A Case Study Using Visualization Interaction Logs and Insight Metrics to Understand How Analysts Arrive at Insights

Hua Guo, Steven R. Gomez, Caroline Ziemkiewicz, David H. Laidlaw *Fellow, IEEE*

Abstract— We present results from an experiment aimed at using logs of interactions with a visual analytics application to better understand how interactions lead to insight generation. We performed an insight-based user study of a visual analytics application and ran post hoc quantitative analyses of participants' measured insight metrics and interaction logs. The quantitative analyses identified features of interaction that were correlated with insight characteristics, and we confirmed these findings using a qualitative analysis of video captured during the user study. Results of the experiment include design guidelines for the visual analytics application aimed at supporting insight generation. Furthermore, we demonstrated an analysis method using interaction logs that identified which interaction patterns led to insights, going beyond insight-based evaluations that only quantify insight characteristics. We also discuss choices and pitfalls encountered when applying this analysis method, such as the benefits and costs of applying an abstraction framework to application-specific actions before further analysis. Our method can be applied to evaluations of other visualization tools to inform the design of insight-promoting interactions and to better understand analyst behaviors.

Index Terms—Evaluation, visual analytics, interaction, intelligence analysis, insight-based evaluation

1 Introduction

In this paper, we present an evaluation of a visual analysis system developed for analyzing document collections structured as spatiotemporal networks. To understand how analysts use this system to generate insights, we used a hybrid evaluation approach, which combines a standard insight-based study with interaction log analysis.

Insight-based evaluation has received increasing attention in the visualization community in recent years [21, 24, 6]. Traditional visualization evaluation metrics such as task accuracy and completion time are straightforward to compute and provide useful benchmarks, but they do not capture the whole story of how analysts use a visualization application to arrive at insights. Insight-based evaluation, on the other hand, lets visualization researchers and designers compare applications based on the insights analysts can gain using the system, which often reflects the practical design goal of visual analytics applications. However, the standard insight-based evaluation methodology, as presented by Saraiya et al. [27], does not prescribe a principled way to evaluate the connection between what actions an analyst performs and the insights she generates. This knowledge is critical for understanding how the design of interactive components in a visual analytics application promotes or inhibits insight generation.

The primary goal of the hybrid evaluation approach is to address this challenge of understanding how application design influences insight generation. In this case study, we performed quantitative analysis of insight characteristics and features extracted from interaction logs. The results were used to orient the subsequent qualitative analysis, where we reviewed interesting usage patterns identified from the quantitative analysis in the context of the user study sessions and derived design recommendations.

The contributions of the paper are threefold: 1) a hybrid evaluation approach to address evaluation goals that cannot be fully achieved using standard insight-based evaluation; 2) a proof-of-concept case study that demonstrates the practical use of the evaluation approach and reveals some of the design choices involved in the approach; 3) results of the evaluation approach applied to a specific visual analytics system, including identified analysis patterns and design recommendations.

- *Hua Guo is with Brown University. E-mail: huag@cs.brown.edu.*
Steven R. Gomez is with Brown University. E-mail: steveg@cs.brown.edu.
Caroline Ziemkiewicz is with Aptima Inc. Email: cziemkiewicz@aptima.com.
David H. Laidlaw is with Brown University. E-mail: dhl@cs.brown.edu.

2 Related Work

Our approach of analyzing insights alongside interaction histories builds on previous evaluations of visualization applications. The most related methodologies include 1) interaction and workflow analysis, and 2) insight-based evaluation. To the best of our knowledge, a systematic investigation of the relationship between insights and interaction in visual analytics has not yet been performed, and we believe this an early step toward designing visual analytics applications that support the goals and workflows of analysts.

2.1 Analyzing Interactions

Studying how analysts use interaction in visualization systems is an important part of evaluating how well these interactions support analysis needs, like generating insights or performing tasks. Histories of user interactions have been used to advance our understanding of tool usage and user goals in a variety of areas (e.g. [2, 7, 17]). For visual analytics tools, user interaction histories contain information about the sequence of choices that analysts make when exploring data or performing a task. They help evaluators identify which of the available interactions in an application are preferred by analysts or are part of a critical path of interactions that is necessary for a task.

In the past, both automatic analyses of interactions and manual reviews have resulted in discovering design improvements for visualization applications and workflows. In some cases, observing interactions is the basis for cognitive and behavioral models of end users, and these models can predict the outcome of design changes to applications and workflows [14], including benchmark tasks for visualization [11]. In other cases, interaction logs have been used to classify users with different task strategies and personality traits, as well as to predict the performance of basic visualization tasks like visual search [5].

In addition to predictive models of behavior, identifying the roles of specific interactions in the analysis process has led to descriptive models of how analysts use visualization to make sense of data, including the *learning loop complex* [26], and Pirolli and Card's cascading *foraging* and *sensemaking* loops [23]. In a case study with the popular visual analytics application Jigsaw, Kang et al. found that analysts' interaction histories showed evidence of these high-level sensemaking processes [18]. Similarly, Boyandin et al. [4] visualized user interaction logs to compare the use of interaction techniques to arrive at findings with flow maps under animation versus small-multiples conditions. Reda et al. [25] approached interaction and sensemaking from a different angle, combining interaction logs and user-reported mental processes into an extended log and modeling the log using transition diagrams to better understand the transition between mental and inter-

action states. Like Kang's case study, we identify and interpret patterns of interactions with a visual analytics application using a qualitative review process. In addition, we describe how a quantitative, automated analysis of interaction histories using methods including but not limited to transition diagram analysis led us toward focused questions to answer during the qualitative review. In Section 3, we describe the quantitative and qualitative stages of interaction analysis.

Another way that interaction histories have been used in visualization applications is to identify states that make navigating or reasoning about an application easier. Heer et al. proposed a model for analyzing visualization interaction histories and used it to identify usage patterns for Tableau [15]. In doing so, they developed "handcrafted", application-specific chunking rules that group low-level interactions into a manageable sequence of states. We build on this idea by proposing an automatic approach for chunking interactions based on frequently occurring patterns in a collection of end users' histories. Another kind of abstraction is categorizing individual actions into top-level categories; this step simplifies the process of identifying interaction sequences with similar semantics but different low-level details (e.g., adjusting the range of a data filter using different interactions). A few taxonomies have been proposed to characterize user interactions with visual analytics systems from a high level. Some of the taxonomies focuss solely on data analysis tasks, such as Zhou and Feiner [34] and Amar et al. [3]. Heer and Shneiderman [16], on the other hand, proposed a taxonomy that captures three types of tasks in iterative visual analysis: data and view specification, view manipulation, and provenance. Our categories are based on a taxonomy of seven general interaction types that Yi et al. describe based on a review of interactive information visualizations [32]. We chose to build on this taxonomy in coding interactions for our case study because it has coverage over most of the behaviors we observed. We describe our coding process in more detail in Section 4.4.

Previous visual analytic applications, such as HARVEST [13], have included features to track insight provenance as end users interact. A benefit of doing this capture at the application-level rather than coding interactions post hoc is that displaying information about semantic actions to analysts could help them perform tasks better, as Gotz et al. found [12]. However, the automatic-capture approach requires instrumenting the visual analytic application, so it cannot be used to evaluate deployed applications, as in our case study.

