The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

Intelligent Assistant for Exploring Data Visualizations

Abhinav Kumar, Jillian Aurisano, Barbara Di Eugenio, Andrew Johnson[1]
{akumar34, jauris2, bdieugen, ajohnson}@uic.edu
[1]University of Illinois at Chicago
Chicago, IL USA

Abstract

Visualization, while an effective tool for identifying patterns and insights, requires expert knowledge due to challenges faced when translating user queries to visual encodings. Research has shown that using a natural language interface (NLI) is effective for these challenges because the user can simply talk to a computer capable of producing the graphs directly. In this paper, we discuss our intelligent assistant which processes speech and hand pointing gestures while also dealing with any number of visualizations on a large screen display. Evaluation of the system shows that it is capable of quickly producing visualizations. It also particularly effective at responding to less ambiguous queries, while in certain cases can handle ambiguous or complex queries.

Exploring large datasets with visualizations makes gathering insights easier due to the ability to quickly identify patterns and compare trends between visualizations. However, for users unfamiliar with visualization, (Grammel, Tory, and Storey 2010) points out the steep learning curve necessary to translate high-level queries into visual representations. While popular tools such as Tableau can help facilitate the process, learning new user interfaces itself presents challenges that can overwhelm the user. Speech recognition and natural language processing, whose advances have in turn led to commercial success of natural language interfaces (NLIs) (e.g., virtual assistants such as Apple Siri, Amazon Alexa, and so on), have been a key focus of the research community in alleviating these challenges. In particular, various interactive visualization systems have been proposed (Cox et al. 2001; Reithinger et al. 2005; Sun et al. 2010; Setlur et al. 2016; Gao et al. 2015; Hoque et al. 2017; Yu and Silva 2019), that implement NLIs for visualization to help process verbal and nonverbal communication, effectively decouples the user interface from the user.

In this paper, we discuss our own intelligent assistant that we claim provides a more supportive environment for data exploration relative to other systems; the user is able to use a large screen display to explore data, can interact with the large screen using free-forming NL as well as pointing gestures, and has the ability to manage multiple visualizations on the screen at once. Our contributions in this paper are: 1) Our assistant is implemented as a dialogue system (such systems are capable of having conversations with humans and have been successful in various domains, for example airline travel (Hemphill, Godfrey, and Doddington 1990; Budzianowski et al. 2018)). It is modeled after our own collected multimodal dialogue data (Kumar et al. 2016; 2017), capturing user interaction with a large screen display while tasked with exploring city crime data for Chicago (called CHICAGO-CRIME-VIS corpus). As a result, rather than making assumptions and enforcing NL templates, our approach is flexible to the different ways that user queries are spoken. Note that only 15% of our data contains such queries, which we refer to as *actionable requests* (AR) directly, while the remaining 85% of the spoken utterances do not specify actions to be taken by the system. 2) Our system is configured to operate on a large screen display, effectively allowing the user to preserve a sizable number of past visualizations on the screen. The user is able to quickly refer to any of these visualizations when forming subsequent ARs or analyzing recently constructed ones. 3) The system manages a dialogue history (DH) of past visualizations, which can be accessed in the future to retrieve visualizations that correspond to references made by the user. For example *"Show me the last heat map."* finds the most recent entry associated with map plots. 4) We conducted an extensive evaluation of the system with 20 subjects in a user study. The evaluation showed that the system is fast at responding to ARs and is also effective in generating satisfactory (according to user feedback) visualizations to ARs that have little ambiguity. One of the primary limitations observed by participants is the tendency by the system to take the meaning of an AR too literally, which can lead to a wrong understanding of the underlying intentions of the user and consequently a misleading visualization.

Related Work

Interactive Systems for Visualization

Advances in the field have led to a number of systems leveraging NLIs for the purpose of exploring visualizations. (Cox et al. 2001) supports multimodal input (speech and mouse clicking) however they templated their commands hence limiting the kinds of NL queries the system can understand. (Reithinger et al. 2005) developed a system to assist users in finding multimedia content via haptic visualization, how-

ever their visualizations are pre-defined and cannot be constructed on the fly. The implementation proposed by (Sun et al. 2010) allows for multiple visualizations on the screen, however they only support speech (unimodal input) and use less sophisticated machine learning methods for language processing. Another system (Gao et al. 2015) only allows a single visualization to display on the screen at a given time however they also provide interactive widgets that allow the user to correct misunderstandings by the system about a given query. The next several systems (Setlur et al. 2016; Hoque et al. 2017) use more sophisticated language processing techniques however only operate on existing visualizations and follow up queries on them rather than focus on creating new visualizations. Finally, (Yu and Silva 2019) is also unimodal, supporting only speech input, however they also focus on sophisticated language processing to improve processing of spoken language.

