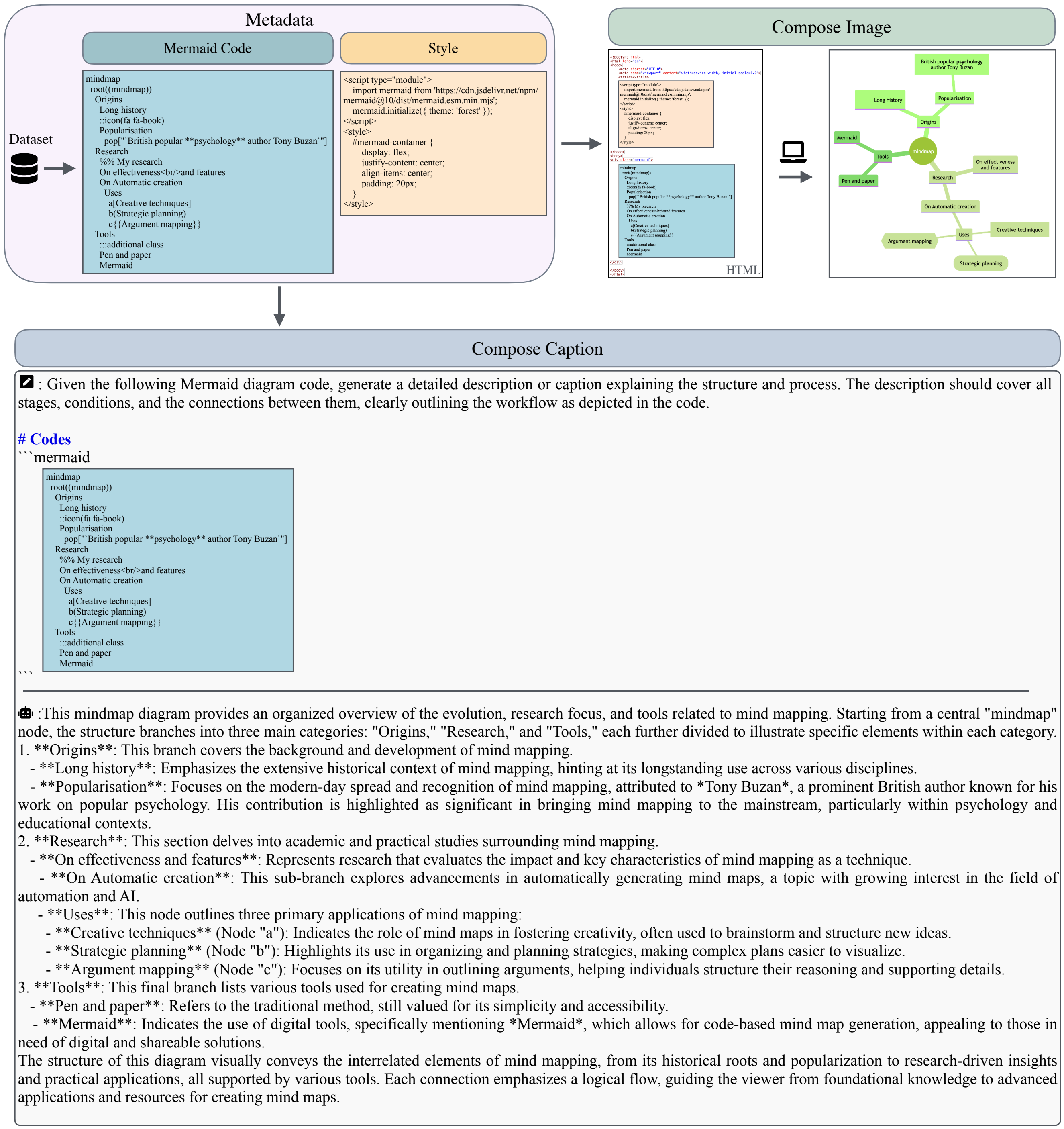


Figure 16. **The Diagram implementation.**

tion. Figure 17 illustrates an example code screenshot and its composed caption.

A.6. Table Implementation

We use Matplotlib [22] to generate table images from tabular data. Similar to our approach for chart visualization, the tabular data for these table images is sourced from the UniChart dataset. Below, we outline the different types of customizations applied to the table images, followed by a discussion of the caption design.

Visualization. The following table customizations were applied:

```java
import javax.swing.*;
import java.awt.*;

public class JTextFieldExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("JTextField Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());

        JTextField textField = new JTextField(20);
        frame.add(textField);

        JButton button = new JButton("Click me");
        frame.add(button);

        frame.setVisible(true);
    }
}
```

The image shows Java code. Here's the code:

````
```
import javax.swing.*;
import java.awt.*;

public class JTextFieldExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("JTextField Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());

        JTextField textField = new JTextField(20);
        frame.add(textField);
        JButton button = new JButton("Click me");
        frame.add(button);

        frame.setVisible(true);
    }
}
```
````

This Java code generates a simple graphical user interface (GUI) using the Swing library. The GUI consists of a window with a text field and a button. The text field allows the user to input text, and the button can be clicked to perform an action (although in this example, no action is defined).

The code creates a `JFrame` object, which is the main window of the GUI, and sets its title, size, and default close operation. It then creates a `JTextField` object, which is the text field, and adds it to the frame. A `JButton` object is also created and added to the frame.

Finally, the `setVisible` method is called to make the frame visible on the screen.

Figure 17. **Caption design for code screenshot.**

1. *Table size*: Varying the table's width and height.
2. *Cell style*: Adjusting the width, height, and color of individual rows or columns.
3. *Border style*: Modifying the style, thickness, and color of table borders.
4. *Font style*: Varying font color, type, and size.
5. *Alignment*: Applying different alignments (left, center, right) to individual columns.

We ensure that the data remains clearly visible despite these customizations. Specifically, we maintain strong contrast

Years	High income: nonOECD Number of refugees	Small states Number of refugees	Greece Number of refugees	Mali Number of refugees
1994	331282	128622	94	172905
1995	456028	125844	222	77219
1996	489315	128000	250	55198
1997	552718	119903	178	18015
1998	514825	121880	174	3702

The image shows a table with data on the number of refugees in various regions and countries from 1994 to 1998. The columns represent the number of refugees in high-income non-OECD countries, small states, Greece, and Mali. The rows represent the years from 1994 to 1998. The refugee counts are shown as numerical values, indicating the number of refugees in each region or country for each year.

Here's the data represented in the table:

Years	High income: nonOECD Number of refugees	Small states Number of refugees	Greece Number of refugees	Mali Number of refugees
1994	331282	128622	94	172905
1995	456028	125844	222	77219
1996	489315	128000	250	55198
1997	552718	119903	178	18015
1998	514825	121880	174	3702

Figure 18. **Caption design for table image.**

between foreground and background colors, and adjust font and table sizes appropriately for readability.

Caption design. The caption design follows similar principles to those used in code image captions, focusing on extracting the table from the image and analyzing the presented data. To obtain the analysis text, we convert the table into markdown format and prompt the Llama-3.1-70B to generate a response that provides a summary and detailed breakdowns. The final caption for the table image is composed by combining the markdown table with the analysis text. Figure 18 shows an example table image and its composed caption.

B. Experiment Details

B.1. Demonstrative Experiment

Collecting VQA data for NIs and CIs. We randomly select 1K samples from the VQAv2 [19] validation set to create VQA pairs for NI. However, to our knowledge, no existing benchmark comprehensively covers the wide variety of CI types. To address this, we curate a toy benchmark comprising 1K VQA pairs for CIs by sampling from datasets ChartQA [44], DocVQA [46], InfoVQA [47], MapQA [9], MME [63], OCRBench [40], MMVet [65], and MMBench [41]. This curated benchmark includes various CI types collages, charts, tables, code, documents, diagrams, infographics, etc, to offer a broad evaluation of

MLLMs' CI comprehension abilities. The composition of this curated benchmark is summarized in Table 6

Benchmark	Covered CI Types	#Samples
ChartQA [44]	Chart	200
DocVQA [46]	Document	100
InfoVQA [47]	Infographic	200
MapQA [9]	Chart (Map)	50
MME [63]	Collage/Code	60
OCRBench [40]	Document/Infographic	100
MMVet [65]	Collage/Chart/Diagram/Table	40
MMBench [41]	Collage/Chart/Diagram/Table	250

Table 6. **Dataset sources for the curated CI benchmark.**

Inference. We evaluate the MLLMs' caption and VQA accuracy using the provided QA pairs. In the caption-conditioned QA pipeline, the captions generated by MLLMs may sometimes lack sufficient context for an LLM to answer the visual question accurately. We instruct the LLM to respond with "I don't know" (IDK) when it identifies insufficient information in the caption. We count the IDK percentages for NIs and CIs. Table 7 shows a higher IDK rate for CIs than for NIs, indicating that CI captions tend to capture less information or information of lesser relevance.

	CI	NI
IDK percentage	58%	12%

Table 7. **IDK percentage.** Percentage of an LLM answering IDK given the visual question and caption context.