Recovering longer reasoning processes by observing interactions is difficult. For example, knowing when one reasoning process ends and another begins may be unclear from a sequence of interaction alone. Previously, Dou et al. [8] demonstrated that interaction logs from a visual analytics tool can be visually examined and coded by humans to recover analysts' reasoning processes, such as specific findings and strategies. Similar to this work, we performed an exploratory user study of a visual analytics application, then used video and a visualization of participants' interactions to recover strategies. Unlike Dou et al.'s study, we first used an automatic analysis of interaction logs in order to focus the qualitative review of participants' interactions. Based on our experience in the case study, we believe this focusing step leads evaluators toward new hypotheses and is helpful in making manual reviews of large datasets more tractable.

2.2 Assessing Insights and Interactions

The purpose of a visualization or visual analysis software is usually to promote the discovery of insights ("an individual observation about the data by the participant, a unit of discovery" [27]) about the underlying data in visual representations [30]. The "complex, qualitative" [21] nature of insights requires evaluation methods beyond using simple benchmark tasks to assess insight generation [22, 6]. Field studies of analysts conducted over long periods of time, like multidimensional long-term case studies (MILCs) [29], can help evaluators identify the effectiveness of a tool and usability issues based on self-reports by analysts and usage histories. This approach results in findings with high ecological validity, but it requires a high level of participation by analysts (e.g., maintaining a journal of insights and frustrations) and making sense of multimodal evaluation data can be difficult. To address this challenge, our work demonstrates a method of correlating self-reported insights and usage histories in a systematic way.

An alternative approach that overcomes some of the challenges of longitudinal field studies is insight-based evaluation, which is aimed at quantifying evidence of insights by analysts during exploratory, lab-based user studies. Saraiya et al. were among the first to demonstrate how to quantify insight characteristics and use them to compare which bioinformatics analysis tools were better suited for certain analysis questions and datasets [27]. A key step in the evaluation involves coding participants' utterances or recorded observations about the data model ("insights") during exploration. Insights can be assigned numeric domain values with the help of domain experts, or categorized by analysis depth. Alternatively, insights have been counted in categories that correspond to types of benchmark tasks in order to compare findings between task-based and insight-based evaluations of the same tool [22]. Liu and Heer [20] and Vande Moere et al. [31] also coded insights recorded during user studies of visualization applications. These coding efforts provide examples for evaluators to follow in performing their own studies, and we referred to the practices in these earlier evaluations in devising our insight coding scheme.

While insight-based evaluations help assess whether one application promotes insight generation compared to another, it does not answer the question of which interactions or features within an application lead to insights. Efforts to characterize behaviors of analysts that result in insights have identified high-level patterns of interaction with visualizations [33]. Our work presents a case study demonstrating that patterns related to quantified insight metrics can be identified systematically from logs, then verified through a qualitative review of the context of the interactions, using screen-captured videos and a visualization of logs. In our case study, all interactions by participants were investigative in nature and did not involve predefined tasks. Doing a head-to-head comparison of insight characteristics to task performance has been done before, in between-subjects [22] and within-subjects [10] study designs, but not at the level of understanding how specific actions relate to insight generation.

3 Evaluating Visual Analytics Systems Using Insights and Interactions

In this section, we describe the evaluation goals that motivated our proposed evaluation approach and present an overview of the approach.

3.1 Goals

In the case study, we aim to answer the following evaluation questions:

- How do users use the system to arrive at insights?
- Which interface design factors are potentially hindering insights generation?

We find it difficult to answer the above questions with a standard insight-based evaluation (e.g. [27]). It is relatively straightforward to compute insight metrics using the standard insight-based evaluation, given the lessons learned from existing insight-based studies; however, it is unclear how to trace how the analysts arrive at insights. One approach would be to watch entire analysis sessions in addition to recording and coding insights and try to summarize observations about what analysis strategies were preferred by analysts, what interface components were easy or difficult to use, and what sequences of actions seem to consistently lead to insights. However, we found that it was difficult to summarize such high-level observations without more targeted objectives in mind while watching the videos. An alternative would be to perform controlled evaluation with individual components and measure how many insights analysts could come up with each. With this approach, however, we won't be able to capture any usage pattern that involves more than one component.

We therefore decided to integrate interaction history analysis into the standard insight-based evaluation given our evaluation goals. Previous works have shown that interaction history can reveal feature usage patterns [15] and analysis strategies [18, 8]. Also, Yi et al. summarized four high-level processes through which users gain insights when using a visualization system [33], supporting the view that interaction

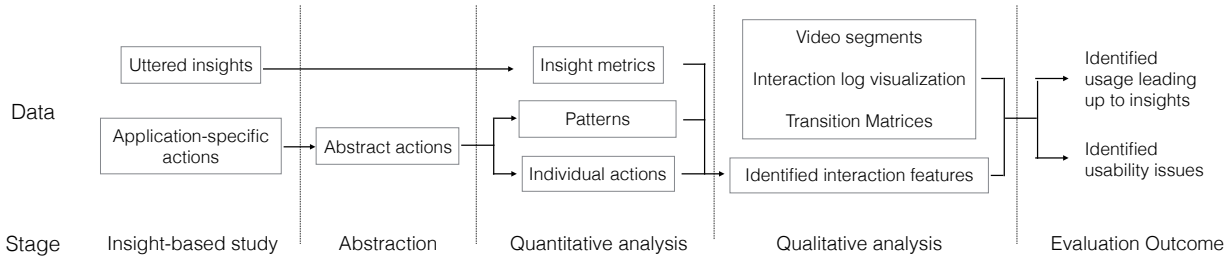

Fig. 1. An illustration of the evaluation pipeline. It starts with a standard insight-based study, in which user-reported insights and application-specific interaction data are collected. The insight data are coded into insight metrics. The interaction data are converted into abstract, generalizable actions, from which patterns (short and frequently appeared strings of individual actions) are extracted. Quantitative analyses such as correlation analysis are then applied on insight metrics, patterns, and individual actions to guide the qualitative analyses that follows. Finally, qualitative analyses are performed using user study video recordings and visualization of interaction logs to answer evaluation questions.

history at least partially captures how users arrive at insights. The potential connections between interaction history and insights led us to the hypothesis behind the proposed evaluation approach: identifying interaction features that are correlated with insight characteristics and reviewing those features in context – using a visualization of logs and screen-capture video recorded during the interaction – will reveal analysis strategies and usability issues that lead to design guidelines for the visual analytics application.

3.2 Evaluation Stages

The evaluation approach consists of four stages, as depicted in Figure 1. It starts with a standard insight-based study, in which we ask the user to complete an open-ended analysis task and collect insights reported by the user while completing the task. In addition, we also collect interaction data users produced when using the visual analysis system to come up with insights.

The insight and interaction data collected from the user study are then further processed in an abstraction stage. The insight data are analyzed and coded into insight characteristics. For the interaction data, a theoretical framework is applied to code the application-specific actions into more abstract, generalizable actions. The purpose of interaction abstraction is so that we could focus on high-level, general visual analysis tasks instead of the application-specific implementations.

