Sneak Pique: Exploring Autocompletion as a Data Discovery Scaffold for Supporting Visual Analysis

Vidya Setlur[1], **Enamul Hoque**[2], **Dae Hyun Kim**[3], **Angel X. Chang**[4]
[1]Tableau Research, Palo Alto, CA, USA [2]York University, Toronto, ON, Canada
[3]Stanford University, Stanford, CA, USA [4]Simon Fraser University, Burnaby, BC, Canada
vsetlur@tableau.com, enamulh@yorku.ca, dhkim16@cs.stanford.edu, angel_xuan_chang@sfu.ca

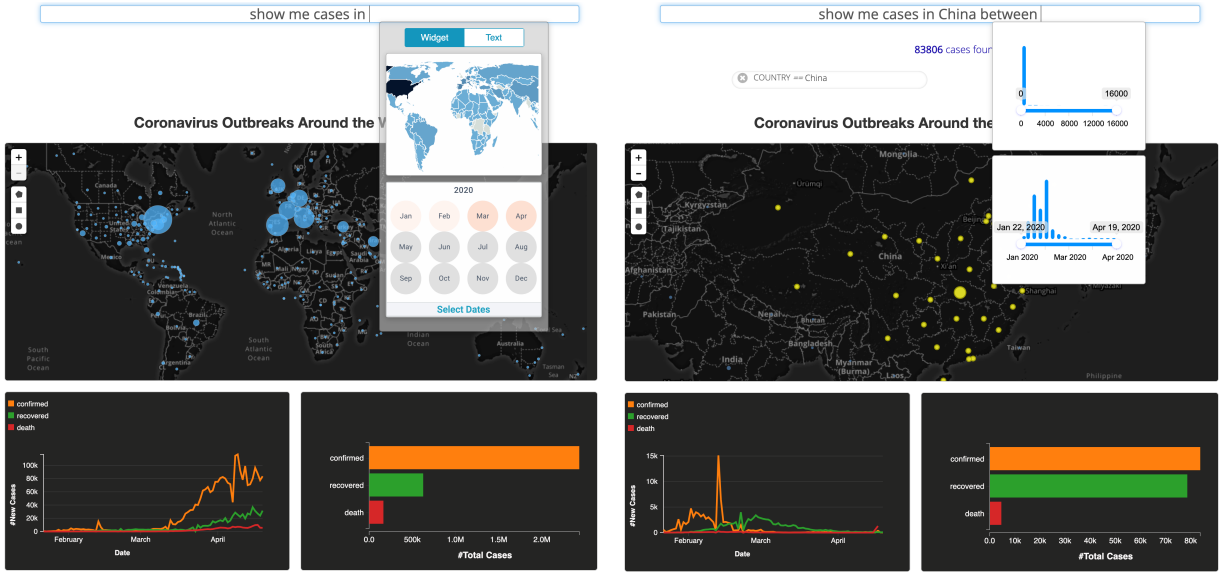


Figure 1. A screenshot of *Sneak Pique* with a dataset of coronavirus cases around the world. Left: A user types the query "show me cases in" and is prompted with map and calendar autocompletion widgets providing previews of the geospatial and temporal data frequencies respectively. The user could toggle to a text autocompletion dropdown to drill down into the geospatial or temporal data. Right: The user then clicks on `China` in the map widget and proceeds to find a range of cases by typing "between." The system displays a pair of date and numerical range widgets with corresponding histograms of data frequencies to help guide the user to pick a valid range based on the underlying data.

ABSTRACT

Natural language interaction has evolved as a useful modality to help users explore and interact with their data during visual analysis. Little work has been done to explore how autocompletion can help with data discovery while helping users formulate analytical questions. We developed a system called *Sneak Pique* as a design probe to better understand the usefulness of autocompletion for visual analysis. We ran three Mechanical Turk studies to evaluate user preferences for various text- and visualization widget-based autocompletion design variants for helping with partial search queries. Our findings indicate that users found data previews to be useful in the suggestions. Widgets were preferred for previewing temporal, geospatial, and numerical data while text autocompletion was preferred for categorical and hierarchical data. We conducted an exploratory analysis of our system implementing this specific subset of preferred autocompletion variants. Our insights regarding the efficacy of these autocompletion suggestions can inform the future design of natural language interfaces supporting visual analysis.

Author Keywords

autocompletion; data preview; natural language interaction; visual analysis.

CCS Concepts

•**Human-centered computing** → **Human computer interaction (HCI);** *Visualization;*

INTRODUCTION

The process of information-seeking has moved away from the traditional paradigm of assuming that the information goal is well-formed; even when users are deliberately seeking information, they do not necessarily know exactly what it is they want [12]. The translation of 'conceptual knowledge' into a query begins with some vaguely-felt need of wanting to know something and gradually evolves to the point where one can describe some attribute of the desired information [51]. Such exploratory search is a complex and cognitively demanding activity that depends on recall and sense-making [36].

Autocompletion is a useful mechanism for supporting this complex task, displaying in-situ suggestions as users type their

queries in the flow of their search tasks. Due to its effectiveness as a scaffold for guiding searchers to be productive, autocompletion is ubiquitous in various search environments [10]. Recently, natural language (NL) interaction in visual analysis tools has garnered interest in supporting expressive ways for users to interact with their data [8, 2, 3, 4].

While information seeking in a visual analysis task bears similarities to other forms of search, there are differences. Visual analysis involves the need to understand the characteristics of the underlying data and the various domains included in the dataset (e.g., range, level of detail of the attributes) [49]. One of the challenges for users in the context of visual analysis tools is the cognitive load of formulating NL queries based on their analytical inquiry [44, 26, 45].

Query reformulation is often based on the data domain being either too broad, narrow, or ill-formulated [13, 45]. Users need guidance to understand whether they are finding new insights with the visualization results returned. A lack of guidance can interfere with an accurate sense of progress toward the analytical goal [16]. Autocompletion in these NL systems is rather basic and tends to focus on *syntactic* completion of queries without any suggestions or helpful previews of the data [19]. There is a need for autocompletion in a visual analysis context to support query formulation with data discovery, guiding the user to make relevancy judgements.

To address this problem, we developed a novel interface system called *Sneak Pique*[1]. Our goal is to help anyone, regardless of skill set to interact with data using NL by bringing the fluidity of in-situ suggestions to analytical expressions typical of visual analysis tasks. We implemented a set of text- and widget-based autocompletion suggestions that provide data previews of the results before they are realized in the visualization. The system evaluates a query as it is being typed and provides data previews that are dynamically updated based on the syntactic structure of the query and the semantics of the tokens. Figure 1 shows examples of autocompletion suggestions generated in *Sneak Pique* as a user explores a dataset of coronavirus cases around the world [7]. Here, the user is prompted with autocompletion widgets providing appropriate previews of the underlying geospatial, temporal, and numerical data. For example, a missing token after "between" prompts a range for the user to choose from. The system also provides a mechanism to toggle from a widget to a corresponding text autocompletion dropdown to drill down into hierarchical data.

Contributions

In this paper, we explore how autocompletion can be used as a *data discovery* scaffold to help users during their visual analysis workflows. In the simplest form of autocompletion, the interaction surfaces a list of precomputed text suggestions to the user. We extend that basic form to previewing a data 'scent' of what a query will retrieve during visual analysis. Specifically, our contributions are summarized as follows:

- We explore a novel design space of autocompletion variants to better understand user preferences for (1) the display of data previews, (2) sort orders for suggestions, and (3) the navigation of data hierarchies during visual analysis.
- We implement an autocompletion system *Sneak Pique* as a design probe for implementing the various autocompletion variants from the design space. Our system employs a look-ahead parser to support basic syntactic completion of partial queries as well as dynamically suggesting missing data values in relevant text- and widget-based suggestions.
- We conduct crowdsourced studies of the autocompletion variants to better understand user preferences and reduce the vast space of design possibilities for these variants.
- The findings from the crowdsourced studies helped inform the subset of autocompletion variants in the final implementation of *Sneak Pique*. A subsequent exploratory evaluation of the system provides useful insights for future system design of NL input systems for visual analysis.

RELATED WORK

The primary goal of autocompletion is to suggest valid completions of a partial query with the intention of minimizing the time and effort for a user during a search task. There are various approaches to how autocompletion achieves this goal and can be categorized into three main categories: (1) Autocompletion to support syntactic query formulation, (2) Autocompletion to support information recall and preview, and (3) Autocompletion to support visual analysis.

