Boosting Text-to-Chart Retrieval with Semantic Insights

Anonymous Author(s)

Abstract

Charts are crucial for data analysis and decision-making. Text-to-chart retrieval systems have become increasingly important for Business Intelligence (BI), where users need to find relevant charts that match their analytical needs. These needs can be categorized into precise queries that are well-specified and fuzzy queries that are more exploratory – both require understanding the semantics and context of the charts. However, existing text-to-chart retrieval solutions often fail to capture the semantic content and contextual information of charts, primarily due to the lack of comprehensive metadata (or semantic insights). To address this limitation, we propose a training data development pipeline that automatically synthesizes hierarchical semantic insights for charts, which produces 207,498 semantic insights for 69,166 charts. Based on these, we train a CLIP-based model named ChartFinder to learn better representations of charts for text-to-chart retrieval. Our method leverages rich semantic insights during the training phase to develop a model that understands both visual and semantic aspects of charts. To evaluate text-to-chart retrieval performance, we curate *the first* benchmark, CRBench, for this task with 21,862 charts and 326 text queries from real-world BI applications, with ground-truth labels produced by the crowd workers. Experiments show that ChartFinder significantly outperforms existing methods in text-to-chart retrieval tasks across various settings. For precise queries, ChartFinder achieves up to 66.9% NDCG@10, which is 11.58% higher than state-of-the-art models. In fuzzy query tasks, our method also demonstrates consistent improvements, with an average increase of 5% across nearly all metrics.

CCS Concepts

• **Information systems** → *Business Intelligence*.

Keywords

Text-to-Chart Retrieval, Contrastive Learning, Hierarchical Semantic Insights

1 Introduction

Visualization charts (charts for short) are essential for various fields, including data analysis and business decision-making [19, 29, 33, 43]. In particular, Business Intelligence (BI) systems heavily rely on charts to present complex data in ways that provide users with intuitive and actionable insights [23, 32]. Most charts are carefully designed to serve specific analytical tasks, with considerations for visual design, chart types, and data relationships.

Given their quality and versatility, these well-crafted charts are valuable resources that can be reused across different tasks and contexts. Given a user-provided text query and a chart repository, the ***text-to-chart retrieval*** task is to find a set of charts that align with the user's query. This process requires understanding both the *visual characteristics* of the charts and the *semantic content of the user query*, ensuring that the retrieved charts are not only relevant but also contextually suitable for the user's analytical task.

Text-to-Chart Retrieval in Real-world BI Systems. Through collaboration with a leading BI company X^1, we analyzed real-world user behavior and identified two primary categories of user queries for chart retrieval: precise and fuzzy queries. These categories reflect the diverse needs of users, ranging from specific, data-driven queries to more exploratory or stylistic tasks.

Precise queries are characterized by data-driven intent and focus on specific analytical targets. These queries aim to retrieve charts that present well-defined information, such as trends, comparisons, or aggregated data points, to answer a clear analytical question. For example, as shown in Figure 1, a user may ask, "Find charts showing the hotel booking distribution across channels in Europe for the year 2022." Such queries demand high semantic alignment to ensure the retrieved charts align closely with the user's specific requirements. To achieve this, retrieval systems must effectively leverage both the data context and the visual features of chart images to establish strong semantic relationships between the query and the chart.

Fuzzy queries, on the other hand, are more open-ended and often focus on exploratory tasks and abstract information. These queries are less constrained and may emphasize stylistic or structural elements of charts rather than specific data content. For example, a user might query, "Use a pie chart to compare different sale methods", as shown in Figure 1. Addressing such queries requires a retrieval system capable of understanding a broader intent and contextual flexibility inherent in exploratory tasks.

Limitations of Existing Methods. Text-to-image retrieval has seen remarkable advancements, leveraging state-of-the-art Multimodal Large Language Models (MLLMs) [16]. These approaches have demonstrated effectiveness in general image retrieval tasks by efficiently capturing visual features from images. However, when being directly adapted to text-to-chart retrieval, these methods face several challenges in addressing the unique aspects of charts.

First, existing methods *fail to capture the semantic depth of charts*. Unlike natural images, which primarily convey visual information,

---

[1]The identity of the company is withheld due to the anonymous submission.
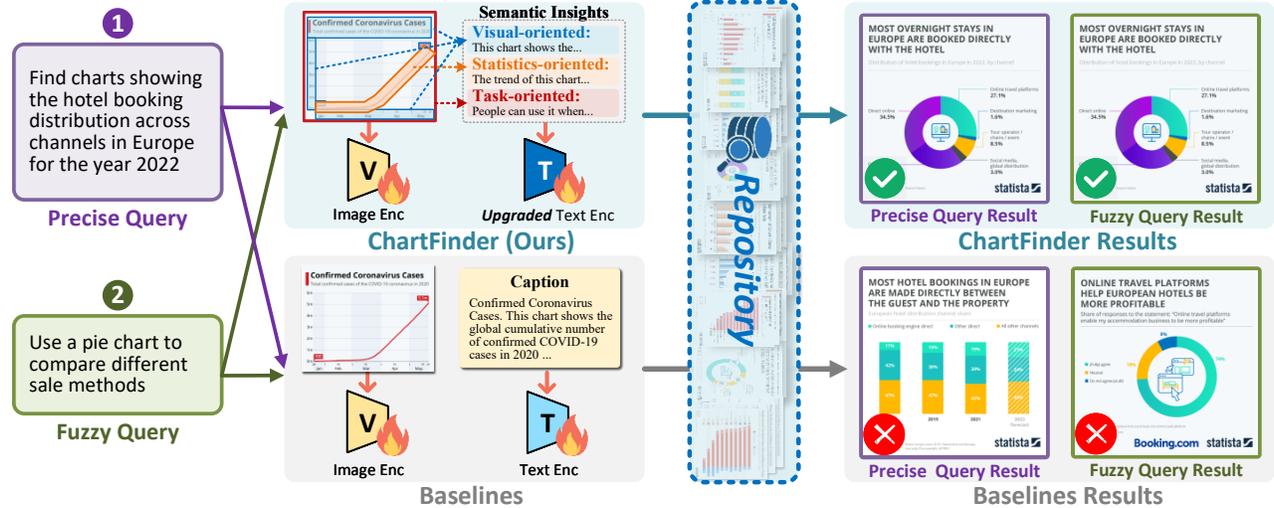
Figure 1: ChartFinder integrates both visual and semantic aspects of charts to improve text-to-chart retrieval.

charts encode complex data relationships, trends, and comparisons. Existing methods that focus solely on visual features often retrieve charts that are visually similar but semantically irrelevant. For example, two charts may share visual characteristics (*e.g.,* both being bar charts) but represent entirely different datasets, topics, or analytical purposes. This limitation is particularly problematic for *precise queries*, which require high semantic alignment between the query and the retrieved chart to meet data-driven needs.

Second, current approaches *fail to address both precise and fuzzy queries* effectively. For precise queries, existing methods that prioritize visual features lack the ability to connect queries with the underlying data or analytical intent, leading to poor retrieval performance. For fuzzy queries, existing methods struggle to interpret broader user intents, such as stylistic preferences or exploratory goals, as they lack mechanisms to capture design elements of charts.

Key Idea. To improve text-to-chart retrieval, it is essential to capture both the visual and semantic aspects of charts. Our *key idea* is to extract semantic insights from chart metadata (*e.g.,* source code) and combine this with the chart images. By training a model that incorporates both the charts (*in the form of images*) and the semantic insights (*in the form of text*), we aim to develop a system that comprehensively understands and bridges the gap between visual representation and the underlying semantic meaning of charts.