The abstract actions then go through a few quantitative analyses. We first identified two types of elements from the interaction logs as the basic units in the analyses. The first type of elements are individual actions. The second type of elements are patterns, which are short strings of individual actions that have frequently appeared in the interaction logs. The patterns are intended to capture blocks of actions that are used to accomplish simple analysis objectives. Therefore, we defined two high-level objectives for the pattern extraction algorithm: 1) the final set of patterns should be small, since the number of basic analysis objectives for a single visual analytics system are limited; 2) the set of patterns should have appeared frequently enough in the interaction history. However, the definitions for "small" and "frequent" will change depending on the application, and we set them as free parameters that the evaluator needs to decide upon in this study, which we discuss in more detail Sections 5 and 7.

We then performed two types of quantitative analyses on each of the two types of elements. For simplicity, we discuss the analyses for actions only, but the version for patterns is similar.

- **Frequency**: computes correlation coefficients between the percentage of each abstract action among all actions and each insight metric, and flags moderate and strong correlations.
- **Composition:** identifies abstract actions that can be achieved using a variety of application-specific actions, but analysts showed strong preference for one application-specific action over the others.

Results from the quantitative analyses are then used to guide focused qualitative analyses. Screen-captured videos of the user study sessions, visualizations of the interaction logs, and action transition matrices are used in the qualitative analyses. Occurrences of actions and patterns flagged in the quantitative analyses are located in the videos and visualizations and reviewed by the evaluators. For negative correlations, the evaulators investigate whether users have run into usability issues when performing those types of actions. For positive correlations, the evaluators aim to understand if the action plays an important role in arriving at insights, and if so, whether it is well supported by the visual analysis system.

4 Data Collection

Given the goals of the evaluation, we first designed and conducted an insight-based study. In this section, we present details on the study setup and how we processed the raw data from the user study.

4.1 The Visual Analysis System

The visual analysis system we evaluated was developed for intelligence analysis, and it supports integrated analysis of textual data from multiple sources and in multiple formats. The analysis engine of the system performs analyses on the incoming textual data and extracts both entities – like documents, people, and keywords – and relationships among entities. The entities and relationships are then represented using a graph structure. The query engine supports complex queries based on entities' attributes and relationships.

The visual front end of the tool consists of information views on two tabs: the Overview tab (Figure 2) and the Explorer tab (Figure 3). In the Overview tab, an analyst can view a list of recommended queries based on graph metrics, such as the vertex degree for entities of a given type. By clicking on a recommendation, the analyst can add the results of the query into the network view (in the Explorer tab) as nodes. The 'Dataset Details' view in the Overview tab shows the entity types in the dataset, the attributes that define each entity type, and how the entities are connected to one another. The analyst can choose to retrieve all entities of a given type using this view. In the Explorer tab, the timeline view shows the distribution of documents over time, and lets the analyst retrieve entities with a timestamp within the selected time range. The timeline can also be stratified vertically given a chosen entity type; in doing so, the analyst can see the distribution of documents for each selected entity of that type. The network view visualizes selected entities and the relationships between them as a force-directed node-link diagram. The analyst can right-click on a node to browse its relationships to other entities and pull these connected entities into the view. Clicking on a node will activate an entity detail window that displays its attributes in detail, including the option of viewing the contents of a document entity. The analyst can choose to show or hide nodes of any given entity type in the network view.

In addition to interacting with the previously mentioned views, the front end provides two other ways of querying data. The 'Data Queries' panel lets the analyst generate a query by specifying attribute values or relationships to chosen entities. The analyst can also use the

search bar in the Explorer tab to input keywords or names and select from a list of related queries.

4.2 Study Procedure

We recruited 10 participants (3 male, 7 female) whose ages ranged from 22 to 52 years ($M = 28.9$, $SD = 9.4$, $Median = 25.5$). Participants were students from a variety of disciplines at a major research university. None of the participants had used visual analysis systems regularly or had experience with investigative data analysis similar to the study task.

We used a subset of the VAST Challenge 2014 dataset [1] in the user study. The dataset contains texts, such as news reports, resumes, and email headers, relevant to a disappearance case that happened in a fictional country. Each participant's task was to analyze the dataset using the visual analysis system and identify possible explanations behind the disappearance case, with supporting evidence from the dataset.

At the beginning of each study session, each participant was given an overview of the task and signed the consent form. We then gave the participant a two-page training manual for the visual analysis tool to study at his or her own pace. Once the participant finished reading the training manual, we asked the participant to complete a set of 18 short training task[1]. An experimenter was present to answer the participant's questions and provided feedback to make sure that the participant understood all the operations available in the tool. The training phase took around 15-20 minutes for all participants.

After the training phase, each participant worked on the analysis task for 45 minutes. With consent from participants, we videotaped participants performing the analysis and collected screen-capture recordings. We used a think-aloud protocol and participants were instructed to explain their analysis processes and report their insights as clearly as possible. An experimenter was present throughout the analysis task, and the participant was free to ask any technical question about the use of the visual analysis tool. The experimenter did not answer questions related to analysis approaches. After the 45-minute analysis, each participant completed an exit questionnaire with a few demographic questions. In total, each session lasted around 1.5 hours.

4.3 Coding Insights

To quantify the reported insights, we identified three types of information in the insights reported by the participants: *facts*, *generalizations*, and *hypotheses*. In the remainder of this paper, we refer to these three terms collectively as *insight characteristics*. The choice of these insight characteristics was made before coding and inspired by previous insight-based studies [27, 10, 20] and work on information discovery and ideation [19]. Definitions and examples for these characteristics are listed in Table 1.

For each participant, we computed the number of unique insights reported for each of the insight characteristics. In general, it may be difficult to count the number of insights since users may report a complex insights which cannot be easily segmented into individual ones. In our case, however, we found that participants almost always reported insights in simple forms, possibly because it is easier to report an insight as soon as it is discovered. Two authors of this paper worked together to come up with the coding scheme and then performed the coding independently. The correlation between the coding results from the two authors is 92.66%, suggesting the two coders are quite consistent. The two coders then discussed the coding results to resolve some of the inconsistencies, and in cases where the coders couldn't come to an agreement, the final insight score was computed by taking the average of the scores from the two coders. In addition, we calculated an *originality* score for each participant. To compute the score, we first counted the number of times each unique fact, generalization, and hypothesis was recorded by any of the participants. We define the originality of each unique piece of information as the inverse of the number of times that information was recorded by anyone. In other words, information identified by all participants has low originality, and information identified by only one participant has high originality. A participant's *originality* score equals the sum of information-specific originality scores over all the information recorded by the person.

[1] User study materials are available at http://bit.ly/1Hd8wxp

4.4 Coding Interactions

User interactions were coded in two passes. In the first pass, two of the authors watched the screen-capture video (including audio from the think-aloud process) of each investigative session and coded each session as a sequence of low-level, application-specific user actions. In the encoding, each action is represented as a tuple containing three elements: *name*, *target*, and *timestamp*. The *name* of an action indicates its function, such as "Adjust time range". The *target* of an action is the interface component that the action is applied to, such as "Timeline". The *timestamp* records the start time of the action since the beginning of the analysis session. Before coding, both evaluators agreed upon a preliminary coding scheme. Each evaluator coded one session independently using the scheme, then both compared notes to resolve any ambiguity and inconsistencies in applying the coding scheme. After, all sessions were divided evenly between the two evaluators, then videos were coded using the finalized scheme.

