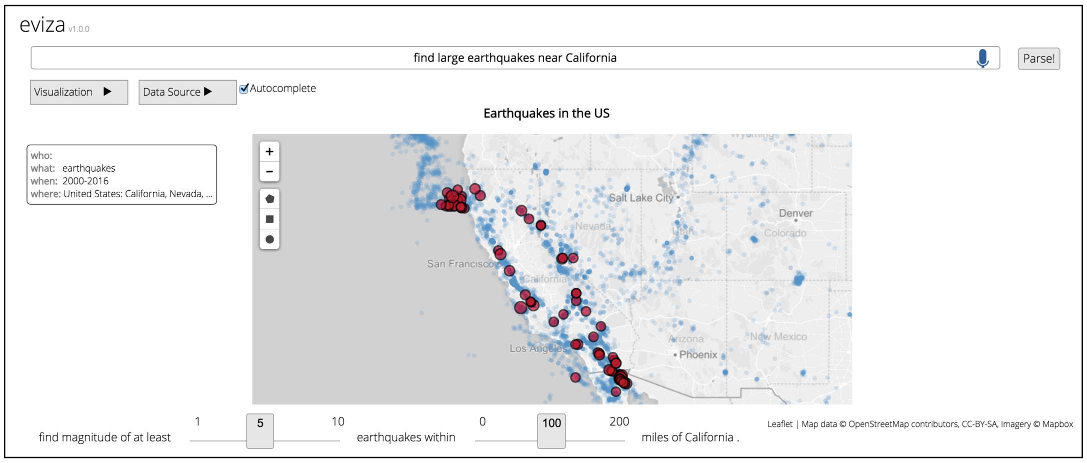
Eviza: A Natural Language Interface for Visual Analysis

Vidya Setlur[1], Sarah E. Battersby[2], Melanie Tory[3], Rich Gossweiler[4], Angel X. Chang[5]

{[1]vsetlur, [2]sbattersby, [3]mtory [4]rgossweiler}@tableau.com, [5]angelx@gmail.com
Tableau Research
Palo Alto, CA 94306


eviza v1.0.0

find large earthquakes near California

Parse!

Visualization ▶ Data Source ▶ ✓ Autocomplete

Earthquakes in the US

who:
what: earthquakes
when: 2000-2016
where: United States: California, Nevada, ...

find magnitude of at least [1 — 5 — 10] earthquakes within [0 — 100 — 200] miles of California .

Leaflet | Map data © OpenStreetMap contributors, CC-BY-SA, Imagery © Mapbox

Figure 1: Eviza's interface showing a map of earthquake data in the United States. The interface supports natural language analytical questions in the context of a visualization. Here, the map shows marks selected in response to the user's query "find large earthquakes near California." The system semantically associates the sized descriptor 'large' to an attribute in the dataset called 'magnitude' that can be associated with size. Eviza finds two ambiguities in the query: 'large' and 'near,' which are fuzzy terms for size and distance. The system sets 'large' to be of magnitude 5 and greater, while 'near' is a 100 mile radius around the border of California. Two ambiguity widgets are added to the interface to allow the user to modify these settings.

ABSTRACT

Natural language interfaces for visualizations have emerged as a promising new way of interacting with data and performing analytics. Many of these systems have fundamental limitations. Most return minimally interactive visualizations in response to queries and often require experts to perform modeling for a set of predicted user queries before the systems are effective. Eviza provides a natural language interface for an interactive query *dialog* with an existing visualization rather than starting from a blank sheet and asking closed-ended questions that return a single text answer or static visualization. The system employs a probabilistic grammar based approach with predefined rules that are dynamically updated based on the data from the visualization, as opposed to computationally intensive deep learning or knowledge based approaches.

The result of an interaction is a change to the view (*e.g.,* filtering, navigation, selection) providing graphical answers and ambiguity widgets to handle ambiguous queries and system defaults. There is also rich domain awareness of time, space, and quantitative reasoning built in, and linking into existing knowledge bases for additional semantics. Eviza also supports pragmatics and exploring multimodal interactions to help enhance the expressiveness of how users can ask questions about their data during the flow of visual analysis.

UIST 2016 October 16-19, 2016, Tokyo, Japan

© 2016 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-4189-9/16/10.

DOI: http://dx.doi.org/10.1145/2984511.2984588

Categories and Subject Descriptors

H.5.m. [**Information Interfaces and Presentation (*e.g.*, HCI)**]: Miscellaneous

Keywords

Natural Language; Visualization; Visual Data Analysis; Parser; Probabilistic Grammar; Ambiguity; Pragmatics

1. INTRODUCTION

There has been a steady growth in systems and interfaces to help users perform data analysis with the aid of visualizations. These systems support a cycle of visual analysis in which a user can fluidly interact with data representations to pose, answer, and refine their questions. Yet, interacting with these analytic tools can be challenging and often requires substantial user practice to become proficient. It has long been known that inexperienced users have difficulty using native database query languages such as SQL to express their data needs [23]. But, even with visual drag-and-drop interfaces, users can still struggle to express their data-oriented questions in terms of tool operations. This is particularly true for novice users [22].

Natural language interfaces have recently received renewed interest for querying and discovering data [35]. Essentially, natural language interfaces take, as input, a question formulated in natural language and return an answer to this question. This approach is promising, as users may be able to express their questions more easily in natural language rather than translating those questions to appropriate system commands (exposed through a query language or interface widgets).

However, it is well-known that robust natural language interfaces are difficult to realize, as they have to handle difficult problems inherent in the task of automatically interpreting natural language. In addition, natural language expressions are often diverse and imprecise, requiring extensive knowledge and sophisticated reasoning for computers to interpret them.

Existing natural language interfaces to data analysis typically return minimally interactive visualizations in response to queries; *i.e.,* the answer needs to be exactly right rather than approximate [4]. In these cases, the user has little to no opportunity to explore the result, gain insight into how their query was interpreted, or revise their query. Most importantly, none of the interfaces are fully integrated with a self-service analysis tool in a manner that allows natural language interactions to become part of a richer visual cycle of analysis.

A rich cycle of visual analysis depends on a user being able to both create new visualizations and interact with those visualizations (*e.g.*, to search, filter, and navigate). While some attention has been devoted to using natural language for the creation aspect (*e.g.*, [35], [21]), much less research has investigated natural language interaction with existing visualizations.

We introduce Eviza, a natural language interface for visual data analysis. Eviza enables a user to have an interactive conversation with their data by emphasizing the ability for a user to continually revise and update their queries in an iterative way to create a 'dialog' with the visualization.

Focusing on interaction with an already existing visualization enables us to explore some new challenges. We examine the types of queries people use to interact with different types of visualizations, how to deal with various types of ambiguity in those queries, and how to ensure natural conversational flow from one query to the next. In addition, by focusing on an existing visualization, we reduce the scope of the natural language interpretation problem, making a useful answer more likely.

2. RELATED WORK

This work builds on a long history in natural language and multimodal input, plus more recent and directly relevant work in natural language interfaces for visualization.

SHRDLU was one of the earliest systems to interface with a spatial environment by combining semantic and syntactic analysis with a body of world knowledge [37]. Another early application for natural language input was to express database queries [11]. These systems depended on hand-crafted semantic grammars tailored to each individual database, which became hard to generalize to other databases [30].

More recently, natural language interfaces for data analysis have emerged as a promising new way of interacting with data and performing analytics. One reason these systems are exploring natural language is to improve usability: while users often know the questions that they want to ask of their data, they can have difficulty understanding how to make a visual analytics system produce the right charts to answer those questions [22]. Yet these individuals can often express their interests and questions quite easily in plain English, as noted in two wizard-of-oz studies ([12], [22]). These studies also revealed that people often specify what they want imprecisely, leaving out details such as dimension names or how to map those dimensions to visual encodings within a chart.

Cox *et al.* [16] demonstrated that multi-modal input is richer than any one modality [27]. Their system dealt with ambiguity by asking a series of questions until the query was fully specified in an unambiguous way. This approach breaks down the flow of analysis since it can take several back and forth interactions before the user can see any view of their data.

A better way to deal with ambiguity is to make a 'best guess' at the user's intent so that a chart can be shown right away. With the Articulate system, Sun *et al.* [35] accomplished this by extracting syntactic and semantic information from a user's query, applying a supervised learning algorithm to translate that into an understanding of their intention, and then generating an appropriate visualization. However, Articulate focused primarily on *generating* a visualization; it enabled very little interaction with the visualization and therefore fell short of supporting cycles of conversation with one's data.