However, implementing this idea presents several **challenges**.

First, manually annotating semantic insights for charts is costly and time-consuming. Thus, we need to develop an efficient and accurate method to derive these semantic insights from chart metadata automatically. Once semantic insights are generated, the second challenge is effectively integrating them with the visual features of the charts. We need to design a model architecture and training strategy that can seamlessly combine these two modalities, *i.e.,* semantic insights and visual features, to generate more accurate representations of the charts. Third, there is currently no readily available corpus containing text queries and their corresponding charts to evaluate the performance of text-to-chart retrieval.

Our Proposal. We propose ChartFinder, a new text-to-chart retrieval model that leverages contrastive learning techniques to align

both the visual features and semantic insights of charts, based on the CLIP architecture [30, 48]. Specifically, our model is trained with both chart images (visual content) and their corresponding semantic insights (textual content), as shown in Figure 1. Once trained, ChartFinder takes a user query and a chart repository as input and retrieves the top-k most relevant charts based on the embedding distance between the text query and the charts.

To train the model effectively, we have developed a *training data generation pipeline* that automatically generates a large number of charts along with their corresponding semantic insights. Inspired by techniques in automatic data visualization [19] and insight generation [24], which create insightful charts from raw data, our pipeline synthesizes three key levels of semantic insights: *(1) Visual-oriented Insight*: a high-level summary of the chart, capturing the overall pattern and visual trends; *(2) Statistics-oriented Insights*: an in-depth analysis of the trends, comparisons, and relationships within the data, highlighting statistical patterns and anomalies; and *(3) Task-oriented Insights*: contextual information on how the chart can be applied for suitable data analysis tasks.

To evaluate the performance of the text-to-chart retrieval task, we introduce CRBench, the first text-to-chart retrieval benchmark sourced from real-world BI scenarios. CRBench contains 326 queries and 21,862 chart images. All query-chart labels are generated by crowd workers in order to capture the diversity of user needs.

Contributions. We make the following contributions:

(1) **Semantic Insights Generation for a New Model.** We develop an automatic pipeline for generating semantic insights from chart metadata, synthesizing three levels of insights. This approach addresses the challenge of manual annotation and enriches the model's ability to understand charts in a comprehensive manner (Section 3.1). We propose ChartFinder, a new text-to-chart retrieval model that combines contrastive learning with the CLIP architecture to align visual features and semantic insights of charts, enabling more context-aware chart retrieval (Section 3.2).

(2) **A New Benchmark.** We curate CRBench, the first benchmark for text-to-chart retrieval sourced from real-world BI scenarios.
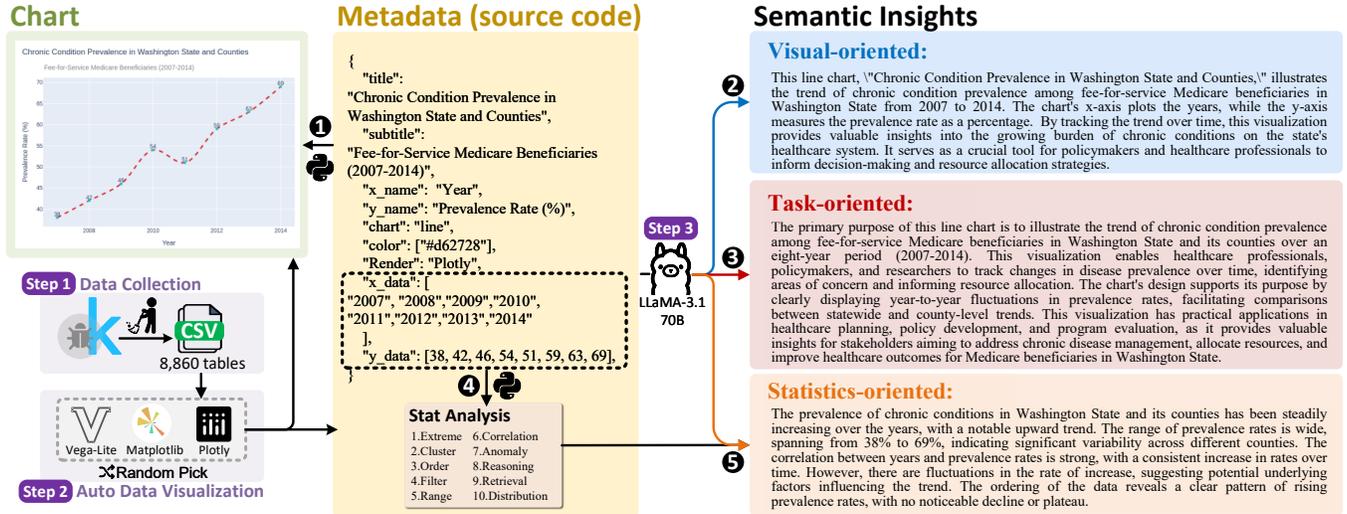
Figure 2: The Pipeline of Automatic Generation of Semantic Insights. We collect 8,860 high-quality tables and automatically generate charts using visualization libraries like Matplotlib and Plotly. The collected data and source code serve as metadata, including chart titles, axis labels, and data points. We analyze metadata and the chart images to compute three levels of semantic insights: visual-oriented insights, task-oriented insights, and statistics-oriented insights.

This benchmark, containing 326 text queries and 21,862 charts, provides a robust dataset to evaluate the performance of text-to-chart retrieval models and helps advance research in this area (Section 4).

(3) **Extensive Experiments.** Experiments show that ChartFinder outperforms existing methods in text-to-chart retrieval tasks across several settings, including different types of user queries and zero-shot scenarios (Section 5).

2 Preliminaries

Charts. A *chart C* is a visual representation of data in the form of rasterized images designed to communicate patterns, trends, or comparisons across one or more variables.

Chart Metadata. Let $M = (T, X, Y, CT, D, E)$ be the associated metadata, where:

- T is the title of the chart,
- X and Y are the labels for the x-axis and y-axis, respectively,
- $CT \in \{line, bar, scatter, pie, \dots\}$ is the chart type,
- $D = \{(x_i, y_i)\}_{i=1}^{n}$ denotes the underlying data, where (x_i, y_i) are data points, and n is the number of data points,
- E represents additional elements such as the legend, data labels, or annotations.

Semantic Insights of Charts. Given a chart C and its metadata M, semantic insights I are higher-level interpretations derived from the chart's visual representation and metadata.

Formally, we define $I = (I^v, I^t, I^s)$, where:

- I^v represents *Visual-oriented Insights*: These insights are derived from visual patterns, trends, or relationships observed in the chart. For example, trends shown in the chart may reveal increasing or decreasing patterns over time

- I^t denotes *Task-oriented Insights*: These insights relate to the practical applications of the chart in specific tasks or decision-making processes. For example, a chart illustrating healthcare trends could help policymakers identify areas of resource allocation.
- I^s represents *Statistics-oriented Insights*: These are quantitative analyses and statistical observations derived from the data. For example, identifying correlations, distributions, or outliers in the dataset used for the chart.

The process of deriving semantic insights is represented as $f : (C, M) \rightarrow I$, where f analyzes the chart C and its metadata M to produce the insights I. Please refer to Section 3.1 for details.