In the second pass, we categorized each low-level action as one of seven top-level actions, which are distinct analysis behaviors not specific to the application. Six of the actions are directly borrowed from Yi et al.'s interaction taxonomy [32]. We added one more action, *Retrieve*, to account for the types of actions that retrieve entities given specific criteria. The definition of the seven top-level actions and categorization of application-specific actions are shown in Table 2.

5 Experiment 1: Quantitative Analysis of Interaction Features

We performed quantitative analyses of the coded insights and interactions to identify potentially interesting interaction features for further qualitative analyses. In this section, we present the methods and results of the experiment. The results are interpreted in Section 6 through qualitative analysis.

5.1 Methods

We first describe details of the methods used in the quantitative analysis, including 1) why and how absolute counts of individual actions and patterns were converted into frequencies before analysis; 2) details about the procedure we used for correlation analysis; and 3) how we extracted common patterns from interaction sequences that warrant follow-up investigation.

5.1.1 Converting absolute counts to frequencies

When analyzing correlations between interaction features and insight metrics, we used frequencies of top-level actions as a percentage of total actions, instead of absolute counts of these actions. This choice was made to account for variations in how quickly and actively each participant performed the analysis. Even though the analysis time was approximately the same for each participant during the user study, we observed that the number of actions performed by each participant in a given session varies significantly between participants ($M = 199.4$, $SD = 73.16$), which suggests that participants were using the system at different paces. To compute the frequency of a pattern used by each participant, we multiplied the number of occurrences of that pattern by its length, and then divided the product by the number of actions performed by each participant.

5.1.2 Correlation analysis

The primary goal of performing the correlation analysis in this paper is not to test hypotheses about the correlations between the two groups of measures (insight metrics and interaction frequencies) but to explore and identify activities worth further qualitative analysis. Therefore, we chose to report all moderate and strong correlations regardless of p-values. We computed pairwise correlations between every interaction feature and insight metric. For pairs with moderate and strong correlations, we created scatterplots to verify the linear relationships between the pairs (e.g., Figure 4). When describing the strength of

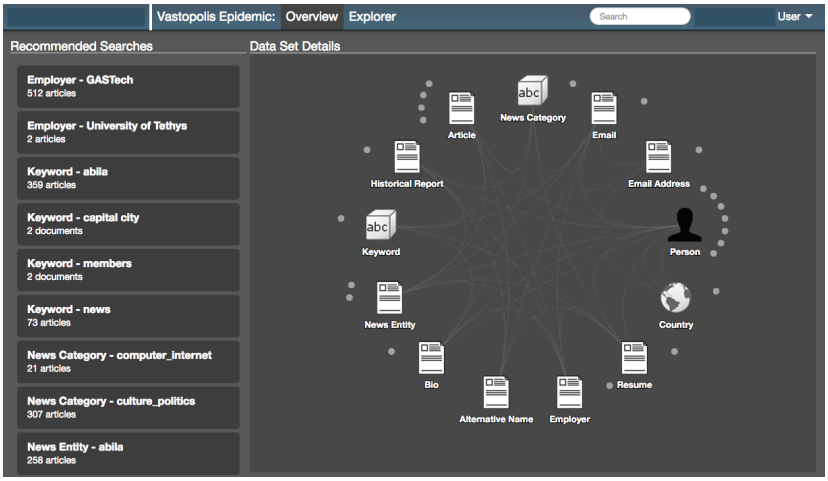

Fig. 2. The Overview tab shows recommended searches (left) and the relationships between information types in the dataset (right). Selecting either a recommended search or exploring an information type launches the Explorer, which displays additional views of the dataset.

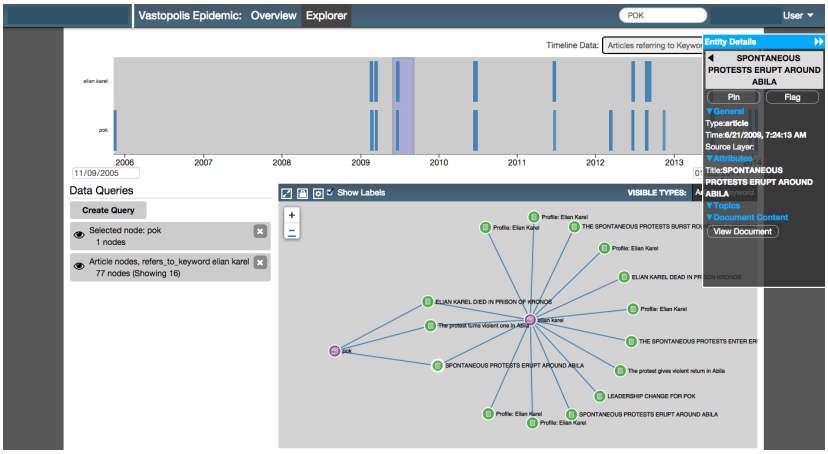

Fig. 3. The Explorer tab shows individual records in the dataset using a network view (bottom). Details for individual records are displayed (right) when the analyst selects a node in the view. The analyst can view active queries and compose new ones (left) and filter retrieved records using an interactive timeline (top).

correlations, we use the guide suggested by Evans [9] and report Pearson's r with absolute value between .40 – .59 as "moderate", .60 – .79 as "strong", and $>$ .80 as "very strong".

5.1.3 Extracting patterns

We define a pattern as a short sequence of consecutive actions that has occurred frequently enough across all interaction sequences collected from one user-study session. For example, participants often performed action sequences where a single query was made, then the result set was immediately examined serially. Later in this paper, we describe this process as "sampling". Before discussing the analyses performed on patterns, we describe how we extracted patterns from the interaction histories and summarize the patterns. Patterns were extracted in two steps.

Step 1: Identify frequently-performed candidate patterns. First, we went through all participants' interaction sequences and counted each unique subsequence of actions that occurred at least once and had length of at least 3. We chose to start with length 3 to capture only non-trivial patterns. In this step, the same action could not be counted twice for the same pattern but may be counted towards different patterns. Once we had generated counts for all the subsequences, we took all those that appeared more than 40 times across all participants' sequences as the candidate patterns. We chose this threshold frequency using the criteria discussed in Section 3.2.

Step 2: Segment interaction sequences into frequent patterns. Next, we used the candidate patterns to segment each interaction sequence into a list of patterns and singletons – individual actions that had not been segmented as part of a pattern. The segmentation step made sure that when generating the final patterns and their counts, no action was counted towards more than one pattern. We used a greedy algorithm during segmentation. For each interaction sequence, we tried to match the head of the sequence to each of the candidate patterns, and longer candidate patterns were chosen for matching first, similar to matching regular expressions. Once the head of the sequence was matched to a candidate pattern, the head was removed and the count for that candidate was incremented. If no match was found, the first action in the sequence was removed and its singleton count was incremented. The process was repeated until the entire sequence had been segmented. After segmenting all interaction sequences, we counted all candidate patterns and singleton actions in the interaction history for that participant.

