LEVA: Using Large Language Models to Enhance Visual Analytics

Yuheng Zhao, Yixing Zhang, Yu Zhang, Xinyi Zhao, Junjie Wang,
Zekai Shao, Cagatay Turkay, Siming Chen

Abstract—Visual analytics supports data analysis tasks within complex domain problems. However, due to the richness of data types, visual designs, and interaction designs, users need to recall and process a significant amount of information when they visually analyze data. These challenges emphasize the need for more intelligent visual analytics methods. Large language models have demonstrated the ability to interpret various forms of textual data, offering the potential to facilitate intelligent support for visual analytics. We propose LEVA, a framework that uses large language models to enhance users' VA workflows at multiple stages: onboarding, exploration, and summarization. To support onboarding, we use large language models to interpret visualization designs and view relationships based on system specifications. For exploration, we use large language models to recommend insights based on the analysis of system status and data to facilitate mixed-initiative exploration. For summarization, we present a selective reporting strategy to retrace analysis history through a stream visualization and generate insight reports with the help of large language models. We demonstrate how LEVA can be integrated into existing visual analytics systems. Two usage scenarios and a user study suggest that LEVA effectively aids users in conducting visual analytics.

Index Terms—Insight recommendation, mixed-initiative, interface agent, large language models, visual analytics

1 Introduction

Visual analytics (VA) combines data analysis techniques with visualizations for effective understanding, reasoning and decision-making on the basis of large and complex datasets [21], [22], [55]. However, the VA processes often require significant effort on the side of the user, resulting in a less efficient data interpretation and analysis process [58]. Several challenges underpin this inefficiency: a steep learning curve with VA systems' onboarding [50], a tendency to lose direction in data exploration [29], and the difficulty of summarizing final insights [8]. Such issues emphasize the need for a more intelligent approach to VA.

To specify the challenges further: firstly, the challenge of onboarding stems from unfamiliarity with a VA system, leading to reduced efficiency in grasping visual mappings and interactions [5], [41]. Secondly, during data exploration, the numerous avenues to analyze data and the lack of structured guidance and direction make it difficult to uncover insights [6]. Thirdly, keeping track of and aggregating key findings is often difficult and summarizing insights is a time-consuming task that demands significant effort and attention to detail [3], [33]. These challenges highlight the need for an intelligent framework to support VA processes and foster efficacy across these pivotal stages.

To address these problems, researchers focus on enhancing VA through visualization onboarding [49], interaction recommendation and guidance [4], [29], [47], and result summarization [8], [44]. However, these methods often do not take advantage of advancements in intelligent algorithms and are not easily adaptable to various VA systems. There is a lack of generalizable intelligent approaches that can work across different VA systems.

The Large Language Models (LLMs) exhibit broader knowledge and problem-solving abilities, making it possible to address the above challenges in the three stages of VA. Firstly, as LLMs can interpret visualizations' declarative grammar [13], [35], we can try to propose the grammar for VA systems and use it as input for LLMs to generate onboarding tutorials. Additionally, LLMs can process various data types [39], making it possible to recommend insights by analyzing both the system status and underlying data, assisting users' exploration. Finally, LLMs exhibit powerful summarization capabilities [17], which may help to summarize the exploration process and generate a report with rich forms.

In this paper, we propose a framework named LEVA (**L**LM-**E**nhanced **V**isual **A**nalytics) that uses LLMs to enhance visual analytics in three stages of the workflow. In the onboarding phase, we provide a solution that enables LLMs to interpret visualizations in each of the views and these views' relationships based on a specification of the VA system. This enables the flexible creation of tutorials for various VA systems. In the exploration phase, we design an insight recommendation strategy that guides LLMs in recommending insights based on the understanding of the system, analytical task, user's interaction, and data to facilitate mixed-initiative exploration [18]. The methodology incorporates a two-step process, including the selection of insight types and assessment of executed insights. Additionally, we integrate insights in the original VA system as annotations instead of textual descriptions, making communication between the LLM and end-users more intuitive. In the summarization phase, LEVA facilitates the user to retrace the analytical history via an interactive stream visualization, allowing the selection of an analytical path for report generation.

- *Yuheng Zhao, Yixing Zhang, Xinyi Zhao, Zekai Shao, Siming Chen are with School of Data Science, Fudan University. E-mail: {yuhengzhao, xinyizhao19, zkshao19, simingchen}@fudan.edu.cn, {yixingzhang23, wangjj23}@m.fudan.edu.cn.*
- *Yu Zhang is with Department of Computer Science, University of Oxford. E-mail: yuzhang94@outlook.com.*
- *Cagatay Turkay is with University of Warwick. E-mail: Cagatay.Turkay@warwick.ac.uk.*
- *Siming Chen and Yu Zhang are the corresponding authors.*

Manuscript received April 19, 2005; revised August 26, 2015.

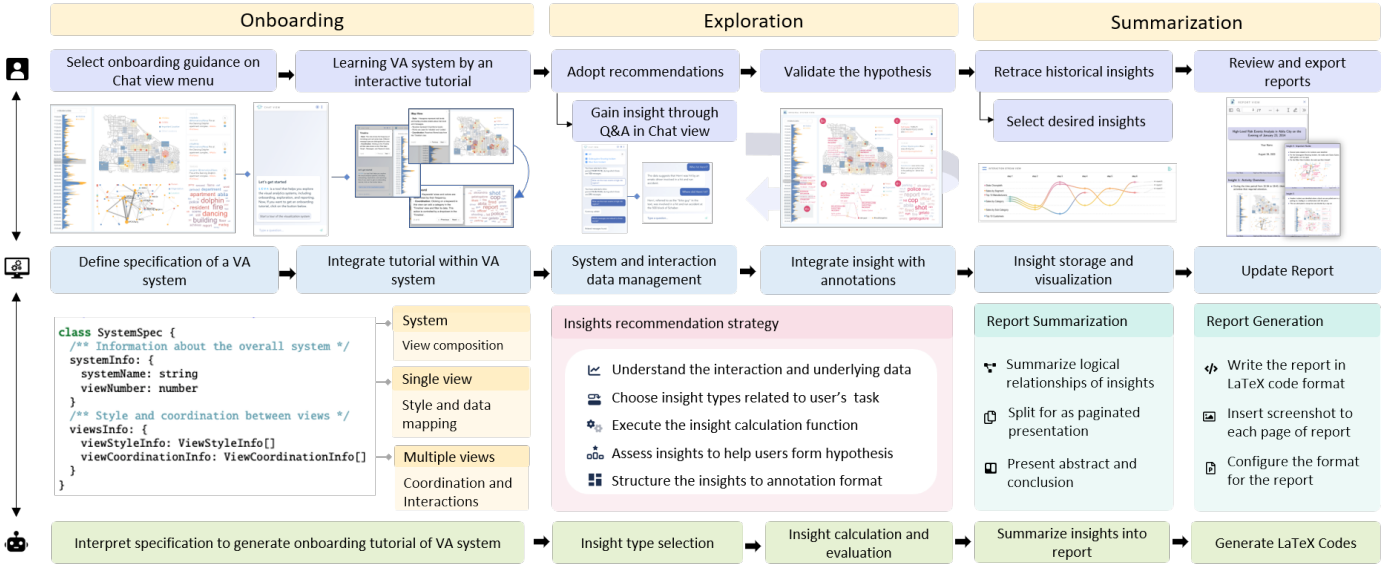


Fig. 1. The LEVA framework proposes strategies for leveraging LLMs to enhance visual analytics workflows, starting from onboarding and exploration to summarization. The architecture (middle) connects analysts (top) and LLMs (bottom) to achieve mixed-initiative exploration through interactive interfaces and guidance strategies of LLMs.

Our strategy involves a report generation method where the LLMs synthesize visualization images and explanations to produce a report using LaTeX code. We have developed two integrated systems combining our framework with two original VA systems. The interface includes a chat view, original system view, interaction stream view, and report view. To the best of our knowledge, LEVA is the first attempt to embed LLMs in complex visual analytics workflows to support users in various analysis stages. Our main contributions are as follows:

- We propose a framework, LEVA, for using LLMs to enhance mixed-initiative exploration in three stages of users' VA workflow.
- We demonstrate how LEVA can be implemented in existing VA systems to facilitate visual analytics by enabling a connection between users, interface and LLMs.
- We report observations and learnings from two usage scenarios and a user study to demonstrate the effectiveness of our framework.

2 Related Work

This section reviews the literature that aims to enhance VA in different stages of users' workflow and the literature using LLMs for VA to examine its abilities.

2.1 Onboarding in Visual Analytics

Visualization onboarding is the process of supporting users in reading, interpreting, and extracting information from visual representations of data [50]. Stoiber et al. [49] found that a VA system may lack low-level information about the data, such as understanding principles of the specific data format, data types, or data structure, which limits data selection and manipulation [22]. Vaishali et al. [12] found that non-expert users often lack visualization literacy to interpret the data and understand the interactions with and between visualizations in a dashboard. They pointed out that it is necessary to bridge the knowledge gap between the system and the user's background before exploring the data. Previous research conducted various onboarding strategies for VA systems. The onboarding form mainly includes textual descriptions, video-based, and step-by-step tours with tooltips and overlays [59]. Kwon et al. [23] conducted a user study to compare these methods and found that an interactive tour is better than others with a more engaging experience. Previous studies have shown that onboarding tools can help users better understand a VA system. However, these tools are limited in their ability to be easily integrated into different systems. To address this, we proposed an intelligent interactive onboarding method to generate tutorials based on the unified grammar of VA systems.

2.2 Insights Recommendation in Visual Analytics

Insight recommendations within visualization methodologies have garnered attention in recent VA research, serving as effective instruments to aid users in their analytical tasks. These methods can broadly be stratified into recommendations encompassing annotations or captions, interactions, and direct visualizations.

A corpus of research has been proposed on generating single visualization [19], [37], [42] or multiple-view visualizations [11]. Typically, these studies deal with data table queries, outputting static visualizations. In contrast, our approach adapts to a VA system, producing an enhanced VA system complete with interactive insight annotations. A separate thread of research, exemplified by Lai et al. [24] and Liu et al. [32], aims at appending annotations to visualizations. However, their emphasis is limited to single-view visualizations without an overarching framework for comprehensive VA systems.

Recognizing the complexity of VA systems, some guidance theories have been proposed [7], [47]. Ceneda et al. [6] characterized interaction guidance along the knowledge gap of the user, the input and output of the guidance generation process, and the degree of guidance that is provided to users. A commonly used implementation method is modeling interaction sequences to predict the next interaction object [29]. While this strategy offers a diverse set of suggestions, it needs to collect large amounts of interaction data, grapple with issues of explainability,

and be applied to specific VA systems. Instead of relying on interaction data, we mine insights directly from data. We let LLMs understand VA systems and dispatch tasks for computation functions. This approach not only ensures explainability but also fosters adaptability to diverse VA systems.

2.3 Insights Summarization in Visual Analytics

In data analysis, the task of translating insights into comprehensive reports necessitates substantial effort from the user for documentation and summarization. Prior research such as DataShot [57] introduced methods for automatic report generation, enabling the derivation of insight-driven reports directly from data along with visuals attached. Li et al. [27] refined this concept by encapsulating data insights within automatically generated notebooks. However, these methods do not support summarization in VA systems. Chen et al. [8] innovated a storytelling approach, incorporating recording and editing utilities in VA. VisInReport [44] offered a tool that fosters manual discourse transcript analysis, and curates reports contingent on user interactions. However, these methods do not automate the visual examination of exploration routes; obtaining insights often requires manual review and refinement. Liu et al. [33] designed an analytical graph depicting the journey of exploratory data analysis. Drawing inspiration from this, we fuse a stream visualization to endorse real-time retrospectives. Our novel methodology harnesses LLMs to discern pertinent records, facilitating automatic summation and report creation.

2.4 Large Language Models for Visualization

There has been a rising interest in employing language models for various downstream applications. Categorizing based on the format of inputs and outputs, LLMs can process code, declarative syntax, natural language, and structured data.

