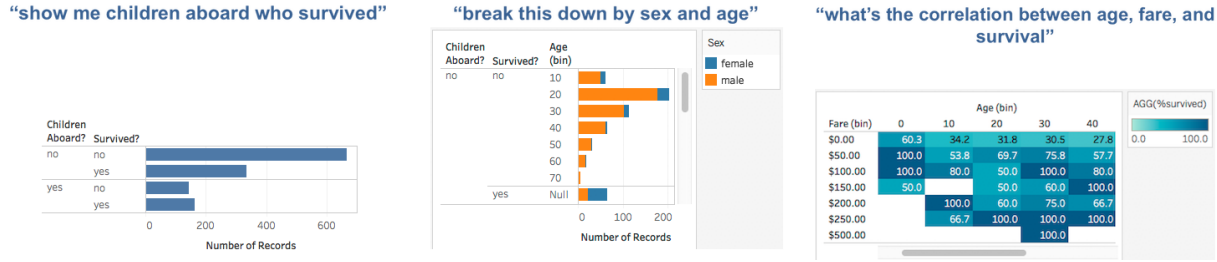
Do What I Mean, Not What I Say! Design Considerations for Supporting Intent and Context in Analytical Conversation

Melanie Tory* Vidya Setlur†
Tableau Software
Palo Alto, California, USA



(a) initial utterance (b) implicit intent to retain context (c) implicit intent asking for a correlation

Figure 1: Example 'analytical conversation' from our study showing how intent could drive visualization responses for a dataset of Titanic passengers. Following an initial utterance (a), an anaphoric reference conveys an implicit intent to retain context (b). Attributes *Children Aboard?* and *Survived?* are retained, while *Sex* and *Age* are added in a way that preserves the previous chart structure. In (c), 'correlation' suggests an implicit intent for a new visualization such as a heat map to depict relationships between attributes *%survived*, *Age*, and *Fare*.

Abstract

Natural language can be a useful modality for creating and interacting with visualizations but users often have unrealistic expectations about the intelligence of natural language systems. The gulf between user expectations and system capabilities may lead to a disappointing user experience. So — if we want to engineer a natural language system, what are the requirements around system intelligence? This work takes a retrospective look at how we answered this question in the design of *Ask Data*, a natural language interaction feature for Tableau. We examine two factors contributing to perceived system intelligence: the system's ability to understand the analytic *intent* behind an input utterance and the ability to interpret an utterance *contextually* (*i.e.* taking into account the current visualization state and recent actions). Our aim was to understand the ways in which a system would need to support these two aspects of intelligence to enable a positive user experience. We first describe a pre-design Wizard of Oz study that offered insight into this question and narrowed the space of designs under consideration. We then reflect on the impact of this study on system development, examining how design implications from the study played out in practice. Our work contributes insights for the design of natural language interaction in visual analytics as well as a reflection on the value of pre-design empirical studies in the development of visual analytic systems.

Index Terms: Human-centered computing—Visualization—Empirical studies in visualization; Human-centered computing—Interaction paradigms—Natural language interfaces

1 Introduction

Natural language (NL) interfaces for visualization [14, 19, 27, 43, 44, 49, 51] can enable effective and engaging interactions with data and may lower the barriers to entry for less-skilled individuals. Designing NL systems for visual analytics is challenging because people often overestimate the system intelligence [27, 43], leading to unrealistic expectations and disappointment when those expectations are not met. This challenge is not limited to visual analytic systems; it is an established trend for all emerging technologies [20, 35, 46].

When engineering such systems, it is nearly impossible to meet the high bar of expectations around their intended behavior, given resource constraints and technology limitations [35, 46, 52]. So then — what are the requirements that an analytical conversation system needs to meet, in order to deliver a delightful user experience? We take a retrospective look at how we answered this question in the design of *Ask Data* [2], a recently developed NL capability for Tableau, shown in Figure 2.

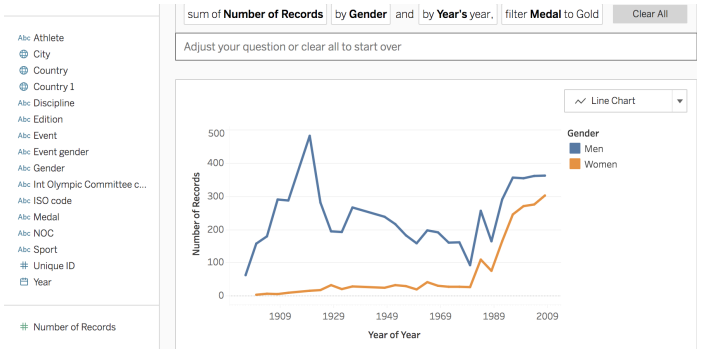


Figure 2: *Ask Data*'s response to "gold medals of each gender over time."

Our previous work on Eviza [43] and Evizeon [27] revealed that people often underspecify details in their NL queries, expecting the system to both fill in the gaps and interpret queries in the context of the current visualization state. Thus, when we began developing *Ask Data*, we knew that two aspects of perceived system intelligence would be its ability to understand analytic *intent* (*i.e.,* what a user is trying to achieve) and its relation to *context* (*i.e.,* the current visual-

*e-mail: mtory@tableau.com
†e-mail: vsetlur@tableau.com

ization state). An open question though, was the extent to which the system would need to understand these aspects of communication to support a positive experience. We were also unclear what visual analytic intents people had in mind and how they would express them. Given the underspecificity of these utterances, what would the system need to infer? A 'smart' system might exhibit forms of computational intelligence to better understand the user's needs and personalize or guide the interaction [42].

Figure 1 illustrates how an analytical conversation system might respond to a series of NL utterances. For a useful transition from (a) to (b), the system needs to infer that utterance (b) represents an intent to retain the existing chart layout but add information. To create a useful visualization in response to utterance (c), the system needs to infer that there is a new line of inquiry and that 'correlation' implies a desire to see the relationship between variables.

Figure 1 represents an ideal interaction that could be difficult to fully realize in a first-generation system, particularly when deployed at scale where it must work with any data source. Could we get away with something simpler and then improve it over time, without severely compromising the experience? At the extreme end, would it be terrible if all the system could do was recognize attributes and values in the input utterance and then use a visual encoding recommender like ShowMe [37] to generate a chart, ignoring all context information and any additional expressions of intent?

To elicit requirements around intent and context for the yet-to-be-built *Ask Data* system, we ran a Wizard of Oz study. The results were deeply influential, ultimately effecting decisions around design principles, system requirements, evaluation criteria, and implementation phasing. We first describe our study and its results. We then reflect on the study's impact on development, examining how design implications from the study played out in practice. Our work contributes insights for the design of NL interaction in visual analytics, a reflection on the value of pre-design empirical studies, and a glimpse into the user experience challenges involved in technology transfer and productization.

2 Related Work

Literature on intent can be classified into: *Intent for search in information retrieval systems* and *Intent for analytical tasks in visual analytics*.

2.1 Intent for search in information retrieval systems

Most research on understanding intent centers on information search, where it is an important aspect of improving precision and recall. Harrison and Dourish recognize the need for determining users' context to better support their activities through appropriate behavior and relevant actions [15]. Broder introduced a taxonomy of search intent with three categories of queries: navigational, informational and transactional [7]. This led to approaches mapping query intent to algorithmically classified search result categories [10, 29, 33, 40, 53]. Hu *et al.* deduced intent from user behavior of URL clicks [28]. Baeza-Yates *et al.*'s approach suggested related queries based on query log data and clustering. Query recommendation and refinement are concepts that help further hone user intent, *i.e.*, transforming an initial query into a more relevant one capable of satisfying the user's information need [4, 32]. Often, information needs evolve. Research has explored interactive intent modeling and reinforcement learning, where search intents are estimated and visualized for interaction [22, 41]. However, the paradigm of intent deduction from search cannot be directly translated to visual analytical workflows, as their goals and the results tend to be different [45].

2.2 Intent for analytical tasks in visual analytics

Numerous systems recommend or automatically create visualizations, based on theoretical foundations such as the data state model [11] and the visualization reference model [8]. Data property based systems (*e.g.,* APT [36] and ShowMe [37]) rely on data characteristics to choose a visual representation. Systems such as Voyager [56] recommend views to reveal data features based on statistical properties, whereas task based systems (*e.g.*, [9, 16]) rely on formal definitions of the user's task. Most recently, Draco [38] combined many of these ideas in a constraint-based programming framework.