EXAMPLE 1. *(Example of Chart, Metadata, and Semantic Insights.) As shown in Figure 2, the line chart C is rendered using the Plotly visualization library. The metadata for this chart is the source code used by Plotly, which contains the title, axis labels, data points, chart type, and other relevant information. We utilize the LLaMA-3.1 model to analyze this metadata and generate semantic insights.*

Text-to-Chart Retrieval. Let $C = \{C_1, C_2, \dots, C_n\}$ denote a repository of charts and a textual query Q provided by a user. The objective of the text-based chart retrieval problem is to retrieve a ranked list of k charts, $C_k = \{C_{r_1}, C_{r_2}, \dots, C_{r_k}\} \subseteq C$, where $k \ll n$ and C_{r_i} represents the i-th most relevant chart to the query Q.

3 Methodology

In this section, we first describe the automatic generation of semantic insights for charts to capture their semantic depth (Section 3.1). Then, we introduce ChartFinder, a model that aligns visual features and semantic insights for text-to-chart retrieval (Section 3.2).

3.1 Automatic Generation of Semantic Insights

Manually annotating charts to generate semantic insights is time-consuming and inefficient. To address this, we propose a training data development pipeline that automatically derives semantic insights from raw data, making the process more efficient and scalable.

Figure 2 shows our proposed pipeline, which consists of three key steps: data collection, automatic data visualization, and semantic insights generation based on the metadata (Figure 2: ❷–❺).

Step 1: Data Collection. The first step in generating high-quality semantic insights is selecting a reliable source of datasets. To meet the above criteria, we selected datasets from Kaggle, which offers several advantages: many datasets are derived from real-world applications or competitions. In addition, Kaggle provides a scoring system for datasets, with those receiving a score of 10 typically offering comprehensive content and high data quality.

We crawled 9,003 datasets with a score of 10 from Kaggle and performed local preprocessing, including removing missing values. After this cleaning process, we retained 8,860 high-quality CSV tables, completing the data preparation for subsequent steps.

Step 2: Automatic Data Visualization. The second step is to create meaningful charts from the collected 8,860 tables. Specifically, we used DeepEye [19], an automatic visualization system, to generate a variety of charts, including bar charts, pie charts, line charts, scatter plots, grouped line charts, stacked bar charts, and grouped bar charts, covering widely-used chart types [18].

To increase the visual diversity of the charts, we re-rendered them using three widely used visualization libraries: Matplotlib, Plotly, and Vega-Lite. For each chart suggested by DeepEye [19], we randomly selected one of these libraries to re-draw the chart (see Figure 2: ❶). Note that DeepEye [19] provides the metadata for each chart, which includes information such as axis labels, chart titles, and data points. We also maintain the metadata within the source code of the visualization, as shown in Figure 2. This metadata is critical for deriving semantic insights in Step 3. In total, we generated **69,166 charts** rendered using Matplotlib, Plotly, or Vega-Lite libraries, along with their corresponding *metadata.*

Step 3: Semantic Insights Generation based on Metadata. The third step is to generate detailed chart semantic insights based on the metadata of the charts. To comprehensively interpret the charts, we design three levels of semantic insights: from the apparent visual patterns observed in the chart, to data-driven statistical insights that uncover trends, comparisons, and relationships, to task-oriented insights that provide context on how the chart can be applied in real-world scenarios for decision-making and resource allocation. This layered approach allows for a holistic interpretation of each chart, supporting diverse user queries and enhancing the retrieval process. As illustrated in Figure 2 (❷–❺), the semantic insights I are computed through the following steps:

(1) *Figure 2-❷: Visual-oriented Insight Generation.* In this step, we generate visual-oriented insights by analyzing the chart's visual patterns and trends. This insight summarizes the chart's overall visual presentation, such as identifying significant trends, distributions, and patterns in the data. For instance, we would identify whether the chart shows a clear upward or downward trend, highlighting its most significant visual characteristics.

(2) *Figure 2-❸: Task-oriented Insight Generation.* Next, we use both the metadata and the visual insights to derive task-oriented insights. This step contextualizes the chart for practical applications, helping users understand how the chart can be used in real-world scenarios. For example, a chart depicting healthcare data might be interpreted in terms of how it could inform resource allocation or support decision-making in healthcare planning.

(3) *Figure 2-❺: Statistics-oriented Insight Generation.* First, we apply a set of 10 statistical analysis tasks [43] on the chart's metadata (*e.g.,* the X/Y-axes), as shown in Figure 2-❹. These tasks help identify statistical properties such as trends, correlations, and anomalies within the data. In the second step, we generate statistics-oriented insights by summarizing this statistical information. These insights provide an in-depth understanding of the chart's data, focusing on aspects such as correlations, distributions, outliers, and other statistical relationships This provides a deeper understanding of the data's behavior and its significance.

After all the processes, we got a total of 69,166 charts and corresponding 207,498 semantic insights.

3.2 The Design Details of ChartFinder

In this section, we first describe the architecture of ChartFinder, followed by the training details. Finally, we outline how the model performs text-to-chart retrieval with ChartFinder.

3.2.1 Model Architecture. The design of ChartFinder is driven by the need to effectively retrieve relevant charts based on a user's text query. This is a cross-modal retrieval task, where the goal is to align the textual queries with the charts' visual and semantic features. Specifically, the challenge is to bridge the two modalities: the chart images (*visual content*) and the corresponding semantic insights (*textual content*). Our model should learn a shared embedding space where both visual and textual information can be effectively aligned.

To achieve this, we adopt a CLIP-based framework [30], which has demonstrated success in cross-modal tasks by aligning visual features with natural language supervision [2, 15, 26]. Since the semantic insights associated with charts can be long and detailed, we utilize Long-CLIP [48] to process these insights. Unlike the original CLIP model, which is limited to encoding short captions (up to 77 tokens), Long-CLIP extends the token limit to 248 tokens, allowing it to handle the longer and more complex textual descriptions required for charts. For encoding the chart images, we use the Vision Transformer (ViT) [4], which produces dense visual embeddings. ViT has shown strong performance in various image-related tasks, making it ideal for encoding the visual features of charts.

Together, the ViT-based image encoder and the Long-CLIP text encoder form the core of ChartFinder. To further explore the model's scalability, we introduce two variants: ChartFinder and ChartFinder-B. Both versions utilize the same Long-CLIP text encoder but differ in their visual encoder. ChartFinder employs the larger ViT-L/14, while ChartFinder-B uses the smaller ViT-B/16.

3.2.2 Training Details. As depicted in Figure 3(a), for each chart C_i, we associate three distinct semantic insights: I_i^v (visual-oriented insights), I_i^t (task-oriented insights), and I_i^s (statistical-oriented insights). The objective is for the model to learn to match the chart C_i with all three semantic insights in a unified embedding space.

(a) Training Process with Semantic Insights of Charts

(b) Retrieval Process based on User Query
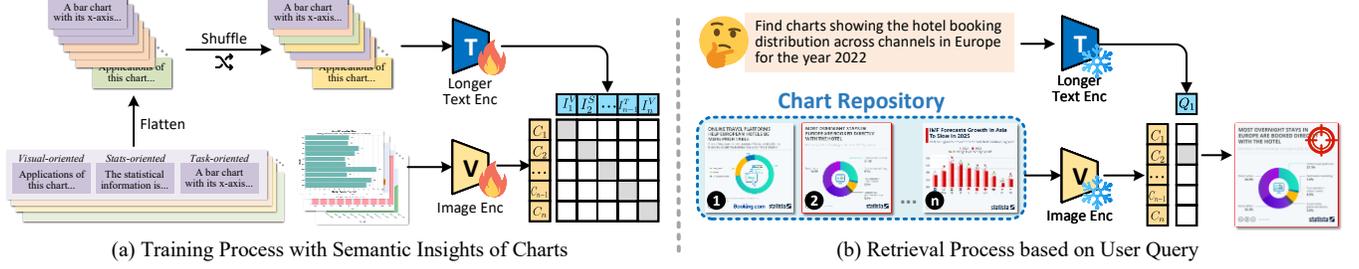
Figure 3: (a) ChartFinder is trained using both chart images and their corresponding semantic insights. (b) ChartFinder takes a query and a set of chart images as input, retrieving the most relevant charts based on embedding similarity.