Related work in the visualization domain focuses on understanding or generating visualizations. A typical application is using natural language to generate data visualizations through an interface [45]. Moreover, codes [36] or declarative grammar for visualizations [13], [35], [38] is often used as input or output as a simplified form of code. This type of work demonstrates code comprehension of language models and basic knowledge of visualizations. Inspired by this, we propose to use declarative grammar to represent VA systems, which are essentially combinations of multiple visualizations.

In addition to visualization tasks, Language models can also analyze data [61] or act as agents to use tools with APIs [43]. Thus, we propose that in addition to using LLMs to process data, some complex computation methods can be used as additional tools, enabling support for a wider range of VA analysis scenarios. The language model is proficient in generating text with format [39]. For example, Liu et al. [34] adopt LLMs for generating semantic input texts based on GUI context. We leverage this prowess to generate and integrate tutorials, insights, and reports to enhance user experience in using VA. Finally, the language model supports dialogue mode, which allows users to ask and answer questions freely, as well as decompose long tasks to support more complex analysis tasks.

To conclude, the development of LLMs has opened up opportunities to enhance VA. Our work is the first trial towards a comprehensive integration of LLMs for intelligent VA.

3 MOTIVATION

Interpreting and analyzing data with VA often requires significant user effort [58]. Various challenges lead to such inefficient analysis, such as the steep learning curve associated with onboarding VA systems [50], getting sidetracked during data exploration [29], and difficulty summarizing the final insights [8]. These challenges underscore the necessity for a more intelligent approach to VA. Combining a review of previous studies, we summarized the following design considerations and propose the design requirements aimed at enhancing the user experience in VA.

3.1 Design Considerations

From our analysis of the VA literature, a recurring concern emerges: despite the advances, users still have to expend excessive effort in various stages of analysis. In each of these stages, we identify and dissect specific challenges that intensify the effort users must exert:

- **C1 System onboarding:** Before users can extract value from VA, they must first grasp the nuances of the system's design. However, two main challenges hinder this. Firstly, although VA systems are often designed with a target user group in mind, the data and visual encoding displayed in visualization might not always align with the users' background. Ambiguities in data information, such as the meaning of data, structure, types, and transformation, may lead to misinterpretations [22]. Secondly, a lack of visualization literacy among users can complicate the interpretation of visual data, making the initial stage of data analysis more laborious than it should be [49]. This emphasizes the need for a more efficient onboarding process that not only caters to users but also provides clear and intuitive information to smoothen the transition and usage.
- **C2 Insights exploration:** When embarking on the journey of insight discovery, users are often bogged down by interactively analyzing complex data and visualization, which is time-consuming [16]. On top of this, the multi-faceted nature of insights means forming hypotheses requires finding parts of many insights that are relevant to the task and then validating them by interacting to find insights in new states, making the exploration stage more challenging [2]. Given these complexities, there emerges a need for VA systems to automatically extract and evaluate insights with users' tasks in mind, ensuring more focused and efficient data analysis.
- **C3 Results summarization:** Summarizing and reporting findings is a final, yet essential step in VA systems. However, it's a process that is often unduly laborious. Users typically navigate through multiple iterative explorations to validate hypotheses [8]. Yet, not every exploration results in valuable knowledge [44]. As a result, analysts frequently spend significant time and effort sifting through, distilling, and manually drafting reports from their interactive exploration outcomes. This highlights the pressing need for automated capturing of findings and allows users to trace back. Furthermore, such automation should provide users with comprehensive illustrated reports [27], sparing them the chore of crafting them themselves.

3.2 Design Requirements

Drawing insights from these design considerations, we have developed a set of targeted design requirements. These requirements are pivotal for enhancing VA, aiming to address the challenges identified earlier.

R1 Visual encoding: To address the barriers of system comprehension and misinterpretations (C1), it's vital that we offer an onboarding tutorial to help users understand how data and visualization are mapped. This could include providing data definitions and encodings.

R2 Interaction and coordination: Considering the challenges users face in discerning the usage of visualizations in a VA system (C1), we should provide an interactive step-by-step tutorial of each view. The content needs to clarify interactions and the relationship between views.

R3 Insight discovery: In light of the repetitive analytical tasks users undergo and the depth of domain knowledge required (C2), our framework should provide automated data analysis that helps users discover diversified insights. This may include analysis for different data types.

R4 Hypothesis formulation and validation: Given that exploration often occurs in an iterative process (C2), our framework should enable the discovery of insights in constantly updated statuses to facilitate hypothesis formulation and validation.

R5 Summarization of exploration results: Considering the labor-intensive nature of the summarization process (C3), our framework should prioritize aiding users in filtering and summarizing their exploration outcomes. This entails the management and visualization of historical insights as well as the capability to generate comprehensive reports.

3.3 LLM-Enhanced User Workflow in Visual Analytics

Our objective is to employ intelligent VA to address the challenges faced by analysts when using VA systems. However, creating distinct models for each exploration stage, workflow, chart type, data type, and domain is labor-intensive and difficult to scale, requiring general frameworks and models. Large language models have emerged for various analytical tasks, offering the potential for continuous assistance throughout the workflow. Thus, we propose a framework that leverages LLMs to enhance visual analytics in three stages of the workflow (Fig. 2). Based on the analysis of considerations and requirements, we summarize the integration into three stages: **onboarding**, **exploration**, insight recommendation in **exploration**, and **summarization** for selective reporting.

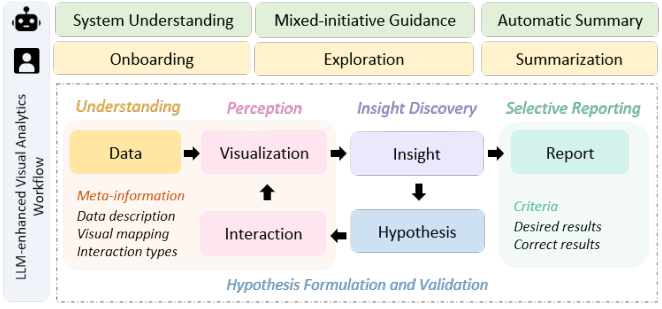


Fig. 2. The LLM-enhanced visual analytics workflow shows how LLMs contribute to the progress of the analysis. The LLMs support visualization understanding, mixed-initiative guidance, and automatic summary while users experience onboarding, exploration, and summarization. Onboarding refers to data understanding, visualization, and interaction perception. Exploration refers to insight discovery, hypothesis formulation, and validation. Summarization refers to selective reporting.

To further detail how we intend to tackle these challenges, we outline the roles and purposes of the LLM-enhanced system. The objective of the LLM-enhanced system is to alleviate the challenges users confront in the three stages. The end-users of this enhanced system are analysts or individuals who frequently work with VA systems. On the other hand, the LEVA implementation itself is intended for design and development professionals who aim to create or augment VA systems. The primary users of LEVA are system developers and designers who wish to leverage large language models to enhance the capabilities of their VA tools.

4 LEVA Framework

In this section, we will introduce our framework (Fig. 1). To enhance the VA workflow, we carefully identify the important steps of users in using visual analytics and propose LEVA to boost the analysis efficiency and enrich the insight exploration (Sec. 3.3). The architecture connects analysts and LLMs to achieve mixed-initiative exploration. In our framework, the original system is a visual analysis system, while our LEVA-enhanced system is added with natural language dialogue, interactive onboarding tutorials, recommended insights, analysis history visualization, and generated reports. The pipeline of our framework goes through three stages: (1) Onboarding: uses LLMs to interpret visualization and views' relationships based on a system specification for users; (2) Exploration: guides LLMs in recommending insights to facilitate user's exploration through the analysis of the system's underlying data. (3) Summarization: allows users to retrace and select their analysis history through visualization and use LLMs to generate insight reports.

4.1 VA System Onboarding

When faced with complex VA systems that contain multiple views, many users experience a steep learning curve. Therefore, we propose an onboarding tutorial generation method, which aims to help users understand two kinds of knowledge in a VA system: visual style and coordination.

The onboarding can follow two types of approaches: one is bottom-up, i.e., introducing the details first, and the other is top-down, which involves providing an overview initially. Following Tanahashi et al., who demonstrated that top-down is more effective in introducing visualizations [54], we take a top-down approach. The tutorial will start from the system-level introduction to the view level. To that end, we leverage LLMs' ability to comprehend the system's information and generate tutorials. Previous studies demonstrate that declarative grammar of visualization is readily understood by language models [35], [36]. To encapsulate automatic tutorial generation, we need a unified specification for different VA systems. To this aim, we propose a specification of the VA system designed in line with the top-down approach, which is used as input for LLMs.

The specification starts from the system-level information `SystemSpec`. We provide the *systemInfo* attribute to detail the system's name and the total number of views it comprises and the *viewsInfo* to describe each view's style and coordination. Examples are available in Appendix A.

```
class SystemSpec {
  /** Information about the overall system */
  systemInfo: {
    systemName: string
    viewNumber: number
  }
  /** Style and coordination between views */
  viewsInfo: {
    viewStyleInfo: ViewStyleInfo[]
```

```
    viewCoordinationInfo: ViewCoordinationInfo[]
  }
}
```

For the view-level information, we introduced the `viewStyleInfo` to delve into the design of each individual view. This information captures the essence of the VA views through the `viewName` attribute, representing its identity, while the *layers* attribute describes its visual components. Each layer details the mark type (e.g., area), and the encoding attributes that provide insights into how data fields are visually mapped onto axes, color scales, and size scales. Additionally, an optional tooltip specification can be employed to enrich the user's interactive experience by displaying supplementary information upon interaction.

```
class EncodingInfo {
  field: string;
  type: string;
  description: string;
}
class ViewStyleInfo {
  viewName: string
  /** Information about styles of multiple encodings */
  layers: {
    markType: string
    encoding: {
      x?: EncodingInfo[];
      y?: EncodingInfo[];
      color?: EncodingInfo[];
      size?: EncodingInfo[];
      /** Detail level data field */
      lod?: EncodingInfo[];
    }
    /** Display prompt information for mouse hover.*/
    tooltip?: EncodingInfo[];
  }[]
}
```

Understanding the coordination between different views is pivotal for a coherent exploration experience. Thus, we introduced the `ViewCoordinationInfo` into the specification. This information bridges the interactions between the source and target views through the *sourceViewName* and *targetViewName* attributes respectively. Furthermore, the *coordinationType* attribute delineates the nature of interactions (e.g., filter, brush), and the *interaction* array provides a granular breakdown of user-triggered events and their subsequent ramifications on the target views.

```
class ViewCoordinationInfo {
  sourceViewName: string
  targetViewName: string | string[]
  /** Coordination type */
  coordinationType: string
  interaction: {
    /** Type of interaction */
    type: string
    /** Interaction's effect on target */
    effect?: {
      /** Action type */
      action: string
      targetViewName: string
      /** Data category for action */
      category: string
      /** Control the result */
      changeby: string
    }
  }[]
}
```

Our specification utilizes a key-value pair approach to determine the choice of fields and their format. This approach could offer flexibility and can be adapted for a variety of visualization types, not just the common visualizations. For example, users may define a customized card using a "title" and "context" to represent the encoding instead of using "x" and "y". Therefore, for special examples, the definition of key-value pairs is flexible, as long as it makes the LLM understand. Moreover, the specification not only helps LLMs generate intuitive onboarding tutorials but also aids in understanding the system to assist subsequent stages.

4.2 Insight Recommendation in Exploration

Visual exploration can be a challenging task that requires significant effort and expertise, particularly when there are no clear focal points to guide the process. Here, we introduce a strategy that channels LLMs towards enhancing insight recommendations. The approach includes three steps: selecting insight types, computing insights, and scoring insights.

4.2.1 Defining Insight

In our framework, insights are considered the basic units in visual exploration. Following previous studies [14], [57], we define insight using four attributes:

$$insight := \langle type, parameters, subject, score \rangle \quad (1)$$

Insight type: We first identify the 15 insight *types* based on previous work [14], [25], [48] including finding extreme, outlier, change point, trend, etc. To cover more complex data types, such as text and graphs, we have added three common types to the list, i.e., text summary and key nodes or key links. More details of the definition of these types are available in the Appendix C. Moreover, some insight types may require the analysis of coordinated views within a VA system. Therefore, from the perspective of complexity, we define an insight to be either aligned with a single view or across multiple views. For example, the trend of "sales" within the "sales view", or identifying positive correlations between "sales" and "profit" in both "sales view" and "profit view". This distinction will help LLMs understand how to calculate insights in VA systems.