Few visualization systems have attempted to *infer* a user's analytical intent. Gotz and Wen [23] used click interactions as implicit signals of intent. Steichen *et al.* [50] and Gingerich *et al.* [21] demonstrated that low-level visualization tasks could be inferred from eye gaze patterns. These results were demonstrated with a small number of pre-defined tasks on known visualizations; a generalization suitable in automated presentation systems does not yet exist. However, the basis for such systems are task models, data-, quality- or interestingness measures.

Recurrent neural networks have been used in conversation-style chatbots [47]. Systems for NL interaction for exploring data [14, 17, 19, 27, 30, 43, 49, 51] depend on understanding user intent and can infer intent since NL utterances may hint at a user's goals. Cook *et al.* guide the user using a mixed initiative approach [12]. However, most systems infer very limited aspects of intent, typically relying on explicitly named data attributes, values, and chart types. Conversational interpretation in Nicky [30] was supported by a domain-specific ontology, which tends not to generalize outside a particular domain. Evizeon [27] and Orko [49] supported follow-on utterances through simplistic models of intent [24], but only for *filters*. Recent research [48] encouraged researchers to study NL utterances to understand user needs. We build on this progress by elucidating ways in which conversational analytics systems may understand and respond to intent.

3 Study Method

Prior to an expensive software development process, we needed to understand the importance of *intent* and *context* in analytical conversations, to define the minimum viable product and plan future improvements. We therefore conducted a Wizard of Oz study, with the goal to gather qualitative data on user expectations related to intent and context behavior. Many other studies followed, but we focus on this one as it was particularly influential. We first document the study and its findings; we then reflect on how the findings impacted development.

We iteratively refined our task, data set, and wizard behavior rules through a pilot study with 10 participants, which also provided wizard practice. We discarded data sets where the data was not easily understood or participants could not consistently achieve meaningful insight within 25 minutes, leading us to the Titanic dataset. We also refined wizard behavior rules to keep users in the flow of analysis as much as possible within each condition (judged qualitatively based on actions users took to correct 'system' behavior as well as users' expressed frustration).

We use the term *utterance* to refer to a participant's typed input to the system (used by the wizard) and *aloud* to refer to a vocalization in conversation with the experimenter (used in our subsequent analysis).

3.1 Conditions

We compared 4 conditions, in which 'system' behavior varied across axes of *context* and *intent* (see Table 1). *B* (baseline) examined whether a simple system that chooses visual encodings based primarily on data attributes (via ShowMe [37]) could be sufficient. The other conditions allowed us to explore the added value of understanding intent and remembering context, plus user expectations surrounding those concepts. A wizard controlled chart creation in

all conditions. We chose a between-subjects design to avoid learning and fatigue effects.

	No Intent	Understand Intent
No Cx	**B (Baseline)** ShowMe default encoding. Add *NumberOfRecords* if no measure given. Fuzzy string match. On explicit request: filters, chart types, calcs, sorting, binning.	**I (Intent)** Understood synonyms & semantics. Custom visual encodings. Automatic binning, calcs, sorting.
Cx	**C (Context)** Prescriptive content retention rules: Retain all on explicit request or anaphoric reference. Reset all on 'reset'. Otherwise: Retain filters, dimensions. Replace measures.	**CI (Context & Intent)** I plus context memory. Wizard judgment of what to retain and when to reset.

Table 1: Study conditions (Cx = context memory). C, I, and CI conditions add functionality beyond the Baseline condition. A more detailed version of this table is available in supplementary material.

Intent: In the no-intent conditions (*B*, *C*), the wizard followed prescribed rules. Initially, we planned *B* to simply use ShowMe plus filters. Pilots demonstrated that this was frustrating, so we added the additional *B* rules in Table 1. The visual encoding was always the ShowMe default except for explicit requests. In intent conditions (*I*, *CI*), the wizard made smarter choices based on their understanding of the user's analytical intent and used their semantic knowledge of the data (*e.g. upper* class = class 1).

Context: In no-context conditions (*B*, *I*), every utterance was treated independently; the wizard cleared the view between visualizations. Context conditions could adapt the existing visualization state (*C* by pre-defined rules, *CI* by wizard judgment). *C* context retention rules were adapted and refined over a series of pilots, as we found it difficult to define prescriptive rules for transitioning the context of attributes without unexpected behavior. Ultimately, unless there was an anaphoric reference or explicit instructions in the input, we retained both dimensions (independent variables) and filters, but replaced numeric measures (dependent variables) when a new one was specified.

Context and intent understanding are not strictly independent. *CI* involved wizard judgment of two types of intent: analytic intent as in *I*, plus context intent of what the user wants to retain from the prior step. Experiencing the wizard role helped us break *intent* into these two components and understand the need for systems to interpret both.

3.2 Participants

We recruited 41 volunteers (18 female, 21 male, 1 male/female pair who walked in together): 26 via an information desk at the Tableau Conference and 15 by email within our organization. All were fluent in English. Participants spanned several industries (retail, education, finance, travel, etc.) and all had analytics experience with spreadsheets and Tableau. They were each randomly assigned to one of 4 conditions.

3.3 Task and Data

We employed an open-ended task with no correct answer. Participants examined the Titanic dataset (1309 records, 10 fields) and attempted to answer the question "Which characteristics made it more likely that a passenger survived?" Participants were instructed to phrase their input naturally, as if they were interacting with a search engine that only knew about the Titanic dataset. They were given a reference page containing data fields and example values. We used only one dataset because the wizard needed to be very familiar with the data. None of our participants reported familiarity with the data set itself, though they were familiar with the Titanic disaster.

3.4 Apparatus and Setting

We used a custom version of Tableau Desktop 10.5 [3] shown in Figure 3 (top). Custom additions were a text input box for the user to type NL input and a red text field for feedback. Input utterances were copied to the feedback field, which could be edited by the wizard (*e.g.* for error messages). Participants interacted with Tableau only in presentation mode, which showed only the visualization, a descriptive caption, filter controls, and legends. The wizard used the full Tableau Desktop interface to produce visualizations.

Three experimenters (including the authors) played the wizard role. All were regular users of Tableau, with a minimum of 1.5 years experience with the tool. Wizards intensively discussed and documented behavior rules during pilots to ensure consistency.

Each session started with a blank screen, mirroring the experience of authoring a visualization from scratch in tools like Tableau Desktop. Subsequent questions were asked with the previous visualization still in view. Filter widgets were always shown for any applied filter. Participants could interact with visualizations through tooltips, filter controls, and by selecting items in a legend to highlight the related subset; these automatic actions did not require wizard intervention.

Figure 3 (bottom) shows the physical setup. Participants interacted with content on a 20" monitor using a mouse and keyboard. A wizard used a MacBook Pro laptop to create visualizations and had a 20" monitor to duplicate the user's screen. Whenever the wizard was manipulating a visualization, the user's screen displayed, "Processing, please wait…". The wizard switched the monitor between the visualization display and the processing message by toggling between extended and mirrored display modes. Due to constraints of the conference setting, the wizard had to be in the same room as the participant; however, the physical setup obscured the participant's view of the wizard's actions. We screen recorded the participant's view plus audio.

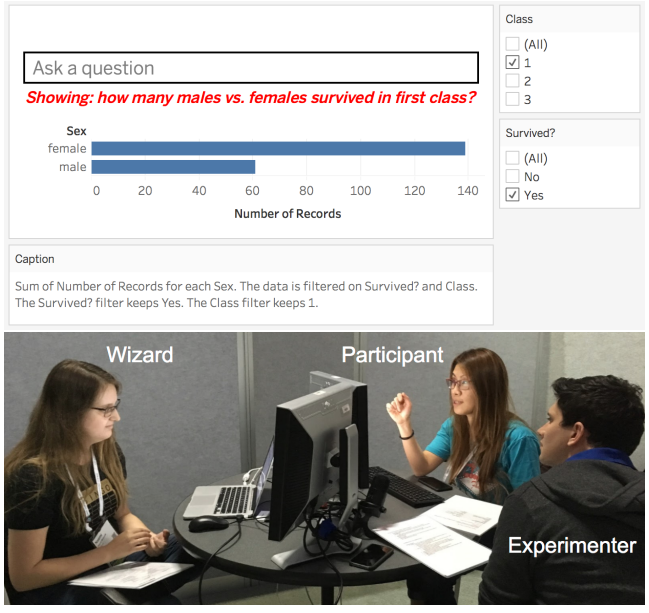


Figure 3: Environment and hardware setup used in the study (bottom) and participant's view of the software (top).