Finally, we treated all candidate patterns that occurred more than 20 times during the segmentation as the final patterns. This frequency threshold was lower than the one in Step 1 because the overall pattern frequencies were reduced when an action could not be counted towards more than one pattern.

Table 1. Terms used for quantifying insights.

Term	Definition	Example
Fact	A statement that is true given the VAST Challenge 2014 dataset and describes the existence or properties of an event or an entity	"The police questioned a Gastech employee named Elian Karel after the disappearance."
Generalization	A statement that describes connections among entities relevant to the disappearance case	"There's one Gastech employee who shares the same last name with a POK member."
Hypothesis	A hypothetical statement relevant to the disappearance case	"Henk might be motivated to join POK because his wife was sick due to the mess of the environment."

Table 2. The mapping between application-specific actions and abstract actions based on Yi et al.'s taxonomy.

Action	Definition	Application-specific actions
Select	Mark something as interesting	Pin entity, Flag entity
Explore	Show me something else	Select recommendation, Retrieve entities by type, Browse network view, Cancel timeline filter, Cancel data type filter
Elaborate	Show me more details	Select search result, Select entity in network, View entity details, View document content
Reconfigure	Show me a different arrangement	Adjust network view, Rearrange network entities, Adjust network view size, Adjust timeline by data type, Adjust timeline range
Filter	Show me something conditionally	Remove query results, Hide entities by type
Connect	Show me related items	Retrieve by selected network entity
Retrieve	Show me matches to a query	View search results, Retrieve by time range, Create a query

5.2 Results

Below we discuss the results for analyses with individual actions and patterns respectively. We focus on the top-level actions since they more directly reflect user intentions. Table 2 lists their definitions and corresponding application-specific actions. We note that due to the small sample size, the exact values of the correlation coefficients can be easily influenced by participants with exceptional performance and interaction patterns, and need to be interpreted in the context of the qualitative analysis that follows in Section 6 instead of taken as generalizable findings. Also, while the actions and patterns are not application-specific, the findings are likely influenced by the types of visualizations available in this application, and may not generalize to applications with other types of visualizations.

For all correlations in this section, we report Pearson's r, bootstrapped 95% confidence intervals (CI), and p-values. CIs were computed using the percentile method with 10,000 bootstrap replicates. Because multiple correlation tests were performed, p-values of the correlations need to be compared with the Bonferroni-adjusted alpha of 0.001.

5.2.1 Analyses with Individual Actions

We report correlations observed between action percentages and insight metrics in Table 3. Among all top-level actions, the composition of *Retrieve* and *Explore* displays much more frequent usage of one application-specific action over the others. The majority of the *Retrieve* actions was *View search results* (59.78%), compared to *Create a query* (21.74%) and *Retrieve by time range* (18.48%). For *Explore* actions, the most frequently used was *Browse network view* (72.09%).

5.2.2 Analysis of Patterns

Using the algorithm described in Section 5.1.3, we extracted six patterns. After merging similar ones, we identified the following four patterns:

- **Orienting** (Reconfigure – Explore – Elaborate): The analyst *reconfigure*s the view to look at the dataset in a different way, and then further *explore*s the dataset and *elaborate*s on the details of some entities.
- **Locating** (Retrieve – Elaborate – Elaborate, Elaborate – Retrieve – Elaborate): The analyst *retrieve*s entities that match some specific criteria and then *elaborate*s to examine the details of the entities.
- **Sampling** (Explore – Elaborate – Elaborate – Elaborate, Explore – Elaborate – Elaborate): The analyst *explores* and adds new entities that might be of interest onto the active view, and then *elaborate*s to gather detailed information about the entities.
- **Elaborating** (Elaborate – Elaborate – Elaborate): The analyst performs a sequence of *elaborate* actions to gather detailed information about one or more entities.

Among the four patterns, **Sampling** has a moderate positive correlation with the number of generalizations ($r = .49$, $p = .15$, CI [-.03, .98]), number of hypotheses ($r = .41$, $p = .23$, CI [-.28, .90]), and originality ($r = .45$, $p = .19$, CI [-.25, .96]). **Elaborating** has a moderate negative correlation ($r = -.44$, $p = .20$, CI [-.89, .11]) with the number of generalizations. The composition analysis shows that in 98.21% of the **Sampling** patterns, the *Explore* action is *Browse the network*, which is consistent with earlier finding that *Browse the network* is the most frequently used *Explore* action. We also found that only one of the **Locating** patterns has *Create a query* as the *Retrieve* action. All the other patterns involve retrieving entities using *View search results*. This contrasts with the ratio of *View search results* actions to *Create a query* actions discussed earlier.

6 Experiment 2: Qualitative Analysis of Video Segments and Interaction Log Visualization

In this section, we discuss the process and results from the qualitative analysis. The results support our initial hypothesis: by viewing interaction features identified during the quantitative analyses in the context of screen-captured video and log visualizations, we were able to uncover interesting analysis strategies and usability issues and use them to inform the application design.

Results from the quantitative analysis suggest the following interaction features are worth further investigation: 1) **Sampling**, 2) **Elaborating**, 3) the use of query builder versus search bar for *Retrieve* and **Locating**, 4) *Filter*, and 5) *Connect*. We conducted a follow-up qualitative analysis to gain an in-depth understanding of how these features relate to successful analysis strategies or obstacles analysts encountered during the sessions.

Table 3. Moderate and strong correlations observed between action percentages and insight metrics

Action	fact	generalization	hypothesis	originality
Explore	($r = .57$, $p = .089$, CI [.17, .90])	($r = .75$, $p = .013$, CI [.16, .96])	($r = .61$, $p = .063$, CI [-.39, .88])	($r = .50$, $p = .142$, CI [-.04, .85])
Elaborate		($r = .44$, $p = .20$, CI [-.07, .78])		
Filter	($r = -.73$, $p = .017$, CI [-.91, -.36])	($r = -.40$, $p = .25$, CI [-.87, .29])		($r = -.54$, $p = .11$, CI [-.93, .17])
Connect		($r = -.59$, $p = .07$, CI [-.94, .10])	($r = -.54$, $p = .11$, CI [-.91, .21])	
Retrieve	($r = -.57$, $p = .09$, CI [-.91, -.16])	($r = -.45$, $p = .19$, CI [-.81, .38])		($r = -.43$, $p = .22$, CI [-.81, .14])

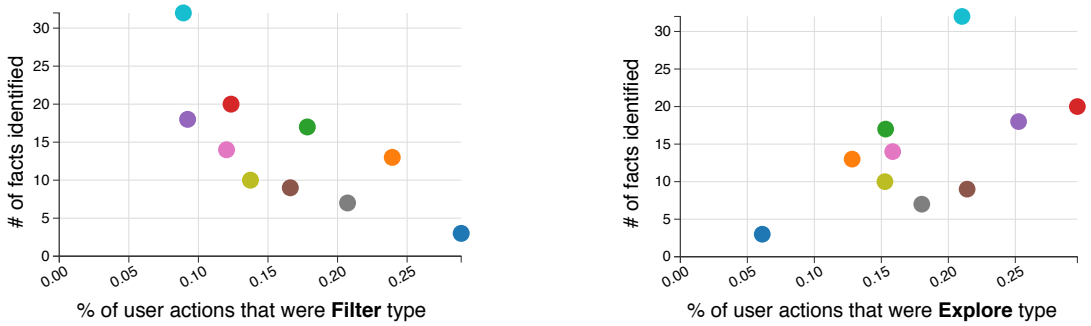

Fig. 4. Scatter plots show the number of facts identified versus the proportion of specific actions used by each participant. Participants are represented by uniquely colored marks that correspond between plots. Left: We found a strong negative correlation between the percentage of actions that were *Filter* type and the number of facts identified ($r = -.73$). Right: We found a strong positive correlation between the percentage of actions that were *Explore* and the number of facts identified ($r = .57$).