Insight parameters: For each insight type, we use *parameters* to describe the characteristics of an insight, such as the direction of correlation and the location or time of a summarized event.

Insight subject: The *subject* in our framework defines the data scope to derive an insight, which includes four attributes:

$$subject := \langle subspace, dimension, measure, context \rangle \quad (2)$$

Each *subject* corresponds to a subset of data in a view. The values of the *dimension* can be mapped to the x-axis, and the *measure* can be mapped to the y-axis. The *subspace* is a filter of a dataset. For example, if a line chart has two products' "sales", the *dimension* is the time, *measure* can be "sales" or "profit", and a *subspace* can be a selected time range or a selected product. For text data analysis, we can use *measure* to refer to the text data column in a dataset. In more complex graph data, the *dimension* can refer to the types of entities and relationships within the graph. The *measure* can signify quantifiable properties or attributes related to nodes or edges.

In addition to the data set in the original view, some additional data may also serve as an important data source. For example, derived insights can be reserved to hint at subsequent analysis. Second, intricate tasks such as finding important nodes associated with particular events surpass the bounds of a conventional *subject* due to the demand for event summaries. For these certain requirements, we define the additional data source as *context*.

Insight score: Different insights are not equally attractive to users. Previous research defines the importance score including impact and significance scores. However, to achieve insight recommendations in the VA system, it is also important to assess what insight types could be used to solve the task. Thus, we consider relevance, impact, and significance scores together to calculate the insight score.

The significance calculation method is adapted from QuickInsight [14]. Specifically, this score is calculated based on hypothesis testing, which takes a value within the range $[0,1]$. Take finding the outstanding number one item in a group as an example. The hypothesis is that the data obeys the null hypothesis, i.e., long-tail distribution. The p-value is used to indicate whether the data excludes the maximum that will go against the null hypothesis. Thus, we get the significance score as $1-p$ (p is the p-value). Other examples of calculation methods are introduced in this specification [1]. While our analysis utilizes p-values to assess the significance of insights, we recognize the concerns around the use of p-values. First, the p-value is volatile and does not convey the magnitude of an effect. Second, the p-value is not suitable for all insight types. To address these issues, our scoring mechanism for insights is designed to be adaptable. We suggest incorporating alternative measures such as effect sizes, Cohen's d, Odds Ratio, or the coefficient of determination [51] for broader applicability. For text analysis, metrics like BLEU and ROUGE or LLMs themselves could serve as more suitable substitutes [10].

To assess the impact score, there are two kinds of aspects that need to be considered. One is the coverage of the data subspace over the entire dataset, and another is the semantics of the data subspace. Previous work defines the impact as the coverage of the data subspace over the entire dataset [57]. Take the social media event analysis (used in Sec. 6.1) as an example, "microblog" and "call center" are two types of messages that represent two distinct *subspaces*. If the amount of "microblog" messages surpasses that of "call center", the insights derived from the "microblog" data are considered to have a greater impact due to their more extensive coverage. However, from the semantic aspect, the message from the "call center" is more impactful because it is more reliable and timely. We assess such impact using LLMs based on their understanding of the dataset and their broad common knowledge of data analysis in different scenarios.

As users typically engage with a VA system with an analytical task in mind, we use a relevance score to quantitatively measure the congruence between the emergent insight and the overarching analytical task. We let LLMs evaluate the insights by considering how closely the insights align with the task. For instance, in event analysis, the task is to detect risk events and regions in the city. Therefore, analyzing messages to summarize events would score highly on relevance due to its direct connection with the task.

Considering three scores jointly, we compute the insight score as $score = \sum_{k \in \{significance, impact, relevance\}} w_k \cdot score_k$. Here, w_k represents the weights for the significance score, impact score, and relevance score, respectively. Since our purpose of getting data insight is to uncover its patterns, the significance score should be given more weight than the others. Therefore, we empirically set the weights to be 0.5, 0.2, and 0.3, respectively. The weights are adjustable to fit different preferences.

1. https://www.microsoft.com/en-us/research/uploads/prod/2016/12/Insight-Types-Specification.pdf

4.2.2 Recommendation Strategy

The insights recommendation strategy should be intuitive, adaptive, and relevant to the user's tasks. We address this with the following two-step strategy:

Step 1 (Insight type selection): Considering the multitude of possible insight types, we prioritize selecting those that are most relevant to the task. When the user makes a selection on a specific view, we use LLMs to find relevant insight types according to the user's selection and task and give a relevance score. The high-relevance score insight types will be used for further calculation. To automatically execute these insight types, LLMs need to schedule the data that compute insight needs (e.g., data tables, column names). The output includes the selected insight types, relevance scores, and the data information for insight calculation.

Step 2 (Insight assessment): This step aims to conduct a comprehensive assessment of insights. First, the last step's selected insights types will be executed to generate the insights. We use the LLMs to translate the calculated results into complete sentences. Second, we calculate the combined score of insights from three aspects. As we defined before, the significance score is calculated by statistical methods, and the impact and relevance scores are assessed by LLMs. The final output of this step is the insights ranked by the combined score and the corresponding data points in the views for annotation.

After the two steps, the insights should be annotated to the relevant views in the original VA system. Users can constantly interact with the view to gain new insights from LLMs.

4.3 Summarization of Exploration Results

Following exploration interactions with LLMs, users may wish to embark on a new exploration round, thereby establishing a human-in-the-loop analysis process for the continuous acquisition of novel knowledge. Given that not every interaction stage will yield the desired insights, we advocate for an interactive strategy to filter exploration results and create comprehensive reports, capitalizing on LLM' text summarization abilities.

Record preservation: Throughout the analytical process, it is imperative to maintain a detailed record of the user's exploration journey, including interactions and insights. We store these elements together because insights and interactions provide key findings that contribute valuable context for report creation. To elaborate, user interactions act as filters that modify the state of the analysis, which defines the subspace of insights. For instance, consider a scenario where a user selects a specific time period. This action prompts the LLM to summarize relevant events in this time period. When presenting the summary of events, it is crucial to provide the time period. This approach ensures that a comprehensive and coherent insight is captured, enhancing the overall understanding and relevance of the insights generated. Each recorded data set is defined to span m analysis rounds. Within each round, there are n distinct steps, representing the user's adopted insights from the LLM or their self-motivated interactions. We can represent the exploration journey as a matrix M of dimensions $m \times n$, where each element $M_{i,j}$ denotes the j^{th} step in the i^{th} round. For each step $M_{i,j}$, the following details are preserved:

- *focused view*: a descriptor of which visualization or data view was in focus.
- *insights*: the specific insights generated by LLM. If the user interacts by themselves, we record the interaction object.

- *screenshots*: a saved screenshot of the focused view crucial for subsequent report illustration.

Selective reporting: To provide users with a cohesive understanding of their analytical journey, our system presents an interactive mechanism. This mechanism graphically illustrates the matrix M, allowing users to retrace the insights for reporting. Once a user selects a round of data, the curated sequence from M serves as input for the LLM. Using this structured input, the LLM crafts a report aggregating both the analytical insights and the visualization, capturing the user's exploration journey.

5 LEVA Implementation

The implementation requirement of our framework includes two parts: the extensions of the original VA system and the LLM-powered components (Fig. 3). The original VA system requires configuration files and data handlers to process user selections and LLM outputs. The LLM-powered components consist of prompt handlers and presentation modules for showcasing outputs across the original system view and three new views.

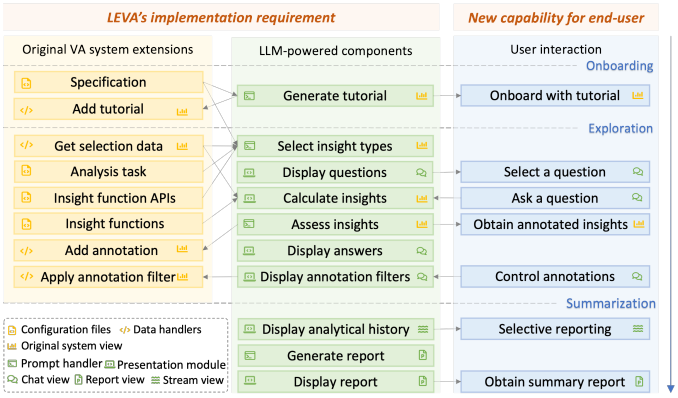


Fig. 3. The integration overview for augmenting a VA system with LLM capabilities involves both existing extensions and LLM-powered components in LEVA's implementation. The enhancement brings new capabilities for end-users at three stages.

In the following, we will introduce the efforts to develop the extensions of the original system and the LLM-powered components LEVA provided, including onboarding tutorial generation, insights recommendation, report generation, and the final integrated interface. For prompt templates used, we provide examples in Appendix B.

5.1 Extensions of Original System for Integration

To integrate our framework within a VA system, we need to prepare the configuration files and data handlers for the original system.

Specification: The specification acts as a blueprint for LLMs, allowing them to decode and understand the design and usage of the original VA system. By comprehending the specifications, LLMs can craft detailed and appropriate tutorials tailored to the VA system. Furthermore, as the specification contains the data and coordination information in each view, it can be used to help LLMs explore data by knowing the system state and determine which view and data column to calculate insights from.

Analysis task: Before recommending insights, it is necessary to propose a user task to describe in natural language sentences. If it is not specified, we can also use LLMs to propose the task that is used to select the relevant insight types.

Insight functions: We propose a list of functions to calculate insights in different types, e.g., get the outstanding top one and find an outlier. These functions correspond to the insight type defined in Sec. 4.2.1. The common insight types we give may not be sufficient in some special cases. Thus, developers can add functions according to their domain-specific analysis requirements. Previous taxonomies summarize complex tasks that might be helpful for proposing the computing functions, such as taxonomies for graph analysis [25], spatial and temporal analysis [1], social media analysis [9].

Insight function APIs: We let LLMs choose the functions from the APIs list to calculate the insight based on their understanding of the system. To achieve this goal, we need to give the definition of a function. It is better to have two attributes to explain the purpose and outcome of the function: $\langle name, description \rangle$. For example, to calculate the outstanding number one item in a group of data, the *name* could be "*get_outstanding_top1*", and the *description* could be "*Calculate the leading value is significantly higher than all the remaining values*". Providing the APIs of insight functions enables LLMs to choose the suitable analytical method and execute them by completing the parameters.

Data handlers: To support the connection between LLMs and the original system, we need to implement these handlers to capture the changes in the system and add the new features from LLMs.

- *Add tutorial:* Receive the tutorials from LLMs and employ a tour guide tool to display the tutorial.
- *Get selection data:* Upon a user interacting with a specific view, the system needs to get the selection data, including the filter and the updated data on each view.
- *Add annotation:* The original VA system needs to highlight the insights generated by LLMs on target views with annotations. This function should enable the selection of visual elements, changing their style, and adding annotations around them.
- *Apply annotation filter:* Allow the selection of the annotation across the views, such as filtering and deleting.

5.2 LLM-powered Components

In addition to the above extensions, the enhancement also needs LLM-powered components to support onboarding, exploration, and summarization within the enhanced VA interface.

5.2.1 Onboarding Tutorial Generation

In the onboarding stage, we use LLMs to generate tutorials by inputting the specifications of a VA system. To design the output of LLMs, considering the text form of the tutorial will increase the user's reading time, we designed the tutorial as an interactive tour guide. Therefore, the output of LLMs is a list of steps in the tour. Each step within the tour includes two attributes, including *title* and *description*. The value of the *title* is `viewName`. The *description* includes visualization *Type*, *Encoding*, and *Coordination*. We let LLMs output *description* as HTML format to paraphrase the long text to fewer lines and set font styles for clear observation. When users select to start the onboarding tour, the tutorial will be triggered and added to the original system to introduce each view. The prompt for generating an onboarding tutorial is shown below:

Prompt template for onboarding:
Here are the specifications of a visual analytics system.
{ `specification data` }
The specification includes the system-level, view-level, and views' coordination information. You need to introduce each

view's style (data meaning, visual mapping) and the relationship between views. Please give your answer in the following format:
{ format requirements }

5.2.2 Insights Recommendation

We implement the interactive recommendation to undergo two steps of conversation with LLMs. The first step is to select appropriate insight types based on the selection data and analytical task, and the second step is to execute and assess the insight to select the final results.