To achieve this, we experimented with two different strategies for training:

One approach was to maintain a similarity matrix for the three different insights during training and compute the average loss across them. However, experiments showed that this method was ineffective in learning meaningful representations.

After further analysis, we decided to *flatten* the semantic insights corresponding to all charts and then *shuffle* them, as illustrated in Figure 3(a). This approach ensures that the model learns to match each chart with its corresponding semantic insights without the interference of rigid similarity constraints. Subsequently, we apply *contrastive learning* based on the *CLIP framework* to optimize the model. The contrastive loss function is defined as:

$$L_i = -\log \frac{\exp^{(sim(C_i, I_i)/\tau)}}{\sum_{j=1}^{N} \exp^{(sim(C_i, I_j)/\tau)}}, \quad I_i, I_j \in [I^v, I^t, I^s]$$

We trained ChartFinder using 4 A800 GPUs with a learning rate of $1e-6$ for 20 epochs. For the base version of the model, we set the batch size to 512, while for the larger version, we reduced the batch size to 256 due to the increased memory usage associated with the larger model.

3.2.3 Text-to-Chart Retrieval with ChartFinder. Text-to-chart retrieval is a cross-modal task where the goal is to retrieve the most relevant charts from a repository $C = \{C_1, C_2, \ldots, C_n\}$ based on a textual query Q provided by the user. As shown in Figure 3(b), this process involves encoding both the text query and the chart images using text and image encoders, respectively.

Let $\vec{Q}$ represent the embedding of the query, and $\vec{C} = \{\vec{C}_1, \vec{C}_2, \ldots, \vec{C}_n\}$ represent the set of chart embeddings, which are generated by the ChartFinder model. These embeddings capture both the visual content of the charts and the semantic insights associated with them. Once the embeddings are generated, the retrieval process involves computing the similarity scores between the query embedding $\vec{Q}$ and all chart embeddings $\vec{C}$. We calculate the similarity using a suitable similarity function, such as cosine similarity: $\text{sim}(Q, C_i) = \frac{\vec{Q} \cdot \vec{C}_i}{\|\vec{Q}\| \|\vec{C}_i\|}$.

Finally, based on these similarity scores, we retrieve the top-k most relevant charts $\{C_{r_1}, C_{r_2}, \ldots, C_{r_k}\}$, where $\{C_{r_1}, C_{r_2}, \ldots, C_{r_k}\}$ are the charts with the highest similarity to the query Q. These top-k charts are presented to the user as the most relevant results based on the provided textual query.

4 CRBench: The First Text-to-Chart Benchmark

In this section, we first discuss the motivation behind the creation of a text-to-chart benchmark (Section 4.1), followed by an introduction to the benchmark curation process (Section 4.2). Finally, we provide an overview of the key characteristics of the first text-to-chart benchmark CRBench (Section 4.3).

4.1 Motivation and Design Consideration

Motivation. Text-to-chart retrieval requires retrieving relevant charts from a repository based on a user's textual query, which involves aligning both the visual features of charts and the semantic insights embedded within the chart images. Unlike traditional image-based retrieval, which focuses solely on visual content, text-to-chart retrieval must also account for the semantic context conveyed by textual descriptions. Although datasets like VisText [40] and Chart-to-Text [8] exist, they primarily focus on chart captioning and description generation rather than text-to-chart retrieval.

Therefore, there is no text-to-chart retrieval benchmark to evaluate how well models handle this cross-modal retrieval task.

Design Consideration. To fill this gap, we aim to propose a real-world dataset designed for text-to-chart retrieval, enabling the fair comparison of retrieval models across a variety of use cases. Several key factors were considered, as follows.

(1) *Precise and Fuzzy Queries*: As discussed in Section 1, the user queries include both precise queries and fuzzy queries to reflect user behavior in real-world BI applications.

(2) *Diversity of Chart Types*: To reflect the variety of data visualizations encountered in real-world applications, the benchmark includes a broad range of chart types, testing model performance across various chart structures.

(3) *Real-world Charts*: The charts included in the benchmark should be sourced from real-world applications, ensuring their applicability in addressing actual user needs and scenarios.

4.2 Benchmark Construction

To build a benchmark that meets the aforementioned considerations, we first collected a large set of charts from real-world BI applications. Next, we used a chart embedding model to the relevant charts to a given text query. We then employed crowdsourcing to perform preliminary annotations based on these text-chart pairs. Next, GPT-4o was used to generate text queries based on the predefined templates. Finally, we conducted a second round

Table 1: Comparison of Existing Benchmarks. The statistics for Chart-to-Text and VisText are from respective test set. The column "*C/Q*" indicates the ratio for charts (C) and text queries (Q). The column "*Dist. of Query Len.*" Shows the distribution of lengths of text queries in characters. The column "*CT*" means the types of charts.

Datasets	CT	#-Charts	#-Queries	C/Q	Difficulty	Dist. of Query Len.	Samples of Similar Charts
VisText [40]	3	882	1764	0.5	Easy		
Chart-to-Text [8]	6	1393	1393	1	Easy		
CRBench (ours)	11	21,862	326	67	Hard		
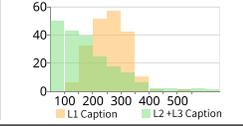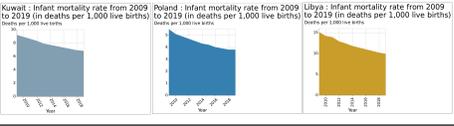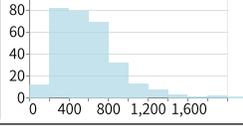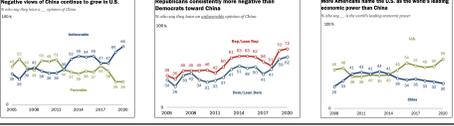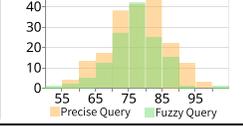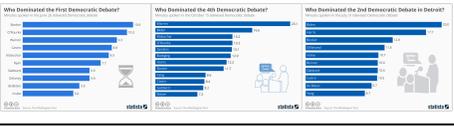
of crowdsourcing to validate the accuracy and relevance of the model-generated text queries.

4.2.1 Charts Collection from Real-world Applications. To ensure CRBench reflects real-world chart retrieval scenarios, we collected charts from reliable sources widely used in business intelligence and data analysis: (i) *Tableau*: A leading data visualization platform for charting and business analytics [39]. (ii) *Statista*: A platform providing statistical data across various sectors [36]. (ii) *Pew Research Center*: A non-profit organization focusing on social science research [31]. (iv) *Our World in Data*: An open-access platform researching global challenges through charts on topics like economics, health, and society [25].

From these sources, we gathered 21,862 charts covering various domains, including finance, business, e-commerce, technology, and more. These fields are closely related to BI applications.

4.2.2 Text Query Generation and Annotation. After collecting charts that meet the BI scenario, our next step is to generate and annotate text queries based on the chart repository.