Autocompletion to support syntactic query formulation

Query autocompletion (QAC) is prevalent in Web search, desktop search, and mobile devices where typing is laborious and error prone [17]. QAC techniques are employed in type-ahead search by providing suggestions that contain prefix characters from the query [55, 31, 33]. There has also been research exploring the utility of word [10] and phrase-level [25, 40] autocompletion. Systems have looked at ways to handle imprecision in search queries by developing error-tolerant QAC or fuzzy type-ahead [17, 32, 58, 20]. Ranking algorithms for generating suggestions have looked at temporal information [47, 56], personalization [46], and diversification to improve user recall in search [15]. While QAC techniques are useful for fact-finding information needs, they tend to be less effective for exploratory search. There is also additional complexity as information preview changes with the context of the query in play. In this paper, we explore how QAC can be extended to support data exploration that dynamically guides users while formulating syntactically correct NL utterances.

Autocompletion to support information recall and preview

Recall-oriented activity through information previews originated in early research in information science [11, 29]. DIALOG kept track of query history and those queries were reused by reference [21]. VOIR displayed the retrieval history of documents using histograms of rank information [23]. Ariadne generated a visual representation of a search trajectory to review earlier actions [53]. NRT implemented a history mechanism that recorded previously-run queries, making it possible for the searcher to scan the results list visually for

[1]The name Sneak Pique is inspired by the premise that autocompletion facilitates a lightweight preview (sneak peak) to pique a user's interest for information in their search journey [24].

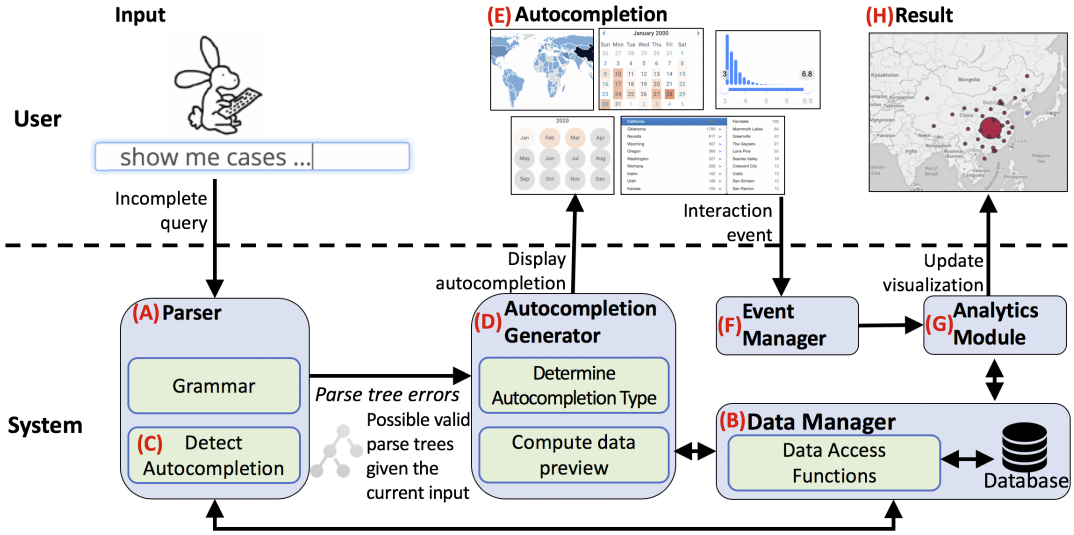


Figure 2. ***Sneak Pique*** **system overview**

new documents [43]. Nandi and Jagadish developed a search interface of a data schema showing the number of records for each possible suggested attribute [39]. AutoG showed possible graphs when the user drew a partial query graph [59]. Qvarfordt et al. designed a query preview widget that provided a visual summary of the results before the query was executed [42]. Their system oriented users in the result page by marking visited content and highlighting changes in the search documents. Inky displays visual feedback for web tasks with contextual information from a target web site [37]. Our work is similar in spirit to these systems, providing a hybrid experience between command line and GUI interfaces. However, prior work focuses primarily on document search goals that are different from visual analysis tasks.

Autocompletion to support visual analysis

While NL interfaces for data visualization have received considerable attention in recent years, they focus on limited syntactic text autocompletion without any preview of the underlying data [22, 44, 30, 48]. Power BI Q&A contextually displays textual suggestions as one types a question [2].Other systems support query reformulation where input utterances are translated into their corresponding canonical analytical expressions that represent the underlying system's language [8, 2, 45, 3, 4]. While these systems focus on guiding the user to type syntactically complete and analytically valid queries, they do not focus on providing a preview of the underlying data when the query involves a filter expression. The closest work to our research demonstrates how some graphical user interface controls called 'scented widgets' can support data analysis tasks [57]. Their system enhanced traditional visual widgets like sliders, combo boxes, and radio buttons with additional embedded visualizations to facilitate navigation in information spaces. In this paper, we implement *Sneak Pique* as a design probe to examine how *both* textual and visual variants of autocompletion with data previews provide users guidance within the context of NL interaction for visual analysis tasks.

SNEAK PIQUE SYSTEM

We introduce a system, *Sneak Pique*, that provides autocompletion suggestions with data preview information in an NL interface during visual analysis. Figure 2 illustrates *Sneak Pique*'s architecture. The system employs a web-based client-server architecture [6]. The input query is processed by an ANTLR [5] parser (A) using a context-free grammar containing predefined rules as well as rules dynamically added based on the attribute values from the underlying dataset [44, 26]. The parser accesses the dataset through the Data Manager (B), which provides functionality to handle data query requests. The Autocompletion Detection module (C) polls the query as the user is typing and triggers grammar parse tree errors when the query is partially complete. These parse errors are passed to the Autocompletion Generator (D) that introspects on the syntactic structure of the partial query along with relevant grammar rules that would be satisfied if the query were complete. The module determines the type of autocompletion suggestion required to resolve the partial query into a complete one. With the help of the Data Manager, the module computes the necessary data preview information that would be displayed in the autocompletion suggestion. The autocompletion suggestion is then rendered in the user interface of the client (E). Any interaction that the user performs with these autocompletion suggestions is captured by the Event Manager (F). The query upon execution, updates the D3 visualization result (H) through the Analytics Module (G) [14].

Autocompletion detection

Sneak Pique employs a left-to-right LL(*) parser [9, 41], performing a leftmost derivation of the input search query. We choose an LL(*) parser for generating autocompletion suggestions as this class of parsers can gracefully throttle up from conventional fixed $k \geq 1$ token lookahead to arbitrary lookahead and is able to backtrack if a suitable parse path cannot be constructed. The input to the parser is a grammar augmented with predicates with corresponding lookahead actions to trigger events being sent to the Suggestion Module. Each

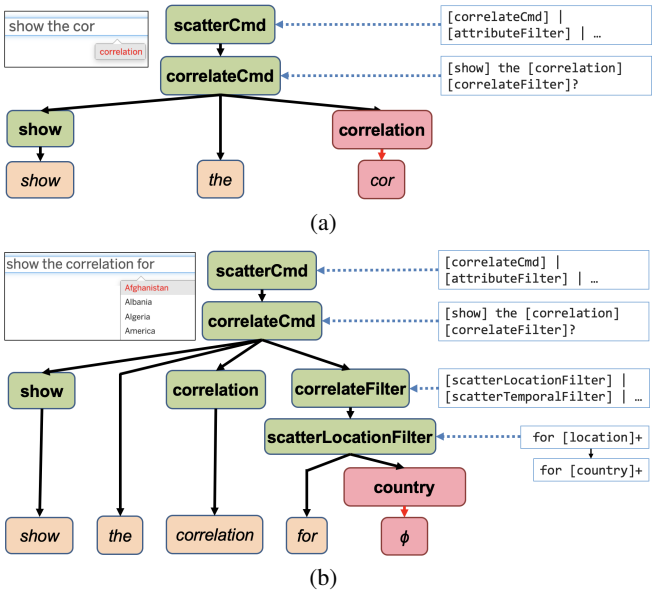


Figure 3. The parser computes lookahead parse trees and predicts tokens for triggering autocompletion. The green nodes depict grammar rules, the orange nodes depict lexicons, and the red nodes indicate parser errors. (a): Autocompletion suggests `correlation` to generate a valid parse for "show the cor." (b): As the user continues typing, based on the underlying data semantics, countries from the location data are suggested for "show the correlation for."