IBM's Watson Analytics features a natural language interface for starting an analysis [3]. Q&A in Microsoft's Power BI allows users to type natural language queries of their data such as "sales per sq ft by store in NC" [4]. ThoughtSpot provides a 'natural language search engine' for data [7]. Narrative Science developed a product to generate natural language summaries of a visualization [5]. Each of these systems is interesting but has fundamental limitations. Most return a minimally interactive visualization in response to queries, meaning the question must be of the type where there is a single exact answer, rather than one where there are a range of relevant and useful ways to answer. Many also require experts to perform modeling before the systems are effective [18, 14].

Most similar to our work is DataTone [21]. Like previous tools, the emphasis in DataTone was on producing a visualization based on a user's typed or spoken query. However, the system improved the analysis flow by making a best guess at the user's intent, producing a chart according to that best guess, and then providing ambiguity widgets through which the user could change their chart if the system's guess was incorrect.

Our work extends DataTone in several substantial ways. First and foremost, our research focuses on techniques for enhancing expressibility of the natural language queries. Further, we emphasize the analytic flow of a user exploring a visualization: instead of focusing on generating a new visualization, our research primarily explores interaction with an *existing* visualization. With this more focused context, we can augment the interface with semantics specific to the visualization in play, allowing for new forms of query expressiveness. We also extend ambiguity widgets to handle ambiguity in quantitative magnitudes as well as time and space (*e.g.*, the query "near Paris" generates a slider for 'number of miles from Paris').

3. CONTRIBUTIONS

The main research goal of this paper is to design and implement a natural language interface that has rich semantics and expressibility and can support the analytical flow of exploring an *existing* information visualization. In particular, our work has the following technical contributions:

- A probabilistic grammar based approach that has predefined syntactic rules that are dynamically updated based on the semantics of the data from the visualization. We believe that this is a more effective approach than deep learning or knowledge based approaches for supporting a range of analytical questions in the given context.
- A template-based autocomplete to offer interpretable suggestions matching the visualization and dataset.
- Extension of ambiguity widgets [21] to quantitative and spatiotemporal ambiguity.
- Built-in rich domain awareness of time, geographic, and quantitative reasoning as well as linking into existing knowledge bases like WolframAlpha [9] for greater query expressibility.
- Supported language pragmatics through a finite state machine that enables a user to have more of a conversation with their data rather than a command-like experience. Users can simply say "How about Texas?" as a follow up to the question "What large customers do I have near California?"
- Multi-modal interactions (*e.g.*, radial selection in a map; asking "what's the distribution of earthquakes *here*") and user context ("show the average house price near me").
- Supported queries that are grounded in empirical data gathered in a user study and validated in an evaluation study.

4. INITIAL STUDY

Type	Examples
Search	Where is Texas?
Filter	Show Africa only
Extremum & Top N	Show me the country with the largest outbreaks Top 3 countries
Navigate	Zoom in to Italy
Sort	Sort by average unemployment
Formatting	Show y-axis label
Change chart type	Can I see this on a map
Grouping	Combine Sat and Sun
Compare	Patterns of New York versus California
Calcs & Stats	What's the nationwide total? Are there any outliers?
Reference lines	What is the best trend line for this plot?
Analytics and trends	Is there a seasonal trend for bike usage?

Table 1: Most common query types observed in the initial study

To guide development of the Eviza prototype, we ran an initial web-based study to understand how people would interact with various types of visualizations using natural language. Our aim was to gather a repository of queries that people would naturally ask, given different types of visualizations. We used this repository to inform the design of our prototype, including the types of functionality it should offer, the keywords to look for, and the grammatical structures we should anticipate.

4.1 Method

Each participant examined five visualizations (randomly selected from a set of 20). For each visualization, they were asked to provide three to five statements or questions that they would use to interact with the view or ask questions of the data. Our study was deliberately open-ended in order to gather a wide variety of query types and mitigate bias against specific types of queries. The visualizations were also deliberately varied; they included maps, bar charts, scatter plots, time series line plots, bubble charts, treemaps, heatmaps, and dashboards.

Our study was run in two parts: text and verbal. Participants were volunteers recruited across all departments of a software company. Seventy five people participated in the text version and 14 in the verbal version. Text participants were sent a web link and completed the study on their own; the system recorded the queries that they typed in. Verbal participants met an experimenter in person and their sessions were audio-taped. In both cases, the session took approximately 10-15 minutes.

4.2 Results

We collected 1235 text queries and 317 verbal queries. Verbal queries were longer on average (12.0 words versus 8.1 words for text queries) but there were no noticeable differences in the types of queries across modality. We categorized the queries through a manual open coding process. The most common types we observed are described in Table 1.

Queries identified in the study directly informed the design and conception of Eviza. They helped us to identify the functionality we needed to support, the keywords and phrasing to expect, and the types of ambiguity that we had to manage.

5. ARCHITECTURE

Figure 2 illustrates Eviza's system architecture. The Eviza system employs a traditional web-based, client-server model. This architecture allows us to support multiple contexts and device types (*e.g.*, mobile, desktop). We deployed a custom node.js® server [6] with an express module to process the request types. The client is a standard web client (HTML5, CSS3, JavaScript) that uses an HTML5 API for voice input.

The user's natural language query input is processed by the autocompletion module, informed by the grammar module containing both predefined rules for processing the query as well as rules dynamically added based on the data attributes in the visualization. The pragmatics module takes all possible parse tree paths from the grammar module, and computes the highest probable path in the finite state machine. The ambiguity module computes both syntactic and semantic ambiguity of the individual tokens in the parsed query. The analytics module then invokes the appropriate analytic functions.

Each analytical function requests an update to the visualization through the data manager. The data manager reads in the requested data from the data files and the corresponding data classes of the visualization. The event manager handles consistency across the various presentation elements in the visualization when there is a change in state of the visualization upon execution of the query. These updates are then rendered by the D3.js library [13].

5.1 Interface

Figure 1 shows the Eviza interface. An input query box accepts both keyboard and voice input and a check box enables or disables autocompletion. Drop-down menus enable a user to choose a visualization type (map, line chart, bar chart or scatter plot) and a data source. An information pane containing 'who', 'what', 'when', and 'where' attributes is populated with corresponding attributes from the visualization. Here, the user has entered a query "find large earthquakes in California." Upon execution of the query, two interactive ambiguity widgets appear below the visualization, to

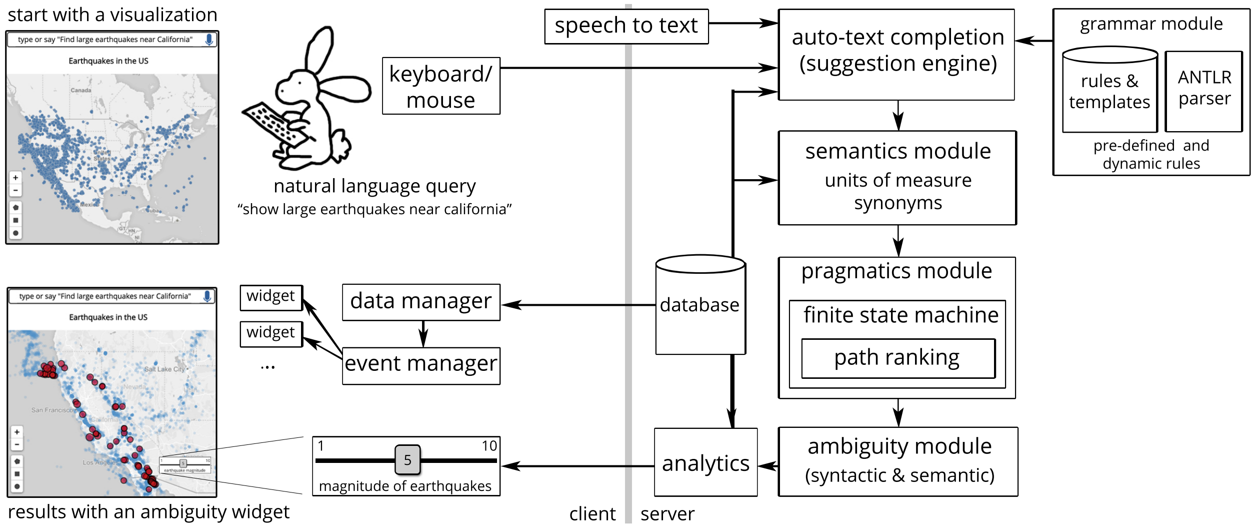


Figure 2: System architecture

indicate the ambiguity for what 'large' means and for 'near', a fuzzy distance term.