Evaluation. We only evaluate MLLMs' caption accuracy on VQA pairs that are answerable by LLMs. We discard original benchmark evaluation guidelines, as each has its own rubric, and instead adopt MMVet's evaluation approach by prompting GPT-4 to score the predicted answers. We demonstrate the prompt used for predicting answer correctness in Figure 19. Figure 2b reports the average prediction score as measures of caption accuracy and VQA accuracy for both CI and NI. For agreement percentage in Figure 2c, we evaluated agreement by checking the correctness scores for each VQA pair; we considered VQA and caption predictions to be in agreement if the absolute difference between their scores was less than 0.2.

B.2. MLLMs Training

Pre-training. We do not pre-train the MLLMs in our experiment. Instead, we directly use the public pre-trained MLLMs checkpoint to initialize the weights and focus on the SFT stage. For xGen-MM, we use the v1.5 checkpoint to initialize the model weights.

: Compare the ground truth and prediction from AI models, to give a correctness score for the prediction. <AND> in the ground truth means it is totally right only when all elements in the ground truth are present in the prediction, and <OR> means it is totally right when any one element in the ground truth is present in the prediction. The correctness score is 0.0 (totally wrong), 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, or 1.0 (totally right). Just complete the last space of the correctness score.

Question	Ground truth	Prediction	Correctness
What is x in the equation?	-1 <AND> -5	x = 3	0.0
What is x in the equation?	-1 <AND> -5	x = -5	0.5
What is x in the equation?	-1 <AND> -5	x = -5 or 5	0.5
What is x in the equation?	-1 <AND> -5	x = -1 or x = -5	1.0
What is the color of the car?	yellow <OR> beige	beige	1.0
What is the largest value in the table?	**401**	**401**	**<FILL>**

Prediction the correctness score for <FILL>

: 1.0

Figure 19. **Prompt for evaluating of MLLMs' predictions.**

SFT data recipe for xGen-MM-inst. Since the dataset used to instruction fine-tune xGen-MM is not released, we curate a similar SFT dataset according to the data mixture mentioned in their paper [60]. Specifically, we include 781K image-text instruction data from various domains [2, 10, 25, 27, 30, 35–37, 42, 44, 46, 50, 55, 61], and 211K pure text instruction following data [14, 51, 70]. We retrain xGen-MM-inst-4B using our own curated dataset for a fair comparison.

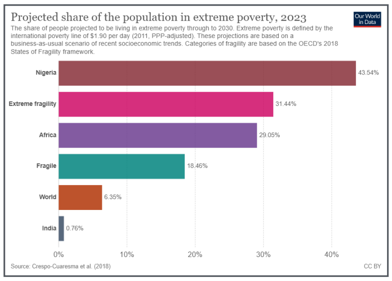
	CompCap-4B	CompCap-7B	CompCap-13B
#vision tokens	128	2880	2880
vision encoder	SigLip [66]	CLIP	CLIP
image aspect ratio	anyres	anyres	anyres
batch size	64	128	128
lr	2e-5	2e-5	2e-5
lr schedule	cosine	cosine	cosine
weight decay	0	0	0
optimizer	AdamW	AdamW	AdamW
#epochs	1	1	1

Table 8. **Hyperparameters for training CompCap series**

Hyperparameters. We show the training hyperparameters for the CompCap series in Table 8. The reproduction of xGen-MM-inst. follows the same hyperparameters as in CompCap-4B. We use 8 Nvidia A100 GPUs to train the 4B MLLMs and 32 Nvidia A100 GPUs for the 7B and 13B MLLMs.

B.3. ChartQA Image Captioning

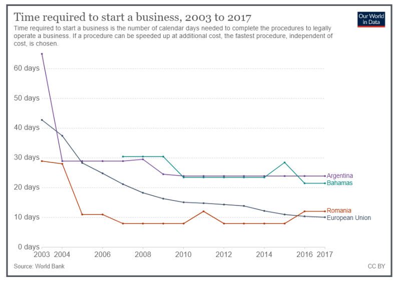
We employ an superior MLLM to generate captions for chart images in the ChartQA training set, using the prompt illustrated in Figure 21. This process produces a total of



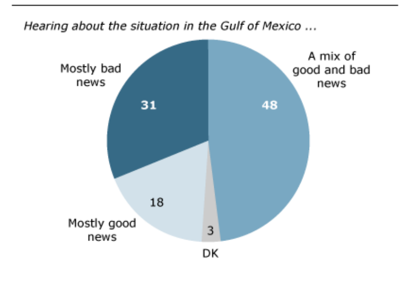
The image is a bar chart titled "Projected share of the population in extreme poverty, 2023." This chart predicts the percentage of people living in extreme poverty, defined as living on less than $1.90 per day (2011, Purchasing Power Parity-adjusted), through 2030. The data shown suggests projections based on a business-as-usual scenario of recent socioeconomic trends. The categories of fragility mentioned are based on the OECD's 2018 States of Fragility framework.
Five entities are represented with different colored bars indicating their respective percentages:

- **Nigeria** has the highest projected share of extreme poverty at **43.54%**, shown with a dark red bar.
- **Africa** (excluding Nigeria) is represented by a dark purple bar, indicating a projected share of **31.44%**.
- Regions with **Extreme fragility** are shown with a pink bar and have a projected share of **29.05%**.
- Those classified as **Fragile** are represented with a teal bar, with **18.46%**.
- The **World** average is shown by a brown bar at **6.35%**.
- **India** has the lowest share among the listed entities, with a dark green bar showing only **0.76%**.

At the bottom, the source of the data is cited as "Crespo-Cuaresma et al. (2018)." On the right side of the image is the logo of Our World in Data with the text "in Data" highlighted in magenta, suggesting this organization provided the visualized data. The bottom right includes a Creative Commons license notation "CC BY," indicating that the chart is licensed for use with attribution.



The image is a line graph titled "Time required to start a business, 2003 to 2017." It shows the number of calendar days needed to complete the procedures to legally operate a business in Argentina, the Bahamas, Romania, and the European Union over the given time period.
The vertical axis represents the number of days, ranging from 0 to 60, and the horizontal axis represents years, from 2003 to 2017. Each country/region is represented by a line of a different color: Argentina (purple), Bahamas (turquoise), Romania (orange), and the European Union (black).
The trend lines indicate a general decrease in the time required to start a business over the years. For example, Argentina's line starts above 50 days in 2003 and ends around 10 days by 2017. The Bahamas shows small fluctuations but stays around 20 days. Romania's line decreases from just below 40 days to approximately 10 days by 2017. The European Union's line shows the least amount of days required throughout the period, starting around 30 days and dropping to slightly above 10 days.
Below the graph, there is a note explaining that if a procedure can be sped up at additional cost, the fastest procedure, independent of cost, is chosen.
In the top right corner, there is a logo with the text "Our World in Data."
In the bottom right corner, there is an attribution to the World Bank as the data source and a "CC BY" icon indicating the Creative Commons license under which the graph is released.



The image displays a pie chart with four segments, representing a survey about public perception concerning the news from the Gulf of Mexico. Each segment is labeled with both a descriptive term and a numeric value indicating the number of responses. The largest segment, colored in a darker blue, represents the majority viewpoint, with "48" indicating that these respondents primarily hear "A mix of good and bad news." The next largest segment, in a lighter blue, shows that "31" respondents hear "Mostly bad news." A smaller grey segment represents "18" respondents who hear "Mostly good news." The smallest segment, in white, indicates "3" respondents are "DK," which typically stands for "Don't Know." Overall, the chart suggests a varied perception of the situation in the Gulf of Mexico, with most people hearing a mixture of good and bad news.

Figure 20. **Examples of ChartQA captions**

18,317 chart-caption pairs. As there are multiple instruction data corresponding to one chart image, we replace data in an image-level in the caption-instruction ablation study. Figure 20 shows some examples of chart caption.

Create detailed captions describing the contents of the given image. Pay more attention to the text or image component and the information they convey.

Figure 21. **Prompt used for captioning ChartQA images.**

C. Additional Results

C.1. Extended Comparison

We provide the full comparison result against SoTA MLLMs in Table 9. We additionally include the results from MLLMs such as GPT-4o [52], Qwen-VL-Max [59], and InternVL-76B [12].

C.2. Full result on the ablation study of CI category

Table 10 provides the scores of all benchmark for the CI component ablation study.

C.3. Ablation on training data sampler

During the SFT stage, MLLMs are typically trained to follow instructions and enhance conversational capabilities. Unlike the usual instruction data, which is often in QA format, CompCap-118K primarily emphasizes caption data, which aims at facilitating vision-language alignment. In this experiment, we explore various strategies for combining our caption data with instructional data during training. Specifically, we set a proportion of caption data from CompCap-118K in each training batch, and adjust this ratio at every training step.