grammar rule encapsulates an analytical intent, similar to other NL visual analysis grammar systems [44, 26, 45]. The rules are composed of lexicons[2] that are either static (i.e., predefined in the grammar for built-in analytical functions such as 'in', 'correlation', and 'average') or dynamic (i.e., computed from the attributes and values from the database at real-time). The parser converts the input grammar to an equivalent augmented transition network (ATN), an efficient graph representation for grammars used in parsing relatively complex NL queries [28]. The state model is computed using a flow analysis that traces the ATN graph representation through all nodes reachable from the top-level node. Given a grammar $G = (N, T, P, S, \Pi, \mu)$, the ATN state machine, $A_G = (Q, \Sigma, \Lambda, E, F)$ has the elements:

- Q is the set of states
- Σ is the set of tokens $N \cup T \cup \Pi \cup \mu$
- Λ is the transition relation mapping $Q \times (\Sigma \cup \varepsilon) \rightarrow Q$
- $E = p_A | A \in N$ is the set of entry states
- $F = p'_A | A \in N$ is the set of final states

A_G is computed for each nonterminal lexical element from the grammar, creating a state model for each Σ. The nonterminal symbols form the syntactic structure of the parse and are replaced by terminal symbols, i.e., the leaf nodes in the parse tree. Nonterminal edges $p \rightarrow p'$ are function calls based on Λ that push the return state p' onto a parse state stack so it can continue from p' after reaching the stop state for the state flow. The parser simulates actions in the ATN to predict the next tokens in Q and computes a lookahead parse tree.

[2]In linguistics, a lexicon is a vocabulary of words and phrases that have known semantic meaning.

Prediction errors occur in the ATN when a sequence does not resolve to a corresponding grammar production rule for the current nonterminal. The LL(*) prediction state model reports an error at the specific token and scans ahead to determine if there are any nonterminals that can resolve the error. For autocompletion to trigger, the shortest lookahead sequences are identified that would generate valid parse trees. The autocompletion detection algorithm is generalizable to both static and dynamic lexicons. For example, in Figure 3a, the query "show the cor" generates a parser error at the nonterminal node `cor` as the token does not match any grammar production rule. The parser computes a lookahead to find the static lexicon `correlation`. In Figure 3b, "show the correlation for" results in an error further down in the parse tree as the query is missing a country name, a dynamically generated lexicon from the dataset. Parse tree error information provides input to the Autocompletion Generator for rendering the suggestions.

Autocompletion generation

After detecting when to trigger autocompletion in the parsing process, we determine how the autocompletion suggestions should be presented in the interface. One of the key guiding principles for designing autocompletion interfaces is recognition over recall: the notion that people are better at recognizing things they have previously experienced than they are at recalling them from memory. Autocompletion also helps with information discovery with unfamiliar data, providing guidance when no recall exists [52].

Design patterns for autocompletion

We draw inspiration from an established set of best practices for implementing autocompletion [38]. The following design patterns emerged while iteratively designing autocompletion suggestions in *Sneak Pique*.

- D1: *Provide suggestions in context of the partial search query*. Autocompletion should assist users when searching by presenting items that match the users input as they type. As the user types in more text into the search field, the list of matching items is narrowed down.
- D2: *Sort order*. Autocompletion should sort items with the most relevant or likely match at the top of the list. This will allow the user to quickly select their match.
- D3: *Semantic grouping*. Autocompletion should group similar items into categories for easy scan and lookup.
- D4: *Reduce visual noise*. Autocompletion should not add cognitive overload and only display information as a preview, not as a final result.
- D5: *Manageable list*. Autocompletion should limit the number of matching items to display, especially when working with a large number of data values.
- D6: *Lightweight interactivity*. Autocompletion should support lightweight interaction allowing users to select an item, saving time and keystrokes. Autocompletion should remain unobtrusive so that users can still type in a complete query.

Determine autocompletion type

Combining autocompletion design patterns with best practices for information visualization display [16, 34], we implement various text and widget-based autocompletion representations

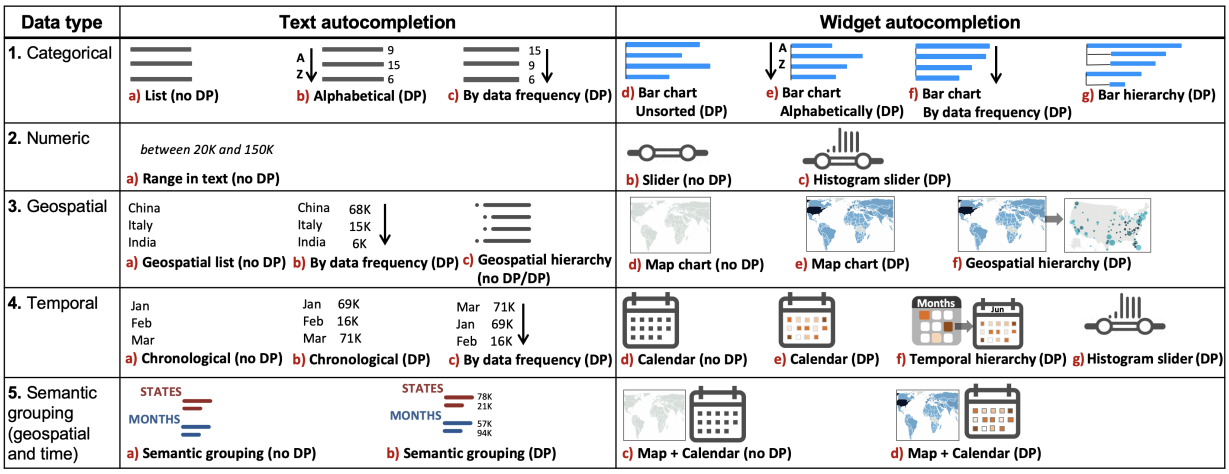


Figure 4. Our design space for autocompletion where each row shows various text and widget-based representations by data type. Data Preview (DP) variants display data frequency numbers of the values.

in D3 [14]. Using *Sneak Pique* as a design probe, we implement autocompletion variants based on the data type of the missing dynamic lexicon in the input query. To explore the utility of displaying the data preview, we generate these variants with and without data frequency numbers that indicate how often the values occur in the dataset. We also generate various sort orders based on the data type. Figure 4 depicts the following set of variants:

- **Categorical**: A text list or a bar chart shows suggestions for a categorical attribute with various sort orders. If a data preview is present, numbers are displayed in the list or encoded as bar lengths (Row 1).
- **Numeric**: Text or a slider widget shows the data value range for an attribute. A variant of the slider widget shows a histogram to encode data frequency (Row 2).
- **Geospatial**: A list or a map chart widget[3] shows location values. If a data preview is present, numbers are displayed in the list or as a visual encoding on the map. For hierarchical data[4], a nested listed view or map widget is provided to drill-down (e.g. from country to city) (Row 3).
- **Temporal**: A list or a calendar widget shows time values. If a data preview is present, the information is displayed in the list or as color encodings in the calendar. Based on temporal intent, the calendar widget defaults to the appropriate level of detail. For example, "cases on" shows a calendar with a date view while "cases in" shows the month view. For a temporal range, the system displays a slider (Row 4).
- **Semantic grouping**: In NL systems, geospatial and temporal intent can be ambiguous [45]. For example, the query "show earthquakes in" could indicate either a missing location or time. We address this ambiguity between the geospatial and temporal filter grammar rules by semantically grouping geospatial and temporal values in the autocompletion suggestions as a text list or as a combination of map and calendar widgets (Row 5).

[3]We employ an Equal Area map projection that tends to be conducive for click interaction in a small display area [54]

[4]Hierarchical data is a tree structure representation of data records.

The system also provides partial text matches to dynamic data values and static analytic concepts. For example, when the user types "ma" the system shows matched suggestions for both `Massachusetts` and `maximum`.