In the first round, the inputs consist of four components, including specification, the current interaction, analysis task and insight function list. We added specifications including `viewstyleinfo` and `viewscoordinationinfo` to the input. The `viewscoordinationinfo` allows LLMs to know the target views after the user's selection. The `viewstyleinfo` contains the data information in the view that will be used as parameters to compose the data for insight calculation. The current interaction is represented as a triplet: $\langle viewName, dimName, value \rangle$. If the user selects non-consecutive elements, such as two locations, California and New York, on the map (Sec. 6.2), the current selection will be two triples in an array. Moreover, if the selection from the previous analysis step is not canceled, it will remain together to provide context for further analysis. Then, LLMs determine the types of insight that can be analyzed based on the `insightfunctionAPIs` and `analytical task` and assess with a relevance score. In order to execute these insight functions, we define the output of LLMs as a quadruple: $\langle functionName, viewName, variableName, dimName \rangle$. The prompt template for insight type selection is shown below:

Prompt template for insight type selection:
When the user makes an action, the system changes. You should analyze data types of connected views based on the coordination information between views.
{ current selection }
{ view style info }
{ views coordination info }
According to the data info in each view and the analytical task, you should select all suitable analytical functions related to the user's task. You also need to give a relevance score to assess how closely related the insight is to the task.
{ analytical task }
{ insight function APIs }
Please give your answer in the following format:
{ format requirements }

In the second round, the selected insight functions are executed to get insights and significance scores, which are the `insight calculationresults` to be the input for further assessment. Then, the calculated results are organized into natural language sentences to describe insights by LLMs. After generating insights and obtaining the significance score, the next step is to assess impact scores. We let LLMs assign an impact score based on the nature of each insight, e.g., potential consequences, urgency and timeliness, and influence on decision-making. The developer can further modify the definition of impact score to fit specific analysis scenarios. The insight will correspond to a triplet: $\langle viewName, dimName, value \rangle$ to locate the insight with highlight effect or annotations on the corresponding Original system view. The prompt template for insight assessment is shown below:

Prompt template for insight assessment:
The selected insights are implemented, and the result is returned, including the value and significance score. You also need to give an impact score. You can consider combining your data analysis experience to evaluate from the following aspects: potential consequences, urgency and timeliness, and influence on decision-making.
{ insight calculation results }
Please give your answer in the following format:
{ format requirements }

After obtaining the generated insight with the structured format, the *Add annotation* function within the original VA system will be executed to link the data objects from the insights to elements within the views. For instance, if the LLM returns insight `{'viewsName':'Sales|ByState','fieldName':'State/Province', 'value':['California','NewYork'],'final_score':0.5}`, the VA system needs to be able to locate the two points on the map, change the style of the target element (e.g., stroke color), and add an annotation at that location. As we described in Sec. 4.2.1, each insight type can correspond to a unique view or multiple views. Thus, if an insight is cross-view, annotations will be added on multiple views as well. Considering that it is possible that not all insight types are what the user would like to see, we prefer to use single-view insights to analyze step by step.

When using LLMs for data analysis directly, it's crucial to consider the strengths and limitations of the language model. We tested the LLMs' data analysis performance on tabular datasets. The results indicated that their accuracy for basic tasks was relatively low, as detailed in Appendix C. Therefore, for tabular data analysis, we use rule-based methods to guarantee accurate results and use LLMs to invoke these functions based on the understanding of underlying data and the user's task. However, LLMs demonstrate exceptional competence when analyzing textual data, as evidenced by recent studies [17], [31]. Thus, we use LLMs to analyze these textual data tasks directly. Furthermore, if the recommended insights are incomplete or inaccurate, we provide an open-question answering function. Users can type follow-up questions to understand the insights in detail. Users can type follow-up questions to understand the insights in detail. The LLMs will analyze the underlying data in the current state and return insights and a highlight of source data, improving the explainability of how the insights are derived.

We adopt two strategies to address the response speed problem of LLMs. One way is to control the output for efficiency. Considering that some computational tasks, like text summaries, can produce lengthy outputs, it's essential to define the output length in the prompt template to ensure it remains concise. Another way is interactive questions and answers. Before executing insight functions, we adopt an interactive approach where LLMs recommend questions based on the assessment of the relevant insight type derived from the insight selection stage. Users can select a desired question to obtain insights. This alternative strategy can reduce the waiting time to calculate all insights.

In summary, the limitation of input and output influences both the calculation method and the interaction paradigm. Depending on the specific analysis scenario, the above strategy can be fine-tuned to strike the right balance. We also discuss these challenges and future directions in Sec. 8.

5.2.3 Report Generation

Based on the strategy introduced in Sec. 4, we implement the methods in two steps. First, the insights's annotations are saved, and the data is a 5-tuple: (*insight*, *type*, *value*, *viewName*, *imageName*). The combination of analysis round *m*, *n* and *viewName* is the *image name*, which will be used in LaTeX code. In general, each step obtains the corresponding screenshot according to *viewName*. Considering the first and last step preferably needs to give an overview of the entire report, we can set up to capture all views. Then, for a round of `historicalanalysisdata`, we let LLMs generate reports in a textual form. The prompt template for summarizing reports is shown below:

Prompt template for report summarization:
Here is a historical analysis of the system data. The data contains insights that need to be reported:
{ `historical analysis data` }
Your task is to write an insight report to present these findings. The amount of insight should be equal to the number of steps for given data. Ensure you include both a cover(report title) and a conclusion.
{ `other requirements` }

Second, given the exceptional performance of LLMs in code generation, the textual report can be transformed into a LaTeX-formatted presentation report. We can add requirements in the output format to set the report styles. The final generated reports are presented to the user through an interactive visualization that supports intuitive reading and markup. The prompt template for LaTeX code generation is shown below:

Prompt template for LaTeX code generation:
Transform the summarized report into LaTeX slides. For each slide, if an insight exhibits a clear hierarchy, segment it using bullet points. Accompany each insight with a screenshot from the system. The filenames for these screenshots can be found in the historical analysis data. Please integrate the following commands for style configuration.
{ `setting requirements` }

5.2.4 LEVA Interface

To support the whole framework, we design an interface to bridge users, LLMs, and the underlying data. The interface comprised of four main components: *Chat view*, *Original system view*, *Interaction stream view*, and *Report view*, as shown in Fig. 4.

Chat view: To receive feedback and control the entire workflow in our framework, this view serves as an interactive interface where users receive feedback and control other views (Fig. 4A). During the onboarding stage, the Chat view initiates a tour guide to introduce the original system. When users start on exploration, this view will showcase questions proposed by LLMs. Users can make selections, and the selected questions and insights are recorded and visualized in the Interaction stream view (Fig. 4C) and to support report generation in Report view (Fig. 4D). Moreover, the view allows for open-question answering, letting users engage in fluid conversations with LLMs to clarify doubts or derive new insights. Due to possible wrong formats generated by LLMs, we allow a feedback mechanism to display error messages in the Chat view and make users aware of failed issues.

Original system view: The original VA system is combined in this view. To augment the system, LEVA introduces annotations as arguments. These annotations, serving as guiding markers, help users identify interesting data patterns. To efficiently manage these annotations, a dedicated control panel in Chat view has been introduced (Fig. 4b2). It offers filtering capabilities and options to clear all annotations on the view. Users can interact with the views, prompting LLMs to propose questions based on their selections. After recommending a cross-view insight, if users select an area with annotations in the source view, the system will show the other part of the insights in the target view.

Interaction stream view & Report view: The Interaction Stream View stands as a historical ledger, cataloging analytical insights the user obtained from LLMs (Fig. 4C). Users can hover over the node to retrace details in each step. Each step's analytical insights are automatically saved. Considering that the user may want to stop the current round and start a new one, we provide an *end button* to enable users to end their current analysis round, signaling the system to start a new round of analysis. Moreover, we set the Interaction stream view as hidden by default. Users can open the interaction view in the *menu* at the top of the Chat view when they need to trace back. Upon selecting an interaction path, the Report view (Fig. 4D) is triggered, presenting a comprehensive report for that round.

6 USAGE SCENARIOS

To evaluate our framework, we demonstrate the LEVA-enhanced VA system in two usage scenarios: one is analyzing multi-facet event data, and the other is analyzing tabular data. We use the OpenAI GPT-4 model in our work.

6.1 Analyzing Multi-faceted Event Data

To illustrate how LEVA aids users throughout the VA workflow, we opted to reproduce a VA system: the recipient of the IEEE VAST Challenge 2021 Mini-Challenge 3 Award [40]. Our motivation for this choice stems from several compelling reasons. Firstly, this system exemplifies the intricate, human-in-the-loop decision-making tasks inherent to visual analytics. Secondly, it incorporates a representative blend of data types and corresponding visualizations, encompassing text, graph, spatial, and temporal data, which is a typical VA system. Lastly, its recognition as an award winner lends credibility and affirms its representativeness.

The challenge's task is to detect and evaluate public risks in Abila City during the evening of January 23, 2014. The provided data include microblog records and emergency dispatch records from a call center. Thus, the system centers around a comprehensive timeline that serves as the main interaction point to detect event evolution. A message view presents messages from specific time periods. A keyword view displays messages within a selected period, allowing users to hone in on particular topics. A graph view reflects occurrence relations between various entities, such as persons and locations. Finally, the system provides a map view of Abila City to show the message distribution and the risk levels.

In this dataset, we mainly follow the guidelines outlined in Sec. 5. However, to provide a multi-faceted analysis of the event, we have made some minor adjustments and additional considerations.

- *insight functions:* We propose some insight types for event analysis. The single-view insight type includes summarizing the events of a certain period in high-risk areas, summarizing the events of a keyword, finding the nodes, messages, and keywords associated with the events, and retrieving values. These functions could be related and combined to address a