3.5 Procedure

Sessions lasted approximately 25 minutes (2-5 minutes of introduction, 15-20 minutes of actual task, 3-5 minutes of wrap-up) and participants created an average of 9 visualizations. An experimenter led the session and employed a question-asking protocol to elicit qualitative information (*alouds*) from the user for subsequent data analysis. The wizard was introduced as "technical support"; their role was revealed during wrap-up.

Input to the system was typed, not spoken, and wizards were trained to respond only to the text utterances. After typing their input, participants saw the processing screen and the wizard created the visualization (or provided a pre-defined error message if the input did not match 'understandability' criteria). Wizard response time was typically $30-60s$. During this time, the experimenter asked the participant what they expected the system to do. The experimenter subsequently prompted them for feedback on the system behavior. Some participants guessed that the system was being operated by a human, but played along; we did not notice a difference in their behavior compared to participants who did not know. The wrap-up interview asked participants to reflect on unexpected system behavior and possible improvements.

3.6 Post-Study Analysis

We created detailed notes for each session, including exact text utterances, timestamps, visualization screenshots, mouse actions such as filtering and sorting, and participants' alouds (*i.e.*, their comments about expectations and feedback). Subsequent analysis was conducted on this video catalog, which is available as supplemental material.

We conducted a qualitative, multi-pass, open coding analysis based on grounded theory [5]. We focused on user and system behaviors around *intent*, *context*, and their relationships. Input utterance / visualization response pairs were the unit of our analysis. Each pass investigated a new concept and refined the coding of previous passes. To mitigate impact of varying wizard behavior (both intentional variation across conditions and occasional unintentional variation within conditions), we explicitly chose to analyze the user's expectations of the response to their input in relation to the *actual* visualization generated (*i.e.*, rather than the expected visualization based on the wizard rules). Experimental condition was also considered as a contextual factor that was likely to impact user expectations.

To understand user intent in relation to visualizations, we categorized unexpected system behaviors based on participants' alouds and actions they took to prevent or recover from system errors. We also examined visualization design elements (beyond the basic condition *B* behavior) that users reported were helpful. To understand context, we used alouds to categorize user intent around transitions, such as whether the user intended to adjust the visualization, elaborate on it, or start over. We compared this expectation against the visualization response to identify instances where context was unexpectedly lost or retained.

Open coding tags were organized through axial coding. At this stage, we realized that our initial groupings corresponded to steps in the visualization reference model [8] and we identified relationships between categories. The axial coding step resulted in our conversational transitions model (next section). Later structured coding passes were done to gather quantitative information and to completely describe expected vs. actual visualization transitions using our model. We note that our focus was on qualitative understanding of system requirements rather than quantitative comparison of conditions. We collected frequency data for our observations (as a rough indicator of prevalence); however, we did not employ statistical hypothesis testing and would not expect these numbers to be representative of real system use.

4 Study Findings

We first introduce the conversational transitions model that emerged from our analysis and helped us to organize and interpret our findings. We then describe what we learned about how VA systems might handle context information and user expressions of intent.

4.1 Conversational Transitions Model

Our conversational transitions model (Figure 4) describes how to transition a visualization state during an analytical conversation. The model is inspired by conversational centering [24], commonly used for identifying structure in human communication. Conversational centering describes how the *context* of a conversation adjusts over time to maintain coherence, through transitional states that retain, shift, continue, or reset discourse elements (in this case, visualization components).

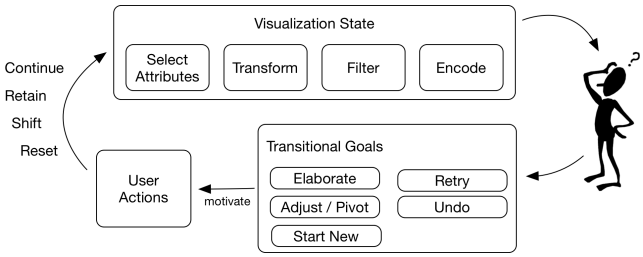


Figure 4: Conversational transitions model.

A key insight during our analysis was that users' *intent* around transitions (how they expected the visualization to change) may apply to any or all aspects of a visualization state, not just filters as in Evizeon [27] and Orko [49]. Applying transitions to filters alone is insufficient for a conversational system that creates *new* visualizations in response to user input, especially one that updates a single visualization at each step. We adopted our state definition from the visualization reference model [8], wherein the *visualization state* is comprised of the data *attributes* in play, *transformations* (*e.g.* calculations to create derived attributes), *filters*, and the *visual encoding* of attributes.

After interpreting a visualization (the thinking human in Figure 4), a user may continue their analytical conversation by formulating a new question. This analytical intent will ultimately drive a user's *transitional goals* (how they wish to transform the existing visualization to answer the new question), which in turn drive *user actions*. We identified the following transitional goals: *elaborate* (add new data to the visualization), *adjust / pivot* (adapt aspects of the visualization), *start new* (create an altogether new visualization), *retry* (re-attempt a previous step that failed), and *undo* (return to the prior state).

This model was derived through the coding analysis. Topical organization of unexpected behaviors revealed categories related to attributes, transformations, filters, and visual encodings. An unexpected *encoding* might show a bar chart when the user expected a crosstab. *Attributes* could be unexpectedly dropped or retained from the prior step or the system could include a different attribute than the user intended in cases of ambiguity. Unexpected *filtering* often occurred when users asked for 'survivors', where they sometimes wanted both *Survived?* = 'yes' and 'no,' but other times wanted only 'yes.' Unexpected behavior around *transformations* included instances where the system showed raw counts instead of an expected survival rate percentage. Organization into these categories also revealed a 1:1 correspondence between system actions and unexpected behaviors: smart system actions prevented unexpected behavior and / or supported error recovery, whereas naïve system actions led to problems. One smart system action (in *I* and *CI*) was to interpret 'survival' as the calculation *%survived*.

Initially, we wondered whether a simple system based on ShowMe [37] could be sufficient for analytical conversation, given a list of attributes extracted from the utterance. Our transitions model made it clear that the answer was no, and helped us articulate *why*. ShowMe automatically creates a visual encoding for selected attributes. However, it does not infer missing attributes or intended transformations, does not address filtering, and does not consider what visual encoding a user might *intend*. Apart from the point case of adding a single attribute to a view, it also does not ensure visual encoding coherence between states. A more intelligent system would infer a user's transitional goals based on their actions and then update the visualization components accordingly. It would also be able to interpret and respond to user intent around each component of a visualization state (*i.e.*, attributes, transformations, filters, and visual encodings).

4.2 Impact of Failing to Understand Intent and Context

What goes wrong when the system fails to correctly understand intent or context? Here we summarize people's reactions to system behavior to examine the impact of these components of intelligence. We focus first on unexpected system responses, as these were often problematic or undesirable. We use the notation [Participant.Condition] to contextualize quotes with the condition the participant experienced.

4.2.1 Unexpected System Behavior

In relation to our model (Figure 4), unexpected system behaviors (Figure 5) mapped to the four visualization state components or an incorrect transition. Most often they related to visual encodings (62 cases) or transformations (44 cases, of which 28 were calculations).

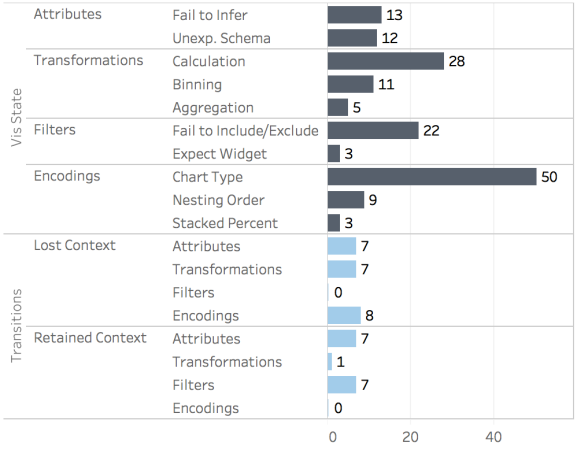


Figure 5: Total unexpected behaviors (gray: vis state, blue: transitions).

For **attributes**, unexpected behaviors involved an *unexpected schema* (*e.g.* misunderstanding the *parents & children aboard?* {yes, no} attribute) or a *failure to infer* an un-named attribute (*e.g.* expecting that the system would also include *NumberOfRecords* even when the user asked for a % calculation). **Transformation** errors included *binning* (failure to bin or unexpected binning), *aggregation* at an unexpected level of detail (or lack thereof), or misunderstood intent around *calculations*. For instance, with the question, "Did younger women survive better than older women?" [P21.I] expected the system to choose a threshold value and group ages above and below the threshold. Instead the wizard showed all 5-year age bins.