6.1 Methods

Each interaction feature was analyzed in two steps. In the first step, we located and marked the occurrence of the target pattern or action during each analysis session by looking at a visualization of the entire interaction log (Figure 6). We also used the log visualization to quickly scan for actions that led up to the target and check if any insight was reported immediately after the target. In the second step, we watched the video segments corresponding to the target pattern or action to gather information not captured in the interaction data. In particular, for each occurrence of a target action or pattern, we took notes of 1) the objects manipulated, 2) the outcome of the interaction, and 3) any user utterance that accompanied the interaction, including both insights and questions or complaints. Two authors of the paper independently completed the analysis and discussed the findings to finalize the results. Both authors agreed on all the findings reported.

In addition, we also computed and visualized transition matrices using the pattern segmentation results to summarize the interaction sequence for each participant. A row or column in the matrix can correspond to either a pattern or an action, and each cell (i, j) contains the number of times the pattern or action j appeared right after the pattern or action i. The transition matrix in Figure 6 shows the average number of transitions for each combination. The transition matrices don't capture as much information as the log visualization, but they make it easy to check which patterns or actions are frequently used together and compare the difference across participants.

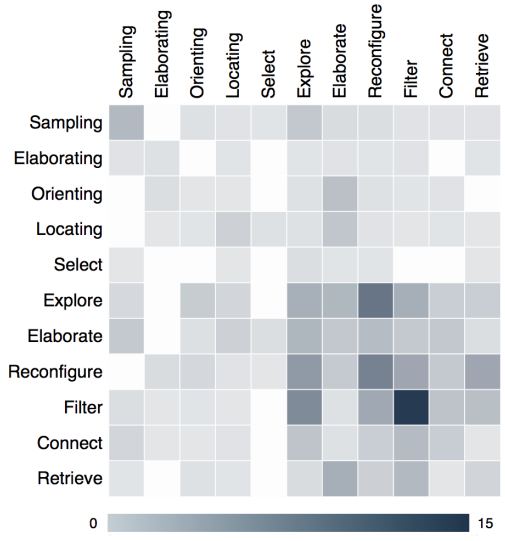

Fig. 5. The transition matrix for patterns performed by all participants. Each cell (i, j) contains the average number of times a pattern or action j appeared immediately after the pattern or action i. The matrix is useful for checking which patterns or actions are frequently used together.

6.2 Results

Here we describe the findings of the qualitative analysis, then summarize design recommendations for the visual analysis system and similar applications that we derived from the results.

6.2.1 Sampling patterns are performed in chunks that lead to insights

Looking at the log visualization, we first observed that multiple **Sampling** patterns often appeared in succession. This is also visible from the transition matrices. When reviewing the video segments, we further observed that a consecutive group of **Sampling** patterns is always part of a larger action structure. In all of the consecutive **Sampling** patterns, the analyst was sampling by browsing the network view. Before starting such a sampling process, the analyst needed to add entities onto the network view using either an *Explore* or *Retrieve* action. Furthermore, when examining the log visualization, we noticed that there are often a mix of *Reconfigure* and *Filter* actions between the initial *Explore* or *Retrieve* action and the group of *Sampling* patterns. This seems to be consistent with Shneiderman's information seeking mantra [28]: the *Explore* or *Retrieve* serves the purpose of creating an overview given a loosely defined condition, the mix of *Reconfigure* and *Filter* actions further refines or adjusts the representation of the overview, and the group of **Sampling** activities obtain details about the entities on-demand. We also observed that more than half of the

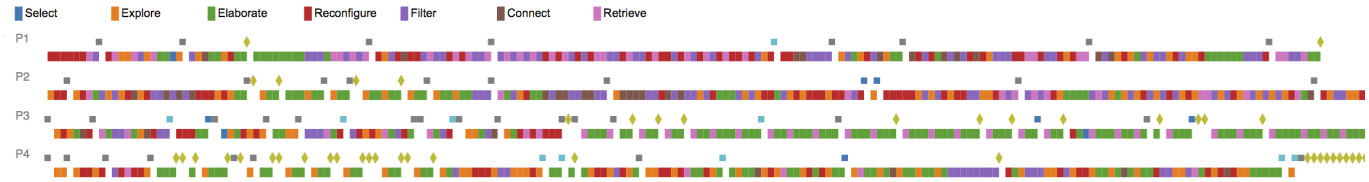

Fig. 6. Visualization of interaction logs from the case study. Each row of colored marks indicates the sequence of top-level actions a participant performed. Patterns of actions across participants were visible from this context. Brushing these marks in the visualization displays details on demand and timestamps that we used to revisit screen-captured video clips of interest.

Sampling activities led to insights directly relevant to the contents examined during those activities, which explains the positive correlation between the amount of **Sampling** and multiple insight metrics.

6.2.2 Elaborating patterns are used in both unguided and targeted ways

Given the negative correlation between the **Elaborating** pattern and the number of generalizations reported, we initially hypothesized that the frequent occurrence of an **Elaborating** pattern might suggest that the analyst was overly focused on a small set of information, losing the big picture, and therefore made fewer generalizations about the dataset. However, when examining the log visualization and video segments, we identified the following distinct scenarios when an **Elaborating** pattern occurred:

S1 – The analyst exhaustively examined the details of all the entities available on the network view. This happened twice with P1. In both cases, he queried for all documents related to a keyword and got only a few documents each time. He then selected and read each document without stopping to browse the network view.

S2 – In two cases, the analysts seemed frustrated and appeared to select entities randomly. One analyst said "I don't know what I'm doing here" while clicking. Both cases appeared near session ends.

S3 – The analyst alternated between the details of two entities several times, possibly comparing the detailed information.

S4 – The analyst added entities to the network view through either *Explore* or *Retrieve*, and did a *Reconfigure* right afterwards to make the network view more readable. The *Reconfigure* separated the *Elaborate* actions from the more targeted *Explore* and *Retrieve* actions, resulting in stand-alone **Elaborating** patterns.

The first two scenarios may have negative impact on insight generation. Exhaustively checking the details of similar entities as in **S1** could be less efficient than sampling diverse information if the task requires connecting dots among a variety of entities and events. In **S2**, frustration may signal dissatisfaction with insights generated so far or the analysis process. However, the latter two cases don't seem to be related to sub-optimal analysis practices or obstacles in analysis.