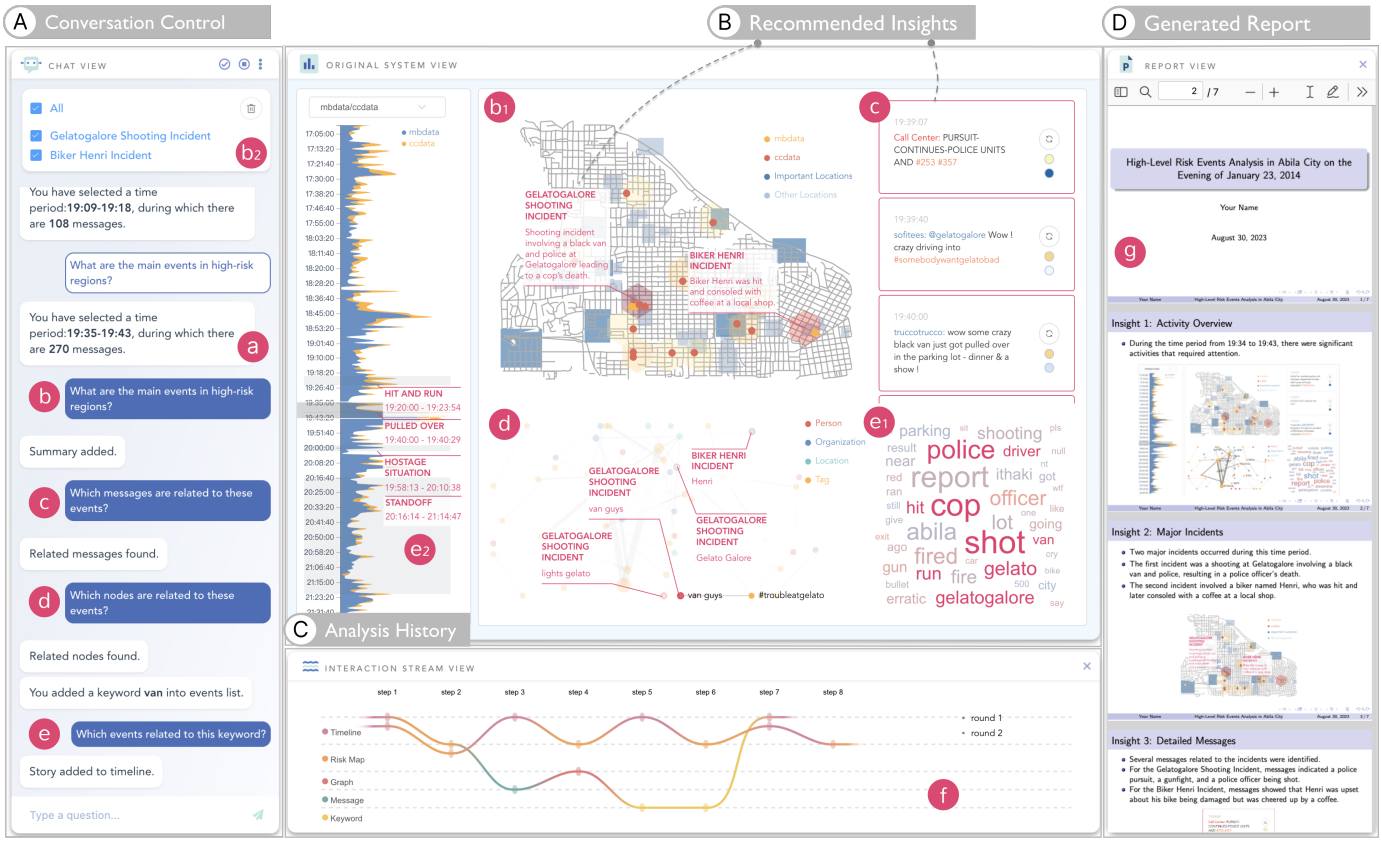


Fig. 4. An implementation of LEVA comprises of four components. Users can communicate with LLMs and control the insight annotations in (a) Chat view; the recommended insights for next step analysis from LLMs are updated in (b) Original system view; Users can retrace the interaction history in (d) Interaction stream view; Once a historical analysis path is selected in (d), the generated insight report will display in (e) Report view.

more complex task. For example, summarizing the events could be recommended as the first and then finding the relevant nodes in the graph. The events serve as a *context* in addition to the graph data, as we defined in Sec. 4.2.1. We also define a cross-view insight to analyze multiple views at once, which is to summarize events with temporal and spatial information. The annotations of this insight type will show on both the timeline view and map view.

- *propose questions:* Considering that textual data analysis may take a long response time, here we let the LLM propose the questions first, and then the user chooses one question on the Chat view to execute the insight functions, as the consideration we described in Sec. 5.2.2.

At the beginning, we click the onboarding button to start a tour guide (Fig. 5a). The tutorial introduces the visualization type, visual encoding, and the coordination between views. This guidance leads us to know the meanings of data analyzed in the timeline view (Fig. 5b). We can also obtain that hexagons on the map denote risk levels of the region (Fig. 5d). The system also tells us that all the other views link to the timeline, which renders based on message type and keywords selected from the keyword view (Fig. 5c). Without this guidance, one needs to take more time to explore the system and might build an inaccurate understanding of the system.

In the next exploration stage, we first look at the timeline as it filters other views. We selected 19:34 to 19:43 since they contain the highest peak. The assistant proposes a question: "What are the main events in high-risk regions?" (Fig. 4b), which is related to our analysis task. Then, the system points out two high-risk regions,

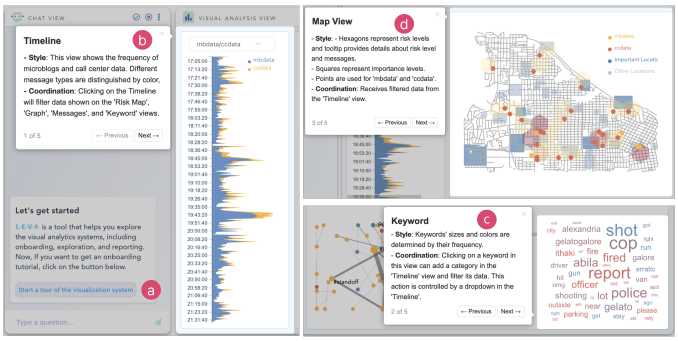


Fig. 5. An onboarding tour example of the VAST challenge system. (a) Initiation via the onboarding button, (b) Introductions to data meanings of "mbdata" and "ccdata", (c) The coordination of keyword view and timeline view based on selected keywords, and (d) The visual encoding of hexagon colors representing risk levels in specific regions.

summarizes two events (Fig. 4b1), and figures out the most relevant nodes (Fig. 4d1) and messages (Fig. 4c1). By briefly scanning the summaries, we know one high-risk incident is a bicyclist who gets hit but is helped out by people from Brew've Been Served. The other is a black van and police shootout in the store. The annotations on the Graph view indicate that the key player in "Gelatogalore Shooting Incident" is the "van guys", which happens in "Gelatogalore", which is a location colored in green. To further know if the black van exits in other time periods, we select "van" from the keyword view (Fig. 4e1). The assistant catches our action and proposes to analyze the related events with "van". We clicked

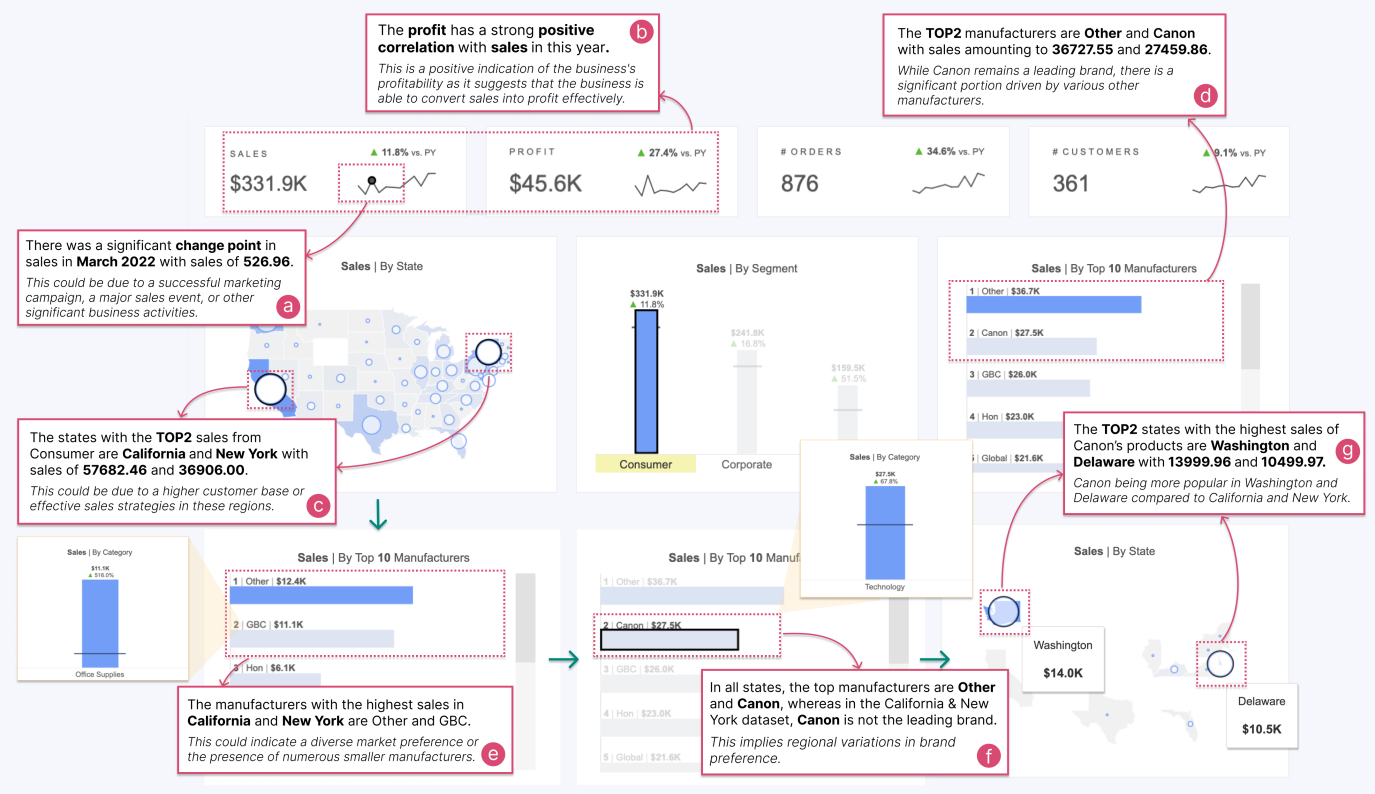


Fig. 6. An example of implementing our framework for tabular data analysis in the exploration stage, which demonstrates the analysis results of each step in a single round (a-d) and the result of multiple rounds (e-g). The annotations represent the insights and potential impact of the LLM output. The analytical process starts with selecting Consumer on Segment view and four exploration steps with LLMs recommendations (a-d). With the help of LLMs, we found that manufacturers with high sales in different regions have a significant distribution of preferences (e-g).

it and four events are summarized along the timeline: "Hit and run", "Pulled over", "Hostage situation", and "Standoff" (Fig. 4e2). After brushing each event on the timeline view, the annotations show up on the map view. Reading the details of each event, we know the whole story of the black van.

In the process of exploration, valuable insights are recorded and can be traced back in the interaction stream graph to avoid forgetting important insights. As incidents related to the black van raise the most public safety concerns, we choose this round to generate the report (Fig. 4f). The generated report explains each insight and retains the pictures of the exploration process, and summarizes the appropriate title and conclusion page (Fig. 4g).

6.2 Analyzing Tabular Data

To demonstrate the generality of the proposed framework, we apply it to tabular data analysis 6. According to our method design in Sec. 4, text data analysis is a task that LLMs are good at, but table data are more suitable for analysis by statistical methods, ensuring accuracy and efficiency. Nonetheless, LLMs remain instrumental, especially in distributing tasks and assessing insights. Thus, this study mainly introduces the exploration stage of tabular data.

For illustrative purposes, we select a dashboard from Tableau that displays superstore sales data from 2022 [2]. This dashboard features nine distinct views: a choropleth map indicating state-wise sales and five bar charts delineating sales across segments, categories, sub-categories, top 10 manufacturers, and top 10 customers. Additionally, there are four line charts that trace the trajectories of sales, profits, orders, and customer metrics.

Before exploration, we prepare specifications, tasks, insight functions, and data handlers and adjust insights content based on the analytics scenario, following implementation guidelines outlined in Sec. 5.1. Some considerations described as below:

- *specification:* As Tableau dashboards' source files (with file extension .twb) are structured in XML format and contain all the system information that can be extracted and converted to the values in specifications, which allows us to extract specifications automatically.
- *analysis task:* While the dashboards display a variety of data, including "sales", "profit", "orders" and "customers", we initially set a task to focus on analyzing the "sales" situation from multiple perspectives.
- *insight functions:* We employ fundamental insight functions as referenced in Sec.4.2.1, such as outstanding number one, change point, trend, and correlation. Given the breadth of insights possible with tabular data, it is easy for users to overlook connections to prior findings. To address this, we also set to compare current insights with those from the previous step, using the earlier results as *context* for computation, as defined in Sec.4.2.1.
- *data handlers:* We integrated the Tableau dashboard into a website using its embedding API [52]. This API permits "get selection data" (Sec. 5.1). However, due to API limitations, tutorials and insights are shown in the chat view, and analytical insights are stored accordingly.
- *propose explanations:* Considering that in sales analysis, users

2. https://public.tableau.com/app/profile/p.padham/viz/SuperstoreDashboard_16709573699130/SuperstoreDashboard

TABLE 1
The questionnaire and the corresponding types, including objective questions and subjective questions.