Filter problems were typically failures to include or exclude data values. Observing a chart of *%survived*, [P12.C] input, "show this by count instead of percent," expecting to see only a count of survivors. Instead the wizard substituted *NumberOfRecords* for *%survived*, producing a total count. Participants also sometimes expected a filter widget to be available even when they had not asked to filter.

Unexpected **visual encodings** occurred most often when ShowMe chose a poor chart type for the task or when the participant simply expected a different chart type. For instance, while looking at a scatterplot, [P17.CI] requested, "split by class," expecting the system to create small multiples rather than add color encoding.

Unexpected behavior around **transitions** meant that either desired context from the prior state was lost, or undesired context was retained, in any visualization state component (attributes, transformations, filters, or encodings). When [P22.C] asked, "What class were people in?" while looking at a bar chart showing counts of survived vs. not, he expected class to be added as an additional variable rather than replacing the *Survived?* attribute. We also observed poor continuity in visual encodings. Figure 6 shows an example where ShowMe substantially shifted the visual encoding when adding two new attributes.

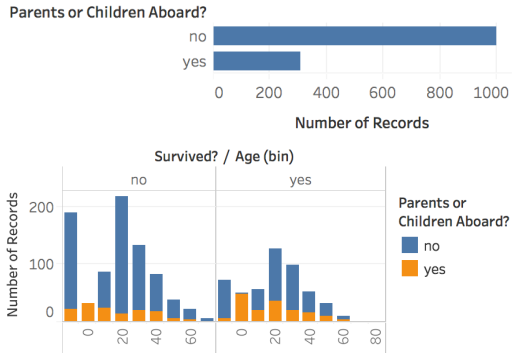


Figure 6: (Top) Response to, "people by parents or children" [P23.C]. (Bottom) Response to follow-up utterance, "count of survival by age (bin) and parents or children." ShowMe substantially changed the view, moving the row attribute to color and changing the bar orientation.

Unexpected visualization states were not always problematic. In 14 instances, unexpected behavior was actually more helpful: 5 times for encodings, 3 times for binning, 3 times for a retained attribute, once for a retained filter, and 2 times for a failure to exclude data values. Figure 1c is a real example from [P21.I], who expected a scatterplot. After examining the chart he remarked several times on its value, *"This is a very interesting chart...This is actually telling the story of what happened. You can see that people who paid the most, and were in the middle age bucket, they survived the most as well...This is the most useful chart that it showed, out of everything."*

4.2.2 Comparison of Conditions

Figure 7 compares total unexpected system behaviors per condition. Unsurprisingly, the 'intelligent' CI condition had the fewest unexpected behaviors, validating the usefulness of both intent and context understanding. Examining the other 3 conditions, though, was helpful to derive minimum system requirements.

B, *I*, and *C* had similar unexpected behavior totals, but the distribution differed. *B* had the most unexpected visual encodings, suggesting that ShowMe's rules were insufficient. *C* (context without intent) had the most transition problems. Even in pilot studies, we found it difficult to prescribe accurate transition rules. *I*'s attribute errors were mostly failures to infer an unnamed attribute (8/13 cases) and its transformation errors were mostly unexpected calculations (13/19 cases). Qualitatively, we observed the most frustration with *C*, as users could not predict what the system would choose to retain from the prior step. In contrast, *B* and *I*'s behaviors were at least *predictable*. Participants learned to repeat and adjust their prior utterances to adapt the view, a strategy that was slightly annoying but

effective. Unexpected visual encodings tended to have less impact on analysis than unexpected attributes, transformations, and filters — it is better to present the correct information non-optimally than to present the wrong information.

		B	I	C	CI
Vis State	Attributes	6	13	3	3
	Transforms	8	19	8	9
	Filters	6	8	6	5
	Encodings	26	10	13	13
Transitions	LostContext	8	4	6	4
	RetainedContext	0	0	13	2
Grand Total		54	54	49	36

Figure 7: Unexpected behaviors by condition.

Additional indicators of system failure are retry, repair, and explicit reset ("start over") actions, summarized in Figure 8. Retries involved rephrasing the prior utterance after it failed to achieve the desired result. Repair actions were explicit corrections (*e.g.*, selecting and excluding an unexpected value, or removing an unexpected attribute as in, "take out fare bin" [P40.C]). Frequent resets suggest that participants lack confidence in the system's ability to transition between states. [P22.C] resorted to this (annoying) strategy when he lost confidence in the system's ability to detect an implicit reset. Most interesting here is the weak performance of *C* (context without intent).

Comparing the conditions revealed two key insights that later informed our system design: (1) Avoid trying to understand context without understanding intent (*i.e.* the poor performance of *C*) and (2) *B* was surprisingly okay since its behavior was predictable.

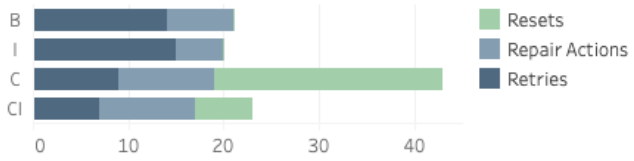


Figure 8: Total reset, retry, and repair actions across all participants.

4.3 Responding to Intent and Context in VA Systems

Conversational analytics systems need to extract user intent from the utterance and then choose how to respond. The supplemental material documents keywords and cues that may help with extracting intent from NL. Here we focus on the latter problem: once we understand intent, how can we produce the most useful visualization?

4.3.1 Supporting Explicit and Implicit Intent

Wizard choices were often better than ShowMe's default because the wizard could understand a user's analytical intent (whereas ShowMe relies only on field types). Here, we explore how a system like ShowMe might be extended, given intent as an additional input.

Prioritize explicit intent requests: Explicit intent requests clearly state user expectations, and therefore should take priority. For example, "color by sex" indicates color encoding, "show only female" indicates a filter, and the utterance in Figure 9(c) specifies a scatter plot.

Visual encoding heuristics for implicit intents: Implicit intents do not directly specify encodings, but visualization best practices can define encoding heuristics. We observed that *actions* and *targets* from Munzner's *why* framework [39] could be identified from utterances and used to model intent. The following examples illustrate how actions and targets can be translated to suitable visualizations by linking them to Few's [18] best practices for data visualization:

- **Numeric analysis**: Figures 9(a) and 10 identify a *distribution* target suited to a histogram. In contrast, a *correlation* target can be revealed in a scatterplot (if the variables are continuous) or highlight table (if discrete), as in Figure 1(c).
- **Categorical data analysis**: For an *overview* of many attributes, as in "Show survival by class, sex, ChildrenAboard? and SpouseAboard?" use the compact heatmap representation. In contrast, Figure 10 implies a *comparison* of target attribute *survived?*: side-by-side views are appropriate for such comparison tasks. A comparator attribute can be redundantly encoded with color if cardinality is low. Alternatively, if the target is an *extreme* as in "Class with the highest survival rate," the target item should be sorted to the top and highlighted.

4.3.2 Transitions: Prioritize Intent Over Encoding Coherence

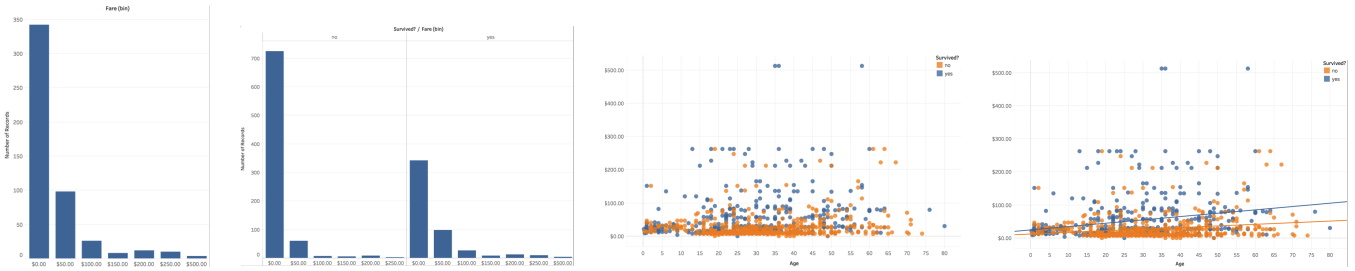
A key insight of our model (Figure 4) was that transition states of continuing, retaining, and shifting need to be applied to *all* visualization state components (attributes, transformations, filtering, and encoding) to maintain conversational coherence. Maintaining coherence in the visual encoding is important, as abrupt changes to the visual representation can be jarring and easily misinterpreted. Figure 11 shows an example where the naïve use of ShowMe in *B* resulted in a misunderstood change. The participant responded in surprise, *"Oh!...Wait a minute, so where...age has disappeared!...Oh shoot, this is not the graph that I wanted...I want my bar chart back with a label."* [P38.B] More coherent transitions could be achieved by adapting the existing visualization state rather than building a new visualization at each step.