The goal of our system is to process multimodal input, allow for visualization construction on the fly, operate on a large screen display, and give the user the ability to manage multiple visualizations on the screen.

Dialogue Systems

A dialogue system is designed to have conversations with humans. They have been widely successful for a variety of domains, such as phone conversations (Williams et al. 2013), tourism (Kim et al. 2017), and chat (Lowe et al. 2015). The classic architecture is comprised of natural language understanding (NLU) to interpret NL utterances, dialogue manager (DM) for maintaining the current state of the conversation and make decisions on the appropriate subsequent actions to be taken by the system, and finally an output generator focused on performing that action. In our case, we deal with spoken dialogue, which typically requires speech-to-text processing prior to executing the NLU. The success of deep learning networks has lead to End-to-end (E2E) neural network based architectures for dialogue systems (Bordes, Boureau, and Weston 2017; Wen et al. 2017), which have produced state-of-the-art results. However, the E2E architecture requires large amounts of data to train, while our manually built corpus is relatively much smaller and hence we elected to implement a classic architecture for our application.

CHICAGO-CRIME-VIS Corpus

We performed data collection with 16 subjects, each tasked with determining when and where to deploy police officers (Kumar et al. 2016). They explored Chicago crime data (obtained from the Chicago data portal[1]) using visualizations on a large screen, interacting with a visualization expert (a human working from a separate room) using speech and hand gestures. Later, we transcribed the dialogues using video recordings of each participant, for a total of 3,179 total utterances, 1,879 word types, and 38,105 word tokens.

In terms of annotation, we identified 449 ARs and coded them with one of 8 possible types (ARs are similar to dialogue acts (DAs)). See Table 1 for details of each AR. The

[1]data.cityofchicago.org

Dialogue Act	Freq	Example
CREATE-VISUALIZATION	198	*"so *uh*, can I see the visualization for crime in the our-city neighborhoods?"*
CLARIFICATION	82	*"Ok, so, what do you mean by non-criminal?"*
WINDOW-MANAGEMENT	57	*"If you want you can close these graphs as I won't be needing it anymore"*
FACT-BASED	43	*"So, what kind of crime is what kin– what kind of crime is maximum?"*
PREFERENCE-BASED	33	*"*Uh* okay I somehow feel the 06-2 and 07-2 they are they're much clearer *uh* to visualize and understand rather than 06-1 and 07-1."*
MODIFY-VISUALIZATION	27	*"*Uh* do– can you show only the bar graph or do you have some any other way of visualizing the same?"*
APPEARANCE	7	*"*Um*, interesting, can I see labels on the data please?"*
HIGH-LEVEL-QUERY	2	*"So, according to you, which areas do I deploy the officers?"*

Table 1: 8 Dialogue Acts in the *CHICAGO-CRIME-VIS* corpus.

intercoder agreement, using 3 coders on 4 of the same subjects, was $\kappa = 0.74$. We also coded 536 total gestures identified using the recorded videos described earlier, of which 368 were pointing gestures (out of these, 159 were pointing to visualizations on the screen while the remaining were targeting entities within visualizations themselves). The intercoder agreement based on 2 coders annotating the same 2 subjects was $\kappa = 0.659$ for differentiating the different types of gestures (pointing, circling, and so on) and $\kappa = 0.639$ for gesture target (pointing to visualizations or pointing to entities within visualizations).

System Architecture

Figure 1 shows our intelligent assistant implemented using the classic dialogue system architecture. Note the three separate modules in the system architecture, including natural language understanding (NLU), dialogue managemement (DM), and visualization execution (VE). The user interacts with a large screen display using speech (Google Speech API on an Android phone app is used for speech-to-text processing) and hand pointing gestures (Microsoft Kinect is used for detecting where on the screen the user is pointing), which are then translated into action and accordingly updated and rendered back to the large screen.