Type	Specific questions
R1: Perceptual visual encoding	**O1:**What do the visual encoding and corresponding data mean for the timeline view?
	S1: Do you know what each data variaiables means?
	S2: Do you understand the meaning of the visual elements?
R2: Interaction and Coordination	**O2:** How the timeline view coordinated with other views?
	S3: Are you clear on how to interact in each view?
	S4: Are you clear on how the views are related?
R3: Data Pattern Discovery	**O3:** What high-risk level events occurred in the peak period?
	S5: Is it easy to get data findings (such as events, key nodes) in these views?
R4: Hypothesis Formulation and Validation	**O4:** What are the key player and location of the summarized event?
	S6: Are you clear about the next step analysis for validation?
	S7: Do you have easy access to rich hypotheses?
R5: Summarization of Exploration Results	**O5:** Discover related events of the keyplayer and summarize them into a report.
	S8: Is it easy to write an analysis report on the interaction results?
	S9: Are you satisfied with the quality of the report you wrote?

may need some hint for reason analysis behind the insights. We let the LLM give additional explanations in the final output based on its board knowledge [26].

In the beginning, we see that the Consumer on sales by segment view was highlighted as the highest value. After selection, the other four bar charts and the map are filtered, and some insights are recommended. The first recommended insight is "a significant change point in March 2022" in the sales line chart (Fig. 6a). The LLMs also suggest potential reasons for the change point could be a successful market campaign or business activities of the superstore. Then, a recommended insight is a strong positive correlation between profit and sales (Fig. 6b). This finding makes us realize that the strategy launched in February succeeds in turning sales into profits, which can continue to be used in the future.

Beyond time series insights, the recommendations also highlight extreme values, specifically, the top two rankings in states or manufacturers (Fig. 6c, d). We notice that the two states with outstanding sales are "California" and "New York". Consequently, our subsequent analytical focus pivots to these states. Within these jurisdictions, the top two manufacturers identify as "Other" and "GBC", the latter being a renowned office supply brand (Fig. 6e). The below explanation suggests a distinct market inclination towards smaller manufacturers in these regions.

An intriguing insight emerges when comparing the top manufacturers at the state and national levels. While "Other" and "Canon" dominate sales across most states, "Canon" does not maintain this lead in "California" and "New York" (Fig. 6f). This variation accentuates the nuanced manufacturing preferences specific to regions. Motivated by this finding, we move to analyze the distribution of "Canon" nationwide. Upon deselecting "California" and "New York", the recommended insight reveals Washington and Delaware as the leading states and reiterates heightened popularity of "Canon" in these regions (Fig. 6g). These findings may help potential adjustments to cater to regional sales predilections.

7 User Study

We conducted a user study to evaluate the effectiveness of LEVA for enhancing VA. Specifically, we wanted to verify if LEVA improves users' understanding of the original system, and aids in insights discovery and summarization more efficiently.

7.1 Study Setup

Here, we introduce the user study by discussing participants and apparatus, questionnaire, procedure, and results analysis.

Participants and Apparatus: We recruited 20 participants with backgrounds ranging from computer science, data science, and mathematics to business analysis, ages from 19 to 25 ($\mu = 22.37, \sigma = 1.79$), denoted as P1-P20. Among them, 4 participants are novices in using VA systems. Participants were randomly assigned to two groups, of which 10 participants used the VA system from the VAST challenge system without LEVA's assistance as the control group and 10 with assistance as the treatment group. The studies were all conducted using a monitor with a resolution of 2560 × 1440, along with a mouse and keyboard.

Procedure: The study was composed of three sessions, beginning with a 10-minute introduction to our framework, the usage of our system, and the original system. Participants can follow the experimenter to use the system and familiarize themselves with system functions and workflow. The formal assessment was conducted using a questionnaire including objective and subjective questions. Each participant in the two groups was first asked to answer objective questions, followed by subjective questions. We also conducted a short interview to collect detailed feedback from the participants.

Questionnaire and Measurements: Based on the five requirements (R1-R5) outlined in Section 3.2, we propose five objective questions (O1-O5) and nine subjective questions (S1-S9) to evaluate the effectiveness of our framework in meeting these requirements (Table. 1). Each requirement was tested with at least one objective question and one subjective question. For objective questions, the O1-O4 assessed users' understanding of data and visual mapping, as well as their ability to discover insights from the VA system. The O5 required participants to find insights and write a report. The S5-S9 is to verify the effectiveness of LEVA in insight recommendation and summarization. Answer times and correct rates were recorded for each question to gauge the influence of LEVA in terms of both efficiency and effectiveness. The correctness assessment for each question is not a strictly binary 0-1 variable but allows for a 0.5 score when answering half of it correctly. For example, for the first question (O1) "*What do the visual encoding and corresponding data mean for the timeline view?*", if the user can comprehend the visual encoding but is uncertain about the data

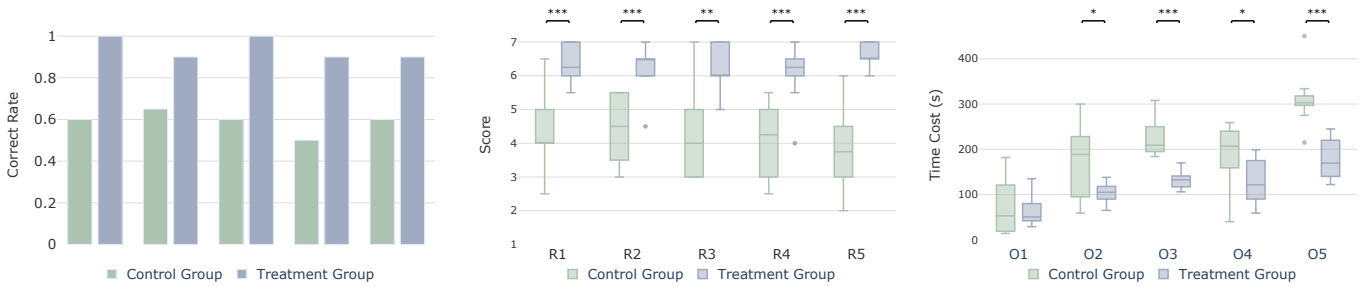


Fig. 7. User Study Results. On the left is the correct rate of five objective questions. In the middle are self-rated scores for subjective questions given by participants. On the right are the answer time for the four timed questions. The number of asterisks (*) in the upper part of the figure indicates the significance level of the test (* : $p < 0.05$; ** : $p < 0.01$; *** : $p < 0.001$). The results suggests our method can improve performance from these three perspectives.

meaning, a score of 0.5 is recorded. Finally, the average score of all participants is calculated as the correct rate for this question. For subjective questions, we included a 7-point Likert scale to allow users to evaluate their level of understanding of the system and whether they encountered any difficulties in gaining data insights and writing analysis reports. The control group was asked, "Do you understand the meaning of the visual elements of each view? Rate your understanding from 1 to 7." For the experimental group, the question was modified with the prefix "With LEVA's assistance" to gauge the impact of LEVA on understanding. In table 1, we present only the core questions, omitting prefixes and suffixes for brevity. Among them, S1-S4 are used to verify whether LEVA helps onboarding. Thus, we need to test their understanding of the UI components of the original VA system. The S5-S9 is to verify the effectiveness of LEVA in insight discovery and summarization.

7.2 Results and Analysis

To compare the answer time and subjective scores in two groups, we first conducted the Shapiro-Wilk test in the user study to verify the assumption of normality, ensuring the validity of subsequent t-tests. The result of the Shapiro-Wilk test confirmed the normal distribution of users' answer times and scores in our samples. After establishing normality, we applied the independent t-test to assess differences in answer times and accuracy between the treatment and control groups, ensuring the reliability of our results. We reported our results of the correct rate, subjective scores and time cost. The detailed result analysis of the user study is presented below.

Accuracy: We reported the results of objective and subjective measures to assess users' understanding of VA. The correct rates of five objective questions are shown in Fig. 7 (left). For each question, the treatment group had a correct rate of over 85%, while the control group's correct rate ranged from 50% to 65%. Most participants can distinguish encoding, but as the legend is not specific, they were unclear about the meaning of "mbdata" and "ccdata" represented by the different colors in the Timeline view (O1). Furthermore, in O2, most participants in the control group found it challenging to know the influence of the keyword view on the timeline, while in the treatment group, there were explanations of the interactions between various views. In O3, all participants in the treatment group answered correctly, while the control group was only 60% correct because the two events with higher risk levels were automatically highlighted in the system exploration in the treatment group. However, the participants in the control group had to switch between the map and message view and read the text repeatedly. The O4 yielded a 50% correctness rate in the control group. This lower accuracy can be attributed to the fact that a significant portion of participants could only identify the key player involved in the event while struggling to pinpoint the event's location within the intricate graph view. In contrast, the treatment group benefited from an automated annotated system that provided clear event location information. For insight summarization (O5), only a few participants in the control group were able to assemble a more coherent understanding of the event (P12, P15, P16), and they had prior experience in social media VA and cost a considerable amount of time (over 5 minutes). In contrast, in the treatment group, only one participant (P4) failed due to their insensitivity to the location name and not noticing the legend.

Score: As shown in Fig. 7 (middle), the treatment group obtained higher scores than the control group, and the average score is improved by approximately 49.21%. This indicates that participants perceived LEVA as an improvement over the original VA system in various aspects. For R1 and R2, while most participants understood brushing the timeline would filter other views, it was easy to overlook the filtering from the keyword view to the timeline view. Thus, they mostly got half of the score in accuracy but gave a lower subjective score due to the confusion in detailed interactions. This suggests that there was a lack of clear explanations of interactions and coordination in the system, underscoring the significance of effective onboarding. The performance in insight discovery varies among participants; only a few people can find some data patterns and form new hypotheses (R3, R4). For report generation (R5), the treatment group scored significantly higher than the control group in the users' own scores on the ease (S8) and the quality (S9) of generating reports ($p < 0.001$). Participants found it difficult to locate previously explored data due to the large amount of information (P18, P19). The results imply that users typically encounter greater difficulty and produce lower-quality insights when tasked with self-directed exploration and manual report generation compared to the generation of reports through an automated process.

Time Cost: We reported the time costs of the five objective problems in Fig. 7 (right). The results indicate that, with the exception of the first question (O1), the treatment group exhibited shorter answer time compared to the control group ($p < 0.05$). For the last four tasks, the average time is reduced by approximately 39.26% in the treatment group. For insight recommendations (O3, O4), the average time is reduced by approximately 41.98% and 31.27%. For the response time of LLMs, summarization events (O3) need more time to return results, about 25s for the long input

and output data of O3, but it is still the one with the most time savings. P2 suggested that *"Even though event summarization took a bit longer than others, it was acceptable given that it saved us even more time than if we had to read and summarize the information ourselves"*. The LLMs's response time of finding key nodes (O4) is faster than O3, only requiring around 5-6s, improving the average 31.27% of time. For the report generation (O5), using the LLM to summarize can save an average of 42.27% of time compared with no assistance. These findings underscore that the utilization of LEVA can substantially diminish the time investment required by users when employing the VA system. In addition, we observed that LLMs have the probability that the text summary is not comprehensive enough. In the study for P7, the LLM did not clearly summarize who the key player was. However, through the free ask in the chat view, P7 got the answer, which took one minute longer than the average time. This shows that when relying on LLMs for data analysis, it is necessary to provide free questions and answers to ensure the acquisition of detailed information.

Feedback: In the final interview, participants were asked about their opinions on the system. On one side, participants in the control group offered feedback pertaining to the VA system itself, highlighting issues such as unclear visual elements and legend (P11, P13), confusing color schemes (P20), excessive textual data (P15, P17, P19), and complex interaction between views (P12), no idea where to click and how to explore (P20). On the other side, participants in the treatment group contributed constructive suggestions for improvement:

For onboarding tutorial generation, many participants agreed that onboarding guidance is required, especially for beginners (P9). To improve the tutorial, P4 suggested that "*The system can further highlight some important keywords in the tutorial*". P7 and P8 recommended incorporating animations, such as arrows, to explain the interactions and relationships between views: "*Some animations like arrows could be used to explain the interactions and associations between views*." This inspires us that the more intuitive tutorial generation could be a future research direction.