6. GRAMMAR

The natural language interface to Eviza is specified by a probabilistic grammar that can be applied algorithmically to provide a structural description of the input query. In particular, we use an ANTLR parser that employs a top-down parsing strategy called $LL(*)$ [28]. Such a parser parses the input from left to right, performing the leftmost derivation of the input query. It can also perform lookahead while parsing for a certain grammar without having to backtrack, allowing for greater flexibility in specifying the grammar rules.

The input to the parser is an extended Backus Normal Form (EBNF) context-free grammar with production rules augmented with both syntactic and semantic predicates based on the current visualization [1]. These rules are associated with embedded actions that trigger an appropriate update in the visualization. Syntactic predicates are statically included in the grammar based on general commands that Eviza supports. The semantic predicates are dynamically added to the production rules based on the context of the visualization, and semantics obtained from third-party corpora.

The syntactic grammar G is a set of recursive rewriting rules (or productions) used to generate patterns of strings defined as a 4-tuple $G = (V_n, V_t, \phi, S)$ where:

- V_t is the set of terminal symbols or constants that appear in the string (*e.g.*, 'June' or a numerical value such as \$53.87).
- V_n is the set of non-terminal symbols which are placeholders for patterns of terminal symbols that can be generated by the nonterminal symbols (*e.g.*, 'month' or 'price').
- ϕ is the set of production rules used to generate patterns of strings, consisting of elements from V_t and V_n.
- S is the starting symbol, a special nonterminal symbol that appears in the initial string generated by the grammar.

Semantic predicates can be represented as a series of EBNF grammars $G^i = (V_n^i, V_t, \phi, S), i = 0..n$ and are dynamically added to the original grammar G as they share the same terminal (V_t) and start (S) symbols. V_n and ϕ evolve by including new elements, and we describe the extension of the grammar as $G^{k+1} = G^k + \Delta G^k$, where $\Delta G = (\Delta V_n, \Delta\phi)$.

Consider a sample production rule in the grammar G^0 for the time-series visualization of stock prices over time:

$$G^0 \longrightarrow \text{'highest <}\textit{value}\text{> in <}\textit{month}\text{>'}$$

Eviza detects that the visualization contains a numerical data attribute called 'price' through its data manager. 'Price' is mapped to the non-terminal symbol 'value' in the grammar since 'value' only accepts numerical terminal symbols. The system then identifies a semantic attribute 'currency' for 'price', and generates a corresponding semantic predicate 'highest **price in <currency>** in <month>.' Note that there could be multiple numeric attributes for 'value', and semantic predicates are computed only if corresponding semantic attributes exist.

$$\Delta G^0 \longrightarrow \text{'in <}\textit{currency}\text{>'}$$

Here <*value*>, <*month*>, and <*currency*> are non-terminal symbols. The updated grammar G^1 is:

$$G^1 \longrightarrow \text{'highest <}\textit{value}\text{> in <}\textit{currency}\text{> in <}\textit{month}\text{>'}$$

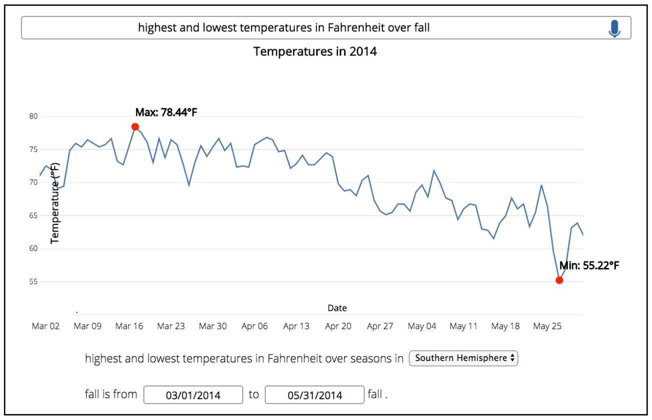
$$V_n^1 \longrightarrow \text{value} = \text{<}\textit{price}\text{>}$$

The next section describes how these semantic values are obtained to extend the grammar.

7. SEMANTICS

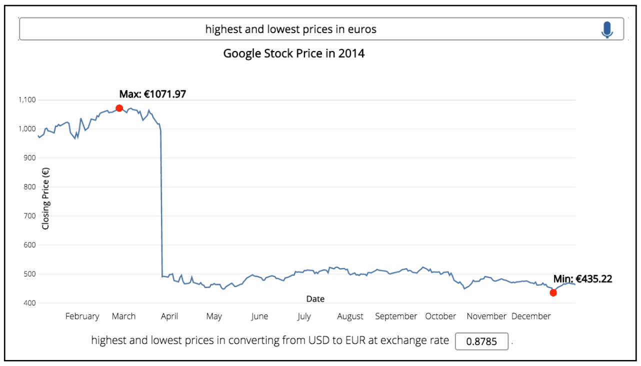
A comprehensive representation of natural language semantics requires access to vast amounts of common sense and domain-specific world knowledge. The process of building such a knowledge base often requires large numbers of corpora and ontologies [26]. We rely on the fact that Eviza has a more *focused* context of the given visualization, and retrieves specific semantic links such as concepts and synonyms for the data attributes that constitute the visualization.

As a user analyzes real-life data, the enrichment of the data's meaning with additional semantics and units helps with the expressiveness of natural language interaction. We augment various semantic types in the underlying grammar such as time, space and entities with existing knowledge bases including WolframAlpha's


highest and lowest temperatures in Fahrenheit over fall

highest and lowest temperatures in Fahrenheit over seasons in Southern Hemisphere

fall is from 03/01/2014 to 05/31/2014 fall .


highest and lowest prices in euros

highest and lowest prices in converting from USD to EUR at exchange rate 0.8785 .

Figure 3: Eviza leverages existing knowledge bases for richer semantics including units. Top: An example query containing seasons and temperature units. Bottom: An example query involving currency.

unit taxonomy [9] and Wordnet's synsets [29]. Semantic enrichment includes basic unit types such as temperature (*e.g.*, Celsius, Fahrenheit), currencies, and quantitative units (*e.g.*, pound, foot, meter, quart). Figure 3 shows example queries involving units. These units are prefetched and added to the semantic predicates of the grammar.

Additional domain-specific semantic enrichment occurs when the visualization is loaded. For example, mapping 'magnitude' in the map of earthquakes (Figure 1) to the Richter scale unit in Wolfram's taxonomy supports richer queries such as "find *large* earthquakes in California." Wolfram considers 5 on the Richter scale to be 'moderate.' Eviza considers an ambiguous 'large' to be 'moderate' in terms of severity. We have ambiguity widgets that come into play if the system over-estimates or under-estimates the severity attached to 'large.'

8. AUTOCOMPLETION

A main challenge of NLP interfaces is in communicating to the user what inputs are supported by the system. Open-ended text boxes provide no information on the types of queries that are supported. This is an important issue to address as any practical system will only support a finite set of operations from the set of all expressible statements.

Autocompletion is a widely used mechanism to get to a desired piece of information quickly and with as little knowledge and effort as possible. As the user types, a list of appropriate completions is returned to the interface. This feature is prevalent in program editors such as Visual Studio, command shells such as the Unix Shell, search engines and desktop search. Autocompletion is also gaining popularity in mobile devices, particularly around exposing the underlying semantic and linguistic model [25].

Similarly, to help users become familiar with the terminology and grammar of *Eviza*, we introduce an autocomplete component. This component provides immediate feedback as the user types, indicating what expressions and values are possible. (Note that we currently do not support autocompletion for voice.) Unlike DataTone, which waits until the user is finished, disambiguation and choices are presented in real time.

Autocomplete was designed to expose the grammar to users in an interpretable manner. The main design choices revolve around selecting which attributes and values to expose from the underlying data. There is substantial prior work in leveraging user history and input statistics to drive suggestions. However, we employ a grammar-based approach to bootstrap when no user history is available. This choice enables organic discovery of features and operations for novice users.

We use templates along with the ANTLR error mechanism to generate candidate options and expose the grammar. The parsing grammar is designed to parse a broad variety of input expressions, including potentially syntactically invalid phrases. To present the user with a clean set of interpretable choices we create a separate set of templates, corresponding to grammar rules, with phrases we want to show. Each template provides easy to interpret patterns for the user to match against, and consists of slots that can be expanded into other templates or lexical choices (see Figure 4 template completions).