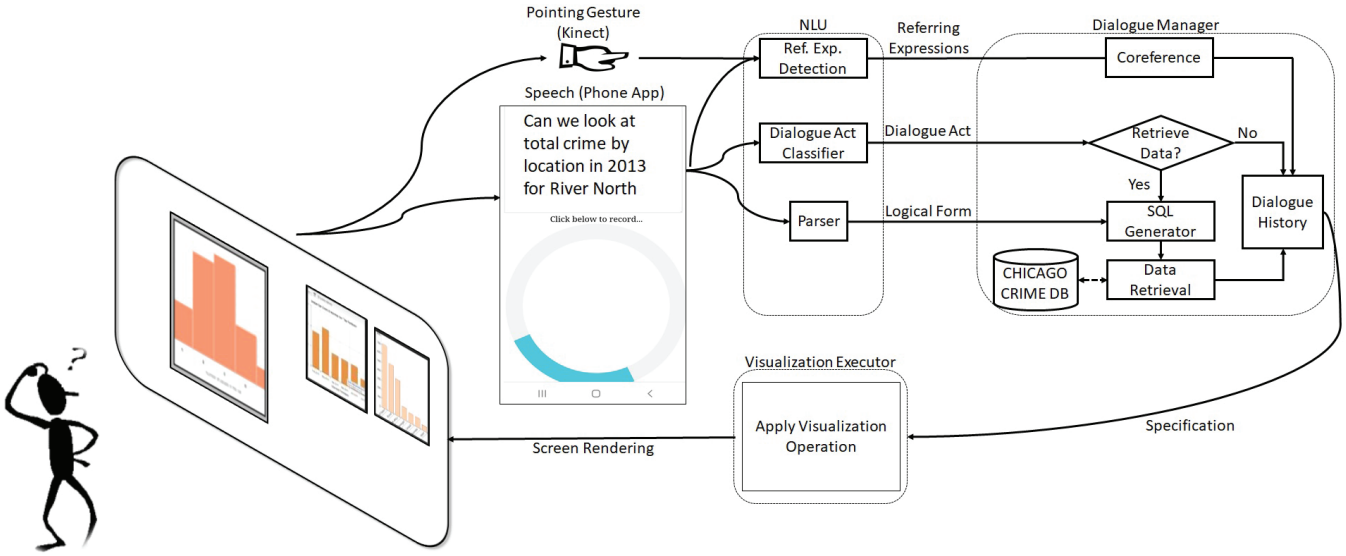

Figure 1: The system pipeline.

Natural Language Understanding

The NLU is comprised of three primary tasks, of which more details on dialogue act classification and parsing can be found in (Kumar et al. 2016) while identification of referring expressions (REs) is a recent addition.

The dialogue act classifier is required to determine what type of AR was spoken by the user for down stream processing (i.e., if the user asks to close a visualization on the screen, the system should look up the existing visualization rather than create a new one). It assigns a semantic label to the AR from the set of 8 DAs described earlier (see Table 1. However, the system implementation only handles *CREATE-VIS*, *MODIFY-VIS*, and *WINDOWS-MANAGEMENT*, while for the other 5 DAs, the system uses text response to asks the user to try something else.

Aside from this, the goal of the parsing component was to identify which information the user desires to filter or aggregate in producing a target visualization (i.e., an intermediate logical form) and subsequently derive the SQL queries for database retrieval. This is an essential step because we do not have access to an existing corpus that maps language to database queries and visualization. Our logical form is represented in the form of a conjunction of clauses, namely $C_{predicate} \cap C_{agent} \cap C_{patient} \cap C_{det} \cap C_{mod} \cap C_{action}$, in which each clause is filled using primarily semantic role labeling (SRL) and syntactic tree parsing. Our system currently relies solely on the information provided in the $C_{predicate}$ and C_{action} clauses, although the other clauses could be leveraged as additional features in the future for improved SQL generation. First,the relevant NPs are identified from tree parsing (we used Stanford parser) to fill the C_{action} clause with nouns, C_{det} clause with determiners and C_{mod} clause with modifiers. The nouns in particular are crucial because they allow us to identify which information to filter and aggregate. Using the example from (Kumar et al. 2016) *"Can I see assaults in the Loop by location type?"*, the *"assaults"* are a filter on the type of crime, *"Loop"* is a filter on a particular neighborhood in the city, and *"location type"* is an aggregate of all the different locations in the city (i.e., restaurant, school, and so on). The SRLs (which refer to a predicate and its arguments) fill the $C_{predicate}$ clause, while also populating the C_{agent}, and $C_{patient}$ clauses (we used ClearNLP (Choi 2014) to derive PropBank (Palmer, Gildea, and Kingsbury 2005) SRLs, which were then further enriched with Verbnet (Kipper et al. 2008) to identify these particular thematic roles). In our current system, the predicate is used to select the highest probable parse tree from the list of Stanford parse trees that also identifies the predicate as a verb. Subsequently, the logical form is then leveraged to form valid SQL queries for database retrieval. Each filter is appended to the *WHERE* clause while aggregates are appended to *GROUP BY* clause.