For insight exploration, participants had varied feedback on insight exploration. Some found that the LLM effectively guided their analysis (P4), quickly leading them to valuable insights (P3). Others mentioned that building on the LLM's analysis, they were inspired to think further and pose new questions by free ask (P7). However, there were also comments about the improvement of more analytical methods. P5 provided a suggestion for improvement: "*In addition to following the original workflow exploration of the system, like the event analysis, some other tasks, such as starting from a person or spreading relationships, could be considered.*" The comment points out the need for more types of insight. Therefore, further study could focus on how to bring more domain knowledge and analytical methods to LLMs.

For selective report generation, most participants appreciated the reports. P6 mentioned that "*not only the comprehensive content with images and texts but also the good formatting*." To improve the report, P10 pointed out that "*If some steps in a stream view could be removed or merged, the generated report would be more useful*." The comments demonstrate that they considered our report generation as a convenient and useful function, but further need to improve the log organization and screening. Additionally, we also collected the comments for scalability. P3 is interested in using LEVA's components as plug-ins for other VA systems. This is a practical suggestion to provide a powerful tool to enhance more VA scenarios we plan to study further. We discuss these valuable suggestions from the user study in the Sec. 8

8 Discussion

In this section, we discuss the generalizability and the performance of LLMs we observed and highlight the lessons learned from the research and future directions.

Generalizability: Our framework is generalizable in four aspects. First, the system specifications we formulate can be expanded and comprehended by LLMs for tutorial generation. Second, our strategy for recommending insights is adaptable, allowing the LLMs to distribute computational tasks and assess insights in VA systems. Third, our interactive reports generation strategy can be extended to other systems by preserving analysis records. During practical implementations, users can fine-tune these strategies based on the VA tasks and the performance nuances of LLMs. Finally, LEVA remains independent of LLMs. Currently, we integrate LLMs's capabilities to support the exploration of VA workflows. Although the future appearance of models with other modal inputs saves efforts on engineering implementations of basic information processing, LEVA's strategy for guiding human intelligence model communication remains unchanged.

The performance of LLMs: Despite their immense strengths, LLMs still have several limitations, and there is a lively and ongoing debate on their merits [53]. The first problem is the accuracy. While one might anticipate that LLMs-enhanced systems will improve as more users interact with them, there are potential risks that LLMs could assist analysts in ways that might not be entirely accurate. Recent research focuses on using fine-tuned LLMs for tool using [43], transforming natural language into code [62], and employing advanced prompting strategies such as self-instruction to compute step-by-step [56]. To enable correct parsing output, we could add an example of constraining the output format of LLMs [60]. Moreover, we could also provide an error reporting strategy to make users aware of the unsuccessful response. We argue that addressing and communicating potential errors remains an exciting and open research challenge and calls for further exploration. The second problem is the response time. Our current approach is to control the output length of the LLMs or invoke alternative computation functions that are faster than LLMs. However, controlling the output length may sacrifice the level of detail and depth in the generated output. One future direction could be exploring acceleration strategies from the perspective of cache mechanisms [15], and predictive analytics might offer speed improvements without compromising the quality of the insights generated.

Human-LLM collaboration in open-ended exploration: In the context of open-ended exploration, the interplay between LLM assistance and human judgment presents a nuanced dynamic. Our user study reveals that the timely questions and insights proposed by LLMs could facilitate efficiency and even prompt users to follow up with their own new questions. However, it is also essential to recognize that users could be over-reliant on LLM guidance. They can thus be steered toward particular directions while missing others. This interplay between guided exploration and autonomous discovery is critical to the design of LLM-supported analytical systems. It warrants careful consideration to balance the benefits of guidance with the freedom of exploration – a challenge also recognized in the visual analytics guidance literature [6].

Domain knowledge integration for LLMs in specific tasks: While ensuring accuracy through the LLM-based insight function invocation mechanisms, there is a need to enhance problem-solving

flexibility in specific domain scenarios. The user study indicates that some users with domain analysis backgrounds will have more profound analytical ideas in open-question answering. LLMs need better insight into the calculation and understanding of the analytical task. This capability needs more domain knowledge. There might be two methods: one is tailoring insight types and functions that calculate insights for a particular problem, and another is to employ a fine-tuned LLM, specifically optimized to serve the needs of the domain. [20]. Both of which are open challenges for further research.

Insight recommendation vs. Interaction recommendation: Our current insight recommendation focuses on extracting essential insights from the underlying data, capitalizing on the inherent patterns and relationships present within the dataset. Another strategy is interaction recommendation, which derives insights from many user interaction data, learning and predicting the next interaction object [29]. Such an approach recommended insights with a more substantial contextual relevance. Looking forward, there's potential to integrate the historical interaction data. This merger can pave the way for more intelligent and contextually aligned insight recommendations.

Interpretation of user interaction for report generation: In report generation, both user selections and LLM-generated insights are pivotal. User selections offer a crucial context for the insights generated by the LLM, making it essential to preserve these selections for a complete exploration record. Currently, our approach can describe the actions taken by the user but falls short in interpreting the underlying motivations of these actions, impacting our capacity to provide comprehensive context in the exploration narrative. Typically, these selections are driven by insights users from their observations of data and combining personal knowledge. Future directions could include mining the related data patterns and combining more domain knowledge to generate more coherent exploration reports.

Narrative-driven report generation: Current methodologies in report generation within our framework primarily focus on compiling logs of exploratory logs into a step-wise report. Looking ahead, future research could pivot towards employing narrative structures and strategies for the automatic summarization of these logs. One direction could be using LLMs for automatic summation. The enhancements would come in two folds. Firstly, refining the narrative by identifying and reorganizing story pieces based on data relationships like temporal and spatial transitions [28], and then crafting coherent explanations [63]. Secondly, enhancing data presentation and narrative flow by considering the transitions between narrative segments [46] with multi-modality expression. These narrative techniques could provide a more intuitive understanding of the explored results.

LLM-based enhancement vs. Rule-based enhancement: The advantage of using LLMs is that it enhances flexibility and scalability in aiding various VA scenarios, offering capabilities beyond what a few lines of traditional coding can achieve. LLMs excel in natural language understanding and generate insights across a broad knowledge domain. While traditional coding is precise, it often becomes cumbersome and inflexible when faced with diverse scenarios and evolving user needs. Specifically, it would require creating an extensive set of detailed rules to break down various query requests, matching computational modules with data, binding views to different types of insights, utilizing numerous manual templates to introduce the system, describing computation results, and guiding the exploration process.

Componentization and plug-in: During our user study, we found out that users expect using our framework to assist them in exploring more VA systems with different tasks. Therefore, it is better to offer a toolkit and divide the functionality of LEVA and the views included in the current implementation into components. This toolkit will include the LLM-powered components and programmatic interfaces. For developers of the original VA system to integrate with this toolkit, the minimum developing cost is to provide the extensions and APIs to receive the LLM's output and modify the prompts to customize the output format of tutorials or annotations. By doing so, we could allow LEVA to be integrated as plugins in different VA systems. Further, we plan to provide more templates to allow users to customize the information (e.g., specification) entered into the LLM and the desired tutorials, reports, and insight recommendation from the LLM according to their needs.

Specification for VA systems: In the extraction of the declarative grammar required in LEVA, we refer to previous work describing basic charts [30] and adding descriptions of the data table, user interaction, and coordination based on the goal of understanding data, view and insight recommendation. An abstract-level description of data and functionality for VA specifications can further benefit various downstream tasks. Such abstractions can greatly facilitate endeavors like the automatic generation of visualization systems and automated storytelling. As we found the interaction between views could be introduced with more intuitive annotations and animations in the tutorial, the future work can research on how to improve the specification and guide LLMs to generate such tutorial.

9 Conclusion

In this study, we introduced LEVA, a framework that integrates LLMs into VA workflows to achieve intelligent VA. LEVA enhances visual analytics through three pivotal stages: onboarding, exploration, and summarization. During the onboarding stage, it interprets visualizations and their relationships, fostering the creation of dynamic tutorials. In the exploration stage, our insight recommendation strategy harnesses LLMs recommend analytical insights based on the interpretation of the system's status and data, enriching visual analysis via annotations. In the summarization stage, LEVA allows users to revisit and select analytical history, streamlining the report generation process. Our integration of LEVA with a VA system led to the development of an interactive interface that fosters a dialogue between users and LLMs. We conducted two usage scenarios and a user study to demonstrate the effectiveness of our framework.

Acknowledgments

The authors want to thank the reviewers for their suggestions. This work is supported by Natural Science Foundation of China (NSFC No.62202105) and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0103), General Program (No. 21ZR1403300), Sailing Program (No. 21YF1402900) and ZJLab.

Supplemental Material

Appendix A introduces the two specifications of the VAST challenge system and the Tableau system. Appendix B describes the prompt examples we used to generate tutorials, insights, and reports. Appendix C describes the evaluation of LLMs' data analysis capability.

REFERENCES

[1] N. Andrienko, G. Andrienko, S. Miksch, H. Schumann, and S. Wrobel. A theoretical model for pattern discovery in visual analytics. *Visual Informatics*, 5(1):23–42, 2021.

[2] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, 2013.

[3] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, C. T. Silva, and H. T. Vo. VisTrails: Visualization meets data management. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 745–747, 2006.

[4] Y. Cao, X. Li, J. Pan, and W. Lin. VisGuide: User-oriented recommendations for data event extraction. In *Proceedings of SIGCHI Conference on Human Factors in Computing Systems*, pages 1–13, 2022.

[5] D. Ceneda, N. Andrienko, G. Andrienko, T. Gschwandtner, S. Miksch, N. Piccolotto, T. Schreck, M. Streit, J. Suschnigg, and C. Tominski. Guide me in analysis: A framework for guidance designers. *Computer Graphics Forum*, 39(6):269–288, 2020.

[6] D. Ceneda, T. Gschwandtner, T. May, S. Miksch, H.-J. Schulz, M. Streit, and C. Tominski. Characterizing guidance in visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):111–120, 2017.

[7] D. Ceneda, T. Gschwandtner, and S. Miksch. A review of guidance approaches in visual data analysis: A multifocal perspective. *Computer Graphics Forum*, 38(3):861–879, 2019.

[8] S. Chen, J. Li, G. Andrienko, N. Andrienko, Y. Wang, P. H. Nguyen, and C. Turkay. Supporting Story Synthesis: Bridging the gap between visual analytics and storytelling. *IEEE Transactions on Visualization and Computer Graphics*, 26(7):2499–2516, 2018.

[9] S. Chen, L. Lin, and X. Yuan. Social media visual analytics. *Computer Graphics Forum*, 36(3):563–587, 2017.

[10] C. Chenghan and H. Lee. Can large language models be an alternative to human evaluations? *arxiv.2305.01937*, 2023.

[11] D. Deng, A. Wu, H. Qu, and Y. Wu. DashBot: Insight-driven dashboard generation based on deep reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):690–700, 2023.

[12] V. Dhanoa, C. Walchshofer, A. Hinterreiter, H. Stitz, E. Groeller, and M. Streit. A process model for dashboard onboarding. *Computer Graphics Forum*, 41(3):501–513, 2022.

[13] V. Dibia. LIDA: a tool for automatic generation of grammar-agnostic visualizations and infographics using large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 113–126, 2023.

[14] R. Ding, S. Han, Y. Xu, H. Zhang, and D. Zhang. QuickInsights: Quick and automatic discovery of insights from multi-dimensional data. In *Proceedings of International Conference on Management of Data*, pages 317–332, 2019.

[15] P. R. Doshi, E. A. Rundensteiner, and M. O. Ward. Prefetching for visual data exploration. In *Proceedings of the Eighth International Conference on Database Systems for Advanced Applications*, pages 195–202, 2003.

[16] D. Gotz and M. X. Zhou. Characterizing users' visual analytic activity for insight provenance. *Information Visualization*, 8(1):42–55, 2009.