We hypothesize that increasing caption data early in training will strengthen alignment, while focusing more on instructional data in later stages will maintain the model's instruction-following proficiency. To explore this, we experiment with four training data samplers:

1. *Truncated.* This sampler first samples from CompCap-118K until all data is used, then shifts to the original downsampled SFT dataset.
2. *Cosine*. For the t-th training step, this sampler returns a batch where $\alpha\text{cosine}(t/T)$ percent is drawn from CompCap-118K, with T representing the total training steps and α adjusted so all CompCap-118K data is covered by training's end.
3. *Linear.* Similar to the Cosine sampler, but the schedule

Model	PT/SFT #Data	SEEDBench ◇	TextVQA ◇	MMBench ◇◆	MME (norm) ◇◆	LLaVABench ◇◆	MathVista ◆◇	OCRBench ◆◇	ChartQA ◆	DocVQA ◆	InfoVQA ◆	WebSRC ◆	Avg.
SoTA MLLMs													
GPT-4o [52]	UNK./UNK.	77.1	-	81.8	83.2	102.0	63.8	73.6	85.7	92.8	-	-	-
Qwen-VL-Max [59]	UNK./UNK.	77.9	85.5	-	88.7	74.9	70.5	85.5	88.3	96.5	84.5	-	-
InternVL-76B [12]	UNK./UNK.	77.6	84.4	-	86.3	96.3	65.6	84.2	88.4	94.1	82.0	-	-
3B - 4B MLLMs													
MM1-3B [48]	2.9B/1.45M	68.8	71.9	67.8	62.9	72.1	32.0	-	-	-	-	-	-
VILA-1.5-3B [34]	32.8M/5.9M	68.0	55.6	62.4	58.2	65.5	30.6	43.7	52.9	-	-	-	-
Phi-3-vision [1]	5B/>8.3M	70.9	63.6	74.2	55.2	**82.2**	**45.1**	**63.7**	**81.8**	**84.3**	**50.0**	**65.2**	**66.9**
xGen-MM-inst.-4B [60]	>25M/UNK.	**71.8**	**72.0**	76.0	64.1	75.7	39.5	54.8	59.5	61.1	31.3	55.8	60.2
xGen-MM-inst.-4B* [60]	>25M/1M	71.3	67.7	75.5	64.0	78.2	32.6	51.6	54.8	55.2	27.6	50.6	57.2
CompCap-4B	>25M/1M	71.6	67.9	**76.2**	**64.7**	81.0	35.0	52.7	57.4	58.1	27.9	55.8	58.9
7B - 8B MLLMs													
VILA-1.5-8B	32.8M/5.9M	65.0	60.2	68.6	60.7	71.7	37.3	43.8	50.9	-	-	-	-
ShareGPT4V-7B [10]	1.2M/665K	69.3	58.3	68.8	**68.4**	66.9	26.5	37.1	21.3	14.4	14.7	36.4	43.8
Qwen-VL-chat-7B [59]	UNK./UNK.	64.8	60.7	60.6	66.4	67.7	34.9	48.8	49.8	62.6	29.7	53.6	54.5
Cambrian-8B [57]	1.2M/7M	**73.3**	**72.6**	**75.9**	64.4	71.0	**47.0**	**61.4**	**72.6**	**77.8**	40.1	68.9	**65.9**
LLaVA-NeXT-Vicuna-7B [39]	558K/779K	71.2	65.2	67.6	66.3	72.4	39.6	55.1	63.5	76.5	39.2	70.4	62.5
CompCap-7B	558K/779K	70.5	65.6	68.9	67.5	**75.5**	41.7	58.5	68.9	77.6	**40.8**	**73.7**	64.5
13B MLLMs													
VILA-1.5-13B [34]	32.8M/5.9M	72.7	61.2	74.3	61.4	73.4	42.5	46.0	74.6	-	-	-	-
ShareGPT4V-13B [10]	1.2M/665K	70.6	52.7	69.0	66.2	69.1	29.3	39.8	24.6	14.5	17.2	39.4	44.8
OmChat-v2.0-13B [69]	>6.5B/20M	**75.2**	**79.8**	**82.1**	**76.1**	66.1	**57.1**	**72.8**	79.9	**88.7**	**58.8**	**88.2**	**75.0**
Cambrian-13B [57]	1.2M/7M	73.2	72.8	75.7	66.8	76.1	47.4	61.0	73.8	76.8	44.6	70.7	67.2
LLaVA-NeXT-Vicuna-13B [39]	558K/779K	71.9	67.6	68.9	68.8	77.1	42.4	57.7	68.5	79.9	43.8	75.3	65.6
CompCap-13B	558K/779K	72.2	67.8	70.8	71.4	**83.4**	45.0	61.4	73.9	81.1	47.0	79.3	68.5

Table 9. **Full comparison on MLLM benchmarks.**

Component	SEEDBench ◇	TextVQA ◇	MMBench ◇◆	MME (norm) ◇◆	LLaVABench ◇◆	MathVista ◆◇	OCRBench ◆◇	ChartQA ◆	DocVQA ◆	InfoVQA ◆	WebSRC ◆	Avg.
Baseline	71.9	67.7	68.9	68.8	77.1	42.4	57.7	68.5	79.9	43.8	75.3	65.6
+ Collage	72.1	67.3	69.9	69.1	78.4	43.2	58.8	70.9	80.4	45.3	75.5	66.4
+ Code	72.3	67.8	70.2	69.3	76.4	43.6	58.7	71.1	80.6	46.1	76.1	66.6
+ Table	72.3	67.8	70.3	69.4	78.7	43.6	59.1	72.0	80.5	46.1	76.6	67.0
+ Diagram	72.2	67.6	70.7	69.9	80.4	43.6	58.3	72.9	81.2	46.9	77.6	67.4
+ Chart	72.4	67.5	70.5	70.0	84.0	46.8	58.6	73.1	81.1	46.1	77.6	68.0
+ Image-Text	72.2	67.8	70.8	71.4	83.4	45.0	61.4	73.9	81.1	47.0	79.3	68.5

Table 10. **Full benchmark result of the ablation study on each CI category.**

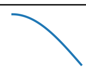
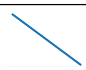
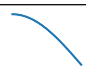
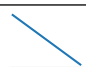
Training schedule	SEEDBench ◇	TextVQA ◇	MMBench ◇◆	MME (norm) ◇◆	LLaVABench ◇◆	MathVista ◆◇	OCRBench ◆◇	ChartQA ◆	DocVQA ◆	InfoVQA ◆	WebSRC ◆	Avg.
CompCap-7B												
 (truncated)	**70.8**	65.4	67.0	65.5	73.2	40.4	54.4	67.0	**78.0**	39.2	72.1	63.0
 (cosine)	**70.8**	65.3	67.9	65.8	71.1	40.2	55.2	67.4	77.3	**40.8**	73.0	63.2
 (linear)	70.6	**65.8**	67.6	67.4	73.9	40.8	54.7	67.4	77.1	39.9	71.8	63.4
 (uniform)	70.5	65.6	**68.9**	**67.5**	**75.5**	**41.7**	**58.5**	**68.9**	77.6	**40.8**	**73.7**	**64.5**
CompCap-13B												
 (truncated)	72.2	**67.8**	70.2	69.8	78.7	44.6	58.9	73.2	**82.0**	45.5	76.6	67.2
 (cosine)	72.1	67.4	69.9	70.6	76.6	44.8	60.4	73.1	80.7	45.8	78.2	67.2
 (linear)	**72.5**	67.5	**71.2**	69.3	79.6	**45.3**	58.6	72.3	80.8	45.3	78.3	67.3
 (uniform)	72.2	**67.8**	70.8	**71.4**	**83.4**	45.0	**61.4**	**73.9**	81.1	**47.0**	**79.3**	**68.5**

Table 11. **Ablation study on train data samplers.**

changes linearly to $1 - t/T$

4. *Uniform.* This sampler uniformly mixes data from CompCap-118K and the original downsampled SFT dataset in each batch.

We evaluate the impact of these samplers using CompCap-7B/13B. We train them on the SFT dataset with each sampler and present the results in Table 11. We find that the uniform sampler consistently outperforms others for both 7B and 13B MLLMs. Unless specified otherwise, all models in our experiments use the uniform sampler for training.

C.4. Diversity Analysis

We demonstrate the diversity of the caption for each CI category in Figure 22. The inner circle of the plot displays the root verbs of the captions, while the outer circle represents the corresponding direct nouns. We display the top 20 root verbs for each CI class, along with the top 5 nouns associated with each root verb.

C.5. Examples of Synthesized CI-caption Pairs

We show examples of the curated image-caption pairs for each CI type in Figures 23, 24, 25, 26, 27, 28.

D. Qualitative Examples

In this section, we show more qualitative captioning results (Figures 29, 30, 31, 32) of CompCap-13B, in comparison with LLaVA-NeXT-Vicuna-13B.

