The Case for a *Visual Discovery Assistant*: A Holistic Solution for Accelerating Visual Data Exploration

Doris Jung-Lin Lee, Aditya Parameswaran
{jlee782,adityagp}@illinois.edu
University of Illinois, Urbana-Champaign

Abstract

Visualization is one of the most effective and widely-used techniques for understanding data. Yet, the growing use of visualizations for exploratory data analysis presses new demands beyond simply the graphical presentation and visualization authoring capabilities offered in existing tools. In particular, many data analysis tasks involve navigating large collections of visualizations to make sense of trends in data; at present, this navigation is done manually or programmatically. We outline a vision for an intelligent, interactive, understandable, and usable tool that can help automate this largely manual navigation: we call our tool VIDA[1]*—short for VIsual Discovery Assistant. We argue that typical navigation tasks can be organized across two dimensions—overall goal and precision of specification. We organize prior work—both our own work, as well as other ongoing work in this area—across these two dimensions, and highlight new research challenges. Together, addressing these challenges underlying* VIDA *can help pave the way for a comprehensive solution for removing the pain points in visual data exploration.*

1 Introduction

With the ever-increasing complexity and size of datasets, there is a growing demand for information visualization tools that can help data scientists make sense of large volumes of data. Visualizations help discover trends and patterns, spot outliers and anomalies, and generate or verify hypotheses. Moreover, visualizations are visceral and intuitive: they tell us stories about our data; they educate, delight, inform, enthrall, amaze, and clarify. This has led to the overwhelming popularity of point-and-click visualization tools like Tableau [27], as well as programmatic toolkits like ggplot, D3, Vega, and matplotlib. We term these tools as *visualization-at-a-time* approaches, since data scientists need to individually generate each visualization (via code or interactions), and examine them, one at a time.

As datasets grow in size and complexity, these visualization-at-a-time approaches start to break down, due to the limited time availability on the part of the data scientists—there are often too many visualizations to examine for a given task, such as identifying outliers, or inferring patterns. Even on a single table, visualizations can be generated by varying the subsets of data operated on, or the attributes (or combinations thereof) that

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

[1] Vida is short for *life* in Spanish.

can be visualized. If we add in various visualization modalities, encodings, aesthetics, binning methods, and transformations, this space becomes even larger.

Thus, there is a pressing need for an intelligent, interactive, understandable, usable, and enjoyable tool that can help data scientists navigate collections of visualizations. We term our hypothesized tool VIDA, short for *VIsual Discovery Assistant*. Data scientists specify their discovery goal at a high level, with VIDA automatically traversing visualizations to provide solutions or partial solutions for the specified discovery goal, thereby eliminating the tedium and wasted labor of comparable visualization-at-a-time approaches.

VIDA Dimensions. In order to be a holistic solution for navigating collections of visualizations, VIDA must be able to support various discovery settings. We organize these settings along two dimensions, displayed along the columns and rows (respectively) of Table 2—first, the overall discovery goal, and second, the degree of specificity of the goal. We also cite references (described later on) for systems that partially provide the necessary functionality for the given setting.

We identify five common discovery goals in visual data exploration, organized along the columns: *finding patterns*, *identifying anomalies/clusters*, *summarizing*, *performing comparisons*, *providing explanations*. These five goals borrow borrow from functionalities in existing systems, as well as related visualization task taxonomies [11, 2]. We omit low-level goals such as filtering or sorting, since these functionalities are common in visualization-at-a-time tools and toolkits. We also omit goals that go beyond visual data exploration, such as extrapolation, supervised learning, and cleaning, among others.

We identify three degrees of specificity for the discovery goal, organized along the rows: *precise*, *fuzzy*, *underspecified*. The degree of specificity characterizes the division of work between how much user has to specify versus how much the system has to automatically infer and aid in accomplishing the discovery goal. At the topmost row, the onus is placed on the user to provide an exact and complete specification of what the solution to their discovery goal must look like; at the middle row, the user can provide a vague specification of what the solution must look like; and finally, at the bottom row, the user provides a minimal specification, or leaves the characteristics of the solution underspecified, leaving it up to the system to "fill in" the rest. Naturally, as we proceed down the rows, it gets harder for the system to automatically interpret what the user might have in mind as a solution to their discovery goal.

Degree of Specificity	Discovery Goals				
	Find Patterns	Identify Anomalies/Clusters	Compare	Summarize	Explain
Precise (Section 2)	Zenvisage				
Fuzzy (Section 3)	ShapeSearch	Scorpion, Natural Language	SeeDB, Natural Language		Natural Language
Minimal (Section 4)		Storyboard			

Table 1: Overview of the systems described in this paper. Columns are organized into discovery goals and rows are ordered by decreasing levels of query specificity and correspondingly increasing levels of autonomous assistance.

At the topmost row, we find that while most discovery goals can be accomplished through exact and complete specification to query languages and programmatic APIs to statistical tools, these systems are often not easily accesible by non-experts.

In Section 2, we discuss how precise visual query systems help accelerate the process of finding desired visualizations, which in turn facillitates other highlighted discovery goals. Precise visual query system addresses the common problem of having to manually examine large numbers of visualizations in search of a desired pattern, which can be error-prone and inefficient. We discuss our work on *zenvisage* which allows analysts to specify their desired visualization through front-end interactions and automatically returns a ranked list of visualizations that closely matches with the input query.