[17] T. Goyal, J. J. Li, and G. Durrett. News summarization and evaluation in the era of gpt-3. *arXiv.2209.12356*, 2023.

[18] E. Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 159–166, 1999.

[19] K. Hu, M. A. Bakker, S. Li, T. Kraska, and C. Hidalgo. VizML: A machine learning approach to visualization recommendation. In *Proceedings of the SIGCHI Conferenceon Human Factors in Computing Systems*, pages 1–12, 2019.

[20] C. Jeong. Fine-tuning and utilization methods of domain-specific llms. *arxiv.2401.02981*, 2024.

[21] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual Analytics: Definition, process, and challenges. In *Information Visualization: Human-Centered Issues and Perspectives*, volume 4950, pages 154–175. 2008.

[22] J. Kohlhammer, D. Keim, M. Pohl, G. Santucci, and G. Andrienko. Solving problems with visual analytics. *Procedia Computer Science*, 7:117–120, 2011.

[23] B. C. Kwon and B. Lee. A comparative evaluation on online learning approaches using parallel coordinate visualization. In *Proceedings of SIGCHI Conference on Human Factors in Computing Systems*, pages 993–997, 2016.

[24] C. Lai, Z. Lin, R. Jiang, Y. Han, C. Liu, and X. Yuan. Automatic annotation synchronizing with textual description for visualization. In *Proceedings of the 2020 SIGCHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[25] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, pages 1–5, 2006.

[26] H. Li, Y. Wang, Q. V. Liao, and H. Qu. Why is AI not a panacea for data workers? an interview study on human-ai collaboration in data storytelling. *arXiv.2304.08366*, 2023.

[27] H. Li, L. Ying, H. Zhang, Y. Wu, H. Qu, and Y. Wang. Notable: On-the-fly assistant for data storytelling in computational notebooks. In *Proceedings of SIGCHI Conference on Human Factors in Computing Systems*, 2023.

[28] W. Li, Z. Wang, Y. Wang, D. Weng, L. Xie, S. Chen, H. Zhang, and H. Qu. GeoCamera: Telling stories in geographic visualizations with camera movements. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023.

[29] Y. Li, Y. Qi, Y. Shi, Q. Chen, N. Cao, and S. Chen. Diverse interaction recommendation for public users exploring multi-view visualization using deep learning. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):95–105, 2023.

[30] Y. Lin, H. Li, A. Wu, Y. Wang, and H. Qu. DMiner: Dashboard design mining and recommendation. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–15, 2023.

[31] C. Liu and B. Wu. Evaluating large language models on graphs: Performance insights and comparative analysis. *arXiv.2308.11224*, 2023.

[32] C. Liu, L. Xie, Y. Han, D. Wei, and X. Yuan. AutoCaption: An approach to generate natural language description from visualization automatically. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 191–195, 2020.

[33] Y. Liu, T. Althoff, and J. Heer. Paths Explored, Paths Omitted, Paths Obscured: Decision points & selective reporting in end-to-end data analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–14, 2020.

[34] Z. Liu, C. Chen, J. Wang, X. Che, Y. Huang, J. Hu, and Q. Wang. Fill in the Blank: Context-aware automated text input generation for mobile gui testing. In *Proceedings of International Conference on Software Engineering*, pages 1355–1367, 2023.

[35] Y. Luo, N. Tang, G. Li, J. Tang, C. Chai, and X. Qin. Natural language to visualization by neural machine translation. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):217–226, 2022.

[36] P. Maddigan and T. Susnjak. Chat2VIS: Generating data visualizations via natural language using chatgpt, codex and gpt-3 large language models. *IEEE Access*, 11:45181–45193, 2023.

[37] D. Moritz, C. Wang, G. L. Nelson, H. Lin, A. M. Smith, B. Howe, and J. Heer. Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):438–448, 2019.

[38] A. Narechania, A. Srinivasan, and J. Stasko. NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):369–379, 2021.

[39] OpenAI. GPT-4 technical report. *arxiv.2303.08774*, 2023.

[40] L. Peng, Y. Zhao, Y. Hou, Q. Wang, S. Shen, X. Lai, J. Gao, J. Dong, Z. Lin, and S. Chen. Mixed-initiative visual exploration of social media text and events. In *Proceedings of the IEEE Conference on Visualization and Visual Analytics*, 2021.

[41] I. Pérez-Messina, D. Ceneda, M. El-Assady, S. Miksch, and F. Sperrle. A typology of guidance tasks in mixed-initiative visual analytics environments. *Computer Graphics Forum*, 41(3):465–476, 2022.

[42] X. Qian, R. A. Rossi, F. Du, S. Kim, E. Koh, S. Malik, T. Y. Lee, and J. Chan. Learning to recommend visualizations from data. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1359–1369, 2021.

[43] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, R. Tian, R. Xie, J. Zhou, M. Gerstein, D. Li, Z. Liu, and M. Sun. ToolLLM: Facilitating large language models to master 16000+ real-world apis. *arxiv.2307.16789*, 2023.

[44] R. Sevastjanova, M. El-Assady, A. Bradley, C. Collins, M. Butt, and D. Keim. VisInReport: Complementing visual discourse analytics through personalized insight reports. *IEEE Transactions on Visualization and Computer Graphics*, 28(12):4757–4769, 2022.

[45] L. Shen, E. Shen, Y. Luo, X. Yang, X. Hu, X. Zhang, Z. Tai, and J. Wang. Towards natural language interfaces for data visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 29(6):3121–3144, 2022.

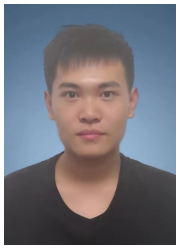
[46] D. Shi, F. Sun, X. Xu, X. Lan, D. Gotz, and N. Cao. AutoClips: An automatic approach to video generation from data facts. In *Computer Graphics Forum*, volume 40, pages 495–505. Wiley Online Library, 2021.
[47] F. Sperrle, D. Ceneda, and M. El-Assady. Lotse: A practical framework for guidance in visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1124–1134, 2022.
[48] J. Stasko, R. Amar, and J. Eagan. Low-level components of analytic activity in information visualization. In *Proceedings of the 2005 IEEE Symposium on Information Visualization*, page 15, 2005.
[49] C. Stoiber, D. Ceneda, M. Wagner, V. Schetinger, T. Gschwandtner, M. Streit, S. Miksch, and W. Aigner. Perspectives of visualization onboarding and guidance in va. *Visual Informatics*, 6(1):68–83, 2022.
[50] C. Stoiber, F. Stoiber, M. Pohl, H. Stitz, M. Streit, and W. Aigner. Visualization onboarding: Learning how to read and use visualizations. In *Proceedings of VisComm Workshop at IEEE VIS Conference*, 2019.
[51] G. M. Sullivan and R. Feinn. Using effect size—or why the P value is not enough. *Journal of graduate medical education*, 4(3):279–282, 2012.
[52] Tableau. Tableau embedding api. 2003.
[53] A. Tamkin, M. Brundage, J. Clark, and D. Ganguli. Understanding the capabilities, limitations, and societal impact of large language models. *arxiv.2102.02503*, 2021.
[54] Y. Tanahashi, N. Leaf, and K.-L. Ma. A study on designing effective introductory materials for information visualization. *Computer Graphics Forum*, 35(7):117–126, 2016.
[55] J. J. Thomas and K. A. Cook. *Illuminating the Path: An R&D Agenda for Visual Analytics*, pages 69–104. 2005.
[56] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi. Self-Instruct: Aligning language model with self generated instructions. *2212.10560*, 2022.
[57] Y. Wang, Z. Sun, H. Zhang, W. Cui, K. Xu, X. Ma, and D. Zhang. DataShot: Automatic generation of fact sheets from tabular data. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):895–905, 2020.
[58] A. Wu, D. Deng, F. Cheng, Y. Wu, S. Liu, and H. Qu. In defence of visual analytics systems: Replies to critics. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1026–1036, 2023.
[59] M. A. Yalcin. *A systematic and minimalist approach to lower barriers in visual data exploration*. PhD thesis, University of Maryland, College Park, 2016.
[60] S. Yousefi, L. Betthauser, H. Hasanbeig, A. Saran, R. Millière, and I. Momennejad. In-Context learning in large language models: A neuroscience-inspired analysis of representations. *arxiv.2310.00313*, 2023.
[61] L. Zha, J. Zhou, L. Li, R. Wang, Q. Huang, S. Yang, J. Yuan, C. Su, X. Li, A. Su, T. Zhang, C. Zhou, K. Shou, M. Wang, W. Zhu, G. Lu, C. Ye, Y. Ye, W. Ye, Y. Zhang, X. Deng, J. Xu, H. Wang, G. Chen, and J. Zhao. TableGPT: Towards unifying tables, nature language and commands into one GPT. *arxiv.2307.08674*, 2023.
[62] W. Zhang, Y. Shen, W. Lu, and Y. Zhuang. Data-Copilot: Bridging billions of data and humans with autonomous workflow. *arxiv.2306.07209*, 2023.
[63] J. Zhao, S. Xu, S. Chandrasegaran, C. Bryan, F. Du, A. Mishra, X. Qian, Y. Li, and K.-L. Ma. Chartstory: Automated partitioning, layout, and captioning of charts into comic-style narratives. *IEEE Transactions on Visualization and Computer Graphics*, 29(2):1384–1399, 2021.

Yuheng Zhao is currently a Ph.D student at the School of Data Science, Fudan University. Her primary research interests are visualization and visual analytics, with an emphasis on foundation model-powered intelligent visual analytics. For more information, please visit https://https://www.yuhengzhao.me/.

Yixing Zhang received a B.S. in Computer Science and Technology from Hefei University of Technology. He is currently a Master's student at the School of Data Science, Fudan University. His primary research interests are visualization and visual analytics, with a specific focus on intelligent and adaptive visualization recommendation and generation.

Yu Zhang received a B.S. degree in Intelligence Science and Technology from Peking University in 2017. Since then, he has been pursuing a Ph.D. at the Department of Computer Science, University of Oxford. His research focuses on intelligent user interfaces in the field of human-computer interaction.

Xinyi Zhao having earned a B.S. in Data Science and Big Data Technology from Fudan University, she is currently advancing her academic journey by pursuing a Master's degree in Data Science at Columbia University. Her primary research interests lie in applied machine learning, with a specific focus on enhancing data visualization techniques.



Junjie Wang is currently a master's student at the School of Data Science, Fudan University. His primary research interests are visualization and visual analytics, with a specific focus on intelligent and adaptive visual analytics.

Zekai Shao is a Ph.D. student at the School of Data Science, Fudan University. His main research interests include large model-powered visualization generation and evaluation, visual analytics, and explainable machine learning. For more information, please visit https://zekaishao25.github.io/.

Cagatay Turkay is a Professor at the Centre for Interdisciplinary Methodologies at the University of Warwick, UK and a Turing Fellow at the Alan Turing Institute, London, UK. His research investigates the interactions between data, algorithms and people, and explores the role of interactive visualisation and other interaction mediums such as natural language at this intersection. He frequently publishes his research on visualisation journals such as IEEE TVCG, CGF, and IEEE CG&A, as well as journals in machine learning and data mining, and also recently co-authored a coursebook titled "Visual Analytics for Data Scientists'. He has been awarded the EuroVis Young Researcher 2019 award and named a EuroGraphics Junior Fellow in 2019.

Siming Chen is an Associate Professor at School of Data Science, Fudan University. Prior to this, he was a Research Scientist at Fraunhofer Institute IAIS in Germany. He received his Ph.D. in computer science Peking University. His research interests are visualization and visual analytics, with the emphasis on Human-AI Collaboration, including LLM-driven visual analytics, social media and autonomous driving visual analytics. He has published 100 papers and more than 40 in top conferences and journals, including IEEE VIS, IEEE TVCG, EuroVis, ACM CHI, UIST, CSCW, etc. He served as multiple organizing chairs, committees and reviewers. He was awarded 10+ best paper/poster awards and honorable mentioned awards in multiple conferences For more information, please visit http://simingchen.me.