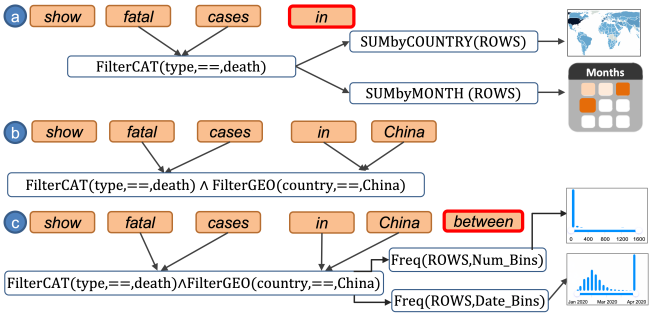


Figure 5. An example of data preview computation in *Sneak Pique*. Widgets with data previews are triggered after the nodes highlighted in red. (a) Here, the user starts with a query "show fatal cases in," which displays map and calendar widgets. In the background, the system applies a filter to select only `death` cases and then applies appropriate aggregation methods to show the `sum of cases` as data previews in the widgets. (b) Next, the user clicks on `China` in the map widget resulting in a complete query. (c) When the user adds "between," the system predicts ranges for `China` in the numeric and temporal widgets.

Compute data preview

The system computes the data frequency for the attributes associated with the predicted dynamic tokens and displays the information. The data preview generation dynamically updates the data preview results based on the context of the current query. The system applies appropriate aggregate and filter analytical functions similar to [26]. Figure 5 illustrates this process. Given the query "show fatal cases in," the system executes the analytical function `FilterCAT(caseType,==,death)`, applying a filter on 'death.' Sum of cases is aggregated by country in the map widget and by month in the calendar widget.

Before generating data previews, the system checks for the presence of attributes in the query to prevent duplicates from being added when interacting with the autocompletion widgets.

In Figure 4c, for the query "show me fatal cases in china and," the system applies `FilterCAT(caseType,==,death)`, but does not allow for `FilterGEO(country,==,China)` as `China` is already present in the query.

EVALUATING AUTOCOMPLETION VARIANTS

No guidelines exist for the appropriate way to show autocompletion suggestions specifically designed for visual analysis. It is unclear what user preferences are for each of these autocompletion variants and how those preferences vary based on data type, sort order, or actual representation. Our goal is to probe some of these characteristics of what an appropriate autocompletion suggestion would look like for formulating sensible defaults in a visual analysis NL interface. Objective measures in terms of speed or accuracy are not applicable for this study, since we are interested in what people think is the most appropriate autocompletion variant. A good outcome of these studies is a strong agreement among participants in certain conditions to help formulate a set of reliable design guidelines. As a step in that direction, we ran three experiments to tease apart user preferences for the various factors that influence the choice for autocompletion suggestions:

- Experiment 1 compares autocompletion variants with and without data frequency information displayed to understand if such data previews are useful to the user.
- Experiment 2 examines the sort order that would be useful to apply to items shown in text autocompletion suggestions.
- Experiment 3 compares autocompletion variants that display data values with and without hierarchies to better understand the handling of hierarchical data in the suggestions.

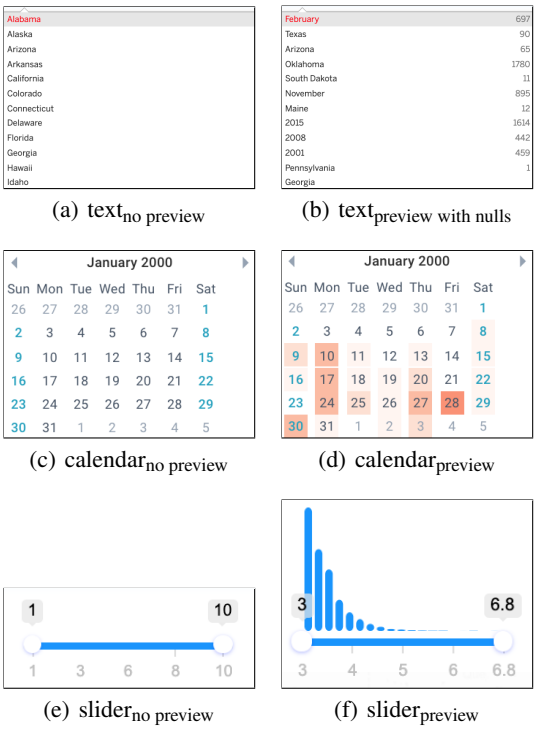
We hypothesize that participants will find data preview information to be useful across all autocompletion variants. We also hypothesize that participants would prefer items sorted in descending order of their data frequencies as more prevalent data values should show up higher in the suggestions; except for temporal items that should be in chronological order. We hypothesize that hierarchical text suggestions would be easier to navigate than widgets. However, participants might find the calendar widget helpful for navigating temporal hierarchies.

Experiment Design

For each of the experiments, participants were recruited from Amazon Mechanical Turk [18]. Participants were English speakers in the U.S. with at least a 95% acceptance rate and 500 approved tasks. We payed a rate equivalent to $1.50 for 10 minutes of effort. The stimuli did not require excluding participants for color deficiencies. Participants could complete only one trial to avoid biases that might arise from repeated exposure to the task.

We used an earthquakes dataset [50] with magnitudes by location and time in the U.S. as this is likely to be understandable to a broad population. The experimental procedure was:

- Training task: A chart shows a scatterplot of the relationship between life expectancies and income for various countries with practice questions to ensure understanding of autocompletion suggestions.



(a) $\text{text}_{\text{no preview}}$ (b) $\text{text}_{\text{preview with nulls}}$

(c) $\text{calendar}_{\text{no preview}}$ (d) $\text{calendar}_{\text{preview}}$

(e) $\text{slider}_{\text{no preview}}$ (f) $\text{slider}_{\text{preview}}$

Figure 6. A sample set of comparison pairs from Experiment 1. (a) Text variant with no data preview, (b) text variant with data frequencies, showing a null value `Georgia`, (c) calendar widget with no data preview, (d) calendar widget with data frequency encoded by color, (e) slider widget to select a numerical range, (f) slider widget with a histogram preview of data frequency within the range.

Comparison 1	Comparison 2
$\text{text}_{\text{no preview}}$	**$\text{text}_{\text{preview without nulls}}$ (75.0%)**
$\text{text}_{\text{no preview}}$	**$\text{text}_{\text{preview with nulls}}$ (66.7%)**
$\text{text}_{\text{preview with nulls}}$	**$\text{text}_{\text{preview without nulls}}$ (75.0%)**
$\text{calendar}_{\text{no preview}}$	**$\text{calendar}_{\text{preview}}$ (91.7%)**
$\text{slider}_{\text{no preview}}$	**$\text{slider}_{\text{preview}}$ (66.7%)**
$\text{map}_{\text{no preview}}$	**$\text{map}_{\text{preview}}$ (83.3%)**

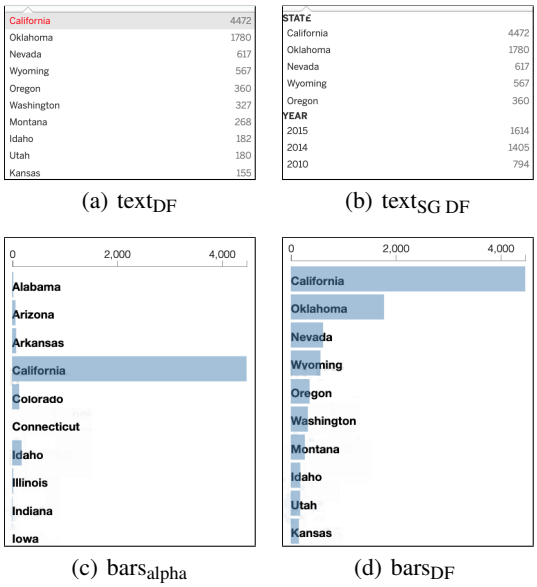
Table 1. Comparisons of autocompletion with and without data previews. The top user preferences are highlighted in bold. In general, participants preferred autocompletion variants with a data preview and nulls removed from the list of text suggestions.

- Presentation of the overall task description and instructions.
- Actual task: A new page showing a search box with the utterance, "find earthquakes [in/between]..." The participant is shown two image autocompletion choices (see Figures 6,7,8 for examples) in randomized order asking the participant to pick their top preferred choice. A freeform text response box is provided for reason for preference.

Due to space constraints, we show a sample set of stimuli for each experiment in the paper. Details including the complete set of comparisons are available in the supplementary material.