Step 1: Similar Charts Selection. In the real world, for example, when searching for charts in a BI system, the number of charts should be much larger than the number of queries. Moreover, each query will correspond to many similar charts. Based on this setting, the first step in annotating a query is to find similar charts. Calculating the similarity of rasterized charts is rather difficult, thus we encode the charts into different embeddings and compare their representation similarity.

We use a pre-trained ViT model to encode chart C_i: $\vec{C_i} = ViT(C_i), i \in [1, 21862]$. $\vec{C_i}$ is the embedding vector of the i-th chart, with dimension 1×768. Next, we concatenate all the single embeddings and have all the tensors T: $T = Concat[\vec{C_1}; \vec{C_2}; ...; \vec{C}_{21862}]$. T shapes like 21862×768. After we get all the embedding tensors, we calculate the similarities matrices between these tensors with cosine similarity: $sim = \cos(T, T^T)$. Based on the similarity matrix, we now can easily get the similarity score of candidate C_j and the

target C_i: $S_{ij} = sim(C_i, C_j), i, j \in [1, 21862]$. Afterward, we will set θ as the similarity threshold. Only if $S_{ik} > \theta, \forall k \in [j_1, j_2, j_3, j_4]$. Then, we combine all four candidate charts and the target chart as one chart group: $Group_a = (C_{a1}, C_{a2}, C_{a3}, C_{a4}, C_{a5}), a \in [1, 274]$.

In this step, we set the threshold as 0.90 and produced 247 groups of similar charts.

Step 2: Text Query Annotation by Crowdsourcing. We collect text queries through the Appen [17] crowdsourcing platform based on groups of similar charts collected in Step 1. In this step, we ask workers to write both precise and fuzzy queries for each chart. To ensure the relevance of the text queries, we provide each worker not only with the chart in question but also with the four most similar charts to the first chart. These additional charts serve as references, ensuring that the queries are contextually aligned with the chart's content and not influenced by unrelated charts.

In the first round of crowdsourcing experiments, we assign five users to annotate queries for each grouped chart. However, due to the inherent variability of crowdsourcing and the complexity of the task, we found that some queries were completely unrelated to the chart. Specifically, we observed that 2 or 3 responses in each group of submissions were irrelevant.

Step 3: Text Query Generation by GPT-4o. Building on the query patterns identified in the first 50 crowdsourcing experiments, we used these insights to generate prompt words for GPT-4. The initial crowdsourcing data was found to have significant variability and low quality, which made it less reliable for further use. To improve the query quality, we turned to GPT-4o to generate queries.

Following the approach outlined in Step 2, we presented GPT-4o with five charts at a time and instructed it to generate two queries, but only for the first chart in the set. This method ensures that the queries generated are tailored to each chart's characteristics and use case, addressing the challenges of crowdsourcing quality and providing more relevant and precise queries for chart retrieval.

Table 2: Experiment results on CRBench.

Models	#Param	Precise Query					Fuzzy Query				
		R@1	R@5	R@10	MRR@10	NDCG@10	R@1	R@5	R@10	MRR@10	NDCG@10
VISTA [51]	196M	8.21	20.00	27.18	13.15	16.42	9.16	19.08	24.43	13.16	15.8
Univl-DR [16]	151M	4.10	12.82	15.38	7.84	9.65	3.82	10.69	14.50	6.90	8.70
MARVEL [52]	310M	1.54	5.13	7.69	3.08	4.63	0.00	0.76	1.53	0.46	2.23
CLIP [30]	151M	12.31	26.15	33.33	19.01	22.43	10.69	21.37	25.95	15.69	18.15
CLIP-DPR [16]	151M	9.74	18.46	23.59	13.60	15.95	5.34	13.74	16.79	8.51	10.48
EVA-CLIP [38]	149M	5.13	19.49	24.10	11.08	14.21	3.05	9.92	13.74	6.40	8.14
CoCa [3]	151M	8.72	21.54	25.1	14.11	16.76	3.05	8.40	9.92	5.19	6.33
Jina-CLIP-v2 [9]	865M	4.10	16.92	25.13	10.01	13.58	3.05	10.69	14.50	5.84	7.87
Long-CLIP-B [48]	149M	26.67	53.85	62.56	37.99	43.90	22.90	40.46	51.91	30.29	35.33
ChartFinder-B	149M	38.97	62.05	70.26	49.30	61.30	29.77	50.38	61.83	38.96	44.41
Long-CLIP-L [48]	427M	41.03	75.90	79.49	55.32	55.32	38.93	67.18	74.81	50.99	56.77
ChartFinder	427M	**47.18**	**80.00**	**84.10**	**61.23**	**66.90**	**41.98**	**75.57**	**80.15**	**55.31**	**61.40**

Step 4: Text Query Annotation by Crowdsourcing. After generating a complete set of 247 precise and fuzzy queries, we noticed that, despite clear instructions to ensure the generated queries were relevant to the first chart and distinct from the other charts in the set, some queries still applied to multiple charts. To address this issue, we conducted a second round of crowdsourcing experiments.

In this round, we showed the generated queries along with their corresponding five charts to nine workers. We asked the workers to select the chart that they believed best matched the query. If five or more workers selected the same chart as the target, we considered the query to be unambiguous. After this crowdsourcing screening process, we retained 195 precise queries and 131 fuzzy queries.

4.3 Benchmark Overview

We compare CRBench with two existing benchmarks, Chart-to-Text [8] and VisText [40], both of which focus on chart-related tasks such as caption generation and chart description. In our study, we treat the captions in these benchmarks as queries for retrieval tasks, enabling us to compare their retrieval effectiveness with CRBench. In VisText, each chart is paired with two types of captions: the L1 captions (*e.g.,* basic properties) and the L2+L3 captions (*e.g.,* trends and statistics). This results in a chart-to-query ratio of 1:2. On the other hand, Chart-to-Text pairs each chart with a single caption, resulting in a 1:1 ratio. In contrast, CRBench includes a higher number of charts than queries, yielding a 67:1 ratio.

Table 1 displays the distribution of query lengths in each of these benchmarks. In VisText, the query lengths for L1 and L2+L3 captions range from 100 to 400 characters. Chart-to-Text features a wider range of query lengths, with some reaching up to 2400 In CRBench, the query lengths are more concise, with both 195 precise and 131 fuzzy queries typically ranging from 0 to 100 characters.

We also analyze examples of similar charts from each benchmark. In VisText, similar charts exhibit strong visual resemblance but significant differences in semantic content. In Chart-to-Text, charts that are considered similar align well semantically but differ visually. CRBench, however, features similar charts that are closely aligned both visually and semantically, reflecting a more holistic match.

Overall, CRBench presents a more challenging and realistic benchmark for text-to-chart retrieval tasks.

5 Experiments
5.1 Experimental Settings

Datasets. To fully test the effectiveness of our trained model in chart retrieval, in the experimental part, we will not only compare the retrieval performance of different models on CRBench but also conduct zero-shot experiments on the two datasets: VisText [40] and Chart-to-Text [8].

Metrics. To evaluate the effectiveness of the model in text-to-chart retrieval, we use the following metrics: Recall@1, Recall@5, Recall@10, MRR@10, and NDCG@10.

Methods. We consider several methods: (1) VISTA [51], (2) Univl-dr [16], (3) MARVEL [52], (4) CLIP [30], (5) CLIP-DPR [16], (6) EVA-CLIP [38], (7) CoCa [3], (8) Jina-CLIP-v2 [9], (9) Long-CLIP [48]. All methods encode the query and candidate charts into embeddings, with retrieval performed using cosine similarity via FAISS [7].