Given partial input by the user, we parse it and align to the available set of templates. Valid templates and rules are updated based on the current visualization and dataset. The parser and autocomplete suggestions share a common lexicon that is dynamically updated during this process. As the user types, potential values are suggested by the system as drop-down options. The user can select a displayed option to complete the input text. For template options, only the first part of the template is filled in, and the user can decide to stick with the template or switch to a different one.

A key design choice is that autocomplete only shows up if there is a parsing error (*i.e.*, the query is incomplete). Therefore, suggestions do not intrusively appear when the user has already formulated a valid query. Suggestions are elaborated in real-time as the user types (see Figure 4). At any time the user can request suggestions by pressing the down arrow.

Suggestions are filtered by prefix matching, with the number of options limited to 10. Then they are sorted by weight, length, and finally alphabetically. Attribute values are read directly from the data to generate candidate suggestions (*e.g.*, countries/regions from a list). For number ranges, the min and max values as computed from the data are provided so that users can get a sense of the range of plausible values.

9. AMBIGUITY

Natural language queries can be ambiguous due to syntactic and semantic variations between the user's mental model and the system's model. We observed many ambiguous queries in our initial study and designed the ambiguity module based on these examples. We employ entropy, a concept commonly used in information retrieval, to measure the uncertainty of a given query [15].

Eviza handles two forms of ambiguity - syntactic and semantic. Inspired by the DataTone system [21], we expose the ambiguity and enable the user to correct default choices through simple GUI widgets in the interface.

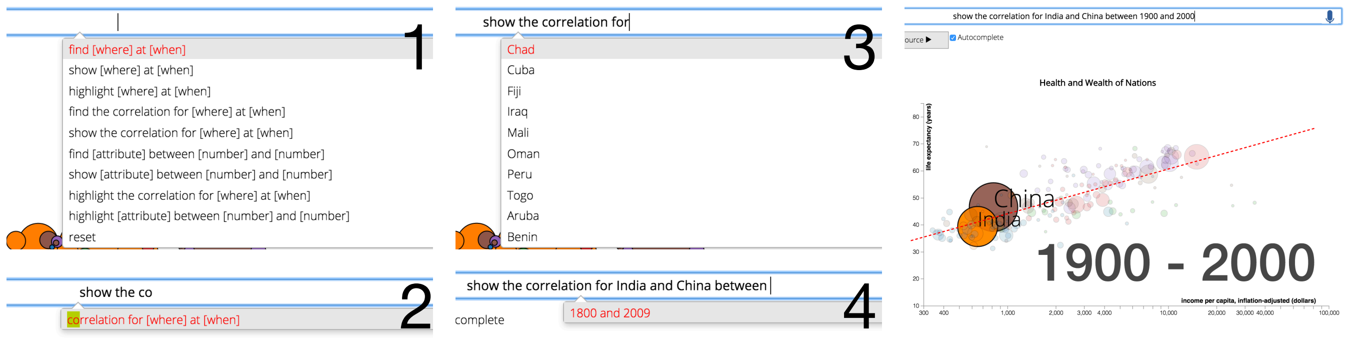

1
find [where] at [when]
show [where] at [when]
highlight [where] at [when]
find the correlation for [where] at [when]
show the correlation for [where] at [when]
find [attribute] between [number] and [number]
show [attribute] between [number] and [number]
highlight the correlation for [where] at [when]
highlight [attribute] between [number] and [number]
reset
2
show the co
correlation for [where] at [when]
3
show the correlation for
Chad
Cuba
Fiji
Iraq
Mali
Oman
Peru
Togo
Aruba
Benin
4
show the correlation for India and China between
complete
1800 and 2009
show the correlation for India and China between 1900 and 2000
ource ▶
Autocomplete
Health and Wealth of Nations
life expectancy (years)
China
India
1900 - 2000
income per capita, inflation-adjusted (dollars)

Figure 4: Autocomplete functionality. User presses the down arrow to get a list of suggestions (1). After the user starts typing, the selection set is reduced and adjusted (2). In steps (3) and (4) dataset-specific values are suggested for template slots. The desired visualization is generated on the right.

9.1 Syntactic Ambiguity

Syntactic ambiguity concerns the uncertainty in the syntactic alignment between input query tokens and data attributes of the visualization due to spelling and plurality variations. Syntactic entropy provides an estimate of the matching complexity based on the set of potential syntactic alignments.

Syntactic entropy, the complement of a cosine similarity metric *sim* between query token q and data attribute a [34] is defined as: $1 - sim(\vec{q}, \vec{a})$.

In practice, we have found that an entropy ≤ 0.3 works well. The top data attribute matches are shown as an ambiguity widget to the user in decreasing order of relevancy. In Figure 5, the top figure shows an ambiguity widget added to resolve the syntactic ambiguity between the input query term 'happy' and the top matched data attribute terms 'HappyLifeYears' and 'HappyPlanetIndex.'

9.2 Semantic Ambiguity

Semantic ambiguity concerns differences in semantic alignment between the input query and data attributes in the visualization. Semantic entropy quantifies an estimate on the amount of synonomy and hyponymy (vagueness) [29].

Semantic entropy is defined as a complement of a semantic distance between the corresponding word senses of q and a defined as $1 - wup(s_1, s_2)]$, where, s_1 and s_2 are synsets of q and a respectively. $wup(s_1, s_2)$ is the Wu-Palmer similarity function [38] that returns a score denoting how similar two word senses are, based on the depth of the two senses in the Wordnet taxonomy and that of their most specific ancestor node called Least Common Subsumer (LCS), with scores returned in the range $\in [0, 1]$.

While several similarity functions exist in the natural language literature, we chose the Wu-Palmer function as it is simple and has good performance. We select the related terms with the top-scoring relatedness to the categorical terms. In practice, we have determined that a threshold score of 0.85 and above tends to lead to an optimal set of semantically related terms.

For instance, consider the query "large earthquakes". We need to identify the field that would tell us whether or not the earthquake is 'large,' and what constitutes a 'large' earthquake. Given the semantic enrichment of units as described in a previous section, the semantic entropy indicates that 'large' and 'magnitude' (an attribute name in the dataset) are close word senses. This ambiguity is expressed as a slider widget set to a minimum magnitude of 5 (a 'moderate' earthquake on the Richter scale in Wolfram's taxonomy) as shown in Figure 1.

We also handle semantic ambiguity in both temporal and spatial domains. For example, the bottom image in Figure 5 shows an ambiguity widget to resolve the semantic ambiguity between the input query term 'near' and the underlying analytical function for computing temporal distance. The widget shows an initial setting of one month before and after 'August' that can be modified by the user if she desires.

Expressing spatial queries in natural language introduces numerous location- and distance-based ambiguities that must be translated into an unambiguous form that can be processed. Geographic locations may be referred to using colloquial or local names, by context-specific reference (*e.g.*, 'in the city' versus the surrounding suburbs), or typed in using abbreviations (*e.g.*, 'NYC' for New York City) which do not match the formal names.

Once locations are disambiguated, there is still the issue that people often think in terms of vague spatial concepts rather than absolute distance [24]. It is more natural to use fuzzy spatial terms such as near, far, and around; *e.g.*, "restaurants near me" as opposed to "restaurants within 1.5 miles" [2]. While fuzzy spatial prepositions are common, spatial queries require metric input. The ambiguity module translates the query to something more concrete, while still allowing the user to adjust to match their model for the term used. Figure 1 shows the query token 'near' mapped to a spatial distance function set to a 100 mile radius around the border of California.

10. PRAGMATICS

One Eviza's goals is to support analytical flow by enabling the user to have a conversation with the visualization. Conversation frequently consists of a series of related utterances, often referring to past references and context [10]. We observed this in our initial query study, where many queries were direct follow-ons to previous ones. As part of this conversational flow, the semantics of a particular query can be influenced by each other via transition probabilities. This approach, where ambiguity in a particular query is disambiguated with respect to context, is called *pragmatics* and helps with understanding the user's intent in a flow of queries [32].

The pragmatics algorithm implements a finite state machine (FSM) [33]. The FSM $W = (\Sigma, Q, E, i, F, \lambda, \rho)$ is defined by a set of query tokens Σ, a finite set of states Q, a finite set of transitions E, an initial state $i \in Q$, a set of final states $F \subseteq Q$, an initial weight λ and a final weight function ρ.