Experiment 1: Evaluating the utility of data preview

The experiment consisted of of six comparison pairs where text, calendar, and map autocompletion variants were shown



(a) text_{DF} (b) $\text{text}_{\text{SG DF}}$

(c) $\text{bars}_{\text{alpha}}$ (d) bars_{DF}

Figure 7. A sample set of comparison pairs from Experiment 2. (a) Text variant sorted by data frequency (DF), (b) text variant showing semantic grouping (SG) and sorted by data frequency, (c) text variant sorted alphabetically with bars showing data frequency, (d) text variant sorted by data frequency with bars showing data frequency.

with (Figures 6b,d,f) and without data frequency information (Figures 6a,c,e). The data frequency is shown as numbers alongside the text items. We also considered variants that included and excluded null data values in order to understand users' preferences for nulls in the previews (Figure 6b). The information is color encoded in both the calendar (Figure 6d) and map widgets, wherein a darker color indicates a higher data frequency. In the slider widget, the data frequency is represented as a histogram (Figure 6f).

Results: Data previews were preferred without null values

We collected data for 72 participants with an even spread across all comparisons. As hypothesized, participants preferred seeing data previews in all of the autocompletion variants except when nulls were shown in the text drop-downs. Feedback from participants indicated that they expected the autocompletion previews to simply exclude the null values to minimize visual clutter. The user preferences across each of these comparisons are shown in Table 1. There was particularly a strong preference for previews in calendars for 11 out of 12 participants (91.7%), followed by maps with 10 out of 12 participants (83.3%). As participant *P*63 stated, "The results show me what's in the data, but the colors on numbers are useful to let me know how much of it is there." Eight out of 12 participants slightly preferred the text dropdown with previews showing nulls compared to the variant without a preview. Participants indicating a preference for no preview found the display of nulls in the drop-down to be not useful. Similarly eight out of 12 participants preferred the slider variant with the data preview shown, with the rest indicating that they liked the simpler design without the preview.

Comparison 1	Comparison 2
$\text{text}_{\text{alpha}}$	**text_{DF} (58.3%)**
text_{DF}	**$\text{text}_{\text{chrono}}$ (58.3%)**
$\text{text}_{\text{SG alpha without labels}}$	**$\text{text}_{\text{SG alpha with labels}}$ (100.0%)**
$\text{text}_{\text{alpha}}$	**$\text{text}_{\text{SG alpha with labels}}$ (75.0%)**
$\text{text}_{\text{alpha}}$ (66.7%)	$\text{text}_{\text{SG alpha without labels}}$
$\text{text}_{\text{SG DF without labels}}$	**$\text{text}_{\text{SG DF with labels}}$ (91.7%)**
text_{DF}	**$\text{text}_{\text{SG DF with labels}}$ (66.7%)**
text_{DF}	**$\text{text}_{\text{SG DF without labels}}$ (75.0%)**
$\text{text}_{\text{SG chrono without labels}}$	**$\text{text}_{\text{SG chrono with labels}}$(91.7%)**
$\text{text}_{\text{chrono}}$	**$\text{text}_{\text{SG chrono with labels}}$ (83.3%)**
$\text{text}_{\text{chrono}}$	**$\text{text}_{\text{SG chrono without labels}}$ (83.3%)**
$\text{bars}_{\text{alpha}}$	**bars_{DF} (83.3%)**

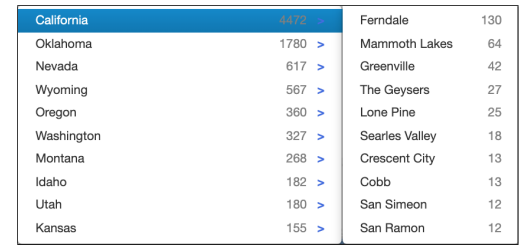
Table 2. Comparisons with top user preferences highlighted in bold for various sort types: alphabetical (alpha), by decreasing order of data frequency (DF) and chronological order (chrono) along with semantic grouping (SG). Participants preferred SG, with labeled preferred over unlabeled. In the absence of SG, DF sort was preferred; although chronological ordering was preferred for temporal items.

Experiment 2: Examining sort order preferences

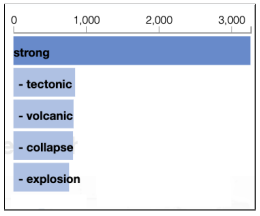
This experiment was conducted to better understand user preferences for sort order in a list of text autocompletion suggestions. Various sorting preferences were evaluated: alphabetical, data frequency, chronological ordering as well as semantic grouping of related items such as places and time concepts (Figure 7). A total of 12 comparison pairs were used as stimuli in the experiment. Given the precedence of research that chronological sorting is preferred over alphabetical sort [35], we did not compare those two variants, keeping the total number of comparisons to a manageable number.

Results: Semantic grouping with labels was most preferred

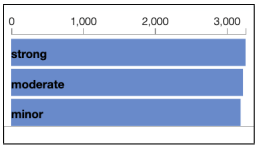
144 participants took part in this experiment with an even spread across all comparisons. The user preferences are shown in Table 2. Participants preferred semantic grouping of related items and found labeling of the groups to be useful. P119 commented, "I wanted to see by both year and country and liked having that choice." P24 stated, "It's easier to have the dates separate from the other areas so I prefer this one." Semantic grouping without labels was preferred when compared with plain text variants without any semantic grouping, sorted by data frequency or chronological order. We did find a surprising result for alphabetical sort, where eight out of 12 (66.7%) participants preferred plain text over semantic grouping without the labels displayed. Participants in their feedback mentioned that the unlabeled semantic groups were difficult to scan. P34 explained their rationale by commenting, "The groups without labels was confusing and the alphabetical sort was easy to read." For temporal data, there was a slight preference for chronological order rather than sorting by data frequency (seven out of 12 (58.3%). For non-temporal data, sorting by data frequency was slightly preferred for both the plain text variant (Seven out of 12 participants) and the text variant showing data frequency bars (Eight out of 12 participants).



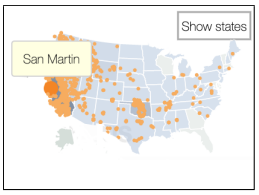
(a) $\text{text}_{\text{cat hierarchy}}$



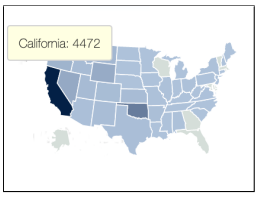
(b) $\text{bars}_{\text{cat hierarchy}}$



(c) $\text{bars}_{\text{no cat hierarchy}}$



(d) $\text{map}_{\text{geo hierarchy}}$



(e) $\text{map}_{\text{no geo hierarchy}}$

Figure 8. A sample set of comparison pairs from Experiment 3. (a) Text variant with a categorical hierarchy for states and cities, (b) text variant with data frequency bars. Clicking on an item expands its children, (c) text variant with data frequency bars showing only the top level of earthquake types, (d) a map widget showing cites overlaid on states with one of the values shown in a tooltip. A user can switch between state and city levels to select a value, (d) a map widget showing only state level information as indicated by the tooltip.

Experiment 3: Evaluating the display of hierarchical data

We ran this experiment to better understand user preferences for both autocompletion display and interaction with and without hierarchical data. We limited the experiment to two-level hierarchies to minimize the complexity of the task. The experiment consisted of 10 comparisons, comparing plain text autocompletion (Figure 8a), text with bars indicating data frequencies (Figures 8b and c), calendar and map widgets (Figures 8d and e) with and without hierarchies. We showed animated gifs to help the participant understand the navigation interaction between the hierarchies rather than static images where the participant would have to view all the hierarchy levels at the same time. We did not have a no-hierarchy variant for the calendar widget as hierarchical temporal interaction is a ubiquitous modality for calendar interaction [27].