Details. Many CLIP-based models apply center cropping during preprocessing to focus on the central part of the image. However, we observed that this operation often crops out crucial chart information, such as the title and axis labels (x-axis and y-axis names). This can negatively impact retrieval performance, introducing bias in the model's ability to understand key chart features. To address this, we removed the center crop operation during testing and instead resized the images directly to a shape suitable for the model's vision encoder input. This adjustment ensures that important chart elements remain intact, reducing potential bias and improving the retrieval process. We applied this image processing strategy consistently across all methods we evaluated.

5.2 Experiments on CRBench

The purpose of this experiment is to evaluate and compare the performance of different models in precise and fuzzy query tasks.

Table 3: Experimental results on VisText.

Models	L1 Caption					L2+L3 Caption				
	R@1	R@5	R@10	MRR@10	NDCG@10	R@1	R@5	R@10	MRR@10	NDCG@10
VISTA [51]	75.85	87.76	89.80	79.85	82.27	44.56	63.95	70.86	37.61	41.80
Univl-DR [16]	63.72	81.75	86.62	71.57	75.22	34.58	52.83	60.77	42.61	46.93
MARVEL [52]	47.51	66.55	72.79	55.61	59.73	24.04	39.80	48.64	31.25	35.37
CLIP [30]	77.44	89.46	92.74	83.05	85.41	48.41	68.25	74.83	56.81	61.14
CLIP-DPR [16]	74.26	88.10	90.93	80.59	83.14	45.01	64.74	72.11	53.51	57.97
EVA-CLIP [38]	71.77	87.07	89.68	78.50	81.26	31.86	49.43	57.26	39.39	43.64
CoCa [3]	75.62	84.35	87.87	79.69	81.65	44.22	58.96	66.44	50.54	54.30
Jina-CLIP-v2 [9]	79.71	90.93	94.67	84.85	87.23	54.20	74.49	80.50	62.72	67.01
Long-CLIP-B [48]	90.48	94.78	95.35	81.88	83.50	60.20	78.12	83.45	67.99	71.73
ChartFinder-B	93.20	97.85	98.75	95.17	96.05	62.59	79.48	84.01	69.64	73.11
Long-CLIP-L [48]	93.20	98.07	99.09	95.21	96.15	66.55	82.99	88.66	73.43	77.09
ChartFinder	**96.15**	**99.43**	**99.66**	**96.90**	**97.44**	**67.80**	**84.58**	**89.12**	**75.20**	**78.78**

Table 4: Experimental results on Chart-to-Text.

Models	R@1	R@5	R@10	MRR@10	NDCG@10
VISTA [51]	51.72	68.46	73.20	58.94	62.39
Univl-DR [16]	49.61	67.91	73.08	57.60	61.35
MARVEL [52]	21.68	38.19	46.30	28.97	33.08
CLIP [30]	74.01	88.16	92.03	80.25	83.11
CLIP-DPR [16]	66.04	82.56	85.86	73.09	76.21
EVA-CLIP [38]	68.05	83.42	87.80	74.73	77.90
CoCa [3]	67.98	83.92	87.65	74.49	77.92
Jina-CLIP-v2 [9]	65.83	83.78	88.23	73.51	77.08
Long-CLIP-B [48]	90.81	98.28	98.85	94.18	95.35
ChartFinder-B	94.90	99.07	99.43	96.89	97.53
Long-CLIP-L [48]	94.97	99.35	99.78	97.06	97.75
ChartFinder	**97.06**	**99.86**	**99.86**	**98.41**	**98.80**

Experimental results are shown in Table 2. From the experimental results, it can be seen that ChartFinder and ChartFinder-B perform best in both query tasks, and multiple metrics are ahead of other methods by more than 5% or even 10%. For example, in the precise query task, ChartFinder's R@1, R@5, and R@10 are 47.18%, 80.00%, and 84.10%, respectively, significantly higher than the second-place Long-CLIP-L's 41.03%, 75.90%, and 79.49%. In the fuzzy query task, ChartFinder also surpasses Long-CLIP-L, with R@1, R@5, and R@10 being 3.05%, 8.39%, and 5.34% higher, respectively. In comparison, although Jina-CLIP-v2 (865M) has the largest number of parameters, it performs poorly in both types of tasks. The R@1 in the precise query task is only 4.10%, and the R@1 in the fuzzy query task is also only 3.05%.

Overall, ChartFinder and ChartFinder-B perform significantly in precise and fuzzy query tasks, leading other methods by more than 5% or 10% in multiple metrics.

5.3 Zero-Shot Experiments

To verify the generalization ability of ChartFinder, we conduct zero-shot experiments on the VisText and Chart-to-Text datasets.

5.3.1 Experiments on VisText Benchmark. We use the test set of VisText [40], comprising 882 charts with each chart corresponding to two captions mentioned in section 4.3, to conduct this group of experiments. We take the two-level caption as the query, and the annotated chart as the retrieval target to perform text-to-chart retrieval tasks.

Overall Results. As shown in Table 3, ChartFinder performs best in all metricss in L1 caption, significantly ahead of other models. For example, Recall@1 reaches 96.15%, far exceeding other models. Recall@10 reaches 99.66%, close to a full match. MRR@10 and NDCG@10 are 96.90% and 97.44%, respectively, further verifying their advantage in ranking quality. ChartFinder-B still achieves excellent performance with fewer parameters. Recall@1 reaches 93.20%, which is comparable to Long-CLIP-L (93.20%) and about 3 percentage points higher than Long-CLIP-B. On Recall@10, ChartFinder-B reaches 98.85%, surpassing all models with similar parameter scales. L2+L3 captions are more inclined to detailed analytical descriptions. Thus, the retrieval difficulty is significantly higher than the L1 task, and the overall metrics have declined. ChartFinder still leads in this task with R@1 reaching 67.80%, R@10 reaching 89.12%, MRR@10 and NDCG@10 reaching 75.20% and 78.78% respectively.

5.3.2 Experiments on Chart-to-Text Benchmark. We use Chart-to-Text dataset [8] to conduct this experiment. Specifically, this dataset has 1,393 text and chart pairs in the test set. We use the same metrics as the Section 5.3.1.