Examples from our *zenvisage* design study demonstrates that precise querying alone is insufficient for addressing all the visual querying demands required in real-world use cases. In addition, we find that users often do not have a good idea of what they want to query for without looking at example visualizations or summaries of the data. To bridge the gap between user's high-level intent and what the system operates as inputs, we advocate that future research needs to look beyond simple precise visual querying by : 1) making visual query system more expressive by supporting a wider class of vague queries (Section 3) and 2) making it easier to know what to query by recommending visualizations that facilitate data awareness (Section 4).

Accordingly, the next row in the table highlights a growing class of *intelligent visual querying system* (IVQS) that interprets the 'vagueness' of queries and allow users to tweak or refine their queries through a feedback mechanism. Doris: I think we need to expand this definition depending on the new content that we will be adding, ShapeSearch, SeeDB, Scorpion?

To address the problem of guiding users to portions of the data that they might be interested in querying, Section 4 introduces systems that help users become more aware of their dataset and visualize where they are in their analysis workflow. The challenge in building these systems involves understanding what types of visualizations should be recommended to facilitate data awareness. As an example, we describe our work on STORYBOARD, a system that provides data summaries and guides users through informative subsets of data. Finally, we discuss related works on how visualizing provenance and situational information can guide users towards more informative analysis actions.

Our paper mainly focusses on the search and recommendation of visualization for data exploration, characterized by the use of visual perceptually-aware objectiveness, but also touches on related work in generic data querying. Related works on recommendation and automatic selection of the visualization design and encoding [31, 16] as well as frameworks and grammar for interactive visualization (Vega, D3) is out of the scope of this paper.

2 Precise Visual Querying

While visual analysis often reveal important anomalies or trends in the data [11, 18], it is often challenging to choose the right piece of data to visualize in order to realize these insights. In this section, we first motivate the use case for visual query systems and describe *zenvisage* as an example of such a system built to address these

	Discovery Goals				
	Find Patterns	Identify Anomalies/Clusters	Compare	Summarize	Explain
	Query Languages + Statistical APIs				
Precise Visual Query System	Zenvisage				
Intelligent Visual Query System	ShapeSearch	Scorpion, Natural Language	SeeDB, Natural Language		Natural Language
Recommendation Guidance System		Storyboard			

Table 2: Overview of the systems described in this paper. Columns are organized into discovery goals and rows are ordered by decreasing levels of query specificity and correspondingly increasing levels of autonomous assistance.

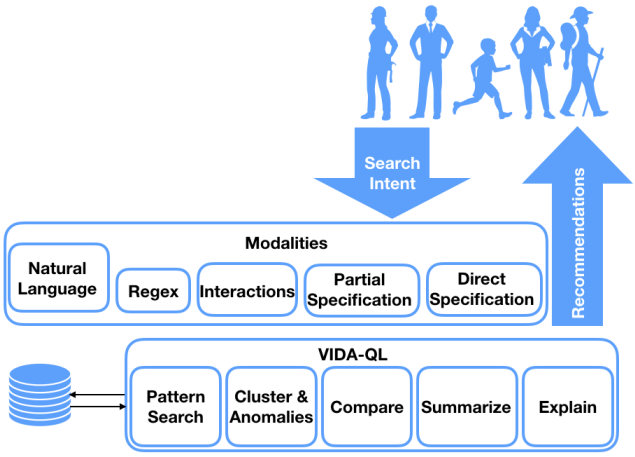

Figure 1: VIDA Architecture

challenges.

2.1 Motivating Example

Astronomers from the the Dark Energy Survey (DES) are interested in finding anomalous time series to discover astrophysical transients (objects whose brightness changes dramatically as a function of time), such as supernova explosions or quasars [6]. When trying to find celestial objects corresponding to supernovae, which have a specific pattern of brightness over time, scientists need to individually inspect the visualizations of each object until they find ones that match the pattern. With more than 400 million objects in their catalog, each having their own set of time series brightness measurement, the process of manually exploring a large number of visualizations is not only error-prone, but also overwhelming for scientists who do not have extensive knowledge about their dataset. The astronomy use case highlights a common challenge in exploratory data analysis, where is often a large space of possible visualizations that could be generated from a given dataset and manual search through this large collection is inefficient. Popular visualization authoring tools such as Tableau and Excel focus on presenting one visualization at a time and there is no systematic way to create, compare, and query large collections of visualizations.

2.2 Effortless Data Exploration with *zenvisage*

The challenges discussed in the previous section point to the need for a tool that enables users to create and search through large collections of visualizations. To this end, we developed *zenvisage*, a *precise visual querying system (PVQS)* that accepts precise queries in the form of a desired pattern specified through frontend interactions and returns a ranked list of visualizations that look similar to the input pattern [26]. *zenvisage* is built on top of a querying language called ZQL, which provides a mechanism for managing collections of visualizations. Contrary to prior work on visualization languages for specifying visual encodings of individual visualizations [27, 29], ZQL supports high-level queries over visualization collections, such as composing, sorting, filtering a collection of visualization. ZQL functionals and primitives can be constructed into rich and expressive query semantics, with functionalties including:

- Finding top-k visualizations whose y values are most or least similar from a queried visualization. (e.g. Find cities with sold price over time similar to Manhattan. Varying along CITY while keeping X = TIME, Y = AVG(PRICE) fixed.)