Results: Text autocompletion was preferred for hierarchies

120 participants took part in this experiment with an even spread across all comparisons. The user preferences across each of these comparisons are shown in Table 3. 10 out of 12 participants (83.3%) indicated that they preferred seeing hierarchies in the text autocompletion variants compared to no hierarchies and those preferences were consistent across categorical, temporal and geospatial hierarchies. As hypothesized,

Comparison 1	Comparison 2
$\text{text}_{\text{cat hierarchy}}$ (83.33%)	$\text{text}_{\text{no cat hierarchy}}$
$\text{text}_{\text{geo hierarchy}}$ (83.33%)	$\text{text}_{\text{no geo hierarchy}}$
$\text{text}_{\text{time hierarchy}}$ (83.33%)	$\text{text}_{\text{no time hierarchy}}$
$\text{text}_{\text{cat hierarchy}}$ (66.67%)	$\text{bars}_{\text{cat hierarchy}}$
$\text{text}_{\text{no cat hierarchy}}$ (83.33%)	$\text{bars}_{\text{no cat hierarchy}}$
$\text{bars}_{\text{cat hierarchy}}$	**$\text{bars}_{\text{no cat hierarchy}}$ (91.67%)**
$\text{text}_{\text{time hierarchy}}$	**$\text{calendar}_{\text{time hierarchy}}$ (100.0%)**
$\text{text}_{\text{geo hierarchy}}$ (83.33%)	$\text{map}_{\text{geo hierarchy}}$
$\text{text}_{\text{no geo hierarchy}}$	**$\text{map}_{\text{no geo hierarchy}}$ (91.67%)**
$\text{map}_{\text{geo hierarchy}}$	**$\text{map}_{\text{no geo hierarchy}}$ (58.33%)**

Table 3. Comparisons of autocompletion with and without hierarchies; top user preferences highlighted in bold. For space considerations, `cat`, `geo`, `time` stand for categorical, geospatial and temporal hierarchies respectively. Plain text variants were generally preferred for showing hierarchies. The map widget was preferred for state-level data and the calendar widget for temporal hierarchies.

when hierarchies were shown, text autocompletion was consistently preferred over bar and map widgets. *P*91 explained, "It was simpler. I figured I could drill down to find what I wanted." *P*8 indicated "I think that it's easier to navigate each level by name than with a map." All participants preferred the calendar widget over the text autocompletion for temporal hierarchies. 11 out of 12 participants preferred the map widget over the text autocompletion when no hierarchy was shown. *P*69 stated, "Not gonna lie, I haven't memorized the US map and seeing where the data is [in the widget], helped." Text autocompletion with bars was not preferred when compared to plain text autocompletion with or without a data hierarchy. Participants said, "It's far easier to compare with the raw totals than shading [*P*23]" and "the numbers are all so close that the bars only served to be a distraction [*P*20]."

Autocompletion variants for the final implementation

User preferences from the three experiments helped identify a subset of autocompletion variants in the final implementation of *Sneak Pique*. Referring to Figure 4, we finalized text (Row 1c), histogram sliders (Rows 2c and 4g), map (Row 3e), calendar (Row 4f) widgets for displaying categorical, numerical and temporal ranges, geospatial, and temporal value completion respectively. Details are as follows:

- **Data previews are useful**: We updated all autocompletion variants to display data previews. Based on user feedback, text autocompletion was shown with nulls filtered out. As none of the participants chose bars for data frequency numbers, that variant was eliminated in the final implementation.
- **Sorting preferences**: Geospatial and categorical text autocompletion was sorted by data frequency. For temporal data, chronological sort was applied (Row 4b).
- **Handling hierarchical data**: Text autocompletion suggestions displayed geospatial and categorical hierarchical data, while the calendar widget showed temporal hierarchical data. We added a toggle button for the map and calendar widgets (Row 5d) to switch to a semantically grouped text view for navigating hierarchical data (Row 5b).

SYSTEM EVALUATION

We conducted a user study of *Sneak Pique* with the following goals: (1) collect qualitative feedback on the autocompletion suggestions for various visual analysis tasks and (2) identify system limitations. This information would help provide insights as to how autocompletion interaction ideas could integrate into a more comprehensive NL visual analysis interface. The study had two parts: Part 1 examined the various autocompletion types that we implemented in the final version of *Sneak Pique*: map, calendar, and range widgets, along with text autocompletion for hierarchical data navigation. Part 2 was observational and aimed to see how people would utilize the autocompletion mechanisms in an open-ended way.

We initially considered a comparative experiment with other NL systems; however, a study of Eviza [44] had already revealed how basic text query autocompletion was a commonly used feature for helping users formulate utterances. Other systems [8, 2, 4] provide complete suggestions rather than token-based autocompletion employed in *Sneak Pique*. Comparing with such systems would only highlight the lack of autocompletion to support data discovery in the former systems rather than leading to new insights. In the future, we will evaluate specific features and the types of queries users ask by comparing *Sneak Pique* to a feature-removed version of itself. Because the main goal of our study was to gain qualitative insight into the advantages of each type of autocompletion, we encouraged participants to think aloud with the experimenter.

Method

Participants

We recruited eight volunteers (four males, four females, age 24-55). All were fluent in English (five native speakers) and all regularly used NL interfaces with autocompletion such as Google. Five used a visualization tool on a regular basis and the rest considered themselves beginners.

Tasks

Each participant used a dashboard showing coronavirus outbreaks for the following parts of the study:

- **Part 1 - Target criteria tasks**: We provided four target tasks; participants interacted with *Sneak Pique* to generate partial queries to trigger the autocompletion for answering questions on place, time, and numerical filtering. These tasks were chosen to cover the usage of all the autocompletion variants in *Sneak Pique*. The questions posed to the participants were intentionally vague so that participants would not simply type the question and find the answer directly in the dashboard without using autocompletion. Each set started with a new query (following a reset by clearing the search box or typing "start over", similar to other NL systems [8]) and each of the remaining tasks transitioned the criteria through one of the task types. To avoid priming participants with specific wording, criteria were presented orally (see supplemental material).
- **Part 2 - Open-ended tasks**: Here, our goal was to qualitatively observe how people would use autocompletion in an unscripted interaction with the dashboard.

Procedure and Apparatus

We began with a short introduction of possible interactions. Participants were instructed to phrase their queries in whatever way that felt most natural, to think aloud, and to tell us whenever the system did something unexpected. We discussed reactions to system behavior throughout the session and then concluded with a semi-structured interview. All the study trials were done remotely over a shared screen videoconference to conform with social distancing protocol due to COVID-19. All sessions took approximately 30 minutes and were recorded.

Analysis Approach

We employed a mixed-methods approach involving qualitative and quantitative analysis, but considered the quantitative analysis mainly as a complement to our qualitative findings. The primary focus of our work was a qualitative analysis of how autocompletion with data previews influenced people's analytical workflows. We conducted a thematic analysis through open-coding of session videos, focusing on strategies participants took. The quantitative analysis consisted of the query lengths and how often participants used the autocompletion variants. Given the remote nature of the study setup, we did not measure the time taken for task completion.

Results

Overall, participants were positive about the autocompletion interaction and identified many benefits. *Sneak Pique* allowed participants to introspect on the data as they were typing ("This is cool...provides me a way to see what I will get while I am typing my question" [$P'1$][5]), helped them proactively discover what was in the data ("By typing, I can already filter to a specific country and see what's in there without having to see the result and try again....I don't have to shoot darts in the dark" [$P'4$]), and could save time ("I could finish the tasks really fast as the autocompletion guided me to see where to look." [$P'7$]).

Part 1 - Target criteria tasks

Six out of the eight participants were able to complete all tasks successfully with all participants interacting with map, calendar, range, and hierarchical widgets to complete the tasks. We observed that tasks were easier to complete when the data frequency information in the autocompletion widgets was visually discernible. One participant struggled to visually compare countries or months when picking values with either a high or low incidence of coronavirus cases. Another participant had difficulty accessing hierarchical data in *Sneak Pique*.

Part 2 - Open-ended tasks

The open-ended task demonstrated how autocompletion was helpful for data discovery while users typed their queries. We observed that participants surfaced autocompletion for both syntactic query completion and for completing filter expressions by place, time, and range. The number of individual queries per participant ranged from 8 to 23 ($\mu = 11.3$) with 46% of them being reformulations of previously typed queries by editing in place. Overall, a good number of partial queries used autocompletion with data previews to help resolve into complete ones (69%).Usage of widgets was roughly split even

[5]We use the notation $P'X$ to indicate participant IDs in these study results to distinguish from those in the Mechanical Turk studies.

across calendar (34%), sliders (33%), and map (29%) widgets; the rest being text autocompletion for accessing hierarchical data. Comments relevant to this behavior included, "It was convenient to type to an extent and rely on the calendar to go to a specific date that was interesting. I hate typing dates" [$P'4$] and "Getting a range right is a hit or miss for me. Helpful to see where most of the data is and pick with slider" [$P'1$]. We also observed participants directly typing the NL query for specific questions such as "show me the cases in New York last month" and "highest cases in India" with 33% of the total number of queries belonging to this category.

The study also revealed several shortcomings. Overall, 18% of the queries were unsuccessful with *Sneak Pique* unable to understand the requests. A small percent of them (8%) were due to related concepts that were not understood. For example, the system was unable to interpret the inflected form of 'death' in "people who died in US" [$P'5$]. The remaining unsuccessful queries consisted of unsupported analytics such as "show me the places where cases are declining" [$P'3$] and "what is the average rate of deaths in march for new york?" [$P'8$].