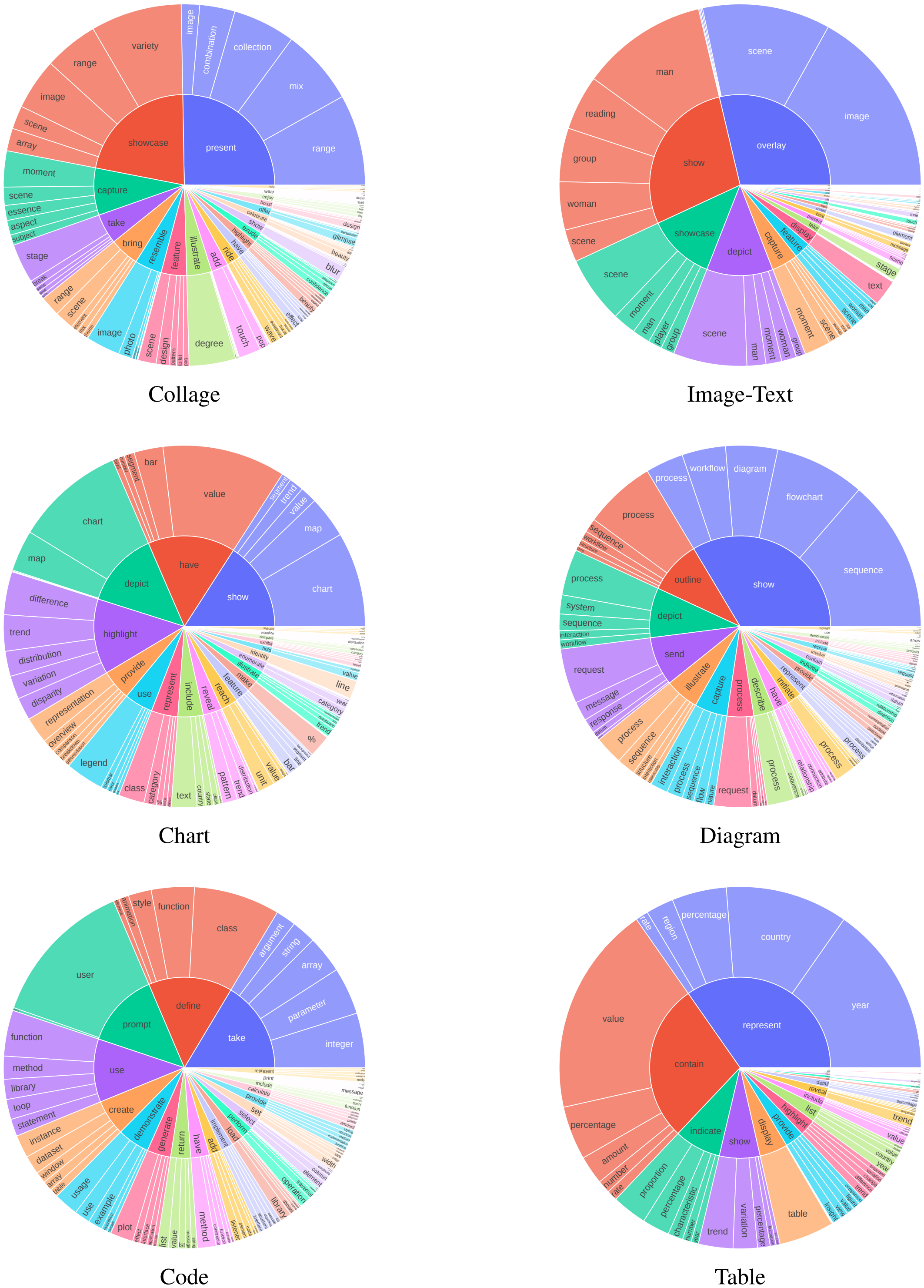


Figure 22. **Diversity analysis of captions for different CI types.**

The image is a collage of four vibrant still-life compositions, showcasing an assortment of fresh apples and oranges in various arrangements. Here are the descriptions of each part of the collage, from left to right:
1. **Left Column:**
- Fresh fruit abounds in this setup, with four groups of stacked apples and oranges displayed on a stone bench, accompanied by a nearby table laden with more fruit and a counter piled high with additional offerings.
- A selection of apples and oranges sit together in harmony on a wooden counter, with two apples featuring colorful stickers nestled beside a rounded pile of oranges.
- A close-up view that reveals a vibrant pyramid of bright green apples, with oranges and other fruits softly blurred in the background, creating a warm and inviting atmosphere.
2. **Right Column:**
- A crisp white ceramic bowl overflows with a delightful arrangement of apples and oranges, set against a simple yet elegant table setting.
The collage celebrates the beauty of fresh fruit, presenting a variety of arrangements that highlight the vibrant colors and textures of apples and oranges. Each image invites the viewer to appreciate the simple joys of nature's bounty.

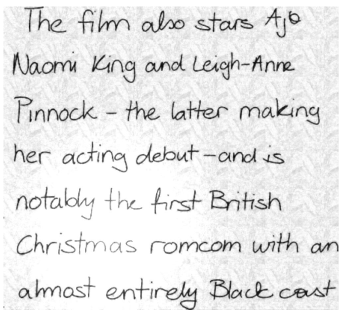
The image is a collage of five serene boat scenes, showcasing various modes of water transportation and leisure activities. Here are the descriptions of each part of the collage, from top to bottom:
1. **Top Row:**
A majestic houseboat transports a smaller motorboat on its back, gliding effortlessly across the tranquil lake.
Two boats are docked in a bustling harbor, with one vessel preparing to set off and another awaiting its turn, while a nearby channel is filled with sailboats drifting gently on the water.
A motorboat, driven by a man wearing a baseball cap, cuts through the calm waters of a lake, leaving a trail of ripples behind.
2. **Bottom Row:**
A forlorn dog sits inside a white boat, gazing longingly out the back as it floats peacefully on the water.
A large white boat, loaded with bicycles at the bow, ferries a group of passengers across the marina, surrounded by other boats enjoying the sunny day.
The collage presents a serene and idyllic portrayal of life on the water, featuring various boats and watercraft, from houseboats to sailboats, and highlighting the joy of leisurely activities on the lake and marina."

The image is a collage of three diverse scenes, showcasing different aspects of daily life and architecture. Here are the descriptions of each part of the collage, from left to right:
- **First Image:** A helpful blue street sign stands near buildings, providing clear directions to visitors and guiding them to nearby attractions and various venues, while a prominent multi-direction parking sign is visible in front of a large building.
- **Second Image:** The stunning red clock tower of Alldredge church rises majestically into the afternoon sky, adorned with a clock on its side, showcasing its beauty as it stands tall and proud.
- **Third Image:** A savory and satisfying meal is laid out on the table, featuring a red bowl of shrimp soup that resembles cauliflower, accompanied by crispy tortilla chips and a refreshing glass of Coke.
The collage presents a unique mix of urban guidance, architectural beauty, and everyday delight, capturing the essence of daily life and its various facets."

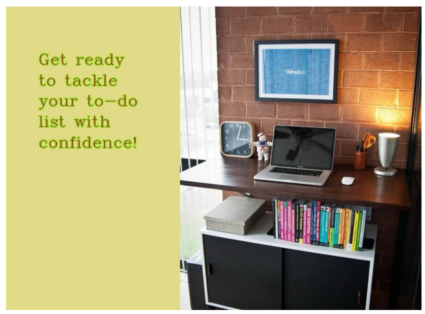
The image is a collage consisting of three distinct sections arranged in two columns:
1. **Left Column**:
- **Top**: A delicious meal is presented on a white plate, consisting of a steak accompanied by a variety of green vegetables, including broccoli, and a side of french fries.
- **Bottom**: A hearty serving of meat, potatoes, and mixed vegetables is laid out on a white plate, featuring a slice of green vegetable and a substantial portion of cauliflower.
2. **Right Column**: A delectable meal is elegantly presented on a white plate, featuring chicken, potatoes, and vegetables, perfectly placed on a table beside a glass of fine white wine.
This collage showcases an assortment of savory dishes, highlighting the versatility of ingredients and presentation styles, from casual comfort food to sophisticated fine dining.

The image is a heartwarming collage of six tender moments showcasing the loving bonds between horses, highlighting their gentle interactions, playful behaviors, and nurturing instincts. Here are the descriptions of each part of the collage, from left to right:
Left Column:
- A sweet display of camaraderie unfolds as a gentle black horse stands alongside a curious baby brown horse, sharing a tender moment of interaction while they graze together.
- A protective large brown horse keeps a watchful eye over its resting foal in a lush green meadow, as another brown horse with a distinctive white stripe on its nose looks on.
- A loving mother horse stands by her baby's side, gently nuzzling and grooming her little one in a serene grassy field.
Right Column:
- Two horses share a playful and intimate moment, with a mischievous brown horse affectionately biting the head of a black horse as they bend down to graze, giving the illusion of a tender hug.
- In a serene and idyllic scene, a small deer tenderly grooms a black horse as it munches on the lush grass, while various horses interact with each other in a heartwarming display of affection, with a young foal showering an older horse with gentle licks, and a couple of horses standing peacefully in a nearby pen.
This collage beautifully captures the strong bonds of friendship, love, and nurturing instincts that exist among horses, showcasing their playful, gentle, and caring nature in various heartwarming moments."

Figure 23. **Image and caption examples of collage in CompCap-118K**

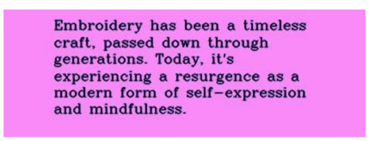


The image shows text reading "The film also stars Aja Naomi King and Leigh-Anne Pinnock - the latter making her acting debut - and is notably the first British Christmas romcom with an almost entirely Black cast".

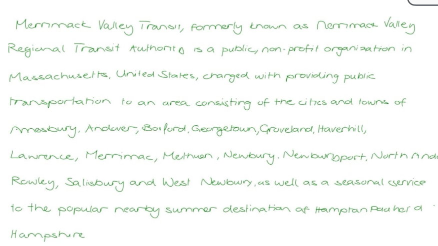


The image shows a laptop placed on a desk in a room, accompanied by various accessories, books arranged on a shelf beneath it, and a lamp illuminating the workspace, all set against a brick wall with a storage bin nearby. On the left side of the image, there is text that reads, "Get ready to tackle your to-do list with confidence!"