6.2.3 Queries often return unexpected selections

By examining the log visualization, we observed that a *creating a query* was frequently followed up by two types of actions: 1) one or more *Reconfigure* actions; 2) a single *Filter* action. From watching the video, we noted that in the first case, analysts usually found useful information from the query. Here, the *Reconfigure* actions were used to adjust the network view so that the analyst could more easily examine the details of the entities retrieved by the query. However, the second case revealed an issue with the query builder: when a *Filter* action was performed immediately after *creating a query*, it was usually because the query added no entities into the network view or, in rare cases, added too many entities that seemed overwhelming to the analyst ("I've got more than what I can deal with here"). Often, the query was then immediately removed. We found similar cases with queries created from the contextual menu in the Network view. We concluded that adding a feature to preview the results of a query when using the builder or a contextual interaction would reduce the occurrence of these interactions and improve the quality of the analysis.

6.2.4 Filter actions are performed in chunks

The transition matrices show that an *Filter* action often transition into another *Filter* action. It was also evident when we scanned the log visualization that *Filter* actions occurred more frequently in groups than individually. Many of the individual *Filter* actions fell into the category described above and was used to remove queries that returned zero entities. Looking at the groups of *Filter* actions, we observed two scenarios: 1) when the Network view became cluttered with too many entities, and 2) when the query list contained too many queries with zero entities and the analyst felt the need to clean up the query list. The second scenario happened more frequently than the first scenario. While the first scenario corresponds to the original design goals of these filtering actions, the second scenario points to the same issue revealed while we examined the query creation action earlier.

6.3 Design Recommendations for the System

An advantage of this evaluation methodology from a developer's standpoint is that it can provide data-driven design recommendations that point to specific interactions and components. Our approach builds on the traditional insight-based evaluation by helping to identify ways to improve a system in addition to measuring its effectiveness.

We summarize below two potential improvements for the system we tested as suggested by our case study. First, since the **Sampling** pattern was prevalent and frequently led to insights, we recommend providing better application support for sampling-type activities, e.g. making a sampled entity more visually distinct from those that have not been examined. Second, we observed that queries created using the query builder and the network's contextual menu frequently returned empty results and caused participants to undo these interactions; we thus recommend providing previews of query results in the context of query tools so that unnecessary interactions can be avoided.

Since the above recommendations focus on the high-level aspects of the system design, they may generalize to similar visual analytics systems which have network views and query builders as core components. However, they may not generalize to systems that consist primarily of other types of visualizations and interface components.

7 Discussion

In this section, we discuss lessons learned from the case study and limitations of this evaluation approach.

7.1 Lessons Learned

7.1.1 Benefits of combining action and pattern analysis

In this case study, including both individual actions and patterns as features in the analysis has yielded additional information. For example, analysis of individual actions shows that both the query builder and the search bar have been used frequently, but the pattern analysis reveals that using the search bar often led to multiple *Elaborate* actions while creating a query rarely did. Similarly, we were able to tell that *browsing a network view* was much more likely to be followed by multiple *Elaborate* actions than other *Explore* actions only by combining the results from analyzing both types of features. Such observations are more valuable than information about action usage or pattern prevalence alone, and enabled more focused qualitative analysis.

7.1.2 Benefits and costs of using abstract, top-level actions

We applied abstraction to actions before performing analyses for three potential benefits. First, using abstract actions unifies application-specific actions that serve similar analysis tasks, reducing unimportant variances in the interaction sequences and allowing more important patterns to emerge. Second, abstract actions are more directly related to user intents and therefore are more meaningful building blocks for analyzing insight generation. Finally, using abstract actions makes it easier to compare and contrast results from multiple user studies. Given the complexity and diversity of visual analytics systems in general, it is unlikely that two systems provide identical sets of application-specific actions. At the same time, abstract actions are not application-specific and enable comparison between systems.

However, using abstract actions also has its costs. First, an existing interaction taxonomy may not apply to a given application as-is. In our case, we had to add an additional category to account for actions that are important to our system but not necessarily to other visual analytics systems. In addition, choosing the most suitable interaction taxonomy to use from existing ones requires effort from the evaluator. Finally, abstraction hides nuances between actions that are classified into the same group, and sometimes it may be important for the evaluator to be aware of those differences. For example, we found that two types of *Explore* actions – *selecting a recommended search* and *browsing network view* – were often performed in different use cases.

7.1.3 Free parameters in the pattern extraction algorithm

Extracting patterns from interaction sequences is not a well-defined task. In the case study, we made assumptions about minimum length and frequency for patterns given the nature of the application and the length of the user study sessions. We expect these assumptions to change when extracting patterns given another applications. For example, with an application that supports more types of actions and more complex analysis goals, it might be desirable to set the minimum pattern length longer. In addition, the minimum frequency for patterns should scale with the average number of actions per session.

7.2 Limitations and Open Questions

7.2.1 Required effort from evaluators

While the proposed approach helps the evaluator to narrow the focus and spend less time during the qualitative analysis, such an evaluation still requires lots of effort from the evaluator. First, executing an insight-based user study and coding insights is difficult and time-consuming. Our approach relies on data from a standard insight-based user study and does not make running the user study easier. Second, since evaluators still need to review videos to confirm or disconfirms patterns, the cost of performing the evaluation scales with the number of participants, as with the standard insight-based evaluation. Finally, it is always possible that certain interesting usage patterns or issues may not manifest themselves in the quantitative stage. If it is important for the evaluator to be comprehensive, a more complete video review may still be necessary.

7.2.2 Temporal aspects of the interaction history

The current case study does not consider temporal information in the interaction data, which could be useful in at least two ways. First, we did not analyze how long participants spent completing individual actions or patterns. It is possible that knowing whether certain actions or patterns take a lot of time on average, or that some participants were much faster at completing them than others, could help identify usability problems in an application. Difference in time allocation on actions may also suggest difference in analysis strategies. It is possible that existing temporal data-mining techniques, such as motif discovery in time series, can be applied to mitigate this issue. Second, we did not align insights with interaction histories to identify sequences of actions that lead up to insights. A major reason this is difficult is that an analyst might perform key interactions that lead to insights they report later on. In this case, it is difficult to identify the time points at which necessary information for an insight was unearthed.

7.2.3 Flexible pattern extraction

The pattern extraction algorithm we presented performs exact matching for top-level action sequences, and is not able to capture more expressive patterns or ones with variable-length action sequences. The limitations of the current matching process are illustrated using two examples from the case study. First, the **Sampling** pattern consists of two action sequences that differ only by one trailing *Elaborate* action. Here, the user's intention is possibly better reflected by the order in which actions are performed instead of the exact number of actions performed. Second, as discussed in Section 6.2.1, **Sampling** patterns often appear in chunks preceded by either an *Explore* or *Retrieve* action, but this pattern is difficult to identify with a frequency analysis when only top-level actions – and not other subpatterns – are parsed. One future direction is to develop a more flexible pattern extraction algorithm with pattern matching similar to regular expressions.