DISCUSSION AND FUTURE WORK

An evaluation of *Sneak Pique* confirmed our intuition that people find data previews useful in autocompletion while performing visual analysis. Results suggest that participants put more thought into the search terms when the preview was present; they were engaged in more sense-making behavior both during query construction and when examining the search results. Observations from the study provide the following design implications for how autocompletion can help with visual analysis tasks, opening new opportunities for future research:

Autocompletion for varied visual analysis workflows: An effect of task intervention during the first part of the study was that there were fewer query reformulations as participants utilized data previews to complete the tasks. The second part of the study was more representative of real world practice, and we did observe change in tactics in how people formulated their queries. Participants used the data previews as a scaffold to construct compound queries where filters were incrementally updated in the original query. $P'2$ said, "I find it convenient to pile more filters in my question as the views in the autocompletion get updated...saves me time when I'm playing around." During data exploration, participants would remove or clear these filter subexpressions if the data previews were not interesting anymore. We observed that for queries where participants already knew the filter values they were interested in, they would type the question quickly without pausing to prompt for data previews. However, the text autocompletion helped guide them while typing the tokens. These observations indicate that autocompletion is used in different ways based on the type of inquiry. Future studies should explore more deeply how autocompletion workflows can adapt to a range of tasks and in the context of established visual analytics systems [8, 4, 2].

Support for more complex previews: While the study participants liked the overall idea and utility of the system, there were limitations in supporting more complex analytics. For example, $P'3$ commented, "I want to type 'show me cases with declining trends...' and get a widget showing me the countries where the cases were going down so I can decide where to look." For queries with both geospatial and temporal intents such as "coronavirus cases in," participants expected the map and calendar widgets to be coordinated during interaction. Balancing interaction simplicity with more complex previews to serve a greater gamut of analytical questions, is worth exploring; there lies a sweet-spot for adding functionality into the autocompletion itself vs. letting the user explore the results in the visualization. The autocompletion behavior would need to be performant to support real-time interaction.

Showing provenance of autocompletion behavior: While most of the participants understood the purpose of the data previews, they described usability issues around understanding autocompletion behavior based on what was in context. When one or more filters are in play, the data previews are dynamically updated to reflect the current data domain. The behavior was not always intuitive and either required clarification by the experimenter or the participant would eventually figure out the functionality after attempting to select a disabled item in the widget. $P'7$ stated - "It would be good if I can see a message appearing saying that I am already looking at April and the data in the autocompletion is for that month." While the dynamic generation of autocompletion suggestions helps provide in-situ guidance to the user, we need to explore ways to show this feedback to set appropriate user expectations.

Personalization of autocompletion suggestions: The topic of personalization of autocompletion behavior came up during the exploratory study. $P'3$, $P'4$, and $P'8$ expressed that they wanted the autocompletion to keep track of their past interaction and update the default views. For example, $P'4$ said "I am interested in the days and not months as these coronavirus cases are changing so much. I don't want to keep switching from month to day view every time." An interesting direction is to monitor user interaction with *Sneak Pique*, record user queries, and update autocompletion preferences.

CONCLUSION

Autocompletion is a useful interaction paradigm for information sense-making. We implemented *Sneak Pique* as a design probe to explore autocompletion in the context of visual analysis and data discovery. The system uses a look-ahead parser to resolve static and dynamic tokens for text and widget autocompletion variants. User study results showed that data previews are indeed useful and informed how such information could be usefully presented in the interface. We implemented a subset of preferred autocompletion variants in the final implementation of *Sneak Pique*. A preliminary evaluation of the system validated our premise that autocompletion can serve as a data scaffold to help users make relevance judgments during visual analysis. We hope that insights learned from our work can identify unique opportunities for striking a balance between lightweight interactivity and rich analytical previews.

REFERENCES

[1] 2019. Watson Analytics, https://www.ibm.com/watson-analytics. (2019). https://www.ibm.com/watson-analytics

[2] 2020. ANTLR (ANother Tool for Language Recognition)). https://www.antlr.org/. (2020).

[3] 2020. Microsoft Q & A. https://powerbi.microsoft.com/en-us/documentation/powerbi-service-q-and-a/. (2020).

[4] 2020. Node.js®. https://nodejs.org/. (2020).

[5] 2020. Novel Coronavirus (COVID-19) Cases, provided by Johns Hopkins University). https://github.com/CSSEGISandData/COVID-19. (2020).

[6] 2020. Tableau's Ask Data. https://www.tableau.com/products/new-features/ask-data. (2020).

[7] 2020. ThoughtSpot. http://www.thoughtspot.com/. (2020).

[8] Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation, and Compiling*. Prentice-Hall, Inc., USA.

[9] Holger Bast and Ingmar Weber. 2006. Type Less, Find More: Fast Autocompletion Search with a Succinct Index. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '06)*. Association for Computing Machinery, New York, NY, USA, 364–371. DOI:http://dx.doi.org/10.1145/1148170.1148234

[10] Nicholas Belkin. 2014. Anomalous States Of Knowledge As A Basis For Information Retrieval. *Canadian Journal of Information Science* (11 2014), 133–143.

[11] Nicholas Belkin, R.N. ODDY, and H.M. BROOKS. 1982. ASK for information retrieval: Part I. Background and theory. *Journal of Documentation* 38 (12 1982), 61–71. DOI:http://dx.doi.org/10.1108/eb026722

[12] Johan Bos. 2004. Computational Semantics in Discourse: Underspecification, Resolution, and Inference. *J. of Logic, Lang. and Inf.* 13, 2 (March 2004), 139–157. DOI: http://dx.doi.org/10.1023/B:JLLI.0000024731.26883.86

[13] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D3: Data-Driven Documents. *IEEE Transactions on Visualization & Computer Graphics (Proc. InfoVis)* (2011). http://vis.stanford.edu/papers/d3

[14] Fei Cai, Ridho Reinanda, and Maarten De Rijke. 2016. Diversifying Query Auto-Completion. *ACM Trans. Inf. Syst.* 34, 4, Article Article 25 (June 2016), 33 pages. DOI:http://dx.doi.org/10.1145/2910579

[15] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. 1999. *Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 579–581 pages.

[16] Surajit Chaudhuri and Raghav Kaushik. 2009. Extending autocompletion to tolerate errors. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 707–718.

[17] Kevin Crowston. 2012. Amazon Mechanical Turk: A Research Tool for Organizations and Information Systems Scholars. In *Shaping the Future of ICT Research. Methods and Approaches*, Anol Bhattacherjee and Brian Fitzgerald (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 210–221.

[18] Giovanni Di Santo, Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2015. Comparing Approaches for Query Autocompletion. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 775–778. DOI: http://dx.doi.org/10.1145/2766462.2767829

[19] Huizhong Duan and Bo-June (Paul) Hsu. 2011. Online Spelling Correction for Query Completion. WWW 2011. https://www.microsoft.com/en-us/research/publication/online-spelling-correction-for-query-completion/

[20] Ina Fourie. 2000. Online Retrieval: A Dialogue of Theory and Practice (2nd ed.). *Electronic Library, The* 18 (12 2000), 448–469.

[21] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G. Karahalios. 2015. DataTone: Managing Ambiguity in Natural Language Interfaces for Data Visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software Technology (UIST 2015)*. ACM, New York, NY, USA, 489–500.

[22] Gene Golovchinsky. 1999. Queries? Link? Is there a difference? (08 1999). DOI: http://dx.doi.org/10.1145/258549.258820

[23] Gene Golovchinsky, Abdigani Diriye, and Tony Dunnigan. 2012. The Future is in the Past: Designing for Exploratory Search. In *Proceedings of the 4th Information Interaction in Context Symposium (IIIX '12)*. Association for Computing Machinery, New York, NY, USA, 52–61. DOI: http://dx.doi.org/10.1145/2362724.2362738

[24] Korinna Grabski and Tobias Scheffer. 2004. Sentence Completion. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04)*. Association for Computing Machinery, New York, NY, USA, 433–439. DOI: http://dx.doi.org/10.1145/1008992.1009066

[25] Enamul Hoque, Vidya Setlur, Melanie Tory, and Isaac Dykeman. 2017. Applying pragmatics principles for interaction with visual analytics. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 309–318.