However, analytical intent may conflict with the goal to maintain visual encoding coherence. Examining instances from the study convinced us that analytical intent should take priority when it is known. Sometimes it is worth the cognitive cost of interpreting a new encoding to gain a better visualization for one's task. For example, in Figure 1, the second utterance can be handled by simply adding a new column and color encoding to the existing view; however, supporting the *correlation* target in the third utterance requires a substantial encoding change. The poor performance of *C* (context without intent) underlines the importance of this prioritization and the need to accurately infer user intent in a system that supports follow-on utterances.

4.3.3 Anticipate User Needs with Proactive Design

We observed that some wizard design choices were *proactive*, *fail-safe* and supported *flexibility*. These design choices enabled users to easily correct misunderstandings or adapt the visualization to answer more questions. Key fail-safe and proactive design choices were:

- **Display filter controls and interactive legends**: Filter controls enable users to recover data that was incorrectly filtered or restore it later for comparison. Interactive legends similarly enable highlighting and filtering.
- **Show data in context**: Instead of filtering to a named value, show the target value in comparison to alternatives. *E.g.* the answer to "how many people survived?" is more interesting in comparison to the number who did not survive.
- **Visually encode filtered attributes**: Including a filtered attribute as an encoded variable supported follow-up actions. By adjusting the filter control, participants could obtain a useful comparison visualization, as illustrated in Figure 12.
- **Add bonus info**: Anticipate future needs by adding more information than requested. *E.g.* [P4.CI] asked how many children were under age 10, and the system responded with an age histogram showing frequency of all age groups.



(a) "Distribution of survivors by fare?" (b) "split this data by survived status" (c) "scatter plot of survival status by age and fare" (d) "add a trend line"

Figure 9: A snippet of analytical conversation in [P17.CI]'s session. "Distribution" in (a) is an implicit intent for a histogram. "split this" in (b) indicates a transition to small multiples. In (c), the full sentence and new attributes suggest a new line of inquiry and "scatter plot" is an explicit encoding request.

- **Apply transformations**: Binning quantitative variables (binned versions of *age* and *fare* were easier to interpret) or creating useful calculations (*e.g.* percentages).
- **Adjust row/column nesting order**: Changing the default attribute order to create a hierarchy suited to the question. *E.g.* "Compare survival by sex for each class" implies a different nesting order than "Compare survival by class for each sex."
- **Redundant color encoding**: Redundantly encoding an important variable, typically the focus of a comparison. *E.g.* in the nesting examples above, redundantly encode *sex* in the first case and *class* in the second. Figure 10 shows another example.
- **Encoding based on semantics**: Using similar encodings and placement for semantically related attributes enhances interpretability (*e.g.*, *parent/child* and *sibling/spouse*).

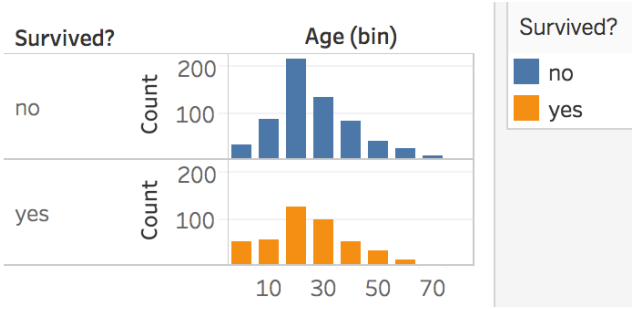
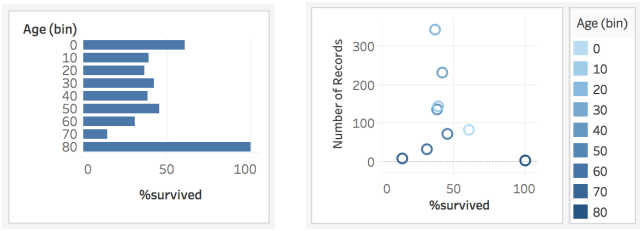


Figure 10: Response to, "What is the age distribution of those who survived and didn't survive?" [P11.I] The *distribution* target and implied *comparison* action suggest two histograms. The target attribute *survived?* is redundantly encoded using position and color.

Anticipating user needs in these ways was nearly always met with praise. Participants expressed excitement and were impressed with how well the system could answer their questions. For example, [P41.I] commented, *"This is better than I expected, because I thought I was just going to get a filter to yes...but I got no as well, so now I have more of the context, which is good."* Similarly, [P34.C] appreciated filter widgets, *"Even though I only asked for males, it has options."*

Proactive behavior is an established concept in intelligent user interfaces, explored in domains such as information systems (*e.g.* [6]), task management (*e.g.* [57]), and mobile interaction (*e.g.* [55]). However, proactivity has been only minimally investigated for analytics and visualization, despite a recent call for more proactive behavior



(a) "percent survived for each age bin" (b) "percent who survived for each age bin, plus the number who survived"

Figure 11: Transition with poor visual coherence, from [P38.B]'s session. Age moves from the y axis to color, which the participant fails to notice.

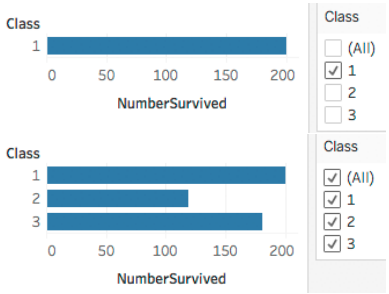


Figure 12: (Top) Response to, "How many passengers in class 1 survived?" [P5.B] (Bottom) Because *Class* is also a row attribute, adjusting the filter control creates a comparative visualization.

in visualization tools [48]. Guo *et al.* explored proactive suggestions for data wrangling, with mixed feedback from users [25] and various visual analytics systems have explored recommendations (*e.g.*, [38, 56]), but none of these systems integrated an explicit understanding of user intent. It is clear from our results that proactive behavior is a worthwhile future direction to explore for analytical conversation.

We are working towards implementing proactive behavior. *Ask Data* includes filter controls and legends, visually encodes filtered attributes in a bar chart, and offers limited support for calculation transformations.

5 Study Takeaways for Ask Data Implementation

This section examines how results of the Wizard of Oz study influenced *Ask Data*, drawing on example utterances from the study as well as subsequent observations of the system in use. We reflect on

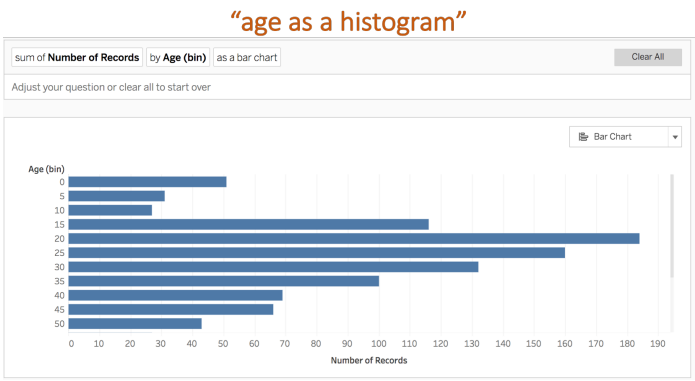


Figure 13: *Ask Data*'s response to "age as a histogram." Because *Age* is a numerical dimension as opposed to a measure, the system infers the binned form of the field and displays a bar chart to provide a reasonable alternative to the user's request.

how our experience developing *Ask Data*, and studies of the system in use, confirmed or contradicted what we learned in our pre-design study.

Participants in the study got into a flow of analysis, employing related utterances in series to investigate a problem. This behavior prompted our most important design principle for *Ask Data*: **no dead ends**. We also observed that when participants were in the flow of analysis, their alouds focused on the data; when their flow got disrupted by undesirable system responses, they focused on system design. This led to the design principle **the interface disappears; it's all about my data**. These design guidelines also served as evaluation criteria, helping us understand what to look for in subsequent usability studies.

Based on these design guidelines, the main technical takeaways for designing the system behavior were:

- Handle *underspecified* utterances and make smart system inferences to keep users in the flow of analysis.
- Maintain *context* to facilitate a conversation with the system based on previous utterances and the current visualization state.