This image showcases a beautiful arrangement of red and pink roses in a glass vase, sitting on a table with flowers and water. Overlaid on the image is a semi-transparent mint green box featuring the text: 'Blossoming into your dream space'.

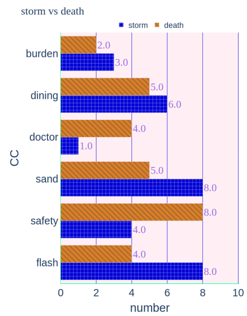


The image contains a block of text on a solid pastel pink background. The text reads:
"Embroidery has been a timeless craft, passed down through generations. Today, it's experiencing a resurgence as a modern form of self-expression and mindfulness."

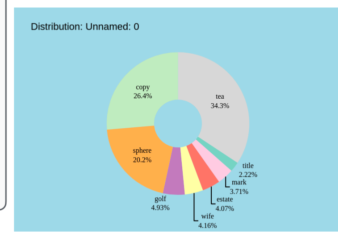


The image shows text reading "Merrimack Valley Transit, formerly known as Merrimack Valley Regional Transit Authority is a public, non-profit organization in Massachusetts, United States, charged with providing public transportation to an area consisting of the cities and towns of Amesbury, Andover, Boxford, Georgetown, Groveland, Haverhill, Lawrence, Merrimac, Methuen, Newbury, Newburyport, North Andover, Rowley, Salisbury and West Newbury, as well as a seasonal service to the popular nearby summer destination of Hampton Beach, New Hampshire".

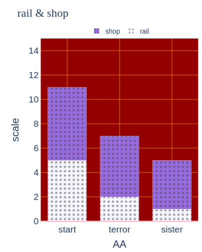
Figure 24. **Image and caption examples of image-text in CompCap-118K**



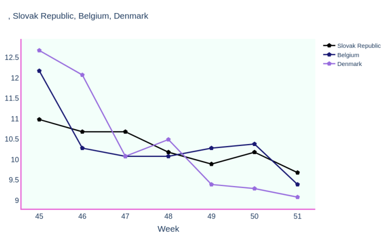
This image shows a grouped bar chart titled "storm vs death". It visually compares the values of two categories, "death" and "storm," across six different groups: "burden," "dining," "doctor," "sand," "safety," and "flash." The chart is horizontally oriented, with the x-axis labeled "number" ranging from 0 to 10, and the y-axis labeled "CC" listing the group categories.
Each group contains two bars: one representing the "death" category, colored in peru, and the other representing the "storm" category, colored in mediumblue. The chart's background is lavenderblush, providing a soft contrast to the bar colors.
For the "burden" group, the "death" bar reaches a value of approximately 2, while the "storm" bar is slightly higher at around 3. In the "dining" group, the "death" bar is at 5, and the "storm" bar is slightly higher at 6. The "doctor" group shows a "death" bar at 4 and a significantly lower "storm" bar at 1. The "sand" group has a "death" bar at 5, with the "storm" bar reaching 8, indicating a notable difference. In the "safety" group, the "death" bar is the highest at 8, while the "storm" bar is at 4. Lastly, the "flash" group shows both bars at 4 and 8 for "death" and "storm," respectively. The chart includes bar text, making the exact values clear and easy to read. This detailed representation allows for a straightforward comparison of the "death" and "storm" categories across the different groups, highlighting the variations and trends within each group.



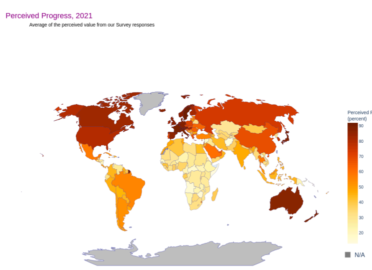
The image depicts a pie chart titled "Distribution: Unnamed: 0". It illustrates the percentage distribution of various classes. The largest segment is represented by "tea" in gainsboro color, accounting for 34.30% of the total. Following this, "copy" in lightgray makes up 26.38%, and "sphere" in sandybrown comprises 20.23%. Smaller segments include "golf" in orchid at 4.93%, "estate" in salmon at 4.07%, "wife" in lemonchiffon at 4.16%, "mark" in mistyrose at 3.71%, and the smallest segment, "title" in skyblue, at 2.22%. Each slice is clearly differentiated by color, providing a visual representation of the proportional data.



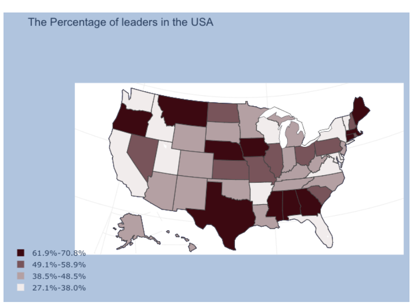
This image shows a stacked bar chart titled "rail & shop", which visually represents the distribution of values across three categories: "start," "terror," and "sister," with each category further divided into two groups: "rail" and "shop." The chart is oriented vertically against a dark red background, providing a striking contrast to the bars.
Each bar is segmented into two colors: ghostwhite for the "rail" group and mediumpurple for the "shop" group. The x-axis, labeled "AA," enumerates the categories "start," "terror," and "sister," while the y-axis, labeled "scale," ranges from 0 to 15, indicating the values of each group.
In the "start" category, the "rail" group has a value of approximately 5, and the "shop" group has a value of roughly 6, making the total height of the bar around 11. For the "terror" category, the "rail" group is about 2, and the "shop" group is around 5, resulting in a total bar height of approximately 7. Lastly, in the "sister" category, the "rail" group is about 1, and the "shop" group is roughly 4, giving a total height of around 5.
The chart effectively highlights the differences in values between the "rail" and "shop" groups within each category, with the "shop" group consistently having higher values than the "rail" group across all categories. The use of distinct colors and clear axis labels ensures that the data is easily interpretable, allowing for quick comparison of the values within and across the categories.



This image shows a line chart titled "Slovak Republic, Belgium, Denmark". It visually represents the weekly data trends for three countries over a span of seven weeks. The x-axis is labeled "Week" and ranges from week 45 to week 51, while the y-axis, which approximately ranges from 9 to 13, indicates the measured values for each country.
Three solid lines, each representing a different country, are plotted against this backdrop. The background color of the chart is mintcream, providing a soft and clear contrast to the data lines.
- **Slovak Republic**: This line starts at around 11.0 in week 45 and shows a general downward trend, dipping to approximately 9.7 by week 51. There is a slight increase around week 50, but the overall trend is a decrease.
- **Belgium**: The line for Belgium begins at about 12.2 in week 45, then drops sharply to around 10.3 by week 46. It continues to fluctuate slightly but generally trends downward, ending at approximately 9.4 in week 51.
- **Denmark**: Denmark's line starts the highest at around 12.7 in week 45. It shows a significant drop to about 12.1 by week 46 and continues to decrease, with minor fluctuations, reaching roughly 9.1 by week 51.
The legend clearly distinguishes each country's line, making it easy to compare their respective trends over the given weeks. The solid line style ensures that the data points are connected smoothly, highlighting the overall trends and variations for each country.

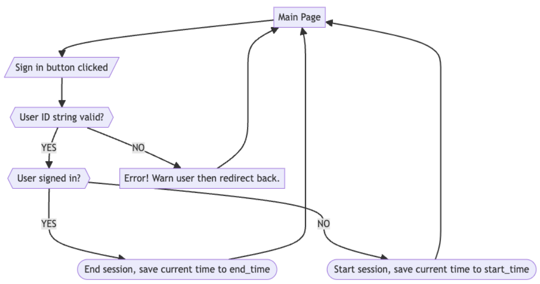


The image depicts a choropleth map titled "Perceived Progress, 2021". It visually represents the perceived progress across various countries worldwide, using a continuous color scale ranging from light yellow to deep brown. The color bar, labeled "Perceived Progress (percent)," transitions from light yellow at the lower end of the scale to deep brown at the higher end, indicating increasing levels of perceived progress.
Countries such as Afghanistan, Somalia, Angola, Zimbabwe, Central African Republic, and Namibia are shaded in light yellow, indicating lower perceived progress. Moving up the scale, nations like Niger, Ethiopia, Democratic Republic of the Congo, Tanzania, Mali, Chad, Kazakhstan, Sudan, Libya, Mongolia, and Algeria are depicted in golden hues, reflecting moderate perceived progress.
Further up the scale, countries including Iran, Colombia, Indonesia, Bolivia, India, Morocco, Peru, Chile, Argentina, Brazil, Türkiye, Mexico, Thailand, and Saudi Arabia are colored in medium orange, signifying higher perceived progress. Dark orange shades represent countries like Ukraine, Romania, China, Greece, Poland, Russian Federation, and the United States, indicating even higher levels of perceived progress.
At the top end of the scale, deep brown shades are used for Finland, Canada, Spain, Australia, Norway, Japan, France, and Sweden, highlighting the highest perceived progress among the countries represented.
This map effectively illustrates the global distribution of perceived progress, with a clear gradient from light yellow to deep brown, allowing for easy visual comparison of progress levels across different regions.