7.2.4 Controlling ordering effects

During the training portion of our study, the application components and interactions were presented to all participants in the same order. In general, it is possible that the presentation order of available interactions may introduce an ordering effect on the participant's preference for certain interactions and analysis strategies. Because the number of user interactions in the application we tested is large, it is impractical to fully control for potential ordering effects through counterbalancing the presentation order of actions; however, more investigation of how training influences analysis strategies could inform evaluators about potential ordering effects and methods to minimize those effects.

8 Conclusion

We presented findings from a case study where we performed an insight-based user study of a visual analytics system, then afterwards looked at quantified insight characteristics alongside patterns in participants' interaction logs. The contributions of this work are threefold. First, we describe an evaluation approach using interaction logs in concert with insight-based evaluation, which lets us answer questions about the relationship between interactions and insights. Second, we contribute a case study with an existing visual analytics application that demonstrates the practical use of the evaluation approach and the experimental design choices involved in using it. Third, we contribute findings about the link between interactions and insights from our case study. We found correlations between insight characteristics, like the number of facts an analyst recovers, and the types of top-level actions she performs, like *explore* actions.

Furthermore, using the logs we identified common analysis patterns composed of these top-level actions – behaviors we call **Orienting**, **Locating**, **Sampling**, and **Elaborating** – and measured correlations between frequencies of these patterns and insight characteristics. Using these quantitative findings and screen-captured video, we identified two design recommendations that are applicable to similar visual analytics applications.

- Design marks representing information so that analysts can distinguish visually between information they have already explored (e.g., during **Sampling** interactions) and new information.
- Provide a preview of query results during interactions that lead to queries (e.g., searching for connected information using a contextual menu). Queries with too many or too few results often lead to extra actions that undo the query and waste time.

This work is a step toward systematically identifying interactions and application features that promote insights in visual analytics systems.

Acknowledgments

This work was supported in part by Aptima, Inc., and NSF award IIS-10-16623 and IIS-10-18769. All opinions, findings, conclusions, or recommendations expressed in this document are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

REFERENCES

[1] VAST Challenge 2014. http://hcil2.cs.umd.edu/newvarepository/benchmarks.php#VAST2014. Accessed: 2015-03-24.

[2] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–26. ACM, 2006.

[3] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 111–117. IEEE, 2005.

[4] I. Boyandin, E. Bertini, and D. Lalanne. A qualitative study on the exploration of temporal changes in flow maps with animation and small-multiples. In *Computer Graphics Forum*, volume 31, pages 1005–1014. Wiley Online Library, 2012.

[5] E. T. Brown, A. Ottley, H. Zhao, Q. Lin, R. Souvenir, A. Endert, and R. Chang. Finding waldo: Learning about users from their interactions. *Visualization and Computer Graphics, IEEE Transactions on*, 20(12):1663–1672, 2014.

[6] R. Chang, C. Ziemkiewicz, T. M. Green, and W. Ribarsky. Defining insight for visual analytics. *Computer Graphics and Applications, IEEE*, 29(2):14–17, 2009.

[7] E. H. Chi, P. Pirolli, K. Chen, and J. Pitkow. Using information scent to model user information needs and actions and the web. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 490–497. ACM, 2001.

[8] W. Dou, D. H. Jeong, F. Stukes, W. Ribarsky, H. R. Lipford, and R. Chang. Recovering reasoning processes from user interactions. *IEEE Computer Graphics and Applications*, (3):52–61, 2009.

[9] J. D. Evans. *Straightforward statistics for the behavioral sciences*. Brooks/Cole, 1996.

[10] S. Gomez, H. Guo, C. Ziemkiewicz, and D. Laidlaw. An insight- and task-based methodology for evaluating spatiotemporal visual analytics. In *Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*, pages 63–72, Oct 2014.

[11] S. Gomez and D. Laidlaw. Modeling task performance for a crowd of users from interaction histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2465–2468. ACM, 2012.

[12] D. Gotz, J. Lu, P. Kissa, N. Cao, W. H. Qian, S. X. Liu, and M. X. Zhou. Harvest: an intelligent visual analytic tool for the masses. In *Proceedings of the first international workshop on Intelligent visual interfaces for text analysis*, pages 1–4. ACM, 2010.

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

[14] W. D. Gray, B. E. John, and M. E. Atwood. Project ernestine: Validating a goms analysis for predicting and explaining real-world task performance. *Human-computer interaction*, 8(3):237–309, 1993.

[15] J. Heer, J. Mackinlay, C. Stolte, and M. Agrawala. Graphical histories for visualization: Supporting analysis, communication, and evaluation. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1189–1196, 2008.

[16] J. Heer and B. Shneiderman. Interactive dynamics for visual analysis. *Queue*, 10(2):30, 2012.

[17] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse. The lumiere project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 256–265. Morgan Kaufmann Publishers Inc., 1998.

[18] Y.-a. Kang, C. Gorg, and J. Stasko. Evaluating visual analytics systems for investigative analysis: Deriving design principles from a case study. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, pages 139–146. IEEE, 2009.

[19] A. Kerne and S. M. Smith. The information discovery framework. In *Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques*, pages 357–360. ACM, 2004.

[20] Z. Liu and J. Heer. The effects of interactive latency on exploratory visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 2014.

[21] C. North. Toward measuring visualization insight. *Computer Graphics and Applications, IEEE*, 26(3):6–9, 2006.

[22] C. North, P. Saraiya, and K. Duca. A comparison of benchmark task and insight evaluation methods for information visualization. *Information Visualization*, 10(3):162–181, July 2011.

[23] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, volume 5, pages 2–4, 2005.

[24] C. Plaisant, J. Fekete, and G. Grinstein. Promoting insight-based evaluation of visualizations: From contest to benchmark repository. *Visualization and Computer Graphics, IEEE Transactions on*, 14(1):120–134, 2008.

[25] K. Reda, A. E. Johnson, J. Leigh, and M. E. Papka. Evaluating user behavior and strategy during visual exploration. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pages 41–45. ACM, 2014.

[26] D. M. Russell, M. J. Stefik, P. Pirolli, and S. K. Card. The cost structure of sensemaking. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 269–276. ACM, 1993.

[27] P. Saraiya, C. North, and K. Duca. An insight-based methodology for evaluating bioinformatics visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):443–456, July 2005.

[28] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

[29] B. Shneiderman and C. Plaisant. Strategies for evaluating information visualization tools: Multi-dimensional in-depth long-term case studies. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–7, New York, NY, USA, 2006. ACM.

[30] J. J. Thomas. *Illuminating the path:[the research and development agenda for visual analytics]*. IEEE Computer Society, 2005.

[31] A. Vande Moere, M. Tomitsch, C. Wimmer, B. Christoph, and T. Grechenig. Evaluating the effect of style in information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2739–2748, 2012.

[32] J. S. Yi, Y. ah Kang, J. T. Stasko, and J. A. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1224–1231, 2007.

[33] J. S. Yi, Y.-a. Kang, J. T. Stasko, and J. A. Jacko. Understanding and characterizing insights: how do people gain insights using information visualization? In *Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization*, page 4. ACM, 2008.

[34] M. X. Zhou and S. K. Feiner. Visual task characterization for automated visual discourse synthesis. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 392–399. ACM Press/Addison-Wesley Publishing Co., 1998.