5.1 Inferencing to Handle Underspecificity

Nearly all utterances in the study were underspecified. Visual encodings and data attributes were frequently incomplete, left out, or specified indirectly as an analytical goal (*e.g.* "sales over time" with a time attribute left out). This motivated us to develop heuristics and inferencing logic to provide defaults for missing inputs [44]. *Ask Data* infers data attributes and encodings for partial analytical expressions to help satisfy the intent in input utterances. Our inferencing logic includes inferring a descending sort order when users ask for "products with highest sales" to show the highest value on top. We also support visualization responses requested by users, with sensible inferencing to map an abstract concept such as 'location' to an appropriate geo attribute or inferring a scatterplot when a user types "show me the correlation."

In addition to supporting vague and underspecified analytical intents, *Ask Data* also supports flow by providing suitable alternatives, in case the system does not support the direct request. For example, in Tableau, one can only create a histogram with a measure, and not a dimension. So, if *Age* is a numerical dimension, and a user types "age as a histogram," Tableau would not return a visualization response. However, the underlying intent is probably to view the number of records per numerical quantity, especially if the user has created a binned form (*Age (bin)*). *Ask Data* can infer the binned field and display *Age (bin)* as a a bar chart, seen in Figure 13. Similarly, based on user feedback, we improved analytical intent for the

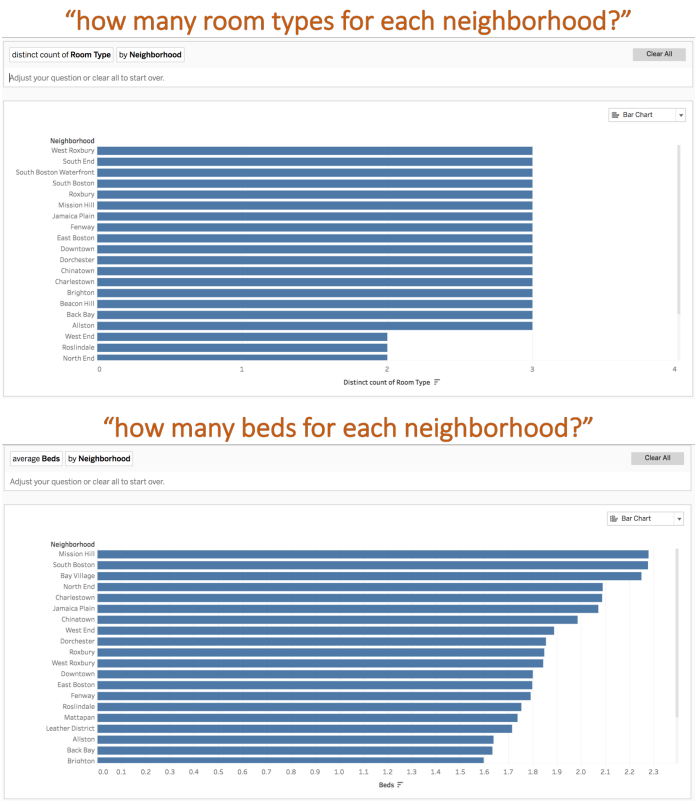


Figure 14: Top: Mapping the intent of "how many" to distinct count for a countable dimension *room types*. Bottom: The default aggregation *average* for a measure *Beds* is inferred for "how many".

phrase "How many" to infer 'distinct count' for countable entities such as dimensions, but the default aggregation such as 'sum' or 'average' for quantitative measures.

5.2 Managing Context in Conversational Flow

Context in a "smart" system is its ability to take into account the information, circumstances and factors surrounding the interaction with a user [54]. Specifically with an NL system such as *Ask Data*, realizing the pragmatic meaning of such an analytical conversation is a matter of matching up the linguistic elements of the utterances with the schematic entities of the context. These entities could be people, places, or objects that are considered relevant to the interaction [13].

A surprising and encouraging finding from the study was that *B*, arguably the least intelligent condition, was not so bad in terms of the user experience. *C* (context) was clearly worse due to its unpredictability. To us, this meant that the implementation of *Ask Data* could be broken into phases — develop a basic system first, iteratively improve its understanding of intent, and then add support for contextual understanding. We adopted an incremental approach to implementing contextual understanding based on the classification of context in ordered degrees of complexity, stemming from linguistic literature [34].

Situational Context: Situational context refers to the environment and information where the interaction occurs [31]. With respect to analytical conversation, that environment is the underlying data source being explored. To provide situational context in the analytical workflow, we surfaced an *explicit* interface, the data pane, that displays information about the attributes in the data source and their data properties. Icons are used to distinguish the various data types (*e.g.,* geo, date time, numeric and text). Hovering over each attribute

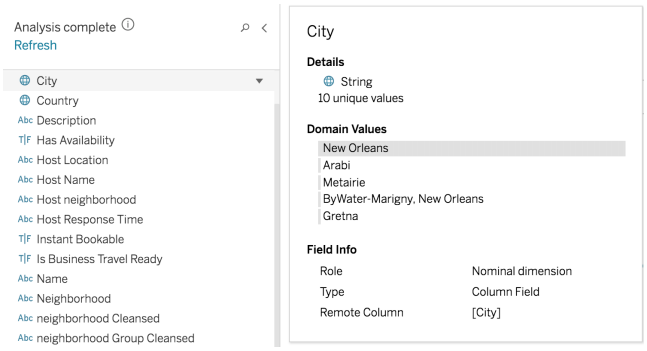


Figure 15: An *explicit* contextual data pane that provides situational context regarding the types of attributes and their properties to help a user type valid analytical expressions in *Ask Data*.

provides additional semantic information such as synonyms and top values in the data domain based on cardinality (refer to Figure 15).

Context through Intentional Interaction: The study also revealed how we could retain the character of conversational interaction (principle of no dead ends), with only limited contextual understanding of follow-on utterances. A key annoyance with the no-context conditions was repeating partial utterances (*e.g.* "survival by sex" followed by "survival by sex and age" to add one attribute). In *Ask Data* we resolved this problem through a refinement user interface (UI). Clicking on any interpreted phrase opens a graphical UI where users can change their query, similar to how study participants used filter widgets to make adjustments. Users could also elaborate by typing a new *scoped query* that would add on to the current interpretation. The top row in Figure 16 shows these various intentional interactions. Using a combination of on-boarding documentation for *Ask Data* and visual treatments to indicate that phrases in the UI textbox were editable, users were encouraged to adopt a mixed initiative approach for repair and refinement.

Linguistic Context: Linguistic context, or cohesion, refers to the relationship amongst tokens in the NL utterances, and how they relate to their predecessors and successors in a discourse [26]. We leverage *Ask Data*'s underlying query language *Arklang* to determine how linguistic context from the previous utterance informs interpretation of a new utterance. ArkLang provides a set of all syntactically valid and semantically meaningful analytical expressions that can be obtained from the semantic model describing the underlying data source, a context-free grammar, along with a fixed set of semantic constraints [44].

Linguistic context is determined by a set of *Add*, *Remove* and *Replace* operations as implemented in Algorithm 1 and shown in Figure 16 (bottom row). For example, if τ = "distinct count of Beds by Neighborhood," and β is the filter expression "Beds at least 2," *Ask Data* will update τ with β, applying a filter to the visualization in context. A *Replace* intent "by Description instead of Neighborhood," will replace the group expression "by Neighborhood" with the group expression "by Description," where α = "by Neighborhood," and β = "by Description." If β is the *Remove* intent for "at least," with α = "Beds at least 2" in play, the update U function will remove the filter on the current visualization.

5.3 Impact on work in progress

Some insights from the study are not yet in the product (at the time of writing) but are influencing work in progress. These include comparison intents (as in Figure 10), additional semantics to inform inferencing, and visual encoding coherence to address jarring changes such as Figure 11.

Algorithm 1: Handling linguistic context in Ask Data

Input: natural language utterance β
Output: VizQL
Let $U = (\tau, \alpha, \beta)$ be the update function that determines the linguistic context to perform *Add*, *Remove* and *Replace* operations to the current contextual set of analytical expressions
τ in the system.
α is an analytical expression that is part of τ.
β is the current utterance in the discourse.
Add determines intent for adding β to τ.
Remove determines intent for removing α from τ.
Replace determines intent for replacing α with β in τ.
1 Perform an $Add(\beta)$ operation if $\beta \notin \tau$.
2 Perform a $Remove(\beta)$ operation if $\beta = \alpha$.
3 Perform a $Replace(\alpha, \beta)$ operation where we apply $Remove(\alpha)$ and $Add(\beta)$.
4 Apply U if $U = (\tau, \alpha, \beta)$ satisfies Arklang constraints.