The purpose of the final component, focused on detecting referring expressions in the current AR, is to identify which existing visualization on the screen is being referred to by the user. It deals with the scenario in which the user would like to create a new visualization based on one that already exists on the screen. For example, the component identifies the RE *"the same chart"* (using keyword matching) along with the hand pointing gesture (using Microsoft Kinect) towards the graph corresponding to *VIS ID 7* when processing the AR *"Can you show the same chart [POINT TO VIS ID 7] for days of the week?"*.

Dialogue Management

The DM maintains a dialogue history of all the visualizations requested so far in the current session with the user, sorted in the order in which they were requested, and leverages it to determine the best visualization operation to apply. First, the DM checks the dialogue act label of the AR. If it is *CREATE-VIS*, the DM translates the parsed logical form into an intermediate SQL query which is then used to retrieve

relevant data, and then the DM adds the visualization object (which we refer to as visualization specification) to the top of the dialogue history. The *MODIFY-VIS* also performs similar to *CREATE-VIS* unless it is a template based query (e.g., *"Can you show the same chart [POINT TO VIS ID 7] for days of the week?"*) or plot type query (e.g., *"Can you go back to the last map?"*) is requested. For the template-based query, the coreference module is subsequently called to look up the visualization object in the DH corresponding to identifier 7 as the initial template from which to construct the new visualization. If for instance, this template contained the filters *"theft"* and *"Loop"* and the temporal aggregate *"month"*, then the processing would copy the filters into the new visualization but replace the aggregate attribute with *"days of the week"*. In the case of plot type query, the DM simply looks up the most recent visualization object that matches the plot type in the AR. Finally, if it is a *WINDOWS-MANAGEMENT* request, the DM retrieves the appropriate previous visualization specification from the dialogue history. Then, the DM sends the visualization specification to the VE, which carries out the action mentioned in the specification. For instance, in the AR *"Close this [POINT TO VIS ID 3] graph."*, the DA classifier identifies this as *WINDOWS-MANAGEMENT*; hence the DM retrieves the visualization object stored in the DH associated with visualization identifier 3 and then indicates in the visualization specification that it should be removed from the large screen.

Visualization Executor

The aim of the VE is to carry out the action described in the visualization specification, by either creating a new visualization or manipulating existing ones already on the screen. The updates are rendered back to the screen for the user. In the case of creating a new visualization, the specification includes data retrieved using the SQL query, x-axis, y-axis, plot type, DA label, and chart title (along with other useful information such as the DH, and so on). The VE uses Vegalite[2] to plot a graphical representation of the visualization specification. We currently support 2-D bar charts, line graphs, and heat maps.

Evaluation

Our purpose for this evaluation was to understand the effectiveness of the system in processing speech and gesture input. A total of 20 human subjects participated in this evaluation, including undergraduate students, graduate students, and other university affiliates.

For speech input, the subject tapped the circle button on the phone app (shown in Figure 1) once to turn on listening and once the user stopped talking, the phone app transmitted the captured text to our system. If the system misheard the AR, the user was allowed to repeat it again. The gesture system (using Microsoft Kinect) detected hand pointing gestures, and also identified the visualization being pointed to on the screen. If the subject pointed to multiple visualizations, our configuration only registered the one pointed to for the longest duration.