- Comparing across a collection of visualizations by iterating over one or more x, y, z attributes while fixing other attributes. (e.g. Find an y attribute that varies with time similarly how average price changes over time)
- Finding a pair of X and Y axes where two specific visualization instances differ the most. (e.g. Finding pairs of attributes where visualizations for the products 'stapler' and 'chair' differ the most.)

Given a ZQL query, *zenvisage* parses the query into a graph of visual component nodes (containing visualization information, such as X, Y columns) and task nodes (common and user-defined primitives for processing visual components, such as sort or filter). *zenvisage* then performs query optimization to merge together multiple nodes, as well as reducing the processing time required for individual visualization components. Using the optimized query plan, the executor compiles visual nodes into SQL queries for retrieving the visualization data and postprocesses the result via the defined operations.

While ZQL provides powerful mechanism for expressively specifying queries on large collections visualizations, writing ZQL queries can be daunting for novice users. Therefore, we extracted a typical workflow of visualization querying (finding top-k most similar visualization from a collection with fixed X,Y while varying Z) to allow users to formulate ZQL queries through interactions. The user can either perform frontend interactions mapped onto ZQL queries or directly input ZQL queries through a table input. The query results are rendered as a ranked list of visualizations in the results panel in the interface. *zenvisage* is a full-fledged visual querying system that supports a variety of querying interactions as illustrated in Figure 2. *zenvisage* also relate to the discovery goal of identifying anomalies and clusters through displaying visualizations of the representative trends and outliers in the dataset. Users can also make comparisons between different subsets of the data by examining visualizations of dynamically created classes in *zenvisage*, described more in our paper [15].

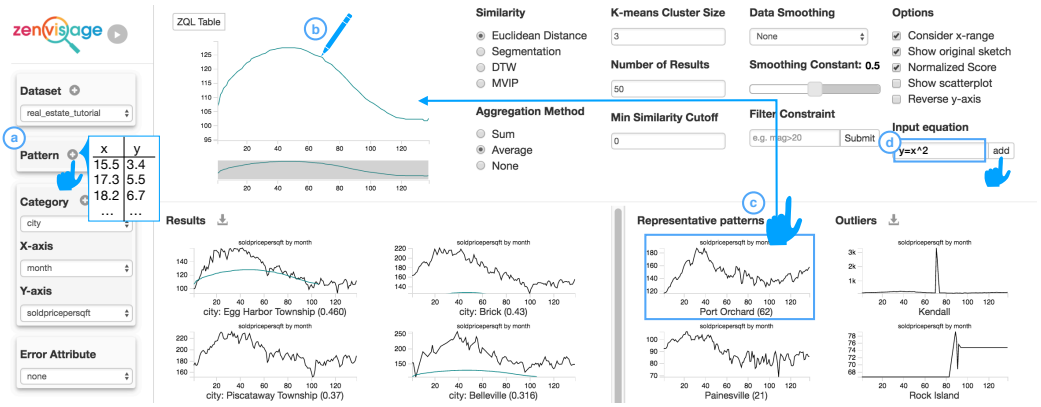

Figure 2: *zenvisage* offers a variety of querying modalities, including: a) uploading a sample pattern from an external dataset as a query, b) sketching a query pattern, c) dragging-and-dropping an existing pattern from the dataset, and d) inputting an equation as a query, from [15].

2.3 Challenges of Precise Visual Querying Systems

In developing *zenvisage*, we collaborated with scientists from astronomy, genetics, and material science in a year-long participatory design process [15]. In particular, we studied how various features impact analysts' ability to rapidly generate new hypotheses and insights and perform visual querying and analysis. Our findings not only offers design guidelines for improving the usability and adoption of next-generation PVQSs, but more importantly, points towards the need for supporting other components in the cycle of visual data exploration.

The Problem of Interpreting Ambiguous, High-level Queries

When users query with a PVQS, they often translate their ambiguous, high-level questions into an plan that consists of multiple interactions to incrementally address their desired query. The expressiveness of PVQSs comes from the multiplicative effect of stringing together combinations of interaction sequences into a customized workflow. Designing features that diversifies potential variations expands the space of possible workflows that could be constructed during the analysis. However, even with many supported interactions, there were still vague and complex queries that could not be decomposed into a multi-step interaction workflow. For example, *zenvisage* was unable to support high-level queries that involved the use of vague descriptors for matching to specific data characteristics, such as finding patterns that are 'flat and without noise', or 'exhibits irregularities'. These scenarios showcase examples of lexical ambiguity, where the PVQS can not map the vague term 'irregular' or 'noisy' into the appropriate series of analysis steps required to find these patterns. In Section 3, we survey the challenges of PVQSs in supporting vague and complex queries and point to several promising directions of ongoing research in this area.