[26] Philipp Hund, John Dowell, and Karsten Mueller. 2014. Representation of time in digital calendars: An argument for a unified, continuous and multi-granular calendar view. *International Journal of Human-Computer Studies* 72 (01 2014), 1–11. DOI: http://dx.doi.org/10.1016/j.ijhcs.2013.09.005

[27] T. P. Kehler and R. C. Woods. 1980. ATN Grammar Modeling in Applied Linguistics. In *Proceedings of the 18th Annual Meeting on Association for Computational Linguistics (ACL '80)*. Association for Computational Linguistics, USA, 123–126. DOI: http://dx.doi.org/10.3115/981436.981472

[28] Carol C. Kuhlthau. 1991. Inside the Search Process: Information Seeking from the User's Perspective. *Journal of the American Society for Information Science* 42, 5 (1991), 361–371.

[29] Abhinav Kumar, Jillian Aurisano, Barbara Di Eugenio, Andrew Johnson, Alberto Gonzalez, and Jason Leigh. 2016. Towards a Dialogue System that Supports Rich Visualizations of Data. In *17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. 304–309.

[30] Guoliang Li, Shengyue Ji, Chen Li, and Jianhua Feng. 2009. Efficient Type-Ahead Search on Relational Data: A TASTIER Approach. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD '09)*. Association for Computing Machinery, New York, NY, USA, 695–706. DOI: http://dx.doi.org/10.1145/1559845.1559918

[31] Guoliang Li, Shengyue Ji, Chen Li, and Jianhua Feng. 2011. Efficient Fuzzy Full-Text Type-Ahead Search. *The VLDB Journal* 20, 4 (Aug. 2011), 617–640. DOI: http://dx.doi.org/10.1007/s00778-011-0218-x

[32] Guoliang Li, Jiannan Wang, Chen Li, and Jianhua Feng. 2012. Supporting Efficient Top-k Queries in Type-Ahead Search. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*. Association for Computing Machinery, New York, NY, USA, 355–364. DOI:http://dx.doi.org/10.1145/2348283.2348333

[33] Jock Mackinlay, Pat Hanrahan, and Chris Stolte. 2007. Show Me: Automatic Presentation for Visual Analysis. *IEEE transactions on visualization and computer graphics* 13 (11 2007), 1137–44. DOI: http://dx.doi.org/10.1109/TVCG.2007.70594

[34] WILLIAM MANN and Sandra Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text* 8 (01 1988), 243–281. DOI: http://dx.doi.org/10.1515/text.1.1988.8.3.243

[35] Gary Marchionini. 2006. Exploratory Search: From Finding to Understanding. *Commun. ACM* 49, 4 (April 2006), 41–46. DOI: http://dx.doi.org/10.1145/1121949.1121979

[36] Robert C. Miller, Victoria H. Chou, Michael Bernstein, Greg Little, Max Van Kleek, David Karger, and mc schraefel. 2008. Inky: A Sloppy Command Line for the Web with Rich Visual Feedback. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. Association for Computing Machinery, New York, NY, USA, 131–140. DOI:http://dx.doi.org/10.1145/1449715.1449737

[37] Peter Morville and Jeffery Callender. 2010. *Search Patterns: Design for Discovery*. O'Reilly. https://www.safaribooksonline.com/library/view/search-patterns/9781449380205/

[38] Arnab Nandi and HV Jagadish. 2007a. Assisted querying using instant-response interfaces. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. 1156–1158.

[39] Arnab Nandi and HV Jagadish. 2007b. Effective phrase prediction. In *Proceedings of the 33rd international conference on Very large data bases*. 219–230.

[40] Terence Parr, Sam Harwell, and Kathleen Fisher. 2014. Adaptive LL(*) Parsing: The Power of Dynamic Analysis. *OOPSLA 2014* 49 (10 2014), 579–598. DOI: http://dx.doi.org/10.1145/2714064.2660202

[41] Pernilla Qvarfordt, Gene Golovchinsky, Tony Dunnigan, and Elena Agapie. 2013. Looking Ahead: Query Preview in Exploratory Search. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13)*. Association for Computing Machinery, New York, NY, USA, 243–252. DOI: http://dx.doi.org/10.1145/2484028.2484084

[42] Mark Sanderson and C. J. van Rijsbergen. 1991. NRT: News Retrieval Tool. *Electron. Publ. Origin. Dissem. Des.* 4, 4 (Dec. 1991), 205–217.

[43] Vidya Setlur, Sarah E. Battersby, Melanie Tory, Rich Gossweiler, and Angel X. Chang. 2016. Eviza: A Natural Language Interface for Visual Analysis. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST 2016)*. ACM, New York, NY, USA, 365–377.

[44] Vidya Setlur, Melanie Tory, and Alex Djalali. 2019. Inferencing Underspecified Natural Language Utterances in Visual Analysis. In *Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI '19)*. Association for Computing Machinery, New York, NY, USA, 40–51. DOI: http://dx.doi.org/10.1145/3301275.3302270

[45] Milad Shokouhi. 2013. Learning to Personalize Query Auto-Completion. In *Proceedings of the ACM International Conference on Research and Development in Information Retrieval (SIGIR)* (proceedings of the acm international conference on research and development in information retrieval (sigir) ed.). ACM. https://www.microsoft.com/en-us/research/publication/learning-to-personalize-query-auto-completion/

[46] Milad Shokouhi and Kira Radinsky. 2012. Time-Sensitive Query Auto-Completion. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*. Association for Computing Machinery, New York, NY, USA, 601–610. DOI: http://dx.doi.org/10.1145/2348283.2348364

[47] Arjun Srinivasan and John Stasko. 2018. Orko: Facilitating multimodal interaction for visual exploration and analysis of networks. *IEEE transactions on visualization and computer graphics* 24, 1 (2018), 511–521.

[48] Chris Stolte, Diane Tang, and Pat Hanrahan. 2002. Polaris: A System for Query, Analysis, and Visualization of Multidimensional Relational Databases. *IEEE Transactions on Visualization and Computer Graphics* 8, 1 (Jan. 2002), 52–65. DOI: http://dx.doi.org/10.1109/2945.981851

[49] U.S. Geological Survey. 2020. Earthquake Facts and Statistics. https://www.usgs.gov/natural-hazards/earthquake-hazards/earthquakes. (2020).

[50] Robert S. Taylor. 1968. Question-Negotiation and Information Seeking in Libraries. *College & Research Libraries* 29, 3 (1968), 178–194. DOI: http://dx.doi.org/10.5860/crl_29_03_178

[51] Jaime Teevan. 2008. How People Recall, Recognize, and Reuse Search Results. *ACM Trans. Inf. Syst.* 26, 4, Article Article 19 (Oct. 2008), 27 pages. DOI: http://dx.doi.org/10.1145/1402256.1402258

[52] Michael Twidale and David Nichols. 1998. Designing Interfaces to Support Collaboration in Information Retrieval. *Interacting with Computers* 10 (05 1998), 177–193. DOI: http://dx.doi.org/10.1016/S0953-5438(97)00022-2

[53] Bojan Šavrič, Tom Patterson, and Bernhard Jenny. 2019. The Equal Earth map projection. *International Journal of Geographical Information Science* 33, 3 (2019), 454–465. DOI: http://dx.doi.org/10.1080/13658816.2018.1504949

[54] Stewart Whiting and Joemon M. Jose. 2014a. Recent and Robust Query Auto-Completion. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*. Association for Computing Machinery, New York, NY, USA, 971–982. DOI: http://dx.doi.org/10.1145/2566486.2568009

[55] Stewart Whiting and Joemon M. Jose. 2014b. Recent and Robust Query Auto-Completion. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*. Association for Computing Machinery, New York, NY, USA, 971–982. DOI: http://dx.doi.org/10.1145/2566486.2568009

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

[57] Chuan Xiao, Jianbin Qin, Wei Wang, Yoshiharu Ishikawa, Koji Tsuda, and Kunihiko Sadakane. 2013. Efficient Error-Tolerant Query Autocompletion. *Proc. VLDB Endow.* 6, 6 (April 2013), 373–384. DOI: http://dx.doi.org/10.14778/2536336.2536339

[58] Peipei Yi, Byron Choi, Sourav S Bhowmick, and Jianliang Xu. 2017. AutoG: a visual query autocompletion framework for graph databases. *The VLDB Journal* 26, 3 (2017), 347–372.