Overall Results. As shown in Table 4, ChartFinder significantly outperforms other models in all metrics. For example, Recall@1 reaches 97.06%, 2.09% higher than the second-best model (Long-CLIP-L, 94.97%). Recall@10 reaches 99.86%, almost close to the full score, showing the excellent performance of the model in the top-k retrieval task. MRR@10 and NDCG@10 are 98.41% and 98.80%, respectively, further verifying the effectiveness of ChartFinder in ranking quality. ChartFinder-B also demonstrates obvious performance, surpassing Long-CLIP-B in all metrics, especially in Recall@1 (94.90% vs. 90.81%) and NDCG@10 (97.53% vs. 95.35%). The above results further prove the effectiveness of ChartFinder.
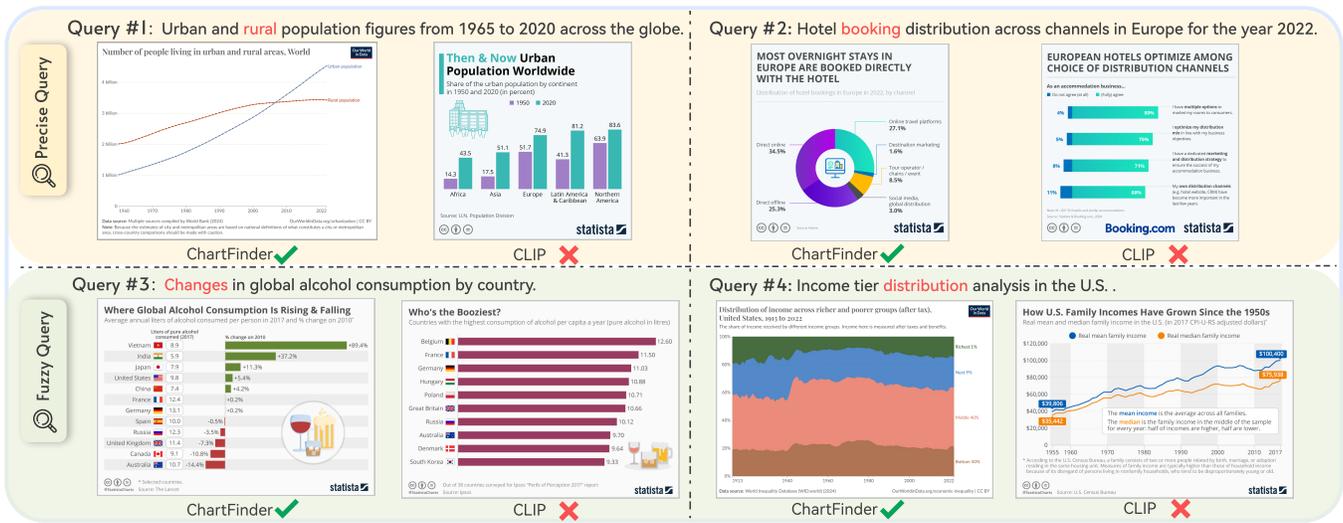
Figure 4: Case study on Top-1 text-to-chart retrieval results of the ChartFinder and CLIP.

5.4 Case Study

Figure 4 compares the top-1 retrieval performance of our model and CLIP on CRBench. In Query #1, ChartFinder successfully retrieves a chart that includes both *urban* and *rural population*, while CLIP retrieves a chart focusing only on *urban population*, failing to meet the query requirements. In Query #2, ChartFinder correctly aligns with the semantic meaning of *booking distribution*, retrieving a pie chart about hotel booking channels. In contrast, CLIP retrieves a chart about optimizing distribution channels, which does not match the query's focus. In Query #3, ChartFinder captures the keyword *change* and retrieves a chart showing trends in alcohol consumption. In contrast, CLIP retrieves a chart ranking alcohol consumption, ignoring the notion of *change*. In Query #4, ChartFinder identifies *distribution* and retrieves a stacked area chart illustrating income distribution. In contrast, CLIP retrieves a chart focusing on income growth trends, failing to address the *distribution* aspect.

6 Related Work

Text-to-Chart Retrieval. Previous work on text-to-chart chart retrieval has primarily focused on two approaches: visualization recommendation systems [27, 28, 42, 46] and natural language interfaces [20–22, 33, 41]. In the context of natural language interfaces, systems like Olio [32] and BOLT [35] allow users to express their analytical intents or design exploration using natural language for chart retrieval. On the other hand, visualization recommendation systems such as SlopeSeeker [1] and Snowy [34] focus on guiding users in exploring and discovering interesting patterns and charts within the data. Although much progress has been made in these two approaches, they still struggle to understand the user's query intent. We believe embedding the query and charts into the same latent space will shine a light on Text-to-Chart Retrieval. With the development of MLLM [12, 13, 44, 47] models, we can see text-to-chart as a cross-modal retrieval task, which has been driven by large-scale image-text paired datasets and pre-trained models like CLIP [30]. Several variants have enhanced CLIP's performance, such as EVA-CLIP [38], which improves retrieval by using EVA [6] as the visual encoder, and Jina-CLIP-v2 [9, 45] which enhances the text encoder [37] for better retrieval. Other approaches, like UniVL-DR [16], MagicLens [49], VISTA [51], and MARVEL [52], fine-tune CLIP on large, diverse datasets to improve model's retrieval abilities. However, MLLM-based methods have limited research focused on chart retrieval. This paper addresses this gap by injecting semantic insights into ChartFinder based on contrastive learning.

Captions with Semantic Insights. Studies show that rich captions can significantly enhance CLIP-based models' performance [14]. For instance, VeCLIP [10] boosts performance by injecting extra visual information into captions, while DreamLIP [50] achieves the same effect as CLIP 300M data using just one-tenth of the long caption data. Another approach to enrich semantic information is to use multiple captions for the same image. Llip [11] suggests that each image should have captions from various angles. VisText [40] introduces a dataset for statistical charts with two levels of semantic information per chart. We extend this idea by proposing three levels of semantic insights to reflect the semantic depth of charts. Moreover, while VisText's manually annotated data is limited, we generate semantic insights using Llama 3.1 [5] 70B based on metadata, ensuring both quality and quantity.

7 Conclusion

In this paper, we introduced ChartFinder, a novel text-to-chart retrieval model that combines contrastive learning with the CLIP architecture to align both visual and semantic features of charts. By leveraging synthesized semantic insights of charts, our model effectively retrieves relevant charts based on a user's textual query. We also presented CRBench, the first benchmark for text-to-chart retrieval in real-world BI scenarios, which enables the fair evaluation of retrieval models. Experiments show that ChartFinder outperforms existing methods, achieving significant improvements in both precise (up to 66.9% NDCG@10) and fuzzy query tasks.

References

[1] Alexander Bendeck, Dennis Bromley, and Vidya Setlur. 2024. SlopeSeeker: A Search Tool for Exploring a Dataset of Quantifiable Trends. In *Proceedings of the 29th International Conference on Intelligent User Interfaces*. 817–836.

[2] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. 2025. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*. Springer.

[3] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. 2023. Reproducible Scaling Laws for Contrastive Language-Image Learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2818–2829. https://doi.org/10.1109/cvpr52729.2023.00276

[4] Alexey Dosovitskiy. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[5] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[6] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. 2024. Eva-02: A visual representation for neon genesis. *Image and Vision Computing* 149 (2024), 105171.

[7] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. arXiv:1702.08734 [cs.CV] https://arxiv.org/abs/1702.08734

[8] Shankar Kantharaj, Rixie Tiffany Ko Leong, Xiang Lin, Ahmed Masry, Megh Thakkar, Enamul Hoque, and Shafiq Joty. 2022. Chart-to-text: A large-scale benchmark for chart summarization. *arXiv preprint arXiv:2203.06486* (2022).

[9] Andreas Koukounas, Georgios Mastrapas, Bo Wang, Mohammad Kalim Akram, Sedigheh Eslami, Michael Günther, Isabelle Mohr, Saba Sturua, Scott Martens, Nan Wang, et al. 2024. jina-clip-v2: Multilingual Multimodal Embeddings for Text and Images. *arXiv preprint arXiv:2412.08902* (2024).

[10] Zhengfeng Lai, Haotian Zhang, Bowen Zhang, Wentao Wu, Haoping Bai, Aleksei Timofeev, Xianzhi Du, Zhe Gan, Jiulong Shan, Chen-Nee Chuah, et al. 2025. Veclip: Improving clip training via visual-enriched captions. In *European Conference on Computer Vision*. Springer, 111–127.

[11] Samuel Lavoie, Polina Kirichenko, Mark Ibrahim, Mahmoud Assran, Andrew Gordon Wilson, Aaron Courville, and Nicolas Ballas. 2024. Modeling caption diversity in contrastive vision-language pretraining. *arXiv preprint arXiv:2405.00740* (2024).