The Problem of Not Knowing What to Query

Another key finding of our study is that users often do not start their analysis with a pattern in mind, so without knowing what the data looks like, they may not always have a intended pattern to query for. For example, we found that many users first made use of the recommended representative trends and outlier visualizations provided by *zenvisage* as contextual information to better understand their data or to query based on these recommended visualizations, then query using these recommended visualizations in a bottom-up approach, rather than coming up with a query in a top-down, prescriptive manner. The takeaway design principle is the need for incorporating visualization recommendations that can help analysts jump-start their exploration. Recommendations facilitate a smoother flow of analysis by closing the loop between the two modalities of querying and exploration, reminiscent of the browsing and searching behaviors on the Web [20], thus ensuring that user is never stuck or out of ideas at any point during the analysis. Typically, visualization recommendation system seeks to accelerate the process of discovering interesting aspects of the data by broadening exploration. In Section **??**, we advocate that recommendation systems should not only focus on data discoverability aspect of exploration, but also contribute towards helping users become more aware of the distributions in their data and the context of their analysis.

3 Towards Intelligent Visual Search

3.1 The Challenge of Usability-Expressivity Tradeoff

The challenge for supporting vague and complex querying stems from the inevitable design trade-off between query expressivity and interface usability in interactive data exploration systems [13, 18]. This tradeoff is observed not only in visual data exploration systems, but also true for general ad-hoc data querying. While querying language such as SQL are highly expressive, formulating SQL queries that maps user's high-level intentions to specific query statements is challenging [13, 14]. As a result, query construction interfaces have been developed to address this issue by enabling direct manipulation of queries through graphical representations [1], gestural interaction [19], and tabular inputs [35, 7]. For example, form-based query builders often consist of highly-usable interfaces that ask users for a specific set of information mapped onto a pre-defined query. However, form-based query builders are often based on query templates with limited expressiveness in their semantic and conceptual coverage, which makes it difficult for expert users to express complex queries. The extensibility of these systems also comes with the high engineering cost, as well as potentially overloading the users with too many potential options to chose from. There is a need for tools that is enables users to formulate rich and complex queries, yet highly usable even for novices.

3.2 Ongoing research and opportunities

Given the tradeoff between expressivity and usability, we can not assume a one-size-fit-all PVQS that could fit the needs for users of different expertise levels and workloads. In this section, we discuss a growing class of *intelligent visual querying system* (IVQS) that works around this problem by taking into account queries of varying degrees of input specificity. There are many open research questions in accelerating the discovery process of answering an imprecise, fuzzy, and complex queries, including : How can we develop better ways to resolve ambiguity by infering the information needs and intent of an analyst? What is the appropriate level of feedback and interactions in IVQSs? How can we develop interpretable visual metaphors that could serve to explain how the query was interpreted and why specific query results are returned to a lay user? We describe three different system challenges in interpreting non-precise queries. Since most visual query systems operates on top of a querying language, we use the linguistical classification scheme to provide an analogy for where the ambiguous aspects of queries may arise during visual data exploration, noting that the use of this analogy by no means limits our analysis to only natural language interfaces.

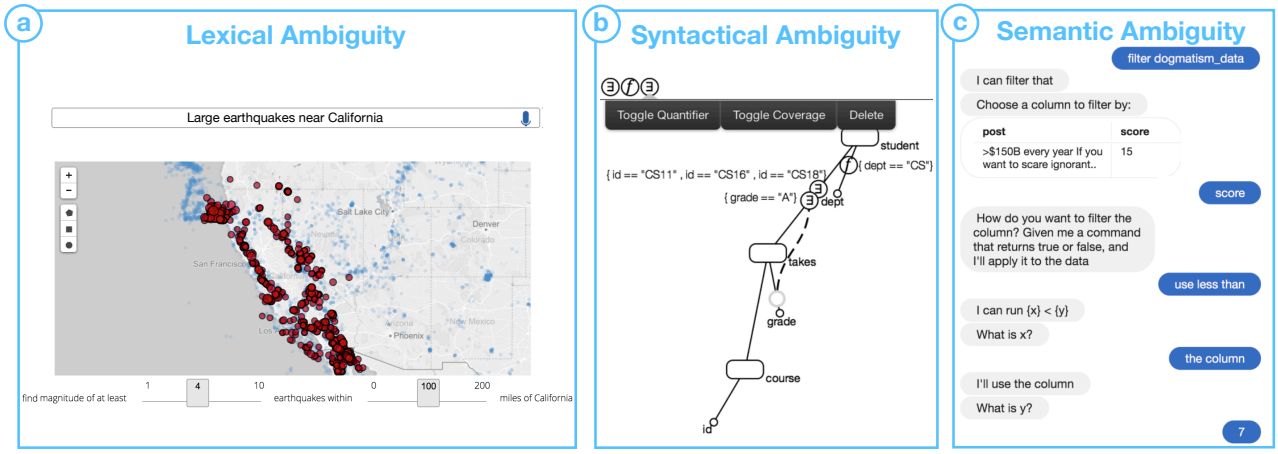

Figure 3: Examples of IVQSs: a) Eviza [25] uses ambiguity widgets to enable users to clarify vaguely-defined terms in their input query to resolve lexical ambiguity; b) DataPlay [1] allow users to toggle between 'for all' and 'at least one' quantifier to control for syntactical ambiguity; c) Iris [8] accepts a vague high-level task description and gather the additional information required through follow-up questions in a nested conversation.