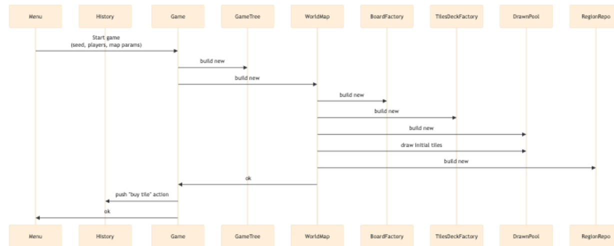


The image shows a choropleth map. It visually represents the distribution of leadership percentages across the United States, categorized into four distinct classes. Each state is color-coded according to its respective class, as indicated by the categorical legend.
- **Gallery (27.1%-38.0%)**: States in this class are shaded in a light gray color. These states are dispersed across various regions, including the Northeast (Vermont, New York), Mid-Atlantic (Delaware, Maryland), Southeastern (Virginia, Florida), East North Central (Wisconsin), South Central (Arkansas), and Western (Utah, Idaho, California, Washington).
- **Martini (38.5%-48.5%)**: Represented by a medium gray color, this class includes states such as New Jersey in the Mid-Atlantic, North Carolina, West Virginia, Kentucky, and Tennessee in the Southeastern region, Michigan and Indiana in the East North Central, Louisiana and Oklahoma in the South Central, Minnesota, South Dakota, and Wyoming in the West North Central, and several Western states including Colorado, New Mexico, Arizona, Alaska, and Hawaii.
- **Russett (49.1%-58.9%)**: States in this class are depicted in a reddish-brown color. These states include Rhode Island and New Hampshire in New England, Pennsylvania in the Mid-Atlantic, South Carolina in the Southeastern region, Ohio and Illinois in the East North Central, Missouri, Kansas, and North Dakota in the West North Central, and Nevada in the Western region.
- **Aubergine (61.9%-70.8%)**: The darkest shade, a deep purple, represents states with the highest percentage of leaders. This class includes Maine, Massachusetts, and Connecticut in New England, Georgia, Alabama, and Mississippi in the Southeastern region, Iowa and Nebraska in the West North Central, Texas in the South Central, and Montana and Oregon in the Western region.
The map reveals a diverse distribution of leadership percentages across the country, with no single region dominating any particular class. The Northeastern states show a mix of all classes, while the Southeastern region has a significant representation in both the Gallery and Aubergine classes. The Western states also display a wide range of leadership percentages, spanning all four classes. This visual representation allows for an easy comparison of leadership percentages across different states and regions in the USA.

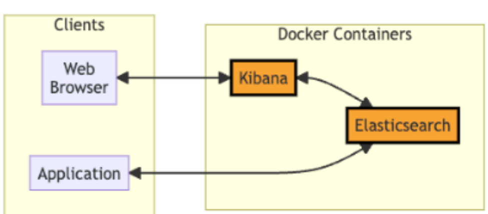
Figure 25. **Image and caption examples of chart in CompCap-118K**



The image shows the process flow for a user interacting with a sign-in system on a main page. Here is a detailed description of the flowchart's structure and process:
1. **Main Page (AA)**: The process begins at the Main Page, represented by the node labeled "Main Page".
2. **Sign-in Button Clicked (A)**: From the Main Page, the user clicks the sign-in button, which triggers the next step in the flow.
3. **User ID String Valid? (B)**: The system checks if the User ID string entered is valid. This decision point is represented by a diamond-shaped node labeled "User ID string valid?".
   - **If the User ID string is valid (YES)**: The flow proceeds to the next decision point to check if the user is already signed in.
   - **If the User ID string is not valid (NO)**: The flow moves to an error handling step.
4. **Error Handling (ER)**: If the User ID string is invalid, the system displays an error message to the user and redirects them back to the Main Page. This is represented by the node labeled "Error! Warn user then redirect back." which then points back to the Main Page (AA).
5. **User Signed In? (C)**: If the User ID string is valid, the system checks if the user is already signed in. This is another decision point represented by a diamond-shaped node labeled "User signed in?".
   - **If the user is already signed in (YES)**: The flow proceeds to end the current session.
   - **If the user is not signed in (NO)**: The flow proceeds to start a new session.
6. **End Session (D)**: If the user is already signed in, the system ends the current session and saves the current time to `end_time`. This is represented by the node labeled "End session, save current time to end_time", which then redirects the user back to the Main Page (AA).
7. **Start Session (E)**: If the user is not signed in, the system starts a new session and saves the current time to `start_time`. This is represented by the node labeled "Start session, save current time to start_time", which then redirects the user back to the Main Page (AA).
In summary, the flowchart outlines a user sign-in process that includes validation of the User ID, handling errors for invalid IDs, checking the user's sign-in status, and appropriately starting or ending sessions based on that status. Each step is interconnected to ensure the user is either redirected back to the Main Page or appropriately handled based on their sign-in status.



The image shows a sequence of interactions between different components involved in starting a game. Here is a detailed description of the flowchart's structure and process:
1. **Initiation from Menu**:
   - The process begins with the `Menu` component initiating the start of a game. This action is represented by the arrow from `Menu` to `Game` with the label "Start game (seed, players, map params)", indicating that the game is started with specific parameters such as seed, players, and map parameters.
2. **Game Initialization**:
   - Upon receiving the start command, the `Game` component proceeds to build necessary structures for the game. The first step is to build a new `GameTree`, as indicated by the arrow from `Game` to `GameTree` labeled "build new".
3. **WorldMap Construction**:
   - Next, the `Game` component initiates the construction of a new `WorldMap`. This is shown by the arrow from `Game` to `WorldMap` labeled "build new".
4. **Board and Tiles Setup**:
   - The `WorldMap` component then takes over to set up the game board and tiles. It first builds a new board using the `BoardFactory`, as indicated by the arrow from `WorldMap` to `BoardFactory` labeled "build new".
   - Following this, the `WorldMap` builds a new tiles deck using the `TilesDeckFactory`, as shown by the arrow from `WorldMap` to `TilesDeckFactory` labeled "build new".
5. **Drawn Pool Initialization**:
   - The `WorldMap` also builds a new `DrawnPool`, as indicated by the arrow from `WorldMap` to `DrawnPool` labeled "build new".
   - After building the `DrawnPool`, the `WorldMap` draws initial tiles into the pool, represented by the arrow from `WorldMap` to `DrawnPool` labeled "draw initial tiles".
6. **Region Repository Setup**:
   - The `WorldMap` then builds a new `RegionRepo`, as shown by the arrow from `WorldMap` to `RegionRepo` labeled "build new".
7. **Completion of WorldMap Setup**:
   - Once all the components of the `WorldMap` are set up, the `WorldMap` signals back to the `Game` component that the setup is complete. This is indicated by the arrow from `WorldMap` to `Game` labeled "ok".
8. **History Logging**:
   - The `Game` component then logs the initial action of "buy tile" into the `History`, as shown by the arrow from `Game` to `History` labeled "push 'buy tile' action".
9. **Final Acknowledgment**:
   - Finally, the `Game` component sends an acknowledgment back to the `Menu` indicating that the game setup is complete. This is represented by the arrow from `Game` to `Menu` labeled "ok".
In summary, the flowchart details the step-by-step process of starting a game, from the initial command in the menu to the setup of various game components and logging the initial action, culminating in a confirmation back to the menu.



The image is a diagram illustrating the interaction between different components within a system, specifically focusing on Docker containers and client applications. Here is a detailed description of the flowchart's structure and process:
1. **Docker Containers Subgraph**:
   - This subgraph contains two key components:
     - **Elasticsearch (ELS)**: Represented by a box labeled "Elasticsearch" with an orange fill and a black border.
     - **Kibana (KIB)**: Represented by a box labeled "Kibana" with the same styling as Elasticsearch.
2. **Clients Subgraph**:
   - This subgraph includes two client entities:
     - **Web Browser (WBR)**: Represented by a box labeled "Web\nBrowser".
     - **Application (APL)**: Represented by a box labeled "Application".
3. **Connections and Workflow**:
   - **Web Browser (WBR) <--> Kibana (KIB)**: There is a bidirectional connection between the Web Browser and Kibana, indicating that the web browser can both send requests to and receive responses from Kibana.
   - **Kibana (KIB) <--> Elasticsearch (ELS)**: Similarly, there is a bidirectional connection between Kibana and Elasticsearch, showing that Kibana interacts with Elasticsearch to fetch and display data.
   - **Application (APL) <--> Elasticsearch (ELS)**: The application has a bidirectional connection with Elasticsearch, suggesting that the application can query Elasticsearch for data and possibly send data to be indexed.
Summary:
The flowchart depicts a system where a web browser and an application interact with a set of Docker containers running Elasticsearch and Kibana. The web browser communicates with Kibana, which in turn interacts with Elasticsearch to retrieve and display data. Meanwhile, the application directly communicates with Elasticsearch for data operations. This setup highlights the central role of Elasticsearch in data storage and retrieval, with Kibana serving as a visualization tool accessible via a web browser.