[2]https://vega.github.io/vega-lite/

Study Design

The study was segmented into four separate parts. In the first 3 parts, the subject was told exactly which ARs to say (taken from real dialogue from the *CHICAGO-CRIME-VIS* corpus). We list the ARs from part 3 in Figure 2 as an example; there are also 6 ARs for part 1 and 6 ARs for part 2 that we do not show here. In the final part, the subjects were encouraged to come up with their own ARs. The speech input was turned off for part 1 (we manually typed into the system each AR that the user had spoken). We enabled speech input for parts 2, 3, and 4. With regards to gesture input, we enabled them for parts 3 an 4 only. For parts 1, 2, and 4, the screen was blank initially (visualizations were all cleared) while in part 3 the last visualization from part 2 was intentionally retained on the screen. The subject was asked to point to that visualization while speaking the first 2 ARs in part 3 (which consequently resulted in the creation of 2 additional visualizations) and then the user was free to point to any visualization on the screen for each of the remaining ARs. After completing each part, the subject was asked to fill out a feedback survey, listed in Figure 3. The first 5 questions asked the subject to simply rate between 1 and 5 while the final 2 questions asked for short essay responses pertaining to best and worst responses produced by the system according to the subject.

Results

1. Can you show the same chart for days of the week?
2. Can you show this graph for months of year?
3. Can you move it
4. Can you minimize it?
5. Can you maximize it?
6. Can you bring up this pic that has been hidden?
7. Yeah don't need this one

Figure 2: *Actionable requests* from part 3 of the study.

We found the speech app to be very effective overall, logging 103 misheard words (including repetitions) out of 3,140 words for parts 2 and 3 (recall in part 1 the ARs are manually typed). A majority of the mistakes were made in part 2, with just 5 words (i.e., *"theft"*, *"wondering"*, *"year"*, *"pic"*, and *"graph"*) making up 36% of the the total mistakes. Note that *"theft"*, *"year"*, and *"graph"* are particularly problematic, because they describe important entities, and hence will result in the wrong visualizations being constructed when misheard. A visualization was successfully constructed for only 31.7% of the 63 ARs in which the speech system misheard the subject (including ARs repeated a second time due to being incorrectly heard the first time)

We also analyzed whether subjects found the behavior of the system significantly better in one of the 4 parts with re-

1. On a scale of 1 to 5, on average how quickly did the system respond to requests?
2. On a scale of 1 to 5, on average how understandable were the responses?
3. On a scale of 1 to 5, on average how well did the responses directly answer the requests?
4. On a scale of 1 to 5, on average how well did the responses provide additional insight not directly requested?
5. On a scale of 1 to 5, how would you rate the responses overall?
6. Which of the responses did you find to be the best and why?
7. Which of the responses did you find to be the worst and why?

Figure 3: The feedback survey questions used in evaluation study.

spect to the other 3. For this, we computed statistical significance when comparing each of the 4 parts based on the ratings we received on the first 5 survey questions (listed in Figure 3). The results are summarized in Table 2. First, note for questions 2 and 4, no part performed significantly better than other parts (hence not included in the table). Question 1 (corresponding to response time) shows that parts 2, 3, and 4 were rated significantly better than part 1. This is expected since part 1 required typing each AR manually. For question 3, which asks whether the system answered the AR being asked, we were surprised to see part 4 perform significantly better than parts 1, 2, and 3 since the subject is free to ask anything of the system (i.e., not scripted). We hypothesize this may have to do with subjects asking more direct questions (i.e., recall that the ARs in the previous parts were taken from the *CHICAGO-CRIME-VIS* corpus and may have been perceived as vague, such as *"Lets start with the activity around university"*, where we are unsure what *"activity"* is referring to in the data). By the same reasoning, again part 4 is found to be significantly better, this time when compared to parts 1 and 3, for question 5 (asking to rate the overall response).

	Q. 1	**Q. 3**	**Q. 5**
Parts	**p-value**	**p-value**	**p-value**
1,2	**<.05**	**<.05**	0.78
1,3	**<.05**	.88	0.66
1,4	**<.05**	**<.05**	**<.05**
2,3	.72	**<.05**	0.19
2,4	.90	**<.05**	0.07
3,4	.38	**<.05**	**<.05**

Table 2: Statistical significance testing (in **bold**) for the rating feedback questions 1, 3, and 5 (no statistical significance for questions 2 and 4) when comparing each of the 4 parts.