Lexical Ambiguity: Lexical ambiguity involves the use of vague descriptors in the input queries. Resolving these lexical ambiguities has been a subject of research in natural language interfaces (NLIs) for visualization specification, such as DataTone [9] and Eviza [25]. These NLIs detects ambiguous quantifiers in the input query (e.g. "Large earthquakes near California"), and then displays ambiguity widgets in the form of a widget to allow users to specify the definition of 'large' in terms of magnitude and the number of miles radius for defining 'near', as shown in Figure 3a. These ambiguity widgets not only serve as a way to provide feedback to the system for lexically vague queries, but also is a way for displaying interpretable explanations of how the system is interpreting the input queries. If we consider IVQSs as a layer on top of PVQS (that performs functionalities such as shape-matching, filtering), an IVQS resolves lexical ambiguity by determining the appropriate *parameters* to the PVQS to achieve the user's desired querying effects. Doris: Describe ShapeSearch here

Syntactic Ambiguity: Syntactic ambiguity is related to the vagueness in specifying how the query should be structured or ordered. For example, DataPlay introduced the idea of syntax non-locality in SQL, in which switching from an existential (at least one) to a universal (for all) quantifier requires major structural changes to the underlying SQL query [1]. As shown in Figure 3b, DataPlay consist of a visual interface that allowed users to directly manipulate the structure of the query tree in tweaking the query to its desired specification.

IVQSs resolve syntactic ambiguities either by mapping portions of the vague queries into to *a series of multi-step workflows* to be executed in the PVQS and allow users to tweak the query representation directly. The query modification is done in a declarative manner in that the underlying mechanism in which the visualized workflow gets translated to the querying language is largely hidden from the end-user.

Semantic Ambiguity: Semantic ambiguity includes addressing 'why' questions that are inherently semantically hard to answer, such as 'why do these datapoints look different from others?'. For example, Wu and Madden [33] have developed Scorpion which traces the provenance of a set of input data tuples to find predicates that explain the outlier behavior. Semantic ambiguity can also arises when the user does not specify their intent completely or explicitly, which is often the case in the earlier stages of the visual data exploration. NLIs for visual data exploration such as Evizeon [12] makes use of anaphoric references to fill in incomplete follow-on queries. For example, when a user says 'Show me average price by neighborhood', then 'by home type', the system interprets the anaphoric reference as continuing the context of the original utterance related to average price on the y-axis. Semantic ambiguity can often be composed of one or more lexical and syntactical ambiguity. For example, in Iris [8], a user can specify a vague, high-level query such as 'Create a classifier', then Iris makes use of nested conversations to inquire about what type of classifier to chose and what features to use in the model to fill in the details of the structure and parameters required. A semantically vague query may or may not be expressible through a single PVQS, since the operations involved in the query may not be covered by the limited workflow combinations in the PVQS.

4 Towards Dataset Understanding

While our focus in the previous sections have been on intention-driven queries, where users have some knowledge of what types of questions he may be interested in, one of the key goals of visual data exploration is to promote a better understanding of the dataset to enable users to make actionable decisions. This section discusses systems that helps users become more aware of their dataset and visualize where they are in their analysis workflow. Such systems can often be useful in situations where there is an absence of explicit signals from the user, which can happen when a user is at the beginning of their analysis (commonly known as the 'cold-start' problem) or when the user does not know what to query for. In this section, we will describe STORYBOARD, a system that provides data summaries and guides users through informative subsets of data, as an example of a system that promotes distribution awareness in a query-free scenario. Then, we will discuss other types of data understanding during dynamic visual data exploration to highlight the challenges and opportunities ahead in this space.

4.1 SEEDB: Exploring Multi-variate Data Dimensions Through Interesting Visualization Recommendations

Identifying interesting trends in high-dimensional datasets is often challenging due to the large combination of possible X,Ys for the generated visualizations. To perform this task, the analyst needs to manually create, examine, and compare the relationships between these multi-variate visualizations to discover interesting insights.

SEEDB is a visualization querying and recomendation environment where user can specify a subset of data that they would be interested in exploring, by indicating the x, y axes and aggregation function of its visualization [28]. Then, SEEDB searches for other views in the reference that are *interesting* with respect to the input view. SEEDB defines a visualization as interesting if it deviates greatly from the specified reference. SEEDB performs smart pruning, sharing optimization to make this process faster. The evaluation user study showed that the SEEDB-recommended visualizations are three times more likely to be interesting compared to manually constructed visualizations and provide an effective starting point for further analysis.

4.2 STORYBOARD: Promoting Distribution Awareness of Data Subsets with Summary of Visualizations

Whereas SEEDB looks at space of possible X,Ys, given a fixed filter and aggregation to recommend interesting visualizations, our next system STORYBOARD explore through the space of possible filters, given a fixed input X,Y and aggregation function. Common analytics tasks, such as causal inference, feature selection, and outlier detection, require understanding the distributions and patterns present in the visualizations at differing levels of data granularity [3, 11, 33]. However, it is often hard to know *what* subset of data contains an insightful distribution to examine. In order to explore different data subsets, an analyst would first have to construct a large number of visualizations corresponding to all possible data subsets, and then navigate through this large space of visualizations to draw meaningful insights. While there are some related work in database literature in constructing informative summaries that help guide users through the complex schema of object-oriented databases[17, 34], these are often focused on table and attribute level information, rather than information about derived from the actual data distributions. The lack of a systematic way to perform these exercises makes the process of manually exploring distributions from all possible data subsets tedious and inefficient [22, 23].