A transition $t \in E$ can be represented as an arc from the source or previous state to the destination or next state $n(t)$, informed by the

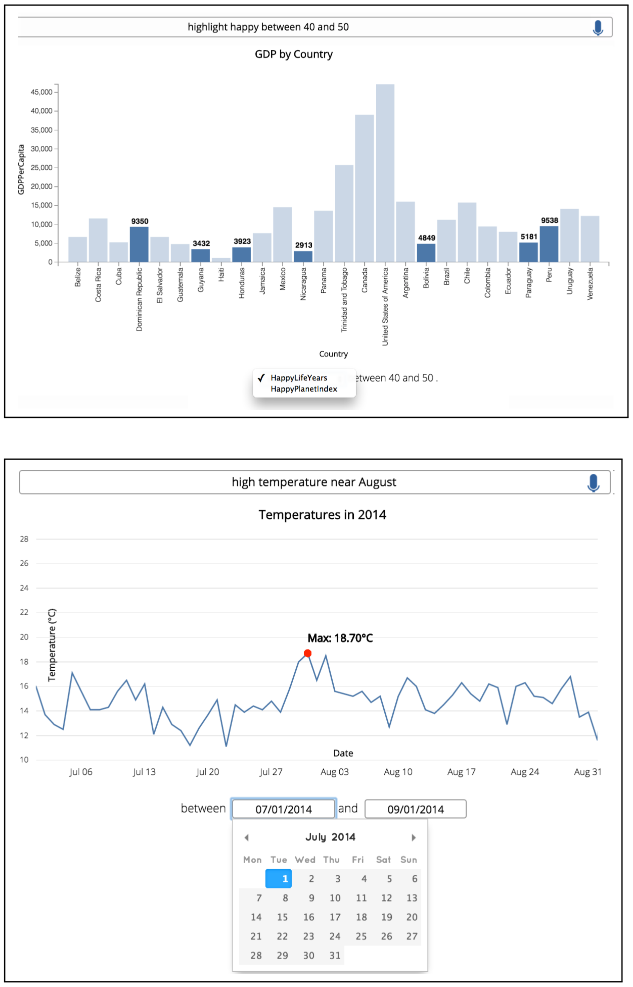


Figure 5: Ambiguity widgets

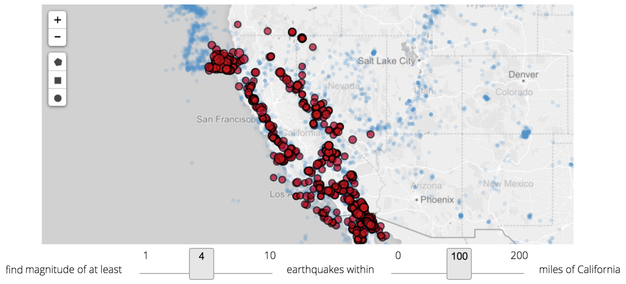
probable parse paths from the grammar. The transition weight $w(t)$ represents its probability p. A path in W is a sequence of consecutive transitions $t_1 \ldots t_n$ with $n(t_i) = p(t_{i+1})$, where $i = 1..(n-1)$.

The algorithm chooses a vector path π with the highest final tensor product weight. The weight associated with π from the initial state i to a final state $f \in F$, is the *otimes*-product of the initial weight, the weight of its consecutive transitions and the final weight $\rho(n(t_n))$ reached by the path, given by:

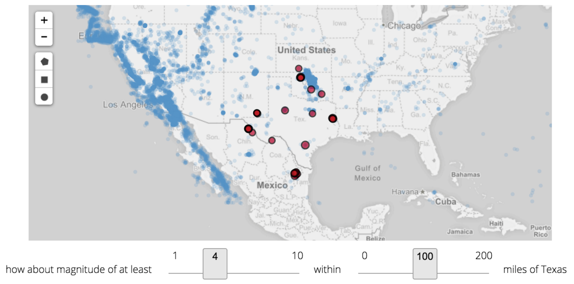
$$w(\pi) = \lambda \otimes w(t_1) \otimes \ldots w(t_n) \otimes \rho(n(t_n)) \qquad (1)$$

Figure 6 shows how the FSM can be applied to queries in Eviza. Given a previous query state "Large earthquakes in California", and a following query state "how about Texas?", there are multiple options for the most probable path. Attributes in the previous query state such as 'large' and 'earthquakes' , as well as user settings where large is set to 4 and above, are all augmented to the following query given the higher product of their corresponding transition probabilities.

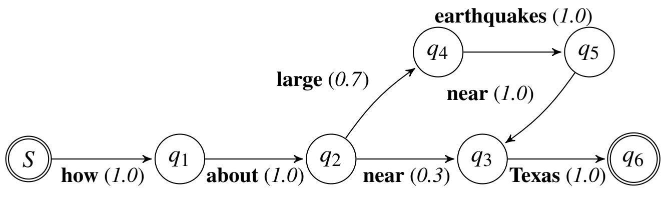
Eviza has built-in queries such as "reset", "show me all the data" to revert back to the original state in the FSM. However, the current system has no mechanism in place for the user to go back to any arbitrary state in the flow.



(a) Previous query: "Large earthquakes near California"



(b) Subsequent query: "how about near Texas?"



(c) Weighted finite-state flow

Figure 6: Pragmatics support in Eviza. (a) An initial result for the query "Large earthquakes in California", showing earthquakes within a 100 mile radius of California of magnitude 5 and greater. The user moves the magnitude slider down to 4. (b) A subsequent query "how about Texas?" resolves to finding earthquakes around Texas with 'large earthquakes' associated with this state. (c) A finite-state transducer showing the probabilistic likelihoods of alternative states for the query. Here, the path "how about large earthquakes near Texas" has a higher probability than "how about near Texas?"

11. ANALYTICS

Table 2 shows some examples of the analytical functions supported in Eviza. These include descriptive statistics (min, max, average), highlighting and coloring based on data attribute values, aggregation, reference lines, and correlation. In addition to these basic functions, we devoted additional attention to the special cases of spatial and temporal analytics. Based on our preliminary study, spatial and temporal queries were extremely common, involved specialized terminology, and introduced some unique types of ambiguity. We further explore the analytics of space and time in the subsections below.

11.0.1 Spatial Analytics

The potential for use of natural language in GIS has long been recognized as an important aspect in making spatial analysis more accessible [19]. A key part of success in natural language GIS is understanding how people conceptualize spatial topological relationships [36, 20] and in formalizing a spatial query language to support querying these relationships [17].

While spatial questions are common and desired for data exploration and analysis, Geographic Information Systems (GIS) are often not readily accessible or understandable to the average user. Eviza facilitates spatial analysis without the user needing to know specialized GIS operations. Our work currently supports several types of common spatial operations:

- **Point-based distance measures:** This includes operations to identify nearest neighbor(s) to any specified point.
- **Point-in-polygon relationships:** We enable identification of spatial overlap for point-in-polygon selection, as well as for disjoint selection of points *outside* a specified polygon (*e.g.*, "largest earthquakes outside California").
- **Spatial aggregation:** We support aggregation using regular spatial bins (*e.g.*, hexbins); we use equal area bins to minimize issues with distortion due to map projection.
- **Attribute-based filtering:** Spatial queries can be combined with numeric and categorical attribute-based filtering or extrema (*e.g.*, top or bottom N features).

Spatial analytic capabilities are managed using the Turf library [8] in conjunction with a geocoding library that we created to provide a framework for spatial queries (*e.g.*, polygonal state and county boundaries, point locations for cities).

We incorporate various capabilities for user interaction with spatial data:

- **Fuzzy spatial prepositions** such as 'near' or 'around': We create buffers of varying size to return selections relevant to the user's query. Adjustments to buffer size can be made through the distance ambiguity widget.
- **Multi-modal input** for richer interaction: Interaction is not restricted to language-based queries. As it is often easier to point or draw to identify geographic areas of interest, we allow for direct manipulation with the map to create polygonal selections (*e.g.*, radial, rectangular, or lasso select).

11.0.2 Temporal Analytics

Temporal queries are a common task during data analysis. The temporal analytics module supports the analytical reasoning of temporal expressions in the input queries.

We developed temporal functions based on the temporal entities and expressions defined by TimeML[31]. The module incorporates the following temporal token types to support the parsing of temporal expressions (Figure 3 and Table 2):

- **Temporal Units:** This includes temporal units that commonly occur in temporal expressions, including months, days of the week, hours and minutes, and their abbreviations. Normalized mappings of these temporal patterns are also considered. For example, 'July' is normalized to '07.' We enrich the basic temporal units with additional ones from WolframAlpha [9] such as quarters, semesters, half-years, seasons and various time zone formats.
- **Temporal Prepositions:** We support temporal prepositions such as 'in', 'during', 'near' and 'around.'
- **Temporal Connectives**: We support connectives such as 'before' and 'after' that connect a temporal anchor. For example, "before summer" starts with 'summer' as the temporal anchor, from which the temporal function for 'before' is computed.