Figure 26. **Image and caption examples of diagram in CompCap-118K**

Country	1970 Amount (in current US$)	1971 Amount (in current US$)
Honduras	6071000	3266000
Indonesia	720000	745000
Nepal	109000	1008000
Panama	4950000	11579000
Paraguay	6443000	7155000

The image shows a table presenting financial data on aid amounts (in current US dollars) received by five countries—Honduras, Indonesia, Nepal, Panama, and Paraguay—over the years 1970 and 1971.
The table columns are as follows:
- **Country**: Lists the names of the countries receiving aid.
- **1970 Amount (in current US$)**: Shows the amount of aid received by each country in the year 1970.
- **1971 Amount (in current US$)**: Shows the amount of aid received by each country in the year 1971.
The rows represent the individual countries and their corresponding aid amounts for the specified years. The data includes specific financial figures for each country, indicating the aid received in both 1970 and 1971.
Here's the data represented in the table:
Country	1970 Amount (in current US$)	1971 Amount (in current US$)
Honduras	6071000.0	3266000.0
Indonesia	720000.0	745000.0
Nepal	109000.0	1008000.0
Panama	4950000.0	11579000.0
Paraguay	6443000.0	7155000.0

Year	Export Cost (US$ per container)	Import Cost (US$ per container)
2008	870	980
2009	870	980
2010	870	980
2011	870	980
2012	825	980
2013	750	800
2014	750	800

The image shows a table with data on the costs associated with exporting and importing a container (in US dollars) for the years 2008 to 2014. The table is organized as follows:
Columns:
- Export Cost (US$ per container)
- Import Cost (US$ per container)
Rows:
- Years from 2008 to 2014
Each cell contains the cost in US dollars for exporting or importing a container for the corresponding year. The data indicates that the export cost remained constant at $870 from 2008 to 2011, then decreased to $825 in 2012, and further dropped to $750 in 2013 and 2014. The import cost remained constant at $980 from 2008 to 2012, then decreased to $800 in 2013 and 2014.
Here's the data represented in the table:
Year	Export Cost (US$ per container)	Import Cost (US$ per container)
2008	870	980
2009	870	980
2010	870	980
2011	870	980
2012	825	980
2013	750	800
2014	750	800

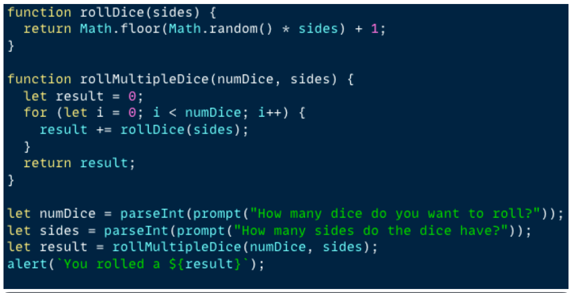
Country	Gaseous fuel Emissions(in percent)	Liquid fuel Emissions(in percent)	Solid fuel Emissions(in percent)
Heavily indebted poor countries	1.2	79.07	10.77
High income	16.03	53.65	26.48
High income: nonOECD	12.6	45.01	3.42
High income: OECD	16.18	54.04	27.52
Latin America(all income levels)	11.64	69.96	5.9

The image shows a table presenting data on emissions from different types of fuel—gaseous, liquid, and solid—expressed as percentages for various country groups.
The table columns are as follows:
- **Country**: Lists the different country groups.
- **Gaseous fuel Emissions (in percent)**: Shows the percentage of emissions from gaseous fuels.
- **Liquid fuel Emissions (in percent)**: Shows the percentage of emissions from liquid fuels.
- **Solid fuel Emissions (in percent)**: Shows the percentage of emissions from solid fuels.
The rows represent different country groups, including:
- Heavily indebted poor countries
- High income
- High income: nonOECD
- High income: OECD
- Latin America (all income levels)
Each cell in the table provides the specific percentage of emissions from each type of fuel for the corresponding country group.
Here's the data represented in the table:
Country	Gaseous fuel Emissions(in percent)	Liquid fuel Emissions(in percent)	Solid fuel Emissions(in percent)
Heavily indebted poor countries	1.2	79.07	10.77
High income	16.03	53.65	26.48
High income: nonOECD	12.6	45.01	3.42
High income: OECD	16.18	54.04	27.52
Latin America(all income levels)	11.64	69.96	5.9

Year	Gaseous fuel Emissions(kt)	Liquid fuel Emissions(kt)	Solid fuel Emissions(kt)
2007	1474.13	7.33	1466.8
2008	2082.86	586.72	1496.14
2009	2112.19	594.05	1518.14
2010	2258.87	715.07	1543.81
2011	2200.2	630.72	1569.48

The image shows a table with data on emissions from different types of fuel (measured in kilotonnes) for the years 2007 to 2011. The table is organized as follows:
Columns:
- Gaseous fuel Emissions (kt)
- Liquid fuel Emissions (kt)
- Solid fuel Emissions (kt)
Rows:
- Years from 2007 to 2011
Each cell contains the amount of emissions in kilotonnes for the corresponding type of fuel and year. The data shows an increasing trend in emissions for all types of fuel over the years, with the highest values recorded in 2010 and 2011.
Here's the data represented in the table:
Year	Gaseous fuel Emissions(kt)	Liquid fuel Emissions(kt)	Solid fuel Emissions(kt)
2007	1474.13	7.33	1466.8
2008	2082.86	586.72	1496.14
2009	2112.19	594.05	1518.14
2010	2258.87	715.07	1543.81
2011	2200.2	630.72	1569.48

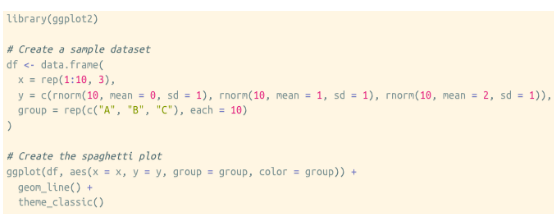
Figure 27. **Image and caption examples of table in CompCap-118K**



The image shows Javascript code. Here's the code:

```
function rollDice(sides) {
  return Math.floor(Math.random() * sides) + 1;
}
function rollMultipleDice(numDice, sides) {
  let result = 0;
  for (let i = 0; i < numDice; i++) {
    result += rollDice(sides);
  }
  return result;
}
let numDice = parseInt(prompt("How many dice do you want to roll?"));
let sides = parseInt(prompt("How many sides do the dice have?"));
let result = rollMultipleDice(numDice, sides);
alert(`You rolled a ${result}`);
```

This code is written in JavaScript and it implements a dice roller functionality. The code defines two functions: `rollDice` and `rollMultipleDice`.
The `rollDice` function takes an integer `sides` as an argument, which represents the number of sides of the dice. It generates a random number between 1 and `sides` (inclusive) using the `Math.random()` function, and returns the result.
The `rollMultipleDice` function takes two arguments: `numDice` and `sides`. It rolls the dice `numDice` times, each with `sides` sides, and returns the total result.
The code then prompts the user to input the number of dice they want to roll and the number of sides the dice have. It calls the `rollMultipleDice` function with the user's input and displays the result in an alert box.



The image shows R code. Here's the code:

```
library(ggplot2)
# Create a sample dataset
df <- data.frame(
  x = rep(1:10, 3),
  y = c(rnorm(10, mean = 0, sd = 1), rnorm(10, mean = 1, sd = 1), rnorm(10, mean = 2,
sd = 1)),
  group = rep(c("A", "B", "C"), each = 10)
)
# Create the spaghetti plot
ggplot(df, aes(x = x, y = y, group = group, color = group)) +
  geom_line() +
  theme_classic()
```

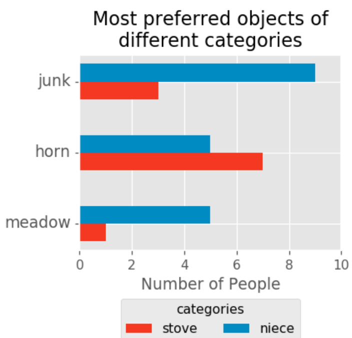
This code generates a spaghetti plot using the ggplot2 library in R. The code first creates a sample dataset with three groups (A, B, and C) and 10 observations each. The x-axis represents the observation number, and the y-axis represents the value of the observation. The group variable is used to color and group the lines.
The ggplot function is then used to create the spaghetti plot. The aes function is used to map the x, y, and group variables to the plot. The geom_line function is used to create the lines, and the theme_classic function is used to apply a classic theme to the plot.
The language used is R, a popular programming language for statistical computing and graphics.

The image shows Java code. Here's the code:

```
import java.util.Comparator;
class Student {
    int rollno;
    String name;
    int age;
    Student(int rollno, String name, int age) {
        this.rollno = rollno;
        this.name = name;
        this.age = age;
    }
}
class AgeComparator implements Comparator<Student> {
    public int compare(Student s1, Student s2) {
        if (s1.age == s2.age)
            return 0;
        else if (s1.age > s2.age)
            return 1;
        else
            return -1;
    }
}
public class Main {
    public static void main(String[] args) {
        Student[] arr = {new Student(111, "bbbb", 20), new Student(131, "aaaa", 22), new
Student(121, "cccc", 21)};
        AgeComparator ageComparator = new AgeComparator();
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (ageComparator.compare(arr[i], arr[j]) > 0) {
                    Student temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
        for (Student student : arr) {
            System.out.println(student.rollno + " " + student.name + " " + student.age);
        }
    }
}
```

This Java code implements a basic comparator example. It defines a `Student` class with attributes `rollno`, `name`, and `age`. The `AgeComparator` class implements the `Comparator` interface and provides a custom comparison logic based on the `age` attribute of the `Student` objects.
In the `main` method, an array of `Student` objects is created and sorted in ascending order of age using the `AgeComparator`. The sorted array is then printed to the console.
The language used is Java.