To this end, we developed STORYBOARD, an interactive visualization summarization system that automatically selects a set of visualizations to summarize the distributions within a dataset in an informative manner. Figure 4 illustrates an example dashboard generated by STORYBOARD from the Police Stop Dataset [21], which contains records of police stops that resulted in a warning, ticket, or an arrest. The attributes in the dataset include driver gender, age, race, and the stop time of day, whether a search was conducted, and whether contraband was found. We requested STORYBOARD to generate a dashboard of 9 bar chart visualizations with x-axis as the stop outcome (whether the police stop resulted in a ticket, warning, or arrest/summons) and y-axis as the percentage of police stops that led to this outcome. First, at the top of our dashboard, STORYBOARD highlights three key data subsets that results in a high arrest rate, which looks very different trend than the overall (where the majority of stops results in tickets). Following along the leftmost branch, we learn that even though in general when a search is conducted, the arrest rate is almost as high as ticketing rate, when we look at the Asian population, whether a search is conducted had less influence on the arrest rate and the trend resembles more like the overall distribution. Doris: Summarize objective in a couple sentences in the above paragraph and point to paper for

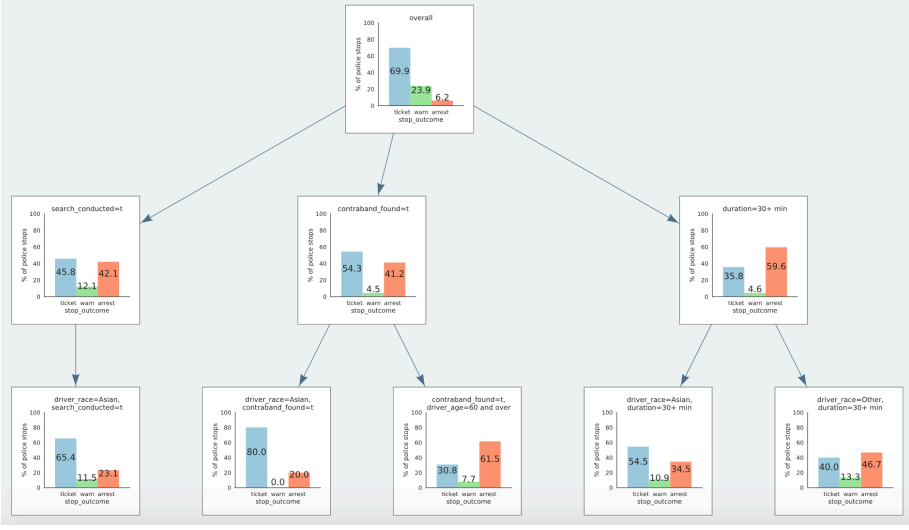

Figure 4: Example dashboard generated by STORYBOARD summarizing the key insights in the Police dataset.

details.

While such summary dashboards are useful for making sense of relationships between data subsets, finding

effective visualizations to summarize a dataset is not as trivial as picking individual visualizations that maximizes some statistical measure, such as deviation [28], coverage [24], or significance testing [3], which can often result in misleading summarizations. The key idea behind our work is understanding how analysts formulate their expectations regarding an unseen visualization in a *data subset lattice*. Applying the idea of data subset lattice from data cube literature [10] to filtered bar chart visualizations, we define a visualization as the *parent* of another visualization if the latter visualization can be derived from the first visualization by adding one additional filter constraint. Our formative user study showed that people naturally form their expectations regarding an unseen visualization based on one or more observed parents and that seeing a parent that well describes the unseen visualization leads participants to better estimate the unseen visualization. More importantly, in the absence of an informative parent or in the presence of multiple parents, participants can be misled to form an inaccurate expectation that exhibit higher variance.

Given these insights, the goal of our system is to select *interestingness* and *informativeness* visualizations that can help them make more accurate predictions regarding the unseen visualizations. To model the informativeness of an observed parent in the context of an unseen visualization, we characterize the capability of the parent in predicting the unseen visualization. Our study shows that a visualization is *informative* if its data distribution closely follows the data distribution of the unseen child visualization, since the visualization helps the analyst form an accurate mental picture of what to expect from the unseen visualization. While informative parents contribute to the prediction of an unseen visualization, the most interesting visualizations to recommend are those for which *even the informative parents fail to accurately predict or explain the visualization*. Our problem of constructing the summary dashboard then becomes the problem of finding the k connected visualizations that are most informatively interesting according to this objective. Detailed treatments of our metrics and algorithms can be found in our technical report. Doris: (CITE PLACEHOLDER) Can we put a version of the STORYBOARD paper on arxiv so that we can cite it?

The effectiveness of STORYBOARD largely comes from how the summary visualizations help analysts become more distributionally aware of the dataset. We define *distribution awareness* as the aspect of data understanding in which analysts make sense of the key distributions across different data subsets and their relationships in the context of the dataset. With distribution awarenes, even though it may be infeasible for an analyst to examine all possible data subsets, the analyst will still be able to draw meaningful insights and establish correlations about related visualizations by generalizing their understanding based on the limited number of visualizations presented in the dashboard. Our evaluation study shows that facilitating distribution awareness through STORYBOARD guides analysts to make better predictions regarding unseen visualizations, ranking attribute importance, and retrieval of interesting visualizations compared to dashboards generated from the baselines.