[12] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*. PMLR, 19730–19742.

[13] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*. PMLR, 12888–12900.

[14] Xianhang Li, Haoqin Tu, Mude Hui, Zeyu Wang, Bingchen Zhao, Junfei Xiao, Sucheng Ren, Jieru Mei, Qing Liu, Huangjie Zheng, et al. 2024. What If We Recaption Billions of Web Images with LLaMA-3? *arXiv preprint:2406.08478* (2024).

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer.

[16] Zhenghao Liu, Chenyan Xiong, Yuanhuiyi Lv, Zhiyuan Liu, and Ge Yu. 2022. Universal vision-language dense retrieval: Learning a unified representation space for multi-modal retrieval. *arXiv preprint arXiv:2209.00179* (2022).

[17] Appen Ltd. 2009. https://www.appen.com/

[18] Yuyu Luo, Xuedi Qin, Chengliang Chai, Nan Tang, Guoliang Li, and Wenbo Li. 2022. Steerable Self-Driving Data Visualization. *IEEE Trans. Knowl. Data Eng.* 34, 1 (2022), 475–490.

[19] Yuyu Luo, Xuedi Qin, Nan Tang, and Guoliang Li. 2018. Deepeye: Towards automatic data visualization. In *2018 IEEE 34th international conference on data engineering (ICDE)*. IEEE, 101–112.

[20] Yuyu Luo, Xuedi Qin, Nan Tang, Guoliang Li, and Xinran Wang. 2018. DeepEye: Creating Good Data Visualizations by Keyword Search. In *SIGMOD Conference*. ACM, 1733–1736.

[21] Yuyu Luo, Nan Tang, Guoliang Li, Chengliang Chai, Wenbo Li, and Xuedi Qin. 2021. Synthesizing Natural Language to Visualization (NL2VIS) Benchmarks from NL2SQL Benchmarks. In *SIGMOD Conference*. ACM, 1235–1247.

[22] Yuyu Luo, Nan Tang, Guoliang Li, Jiawei Tang, Chengliang Chai, and Xuedi Qin. 2022. Natural Language to Visualization by Neural Machine Translation. *IEEE Trans. Vis. Comput. Graph.* 28, 1 (2022), 217–226.

[23] Yuyu Luo, Yihui Zhou, Nan Tang, Guoliang Li, Chengliang Chai, and Leixian Shen. 2023. Learned Data-aware Image Representations of Line Charts for Similarity Search. *Proc. ACM Manag. Data* 1, 1 (2023), 88:1–88:29.

[24] Pingchuan Ma, Rui Ding, Shi Han, and Dongmei Zhang. 2021. MetaInsight: Automatic Discovery of Structured Knowledge for Exploratory Data Analysis. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD '21)*. Association for Computing Machinery, New York, NY, USA, 1262–1274. https://doi.org/10.1145/3448016.3457267

[25] OWID. 2011. Our World In Data. https://ourworldindata.org/

[26] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*. 2641–2649.

[27] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. 2018. DeepEye: An automatic big data visualization framework. *Big Data Min. Anal.* 1, 1 (2018), 75–82.

[28] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. 2018. DeepEye: Visualizing Your Data by Keyword Search. In *EDBT*. OpenProceedings.org, 441–444.

[29] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. 2020. Making data visualization more efficient and effective: a survey. *VLDB J.* 29, 1 (2020), 93–117.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[31] Pew Research. 2004. Pew Research Public. https://www.pewresearch.org/

[32] Vidya Setlur, Andriy Kanyuka, and Arjun Srinivasan. 2023. Olio: A Semantic Search Interface for Data Repositories. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–16.

[33] Leixian Shen, Enya Shen, Yuyu Luo, Xiaocong Yang, Xuming Hu, Xiongshuai Zhang, Zhiwei Tai, and Jianmin Wang. 2022. Towards natural language interfaces for data visualization: A survey. *IEEE transactions on visualization and computer graphics* 29, 6 (2022), 3121–3144.

[34] Arjun Srinivasan and Vidya Setlur. 2021. Snowy: Recommending utterances for conversational visual analysis. In *The 34th annual ACM symposium on user interface software and technology*. 864–880.

[35] Arjun Srinivasan and Vidya Setlur. 2023. BOLT: A Natural Language Interface for Dashboard Authoring.. In *EuroVis (Short Papers)*. 7–11.

[36] statista. 2007. Chart of the Day. https://www.statista.com/chartoftheday

[37] Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, et al. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *arXiv preprint arXiv:2409.10173* (2024).

[38] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. 2023. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint: 2303.15389* (2023).

[39] Tableau. 2003. Tableau Public. Website. https://public.tableau.com/app/discover.

[40] Benny J Tang, Angie Boggust, and Arvind Satyanarayan. 2023. Vistext: A benchmark for semantically rich chart captioning. *arXiv preprint: 2307.05356* (2023).

[41] Jiawei Tang, Yuyu Luo, Mourad Ouzzani, Guoliang Li, and Hongyang Chen. 2022. Sevi: Speech-to-Visualization through Neural Machine Translation. In *SIGMOD Conference*. ACM, 2353–2356.

[42] Manasi Vartak, Silu Huang, Tarique Siddiqui, Samuel Madden, and Aditya Parameswaran. 2017. Towards visualization recommendation systems. *Acm Sigmod Record* 45, 4 (2017), 34–39.

[43] Yifan Wu, Lutao Yan, Leixian Shen, Yunhai Wang, Nan Tang, and Yuyu Luo. 2024. Chartinsights: Evaluating multimodal large language models for low-level chart question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. 12174–12200.

[44] Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. 2024. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4818–4829.

[45] Han Xiao, Georgios Mastrapas, and Bo Wang. 2024. Jina CLIP: Your CLIP model is also your text retriever. In *Multi-modal Foundation Model meets Embodied AI Workshop@ ICML2024*.

[46] Yilin Ye, Jianing Hao, Yihan Hou, Zhan Wang, Shishi Xiao, Yuyu Luo, and Wei Zeng. 2024. Generative AI for visualization: State of the art and future directions. *Vis. Informatics* 8, 1 (2024), 43–66.

[47] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. 2021. Florence: A new foundation model for computer vision. *arXiv preprint: 2111.11432* (2021).

[48] Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. 2025. Long-clip: Unlocking the long-text capability of clip. In *European Conference on Computer Vision*. Springer, 310–325.

[49] Kai Zhang, Yi Luan, Hexiang Hu, Kenton Lee, Siyuan Qiao, Wenhu Chen, Yu Su, and Ming-Wei Chang. 2024. Magiclens: Self-supervised image retrieval with open-ended instructions. *arXiv preprint arXiv:2403.19651* (2024).

[50] Kecheng Zheng, Yifei Zhang, Wei Wu, Fan Lu, Shuailei Ma, Xin Jin, Wei Chen, and Yujun Shen. 2025. Dreamlip: Language-image pre-training with long captions. In *European Conference on Computer Vision*. Springer, 73–90.

[51] Junjie Zhou, Zheng Liu, Shitao Xiao, Bo Zhao, and Yongping Xiong. 2024. VISTA: Visualized Text Embedding For Universal Multi-Modal Retrieval. *arXiv preprint arXiv:2406.04292* (2024).

[52] Tianshuo Zhou, Sen Mei, Xinze Li, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, Yu Gu, and Ge Yu. 2024. MARVEL: Unlocking the Multi-Modal Capability of Dense Retrieval via Visual Module Plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 14608–14624.