Figure 28. **Image and caption examples of code in CompCap-118K**



👤: What is shown in the image?

LLaVA-NeXT-Vicuna-13B: The bar chart provides a visual representation of the preferences of people for different objects categorized as "junk", "horn", and "meadow". The data suggests that "junk" is the most preferred object, with the highest number of people indicating a preference for it. "Horn" is the second most preferred object, followed by "meadow". The chart indicates that "junk" is preferred by approximately 9 people, "horn" by about 6 people, and "meadow" by about 2 people. The chart is a straightforward depiction of the data, with no additional context provided.

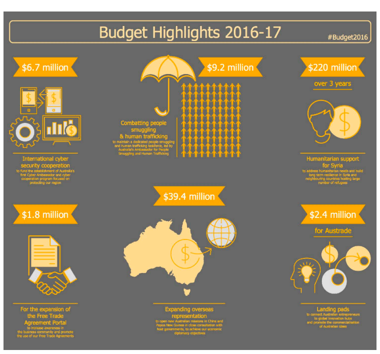
CompCap-13B: The image is a grouped bar chart titled "Most preferred objects of different categories". It visually compares the preferences for three categories: "junk," "horn," and "meadow," across two groups: "niece" and "stove." The chart is oriented horizontally, with the x-axis representing the "Number of People" and the y-axis enumerating the categories.

Each category is represented by two bars: one for "niece" and one for "stove." The "niece" bars are colored in steelblue, while the "stove" bars are colored in orangered. The background of the chart is ghostwhite, providing a clear contrast to the bar colors.

In the "junk" category, the "niece" group has a significantly higher preference, with a value close to 9, compared to the "stove" group, which has a value around 3. For the "horn" category, the "stove" group shows a higher preference, with a value near 7, while the "niece" group has a value around 5. In the "meadow" category, the "niece" group has a value close to 5, whereas the "stove" group has a much lower value, around 1.

The x-axis is labeled "Number of People," and the y-axis is labeled "Category," clearly indicating the data being compared. The chart includes bar text, making it easy to read the exact values directly from the bars. This detailed representation allows for a clear comparison of preferences across different categories and groups.

Figure 29. **More examples of MLLMs on CI captioning (part 1).**

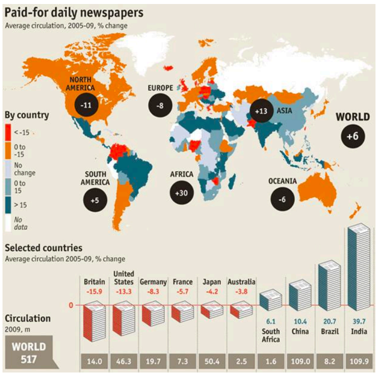


👤: What is shown in the image?

LLaVA-NeXT-Vicuna-13B: The image presents a budget highlights report for the fiscal year 2016-17. It outlines various budget allocations and expenditures across different sectors. The largest allocation is for the "Combating people smuggling & human trafficking" sector, which received $9.2 million. This is followed by the "International cyber security cooperation" sector, which received $6.7 million. The "Humanitarian support" sector received $220 million, with a note that this is over a 3-year period. The budget also includes allocations for "Expanding overseas representation" and "Landing pads" for Australian entrepreneurs, with specific amounts not provided in the image. The report also mentions the "Budget 2016-17" hashtag, indicating that this is a social media post or a report related to the Australian budget for the fiscal year 2016-17. The overall tone of the image is informative, providing a snapshot of the budget allocations in a clear and concise manner.

CompCap-13B: The image is a budget highlights infographic for the year 2016-17. The infographic is structured with a title at the top and a series of budget highlights listed below. Each highlight is accompanied by a yellow umbrella icon and a specific dollar amount, indicating the budget allocation for that particular item. The budget highlights are as follows:
1. **Combating people smuggling and human trafficking**: $9.2 million.
2. **International cyber security cooperation**: $6.7 million.
3. **Humanitarian support for Syria**: $220 million.
4. **Expanding overseas representation**: $39.4 million.
5. **Landing pads for Australian entrepreneurs**: $2.4 million.
6. **For the expansion of the Free Trade Agreement Portal**: $1.8 million.
The infographic also includes a hashtag at the top, indicating the source of the information: #Budget2016. The overall design is simple and straightforward, with a focus on the financial figures and their respective purposes. The use of yellow for the umbrella icons and the dollar amounts provides a visual contrast against the gray background, making the information easily readable and understandable.

Figure 30. **More examples of MLLMs on CI captioning (part 2).**



: What is shown in the image?

LLaVA-NeXT-Vicuna-13B: The image presents a comparative analysis of the average daily newspaper circulation in various countries across the world. The data is measured in millions of copies and is presented for the years 2005 and 2009. The countries are categorized into different regions, including North America, Europe, Asia, and the World.

In 2005, North America had the highest average newspaper circulation at 14.0 million copies, followed by Europe at 11.1 million, Asia at 10.9 million, and the World at 51.7 million. By 2009, the World's average newspaper circulation had increased to 90.9 million copies, while North America's had decreased to 11.6 million, Europe's had decreased to 10.4 million, Asia's had increased to 10.9 million, and South America's had increased to 10.9 million.

The data suggests a general decline in newspaper circulation in North America and Europe, while Asia and South America experienced growth. The World's average newspaper circulation increased significantly, indicating a global trend of increasing newspaper readership.

The bar chart provides a clear visual representation of the data, with each country's bar showing the change in newspaper circulation from 2005 to 2009. The use of color coding helps to quickly identify which countries experienced growth or decline in newspaper readership.

Overall, the image provides a comprehensive overview of the state of newspaper circulation across different regions of the world, highlighting both the challenges and opportunities faced by the newspaper industry.

CompCap-13B: The image is a bar chart titled "Paid-for daily newspapers," which presents data on the average circulation percentage change for the year 2009 across various countries and the world. The chart is divided into two main sections: a map showing the percentage change by country and a bar chart detailing the average circulation percentages for selected countries.

Map Section:
- **North America:** Marked with a significant negative change of -11%.
- **Europe:** Marked with a slight negative change of -8%.
- **Asia:** Marked with a positive change of +13%.
- **Oceania:** Marked with a negative change of -6%.
- **South America:** Marked with a positive change of +5%.
- **Africa:** Marked with a positive change of +30%.
- **World:** Marked with a positive change of +6%.

Bar Chart Section:
- **United Kingdom:** Marked with a negative change of -15.9%.
- **United States:** Marked with a negative change of -13.3%.
- **Germany:** Marked with a negative change of -8.3%.
- **France:** Marked with a negative change of -5.7%.
- **Japan:** Marked with a negative change of -4.2%.
- **Australia:** Marked with a negative change of -3.8%.
- **South Africa:** Marked with a positive change of 6.1%.
- **China:** Marked with a positive change of 10.4%.
- **Brazil:** Marked with a positive change of 20.7%.
- **India:** Marked with a positive change of 39.7%.

The chart effectively highlights the varying trends in paid-for daily newspaper circulation changes across different regions, with notable negative changes in North America, Europe, and parts of Asia, while positive changes are observed in Asia, Africa, Oceania, South Africa, China, Brazil, and India.

Figure 31. **More examples of MLLMs on CI captioning (part 3).**

Amount Manny spent on lunch	
Day	Amount spent
Tuesday	$10
Wednesday	$2
Thursday	$7
Friday	$9
Saturday	$8

: What is shown in the image?

LLaVA-NeXT-Vicuna-13B: The image presents a budget for a week's worth of lunches for a person named Manny. The budget is broken down by day of the week, with the amount spent on lunch for each day listed next to the day. The total amount spent for the week is $8. The days are listed in a row, with the amount spent on each day in a column to the right. The budget is simple and straightforward, with no additional information provided.

CompCap-13B:The image shows a table detailing the amount of money spent on lunch by a person named Manny over a week. The table is organized as follows:
Columns:
- Day: Lists the days of the week from Tuesday to Saturday.
- Amount spent: Indicates the amount of money spent on lunch for each day.
Rows:
- Each row corresponds to a specific day of the week.
Details:
- On Tuesday, Manny spent $10 on lunch.
- On Wednesday, Manny spent $2 on lunch.
- On Thursday, Manny spent $7 on lunch.
- On Friday, Manny spent $9 on lunch.
- On Saturday, Manny spent $8 on lunch.
The table provides a clear breakdown of Manny's lunch expenses over the specified week.
Here's the data represented in the table:
Day	Amount spent
Tuesday	$10
Wednesday	$2
Thursday	$7
Friday	$9
Saturday	$8

Figure 32. **More examples of MLLMs on CI captioning (part 4).**