4.3 From Distributional to Contextual Awareness: Challenges and Opportunities

The notion of distribution awareness is useful when considering the scenario at one static point in time of the analysis, such as during cold-start. In this section, we introduce a complementary notion of data understanding called *contextual awareness*, which is essential when considering a dynamic analytic workflow for visual data exploration.

Contextual awareness is the aspect of data understanding related to the *situation* (what is the information that I'm currently looking and how did it come about?) and *provenance* (what have I explored in the past and where should I look next?) of data. Situational understanding involves recognizing what data is in the current scope of analysis, including making sense of the data attributes and schema and keeping track of what filter or transformations have been applied to the displayed data. Provenance understanding is associated with the analyst's past analysis actions on the data. As an example, an analyst may be interested in how the sales price of a product changes as a function of other dimensions variables, such as geographic location, year sold, and product type. Situation information informs him that he is looking at a bar chart with X=TYPE, Y=AVG(PRICE),

whereas provenance information points to the fact that he should explore the geographic dimension, since he has already explored the temporal attribute YEAR.

Within a dataset, provenance is essential in helping users navigate through the space of possible analysis actions and provide users with sense of coverage and completion. While the problem of data provenance has been well studied in database literature [4, 5, 32], the effects of showing provenance information to users during data analysis is an important but underexplored area. The notion of adding navigational cues to guide exploration in visual information spaces was first proposed in Willet et al.'s work on *scented widgets* [30]. In Pirolli and Card's theory of information foraging, scents are cues that signifies the percieved benefit that one would recieve during a search. Scented widgets adds to existing search interfaces by embedding visualizations that provide informational scents, such as histogram distributions of how popular a particular value is among users or using color to encode the size of a dataset in a drop-down menu. Recently, Sarvghad et al. have extended the idea of scented widgets to incorporate dimension coverage information during data exploration, including which dimensions have been explored so far, in what frequency, and in which combinations [24]. Their study shows that visualizing dimension coverage leads to increased number of questions formulated, findings, and broader exploration. Interpretable and non-disruptive cues that enables users to visualization provenance history helps sustain contextual awareness and guides users towards more informative next steps in their analysis.

Mechanisms that facilitate distribution awareness for users can effectively couple with contextual awareness in dynamic exploration situations to help update the user's mental model on the current data context. For example, the representative and outlier patterns in *zenvisage* provides summaries of data in context. When a dataset is filtered, the representative trends are updated accordingly. By being aware of both the context and the distributions, the users becomes distributionally aware of how the typical patterns and trends of the distributions changes in a particular context.

While our discussion above have been focused on how to design systems that can help facilitate these aspects of user's awareness in dataset understanding, these ideas can be generalized to principles in designing IVQS discussed in Section 3. An IVQS needs to be distributionally and contextually aware, by make use of information about the data (distribution awareness), the analytic context, and situation jointly in making timely recommendations. In other words, these systems should not only facilitate these aspects of data awareness, but also need to make use of this information to make recommendations that can guide analysts towards meaningful stories and insights for further investigation.

5 Concluding Remarks

Data is inherently agnostic to the diverse information needs any user may have. Visual data exploration systems help bridge the gap between what users want to get from the data through querying and what insights the data has to offer through recommendations. To more facilitate a more productive collaboration, in this paper, we discuss how precise visual query systems provide informative visualizations to accelerate the process of data discovery, how intelligent visual query systems account for vague and complex query through query refinement feedback, and how recommendations could promote better distributional and contextual awareness for users. We hope that the agenda sketched out in this paper sheds light to the many more exciting research questions and opportunities to come in this nascent growing field.

Acknowledgement

A. P. acknowledges support from grants IIS-1513407 and IIS-1633755 awarded by the National Science Foundation, grant 1U54GM114838 awarded by NIGMS and 3U54EB020406-02S1 awarded by NIBIB through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov), and funds from Adobe, Google, and the Siebel Energy Institute. The content is solely the responsibility of the authors and does

not necessarily represent the official views of the funding agencies and organizations. Doris: copied this from the crowdsourcing vision paper, might have to change grant number and sources accrodingly. Might also want to include acknowledgement for collaborators here too.

References

[1] Azza Abouzied, J Hellerstein, and A Silberschatz. Dataplay: interactive tweaking and example-driven correction of graphical database queries. *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 207–217, 2012.

[2] Robert Amar, James Eagan, and John Stasko. Low-level components of analytic activity in information visualization. *Proceedings - IEEE Symposium on Information Visualization, INFO VIS*, pages 111–117, 2005.

[3] Anushka Anand and Justin Talbot. Automatic Selection of Partitioning Variables for Small Multiple Displays. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):669 – 677, 2015.

[4] Peter Buneman, Adriane Chapman, and James Cheney. Provenance management in curated databases. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, SIGMOD '06, pages 539–550, New York, NY, USA, 2006. ACM.

[5] Y. Cui and J. Widom. Lineage tracing for general data warehouse transformations. *Proceedings of the VLDB Endowment*, 12(1):41–58, May 2003.

[6] Drlica Wagner et al. Dark energy survey year 1 results: The photometric data set for cosmology. *The Astrophysical Journal Supplement Series*, 235(2):33, 2018.