Another aspect of handling natural language queries is the fact that these temporal expressions could be relative or underspecified (*e.g.* 'next year', 'June', 'this year.' Eviza considers both the temporal context of the visualization and temporal ordering based on pragmatics collected from previous queries to resolve these ambiguities.

12. EVALUATION

We conducted a preliminary user study to assess Eviza's natural language approach for interacting with visualizations. We had two main goals: (1) collect qualitative feedback on Eviza's design features and (2) gain insight into the situations in which people would prefer to use natural language interaction versus direct manipulation (DM); this information should help us integrate the interaction ideas into a more comprehensive analytics interface. To achieve our second goal, we wanted participants to experience comparable tasks using both natural language input and DM, so that we could discuss the advantages of each. We therefore compared Eviza to a typical DM visual analytics tool, specifically Tableau Desktop. Comparing to Tableau did not allow us to directly measure the value of specific features, but did support a qualitative discussion around the strengths and limitations of natural language interaction. In future user studies we will evaluate specific features by comparing Eviza to a feature-removed version of itself.

Because the main goal of our comparison was to gain qualitative insight into the advantages of each type of input, we encouraged participants to think aloud and have a conversation with the experimenter. To assess overall efficiency, we also measured time to complete each task set.

12.1 Method

Twelve volunteers completed a series of five tasks on an existing visualization using both Eviza and Tableau Desktop 10.0. Participants were recruited from all departments of a software company. To mitigate the influence of Tableau's learning curve, we aimed to recruit people who had experience with Tableau Desktop: two used Tableau Desktop on a regular basis and the rest used it occasionally and considered themselves beginners. Approximately half of the participants were native English speakers and the others were fluent.

Tasks involved searching, filtering, or manipulating a visualization to reveal specified target information (*e.g.*, countries with life expectancy < 50 in 1998). See the supplementary material for task details. Target information was presented in a table so that participants would not directly copy the instruction text as their query. Each participant interacted with one randomly chosen visualization (map, scatterplot, line chart, or bar chart) with both tools, in counterbalanced order. Charts in the two tools used the same data and same visual encoding. For consistency, all participants interacted with Eviza using text input rather than speech. Each session took 30 minutes and concluded with a semi-structured interview. We recorded screen capture video plus audio.

12.2 Results

We first present results of the time comparison between the two interfaces, followed by our qualitative results.

12.2.1 Task Time

Task time was measured from the screen capture video. We marked the start of a task as soon as the participant began reading the task instructions and the end once the target view had been created. Any discussion with the experimenter was excluded from the time measure.

Figure 7 shows that overall Eviza (mean 167*s*) was about twice as fast as Tableau (mean 381*s*). After log transforming the data to improve the fit to a normal curve, we compared the means using

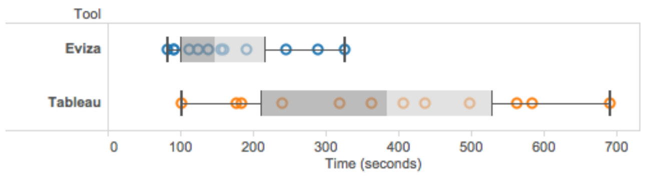


Figure 7: Boxplots of total task time (seconds).

a paired-samples t-test. The result showed a significant difference (t=6.35, df=11, p <0.001).

12.2.2 Natural Language vs. Direct Manipulation

Participants were very positive about natural language, *e.g.*, (P7) "Nice! That is so smart" and "pretty impressive. You can just type in a question and get the answer." All users expressed that Eviza was more natural but Tableau was more powerful. Even P15, who was a Gold certified Tableau user, stated, "Almost nothing to me seemed more natural to me in Tableau. The only reason it seemed expedient, if it was expedient, was because I have a lot of experience with Tableau." P9 commented that natural language enabled her to focus more on the data whereas with the DM interface she focused more on how to use the tool. Two participants commented that certain queries were easier to express as text phrases than in the mathematical thinking that underlies a DM interface. P2 expressed that NL might be most useful for complex actions like calculations & filters, when you are unsure how to do the action yourself (*e.g.*, calculating year over year growth).

We observed that tasks were easier with Eviza when any of the following were true: the DM interface required many clicks to complete the task (*e.g.*, having to precisely adjust a slider to get an exact value, or completing a long series of actions three times to create reference lines for min, max, and average), the user did not know how to do the task in the DM interface (*e.g.*, complex filter settings), the user did not know where to find the control for the function (*e.g.*, finding the regression line function), or the user did not know the name of the dimension to which a target item belonged (*e.g.*, it is not obvious that 'South America' is found within 'SubRegion').

Participants struggled to some degree with both interfaces. With the DM interface, participants struggled with finding the right function and configuring its options, due to the large volume of options and settings. In contrast, with natural language, people were unsure what commands the system would understand and did not always receive sufficient feedback to know whether their question had been understood.

Participants reported that natural language and DM were complementary. Natural language was easier to learn and good for focused queries. But Tableau's DM interface was more powerful and expressive, making it potentially more suitable for complex or exploratory tasks. One participant felt that natural language was easier for him as a novice user, but that power users would more likely prefer using DM. He suggested incorporating both forms of interaction into one interface.

12.2.3 Feedback on Eviza Features

Pragmatics: Reactions to system memory of previous queries was mixed, with some participants finding this behavior very helpful and others finding it unexpected. There were very few examples of true follow-up queries because they were not necessary for our study tasks, but two participants took advantage of this functionality to break a task into steps. P14 particularly appreciated this feature and used it in three out of five tasks. For example, to find the extreme temperatures in May 2014, he entered two queries in sequence: "Display temperatures in May 2014" followed by "What are the high and low temperatures." During the interview, he stated, "I was able to break down tasks into pieces. It was nice that the graphs and interface didn't revert back to some previous state." Similarly, P7 attempted to use a long, complex query that Eviza could not fully recognize ("show highest average GDPPerCapita of all countries between 1950 to 1960"); when that failed, she broke the task into parts. Clearly in this case it would have been better to support the original query, but the pragmatics provided a useful fallback approach.

In contrast, P13 and P15 got unexpected results because of pragmatics. P15 asked for a trend line and was surprised that Eviza kept countries highlighted from the previous query. Similarly, P13 was surprised to see min and max values still highlighted from a previous query after she asked for the average: "Oh! Now you still have min and max...I didn't expect it to be there. I prefer it to go away." These experiences suggest that we need to establish better criteria for deciding when to remember information from prior queries, and support flexibility for users to correct poor system choices.

Handling ambiguity: As expected from the initial study, handling ambiguous queries was essential. Eight out of twelve participants expressed queries that required some ambiguity handling. Some examples include: "show *recent* earthquakes", "earthquakes *near* Rhode Island", and "show *high*, *low*, *avg*, *C* in *spring*". Participants appreciated not having to precisely specify their queries. For example, P5 stated, "eviza is very straightforward, you don't need to know the dimension names." Nonetheless, Eviza's handling of ambiguity was limited and participants sometimes needed to try two or three different phrases. For instance, Eviza could not relate "country with most people" to the *population* dimension and could not relate "C" and "F" to temperature units *Celsius* and *Fahrenheit*.

Participants also appreciated and used the ambiguity widgets to increase precision and to explore. For instance, after entering a query for "large" earthquakes, P12 expressed, "Oh, look, there's a slider that showed up! Nice!"; she then used the slider to adjust the magnitude threshold. However, at least two participants failed to notice the widgets at first due to their position below the visualization. In addition, occasionally the ambiguity widgets expressed a result different from how the user had conceptualized their query. After P12 typed, "find recent earthquakes", the system responded with an ambiguity widget for the "N most recent earthquakes" (N is variable). However, in this instance the user expected to be able to specify a date range.

Autocomplete: Autocomplete was a heavily used feature. Nine out of twelve participants actively used this feature, often repeatedly. The remaining three participants may have used it but we could not be certain: when the feature appeared, they often typed words that matched it, but they never directly selected items from the list. Several participants stated during the interview that this feature was helpful. For example, P10 said, "I did find myself trying to think about how to phrase things...the hints really helped to get that language right," and P11 stated, "I ended up coming with something I didn't plan but I was kind of guided. It helped me to phrase this search."