6 Discussion

6.1 What We Missed

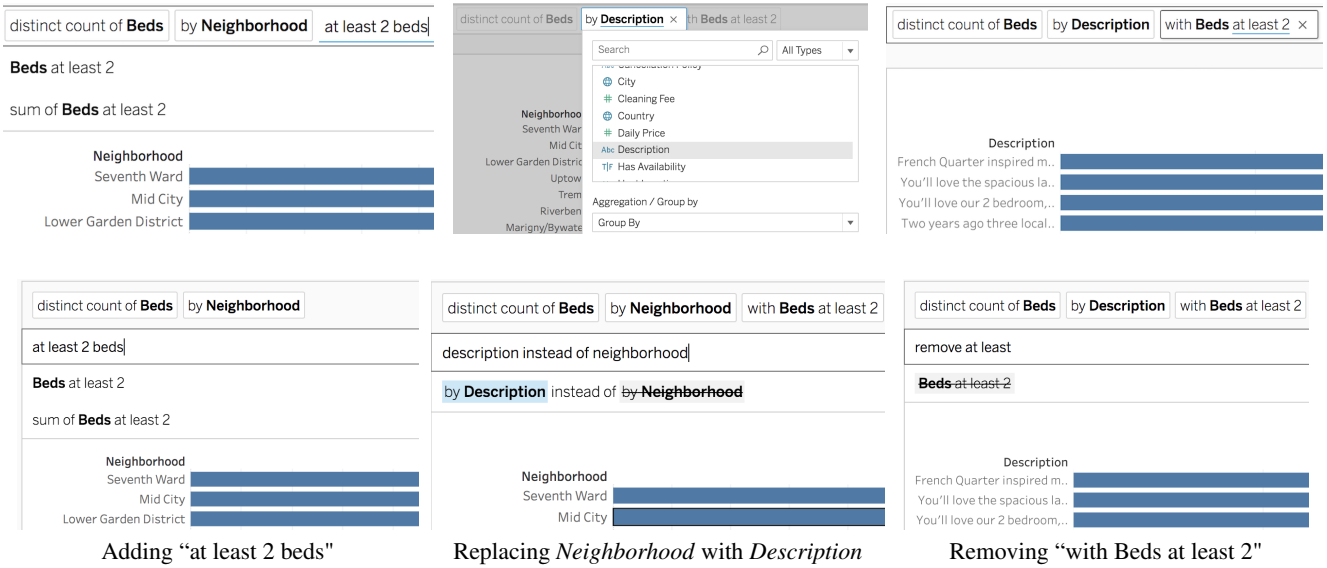
The study presented here naturally missed some insights and identified others that turned out to be less important. One observation was that breaks in analytical flow could be detected via overlap of concepts in subsequent utterances (see supplemental material). This turned out to be irrelevant because we designed the interface to retain prior inputs until a user explicitly removed them. Additionally, while the study identified a need for smart inferencing, details of what to infer (and when) required substantial follow-up. The study also did not offer insight into user learning or skill development.

Because the Titanic data set had only one measure (*NumberOfRecords*), we also saw little diversity in calculations or numerical targets like outliers. Even within the limited context of the Titanic data set (where most transformations were % calculations), the ways in which people expressed intent around calculations were varied and complex. We later repeated the Wizard of Oz approach in a follow-on study specifically focused on understanding calculation intents.

6.2 Limitations and Future Work

Our study design has several limitations that restrict the generalizations we can draw from our results. Most notably, we chose to sacrifice some internal validity by running the study in a noisy conference setting with the wizard in the same room as the participant; we made this choice for external validity since it enabled access to our target user population. However, wizards may have been influenced by overhearing the conversation and participants may have been influenced by the wizard presence. Additionally, wizard judgment played a large role in the 'system responses' and was not strictly controlled, making system responses less machine-like and subject to human bias and interpretation. We mitigated some of these effects by having the wizard follow strict rules in the no intent conditions and training them to respond only to text utterances. We also analyzed what users said they expected in relation to their input, regardless of the actual system response, focusing on what system behavior they *intended* with their utterance.

Absence of autocompletion and delayed response time reduced realism, as expected in this type of study. At the same time, the system delay enabled us to ask people about their expectations, generating rich qualitative data. The study also used only one data source that was somewhat simplistic. Later investigations for *Ask Data* elaborated on this work by exploring more complex data sources and utterances.



Adding "at least 2 beds" | Replacing *Neighborhood* with *Description* | Removing "with Beds at least 2"

Figure 16: A user interacting with the New Orleans AirBnB [1] data source in *Ask Data*. Top row: First version of *Ask Data* supporting intentional interaction to explicitly add, replace and remove utterances as scoped queries. Bottom row: Newer version of *Ask Data* that additionally supports natural language utterances to add, replace and remove interpreted phrases using linguistic context.

Future work might investigate changes in interaction patterns over the longer-term course of an analysis session, intents around learning system capabilities, and handling query over-specification (*e.g.* suggesting reduced constraints when no results are returned). We would also like to explore future work on voice and multi-modal interaction.

6.3 Studying Smart Systems

Studying analytical conversation systems is notoriously difficult. Any structured (*i.e.,* quantitatively measurable) task is nearly impossible to articulate without biasing users' NL input. Such tasks are also poor representatives of real world use. When we evaluated Eviza [43], we noticed substantial behavior differences on structured tasks compared to open-ended analysis. Yet without structured tasks, it is difficult to define concrete indicators of success. Observing that when users are in the flow of analysis, they focus on the data, not the interface, was a key insight. We looked for this throughout future studies of *Ask Data*.

The Wizard of Oz study enabled us to test ideas and make key decisions early. Despite the study's limitations, the results were impactful, reducing uncertainty around requirements and design choices, undoubtedly reducing costly development time. We repeatedly found ourselves referring back to examples from this pre-design study to answer small questions that arose throughout development.

7 Conclusion

We presented a pre-design empirical study that informed design considerations for *Ask Data*, a deployed analytical conversation system. Results of the study gave us a systematic way to think about *intent* and *context* understanding in analytical conversations, suggested approaches to interpret and respond to intent, and revealed how varying levels of system understanding might effect the user experience. Findings influenced our design principles and prompted us to develop inferencing to handle underspecification and strategies to manage user expectations around context. Overall, the study narrowed the space of design options under consideration, reducing uncertainty around timing and feasibility. We hope that others may find value in our insights around the design of intelligent visual analytics systems, the value of pre-design studies, and the challenges of productizing research.

Acknowledgments

We thank Jeff Ericson and Naomi Bancroft for developing the software needed for this study. We also thank Anthony Chen for his assistance running the study and Marti Hearst for her suggestions on the draft.

References

[1] AirBnB. `https://www.airbnb.com`. Accessed: 2019-06-21.

[2] Tableau's Ask Data. `https://www.tableau.com/products/new-features/ask-data`. Accessed: 2019-03-24.

[3] Tableau Desktop, https://www.tableau.com/products/desktop, Mar. 2019.

[4] R. Baeza-Yates, C. Hurtado, and M. Mendoza. *Query Recommendation Using Query Logs in Search Engines*, pp. 588–596. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. doi: 10.1007/978-3-540-30192-9_58

[5] G. Barney. Basics of grounded theory analysis. *Emergence vs Forcing. Sociology press*, 1992.

[6] D. Billsus, D. M. Hilbert, and D. Maynes-Aminzade. Improving proactive information systems. In *Proc. Conf Intelligent User Interfaces*, pp. 159–166. ACM, 2005.

[7] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, Sept. 2002. doi: 10.1145/792550.792552

[8] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[9] S. M. Casner. Task-analytic approach to the automated design of graphic presentations. *ACM Trans. Graphics*, 10(2):111–151, 1991.

[10] H. Chen and S. Dumais. Bringing order to the web: Automatically categorizing search results. In *Proc. SIGCHI Conf. Human Factors in Computing Systems*, CHI '00, pp. 145–152. ACM, New York, NY, USA, 2000. doi: 10.1145/332040.332418

[11] E. H. Chi. A taxonomy of visualization techniques using the data state reference model. In *Proc. IEEE Symp. Information Visualization*, INFOVIS '00, pp. 69–. IEEE Computer Society, Washington, DC, USA, 2000.

[12] K. A. Cook, N. O. Cramer, D. Israel, M. J. Wolverton, J. R. Bruce, E. R. Burtner, and A. Endert. Mixed initiative visual analytics using task-driven recommendations. doi: 10.1109/VAST.2015.7347625

[13] A. K. Dey. Understanding and using context. *Personal Ubiquitous Comput.*, 5(1):4–7, Jan. 2001. doi: 10.1007/s007790170019

[14] K. Dhamdhere, K. S. McCurley, R. Nahmias, M. Sundararajan, and Q. Yan. Analyza: Exploring data with conversation. In *Proc. Conf. Intelligent User Interfaces*, IUI 2017, pp. 493–504, 2017.