[7] David W. Embley. Nfql: The natural forms query language. *ACM Transactions on Database Systems (TODS)*, 14(2):168–211, June 1989.

[8] Ethan Fast, Binbin Chen, Julia Mendelsohn, Jonathan Bassen, and Michael S. Bernstein. Iris: A conversational agent for complex tasks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 473:1–473:12, New York, NY, USA, 2018. ACM.

[9] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G. Karahalios. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, UIST '15, pages 489–500, New York, NY, USA, 2015. ACM.

[10] Venky Harinarayan, Anand Rajaraman, and Jeffrey D. Ullman. Implementing data cubes efficiently. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, SIGMOD '96, pages 205–216, New York, NY, USA, 1996. ACM.

[11] Jeffrey Heer and Ben Shneiderman. Interactive Dynamics for Visual Analysis. *ACM Queue*, 10(2):30, 2012.

[12] Enamul Hoque, Vidya Setlur, Melanie Tory, and Isaac Dykeman. Applying Pragmatics Principles for Interaction with Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics*, (c), 2017.

[13] H. V. Jagadish, Adriane Chapman, Aaron Elkiss, Magesh Jayapandian, Yunyao Li, Arnab Nandi, and Cong Yu. Making database systems usable. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, pages 13–24, New York, NY, USA, 2007. ACM.

[14] Nodira Khoussainova, YongChul Kwon, Magdalena Balazinska, and Dan Suciu. Snipsuggest: Context-aware autocompletion for sql. *Proceedings of the VLDB Endowment*, 4(1):22–33, 2010.

[15] Doris Jung-Lin Lee, John Lee, Tarique Siddiqui, Jaewoo Kim, Karrie Karahalios, and Aditya G. Parameswaran. Accelerating scientific data exploration via visual query systems. *CoRR*, abs/1710.00763, 2017.

[16] Jock D. Mackinlay, Pat Hanrahan, and Chris Stolte. Show Me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, 2007.

[17] J McHugh, S Abiteboul, R Goldman, D Quass, and J Widom. Lore: A database management system for semistructured data. *SIGMOD Record*, 26(3):238–247, 1997.

[18] Kristi Morton, Magdalena Balazinska, Dan Grossman, and Jock Mackinlay. Support the Data Enthusiast: Challenges for Next-Generation Data-Analysis Systems. *Proceedings of the VLDB Endowment, Volume 7, pp. 453–456, 2014*, 7:453–456, 2014.

[19] Arnab Nandi, Lilong Jiang, and Michael Mandel. Gestural query specification. *Proceedings of the VLDB Endowment*, 7(4):289–300, 2013.

[20] Christopher Olston and E D H Chi. ScentTrails: Integrating browsing and searching on the Web. *ACM Transactions on Computer Human Interaction TOCHI*, 10(3):177–197, 2003.

[21] E. Pierson, C. Simoiu, J. Overgoor, S. Corbett-Davies, V. Ramachandran, C. Phillips, and S. Goel. A large-scale analysis of racial disparities in police stops across the united states, 2017.

[22] Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. Discovery-driven exploration of olap data cubes. In *Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '98, pages 168–182, Berlin, Heidelberg, 1998. Springer-Verlag.

[23] S. Sarawagi. User-adaptive exploration of multidimensional data. *Proceedings of the VLDB Endowment*, pages 307–316, 2000.

[24] Ali Sarvghad, Melanie Tory, and Narges Mahyar. Visualizing Dimension Coverage to Support Exploratory Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):21–30, 2017.

[25] Vidya Setlur, Sarah E Battersby, Melanie Tory, Rich Gossweiler, and Angel X Chang. Eviza: A Natural Language Interface for Visual Analysis. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16*, pages 365–377, 2016.

[26] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. Effortless data exploration with zenvisage: An expressive and interactive visual analytics system. *Proceedings of the VLDB Endowment*, 10(4):457–468, November 2016.

[27] C. Stolte, D. Tang, and P. Hanrahan. Polaris: a system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–14, 2002.

[28] Manasi Vartak, Samuel Madden, and Aditya N Parmeswaran. SEEDB : Supporting Visual Analytics with Data-Driven Recommendations. 2015.

[29] Leland Wilkinson. *The Grammar of Graphics (Statistics and Computing)*. Springer-Verlag, Berlin, Heidelberg, 2005.

[30] Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1129–1136, 2007.

[31] Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. Voyager 2 : Augmenting Visual Analysis with Partial View Specifications. 2017.

[32] Allison Woodruff and Michael Stonebraker. Supporting fine-grained data lineage in a database visualization environment. In *Proceedings of the Thirteenth International Conference on Data Engineering*, ICDE '97, pages 91–102, Washington, DC, USA, 1997. IEEE Computer Society.

[33] Eugene Wu and Samuel Madden. Scorpion: Explaining Away Outliers in Aggregate Queries. *Proceedings of the VLDB Endowment*, 6(8):553–564, 2013.

[34] Cong Yu and H. V. Jagadish. Schema summarization. In *Proceedings of the VLDB Endowment*, VLDB '06, pages 319–330. VLDB Endowment, 2006.

[35] Moshe M Zloof. Query by Example. *National Computer Conference*, pages 431–438, 1975.