Autocomplete served three distinct functions. First, it helped with difficult-to-spell attribute names or words (*e.g.*, "Fahrenheit" or "GDPPerCapita"). Second, it reduced the need to type long words and phrases (*e.g.*, type "rhod" then pick "Rhode Island" from a list of states). Third, it helped users learn what language the system understood. P12 formulated one query almost word for word via autocomplete. First he typed “find”, then he saw the word “earthquake” and selected it, then he saw the word “in” and selected it, and finally he typed “C” and then selected “California” from the list. As another example, P13 was attempting to change units in a line chart. After two failed attempts to create a query that Eviza would understand, she started using autocomplete. She first typed “show” and then looked at the suggestions, eventually typing “unit in” to match one of the suggested phrases. She then typed “F” and completed the word “Fahrenheit” by selecting it from the list.

Although autocomplete was heavily used, it led at least one participant to think that only those example queries were supported. Repeatedly typing similar queries was also annoying, leading two users to suggest a history list of past queries in addition to autocomplete.

12.2.4 Empowerment

Interviews also revealed some interesting issues around empowerment, ownership, transparency, and trust when using natural language. Although all of the users found the tasks easier to complete with Eviza, two of them expressed that the ease of the task led to a loss in empowerment and ownership. One stated that “it takes away the pleasure of doing it yourself”, and the other used the analogy, “it’s like buying things from the store versus making the pasta yourself." A related concept was the level of control. P13 was particularly concerned about the lack of control, stating, “To be honest I don’t mind being more explicit. I’m actually more irritated by the assumptions the machine makes...when I say max I mean max, not max & average...It’s almost like amazon adding stuff into your cart that other people bought."

Several participants were also concerned about what the system was doing to their data in the back end. The ‘black box’ nature of the analytical functions led to concerns about transparency and trust. One user successfully completed a task that required her to average the data across a ten year period, but it was not obvious to her whether Eviza had aggregated the data correctly. Another participant felt that Tableau filters, while not easy, at least made the filter settings transparent, and a third participant complained that it was not clear what was in the data or how Eviza was calculating things like currency conversion. This feedback indicates that natural language interfaces need to expose some details about the data provenance so that users can understand what has happened to their data.

13. DISCUSSION AND FUTURE WORK

Results of the evaluation confirmed our intuition that people would find it natural and intuitive to interact with visualizations using natural language. The keywords and grammars supported by Eviza were discoverable and natural, and the ability to express a command in the way you think about it made the approach both fast and easy to learn.

Natural language was also complementary to DM, so integrating both into an analytics interface is a promising design direction. Natural language interaction might even help people learn a DM interface, if recognized natural language commands were translated to highlighted actions in the interface.

At the same time, natural language interpretation would need to work very well for a broad range of queries for it to be effective in a non-prototype implementation. Interestingly, since Eviza could understand natural language, participants also expected it to be smart. For example, they expected it to have access to commonly available data (*e.g.*, dates of World War II), and to understand references like ‘biggest brown circle’, vague geographic areas like ‘Eastern states’, and domain-specific words such as ‘coldest.’ While people will adapt and rephrase their questions if they are not immediately understood, having to do so very many times can make the interactive experience frustrating and slow.

By constraining the context, Eviza has made several significant steps towards greater natural language expressibility through richer semantics, pragmatics and expressibility. However, there are several interesting directions for future work.

While the study participants liked the overall idea and utility of the system, there were some issues. Eviza had trouble recognizing all parts of long and complex queries and could not recognize all grammatical constructs (*e.g.*, “highlight those whose magnitude is greater than 4” and “in the eastern part of the US”). An extension to our work is to explore more robust semantic parsers. We would like to explore how longer, complicated queries could be broken down into simpler components through mixed initiative approaches such as clarification dialogs presented to the user.

Participants appreciated the simplicity of accessing analytic features with natural language, but understood that Tableau was much more powerful. Apart from use of words and grammar that Eviza did not understand, most of the other natural language “failure” cases related to analytics functionality that had not been built in. Eviza’s temporal analytics were very limited, making it unable to deal with requests for seasonal trends or queries such as “what country has changed the most between 1900 to 2000.” Eviza’s geographic capabilities could also be extended (*e.g.*, to understand which regions are ‘islands’). Most of these failures occurred when users explored Eviza after completing the study tasks.

One place where natural language interaction may shine is for complex analytics. Eviza enables some analytical functions (*e.g.*, complex filters, correlation, aggregation), but this could be taken much further to make complex analytics accessible to people with limited knowledge of statistics or calculations. People may be able to easily express natural language questions such as “is there a seasonal trend?” without knowing how to calculate the answer. Extending the semantics of the system with additional corpora and grammar mappings to the visualization, including user-defined ones, could further help improve the user experience.

While our current system focuses on a single visualization, extending the behavior to support queries across multiple visualizations, such as a dashboard or a set of co-ordinated views, could lead to interesting research problems in query parsing and ambiguity. Also, extending the system to generate new views (complementing the current visualization) could better support analytical flow questions such as “compare average earthquake magnitudes in California and Oregon", perhaps by complementing a map view with a histogram of magnitudes.

Lastly, we believe that for natural language to be effective in visual analytical tasks, it needs to work well with other modalities such as drag-and-drop, touch, voice and user context. While we have explored some initial ideas around using voice and context (*e.g.*, location), future research could further explore pragmatics and ambiguity challenges related to different input modalities.

14. CONCLUSION

Eviza is a prototype system that enables a user to have a conversation with their data using natural language. Whereas previous work has focused on generating a new visualization based on a user’s typed or spoken query, Eviza tackles a different problem: enabling an interactive conversation with an existing visualization. In building Eviza, we have made several novel contributions to natural language interaction with visualizations: use of a probabilistic grammar-based approach that dynamically refines the parsing rules based on the data in the context, the use of template-based autocomplete to help users gain an understanding of available commands, richer language pragmatics, extension of ambiguity widgets to deal with quantitative and spatiotemporal ambiguity, and integration of multimodal interactions and spatial, temporal, and quantitative analytics. Our preliminary evaluation validated the intuition that natural language interfaces are a promising approach to interacting with data and making analytics accessible to a broad audience. By enabling users to ask questions in the same way they think, natural language has strong potential to support the flow of visual analysis. In future work, we plan to continue developing Eviza to further explore the potential of natural language and its integration with other input modalities.

15. ACKNOWLEDGMENTS

The authors wish to thank Heidi Lam, Terry Roberts and the anonymous reviewers for their helpful feedback.

16. REFERENCES

[1] Extended Backus-Naur Form. https://en.wikipedia.org/wiki/Extended_Backus%E2%80%93Naur_Form.

[2] I-want-to-go moments: From search to store. https://www.thinkwithgoogle.com/articles/i-want-to-go-micro-moments.html.

[3] IBM Watson Analytics. http://www.ibm.com/analytics/watson-analytics/.

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

[5] NarrativeScience. https://www.narrativescience.com/quill.

[6] Node.js®. https://nodejs.org/.

[7] ThoughtSpot. http://www.thoughtspot.com/.

[8] Turf: Advanced geospatial analysis for browsers and node. http://turfjs.org.

[9] WolframAlpha. https://www.wolframalpha.com/.

[10] Allen, J. Recognizing Intentions from Natural Language Utterances. In *Computational Models of Discourse*, M. Brady, Ed. M.I.T. Press, Cambridge, Massachusetts, 1982.

[11] Androutsopoulos, I., Ritchie, G. D., and Thanisch, P. Natural language interfaces to databases—an introduction. *Natural Language Engineering 1*, 1 (1995), 29–81.

[12] Aurisano, J., Kumar, A., Gonzales, A., Reda, K., Leigh, J., Di Eugenio, B., and Johnson, A. Show me data? observational study of a conversational interface in visual data exploration. IEEE VIS (2015).

[13] Bostock, M., Ogievetsky, V., and Heer, J. D3: Data-driven documents. *IEEE Transactions on Visualization & Computer Graphics (Proc. InfoVis)* (2011).

[14] Carbonell, J. G., Boggs, W. M., Mauldin, M. L., and Anick, P. G. The xcalibur project, a natural language interface to expert systems and data bases. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (1985).

[15] Cover, T. M., and Thomas, J. A. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991.

[16] Cox, K., Grinter, R. E., Hibino, S. L., Jagadeesan, L. J., and Mantilla, D. A multi-modal natural language interface to an information visualization environment. *International Journal of Speech Technology 4*, 3 (2001), 297–314.