[15] P. Dourish. What we talk about when we talk about context. *Personal Ubiquitous Comput.*, 8(1):19–30, Feb. 2004. doi: 10.1007/s00779-003-0253-8

[16] M. Fasciano and G. Lapalme. Intentions in the coordinated generation of graphics and text from tabular data. *Knowledge and Information Systems*, 2(3):310–339, 2000.

[17] E. Fast, B. Chen, J. Mendelsohn, J. Bassen, and M. Bernstein. Iris: A conversational agent for complex tasks. *arXiv preprint arXiv:1707.05015*, 2017.

[18] S. Few. *Show me the numbers: Designing tables and graphs to enlighten*. Analytics Press, 2012.

[19] T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proc. ACM Symp. User Interface Software Technology*, UIST 2015, pp. 489–500. ACM, New York, NY, USA, 2015.

[20] Gartner. 5 trends emerge in the gartner hype cycle for emerging technologies, 2018. `https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging\-technologies-2018/`. Accessed: 2019-03-24.

[21] M. Gingerich and C. Conati. Constructing models of user and task characteristics from eye gaze data for user-adaptive information highlighting. In *AAAI*, pp. 1728–1734, 2015.

[22] D. Glowacka, T. Ruotsalo, K. Konuyshkova, k. Athukorala, S. Kaski, and G. Jacucci. Directing exploratory search: Reinforcement learning from user interactions with keywords. In *Proc. Conf. Intelligent User Interfaces*, IUI '13, pp. 117–128. ACM, New York, NY, USA, 2013. doi: 10.1145/2449396.2449413

[23] D. Gotz and Z. Wen. Behavior-driven visualization recommendation. In *Proc. Conf. Intelligent User Interfaces*, pp. 315–324. ACM, 2009.

[24] B. J. Grosz and C. L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, July 1986.

[25] P. J. Guo, S. Kandel, J. M. Hellerstein, and J. Heer. Proactive wrangling: mixed-initiative end-user programming of data transformation scripts. In *Proc. Symp. User interface software and technology*, pp. 65–74. ACM, 2011.

[26] M. Halliday and R. Hasan. *Cohesion in English*. English Language Series. Taylor & Francis, 2014.

[27] E. Hoque, V. Setlur, M. Tory, and I. Dykeman. Applying pragmatics principles for interaction with visual analytics. *IEEE Trans. visualization and computer graphics*, 24(1):309–318, 2018.

[28] Y. Hu, Y. Qian, H. Li, D. Jiang, J. Pei, and Q. Zheng. Mining query subtopics from search log data. In *Proc. ACM SIGIR Conf. Research and Development in Information Retrieval*, SIGIR '12, pp. 305–314. ACM, New York, NY, USA, 2012. doi: 10.1145/2348283.2348327

[29] B. J. Jansen, D. L. Booth, and A. Spink. Determining the informational, navigational, and transactional intent of web queries. *Inf. Process. Manage.*, 44(3):1251–1266, May 2008. doi: 10.1016/j.ipm.2007.07.015

[30] R. Kincaid and G. Pollock. Nicky: Toward a virtual assistant for test and measurement instrument recommendations. *2017 IEEE 11th Intl. Conf. Semantic Computing (ICSC)*, pp. 196–203, 2017.

[31] L. Koved and B. Scneiderman. Embedded menus: Selecting items in context. *Commun. ACM*, 29(4):312–318, Apr. 1986. doi: 10.1145/5684.5687

[32] R. Kraft and J. Zien. Mining anchor text for query refinement. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pp. 666–674. ACM, New York, NY, USA, 2004. doi: 10.1145/988672.988763

[33] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *Proc. Conf. World Wide Web*, WWW '05, pp. 391–400. ACM, New York, NY, USA, 2005. doi: 10.1145/1060745.1060804

[34] S. Lichao. The role of context in discourse analysis. *Journal of Language Teaching and Research*, 1, 11 2010. doi: 10.4304/jltr.1.6.876-879

[35] G. López, L. Quesada, and L. A. Guerrero. Alexa vs. siri vs. cortana vs. google assistant: a comparison of speech-based natural user interfaces. In *Intl. Conf. Applied Human Factors and Ergonomics*, pp. 241–250. Springer, 2017.

[36] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. Graphics*, 5(2):110–141, 1986.

[37] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Trans. Visualization and Computer Graphics*, 13(6), 2007.

[38] D. Moritz, C. Wang, G. L. Nelson, H. Lin, A. M. Smith, B. Howe, and J. Heer. Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco. *IEEE Trans. Visualization and Computer Graphics*, 2018.

[39] T. Munzner. *Visualization analysis and design*. CRC press, 2014.

[40] F. Radlinski, M. Szummer, and N. Craswell. Inferring query intent from reformulations and clicks. In *Proc. Conf. World Wide Web*, WWW '10, pp. 1171–1172. ACM, New York, NY, USA, 2010. doi: 10.1145/1772690.1772859

[41] T. Ruotsalo, J. Peltonen, M. Eugster, D. Glowacka, K. Konyushkova, K. Athukorala, I. Kosunen, A. Reijonen, P. Myllymäki, G. Jacucci, and S. Kaski. Directing exploratory search with interactive intent modeling. In *Proc. ACM Conf. Information & Knowledge Management*, CIKM '13, pp. 1759–1764. ACM, New York, NY, USA, 2013. doi: 10.1145/2505515.2505644

[42] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd ed., 2009.

[43] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang. Eviza: A natural language interface for visual analysis. In *Proce. ACM Symp. on User Interface Software and Technology*, UIST 2016, pp. 365–377. ACM, New York, NY, USA, 2016.

[44] V. Setlur, M. Tory, and A. Djalali. Inferencing underspecified natural language utterances in visual analysis. In *Proc. Conf. Intelligent User Interfaces*, pp. 40–51. ACM, 2019.

[45] B. Shneiderman, D. Byrd, and W. B. Croft. Sorting out searching: A user-interface framework for text searches. *Communications of the ACM*, 41(4):95–98, Apr. 1998.

[46] N. Smith Auld. A siries of unfortunate events: Users' emotional responses during the first month of siri use. `https://blinkux.com/assets/Blink_Siri_White_Paper.pdf`. Accessed: 2019-03-27.

[47] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 196–205. Association for Computational Linguistics, Denver, Colorado, May–June 2015. doi: 10.3115/v1/N15-1020

[48] A. Srinivasan and J. Stasko. Natural language interfaces for data analysis with visualization: Considering what has and could be asked. In *Proceedings of EuroVis*, vol. 17, pp. 55–59, 2017.

[49] A. Srinivasan and J. Stasko. Orko: Facilitating multimodal interaction for visual exploration and analysis of networks. *IEEE Trans. Visualization and Computer Graphics*, 24(1):511–521, 2018.

[50] B. Steichen, G. Carenini, and C. Conati. User-adaptive information visualization: using eye gaze data to infer visualization tasks and user cognitive abilities. In *Proc. Conf. Intelligent User Interfaces*, pp. 317–328. ACM, 2013.

[51] Y. Sun, J. Leigh, A. Johnson, and S. Lee. Articulate: A semi-automated model for translating natural language queries into meaningful visualizations. In *Intl. Symp. Smart Graphics*, pp. 184–195. Springer, 2010.

[52] J. D. Weisz, M. Jain, N. N. Joshi, J. Johnson, and I. Lange. Bigbluebot: teaching strategies for successful human-agent interactions. In *Proc. Conf. on Intelligent User Interfaces*, 2019.

[53] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine. In *Proc. Conf. World Wide Web*, WWW '01, pp. 162–168. ACM, New York, NY, USA, 2001. doi: 10.1145/371920.371974

[54] H. Widdowson. On the limitations of linguistics applied. *Applied*

Linguistics, 21(1):3–25, 03 2000. doi: 10.1093/applin/21.1.3

[55] W. Woerndl, J. Huebner, R. Bader, and D. Gallego-Vico. A model for proactivity in mobile, context-aware recommender systems. In *Proc. ACM Conf. Recommender systems*, pp. 273–276. ACM, 2011.

[56] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans. Visualization and Computer Graphics*, 22(1):649–658, 2016.

[57] N. Yorke-Smith, S. Saadati, K. L. Myers, and D. N. Morley. The design of a proactive personal agent for task management. *Intl. J. Artificial Intelligence Tools*, 21(01):1250004, 2012.