[17] Egenhofer, M. Spatial sql: A query and presentation language. *IEEE Transactions on Knowledge and Data Engineering 6*, 1 (1994), 86–95.

[18] Finin, T., Joshi, A. K., and Webber, B. Natural language interactions with artificial experts. *Proceedings of the IEEE 74*, 7 (June 1986), 921–938.

[19] Frank, A. U., and Mark, D. M. Language issues for geographical information systems. In *Geographical Information Systems: Principles and Applications, vol 1*, D. Maguire, M. Goodchild, and D. Rhind, Eds. Longman, London, 1991, 147–153.

[20] Freeman, J. The modelling of spatial relations. *Computer Graphics and Image Processing 4*, 2 (1975), 156–171.

[21] Gao, T., Dontcheva, M., Adar, E., Liu, Z., and Karahalios, K. G. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software Technology*, UIST '15, ACM (New York, NY, USA, 2015), 489–500.

[22] Grammel, L., Tory, M., and Storey, M. A. How information visualization novices construct visualizations. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov 2010), 943–952.

[23] Li, F., and Jagadish, H. V. Constructing an interactive natural language interface for relational databases. *Proc. VLDB Endow. 8*, 1 (Sept. 2014), 73–84.

[24] Montello, D., Goodchild, M., Gottsegen, J., and Fohl, P. Where's downtown? behavioral methods for determining referents for vague spatial queries. *Spatial Cognition and Computation 3*, 2&3 (2003), 185–204.

[25] Myers, K. L., and Yorke-Smith, N. A cognitive framework for delegation to an assistive user agent (2005).

[26] Ng, H. T., and Zelle, J. Corpus-based approaches to semantic interpretation in natural language processing. *AI Magazine Winter 1997*, 45–64 (1997).

[27] Oviatt, S., and Cohen, P. Perceptual user interfaces: Multimodal interfaces that process what comes naturally. *Commun. ACM 43*, 3 (Mar. 2000), 45–53.

[28] Parr, T. *The Definitive ANTLR 4 Reference*, 2nd ed. Pragmatic Bookshelf, 2013.

[29] Pedersen, T., Patwardhan, S., and Michelizzi, J. Wordnet::similarity: Measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, HLT-NAACL–Demonstrations '04, Association for Computational Linguistics (Stroudsburg, PA, USA, 2004), 38–41.

[30] Popescu, A.-M., Etzioni, O., and Kautz, H. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, IUI '03, ACM (New York, NY, USA, 2003), 327–327.

[31] Pustejovsky, J., Castaño, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., and Katz, G. Timeml: Robust specification of event and temporal expressions in text. In *in Fifth International Workshop on Computational Semantics (IWCS-5* (2003).

[32] Reinhart, T. *Pragmatics and Linguistics: An Analysis of Sentence Topics*. IU Linguistics Club publications. Reproduced by the Indiana University Linguistics Club, 1982.

[33] Roche, E., and Shabes, Y., Eds. *Finite-State Language Processing*. MIT Press, Cambridge, MA, USA, 1997.

[34] Salton, G., and McGill, M. J. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[35] Sun, Y., Leigh, J., Johnson, A., and Lee, S. *Articulate: A Semi-automated Model for Translating Natural Language Queries into Meaningful Visualizations*. Springer Berlin Heidelberg, 2010, 184–195.

[36] Talmy, L. How language structures space. In *Spatial Orientation: Theory, Research, and Application*, H. Pick and L. Acredolo, Eds. Plenum, New York, 1983.

[37] Winograd, T. *Procedures as a Representation for Data in a Computer Program for Understanding Natural Language*. PhD thesis, 1971.

[38] Wu, Z., and Palmer, M. Verbs semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, ACL '94, Association for Computational Linguistics (Stroudsburg, PA, USA, 1994), 133–138.

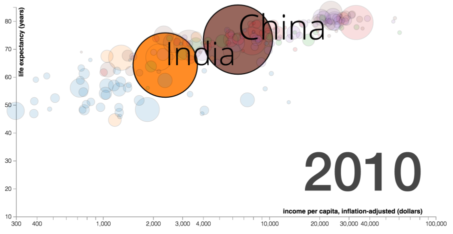
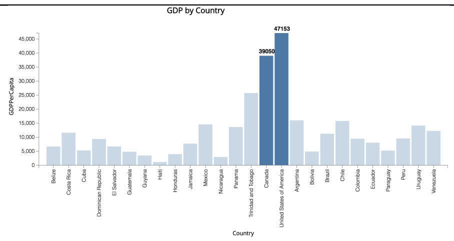
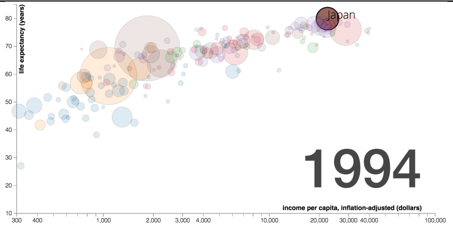
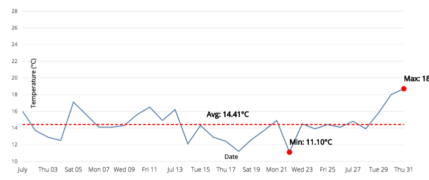
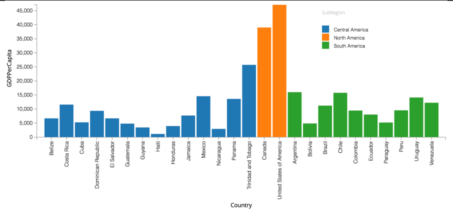
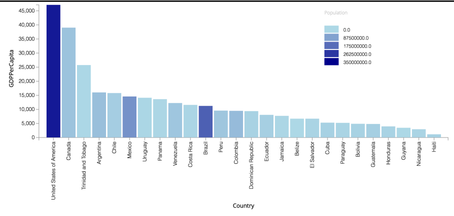
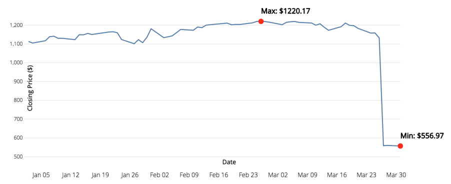
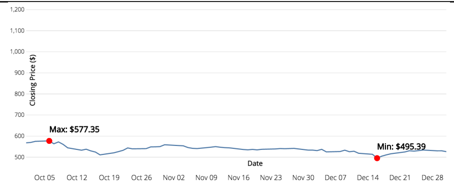
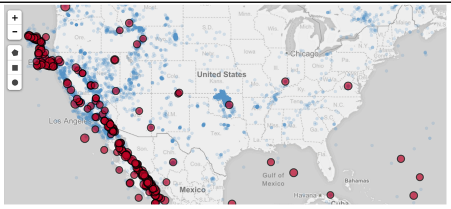
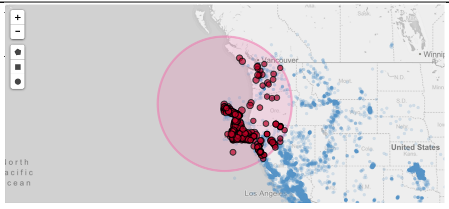
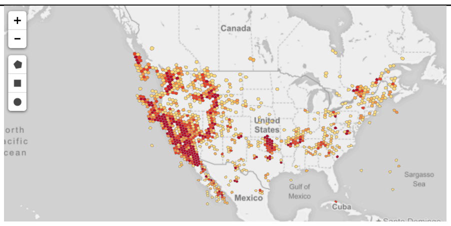
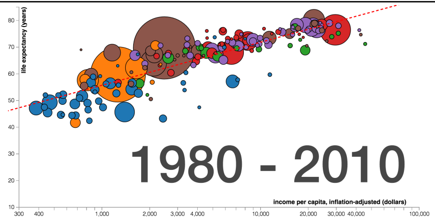
Analytic functions	**Examples**	
highlight	 "highlight India and China"	 "find North America"
quantitative comparisons	 "show life expectancy greater than 83"	 "high low and average in July"
color encoding	 "color by region"	 "color by population in descending"
temporal	 "prices in the 1st quarter with high and low"	 "what about around November?"
spatial	 "large magnitudes outside NY"	 "earthquakes with magnitudes between 4 and 6 here"
aggregation	 "aggregate all these earthquakes"	 "find correlation between 1980 and 2010"

Table 2: Types of analytical functions supported in Eviza along with some examples.