Aside from ratings analysis, we also studied the final 2 short essay questions pertaining to the best and worst system responses. For parts 1 and 2, the subjects generally were impressed with the system responses for several questions, one of which was vague (*"Could I look at when crimes happen for each neighborhood"*), another was a complex query (*"If I was walking to the EL, would there be any areas that are particularly dangerous to walk through due to theft or battery"*) and also an oddly worded AR (*"I would just like to see the total amount of crimes that happened divided by the three main areas, university, River North and Near West Side"*. Aside from these, for another AR asking about data aggregated by location, the system produced the results in descending order, which the subjects noted made it easy to identify the locations with the most crime. Next, summarizing negative feedback, a common complaint was that the system took the AR too literally for the AR *"Can you show the location type for the crimes that occur between noon and 6 and 6 and midnight"*, producing a visualization with one time interval from noon to midnight in a single visualization. Additionally, for heat-maps produced by the system, some of the subjects noted a lack of a legend for the different colors, hence making it harder to interpret by the users.

Moving on to part 3, we found that the subjects liked the windows management features overall because they facilitated managing many visualizations on the screen at once. In particular users noted minimizing visualizations to the upper right corner and moving partially covered visualizations to the foreground. The subjects disliked the response for the AR dealing with moving (i.e., *"Can you move [POINT TO VIS] it"*) because the destination position is random and cannot be specified.

Finally, in part 4, several limitations were identified. First, due to a small dialogue corpus, the system had limited vocabulary (e.g., it could understand *"close the graph"* however *"remove the graph"* failed). The feedback also pointed out a few instances in which the system would become confused when the user is pointing to a screen location consisting of multiple visualizations overlayed on top of each other. Also they did not see the purpose of generating empty visualizations in the absence of relevant data, rather than avoiding producing such graphs. In terms of positive feedback for part 4, subjects generally agreed that the system responses were quickly generated and also noted that the system produced expected responses to more direct ARs (which supports our hypothesis for question 3 results in the rating analysis discussion earlier).

Conclusion

In this paper, we presented our new intelligent assistant for exploring visualizations, implemented as a dialogue system and modeled on our own multimodal dialogue corpus. Subjects interacted (i.e., by using speech and gestures) with a large screen display. An extensive evaluation of the system with 20 users was also done, and indicated overall that the system did well on response time; also, users were impressed in some cases when the system was able to successfully produce visualizations for oddly worded, vague, and complex queries. The results also noted that the system can take ARs too literally and produce unsatisfactory visualizations.

In terms of future work, we plan to focus on building a more sophisticated machine learning based approach to detecting and resolving referring expressions rather than

keyword matching as being done currently. We also plan to study the findings of the evaluation study to determine how we could improve the system implementation for future users.

Acknowledgments

This work was supported, initially, by NSF award IIS-1445751; and currently, by NSF award CNS-1625941, and by a UIC University Scholar award to Barbara Di Eugenio.

References

Bordes, A.; Boureau, Y.; and Weston, J. 2017. Learning end-to-end goal-oriented dialog. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Budzianowski, P.; Wen, T.-H.; Tseng, B.-H.; Casanueva, I.; Ultes, S.; Ramadan, O.; and Gasic, M. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 5016–5026.

Choi, J. D. 2014. *Optimization of Natural Language Processing components for Robustness and Scalability*. Ph.D. Dissertation, University of Colorado Boulder.

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

Gao, T.; Dontcheva, M.; Adar, E.; Liu, Z.; and Karahalios, K. G. 2015. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, 489–500. ACM.

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

Hemphill, C. T.; Godfrey, J. J.; and Doddington, G. R. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Hoque, E.; Setlur, V.; Tory, M.; and Dykeman, I. 2017. Applying pragmatics principles for interaction with visual analytics. *IEEE Transactions on Visualization and Computer Graphics* 24(1):309–318.

Kim, S.; D'Haro, L. F.; Banchs, R. E.; Williams, J. D.; and Henderson, M. 2017. The fourth dialog state tracking challenge. In *Dialogues with Social Robots*. Springer. 435–449.

Kipper, K.; Korhonen, A.; Ryant, N.; and Palmer, M. 2008. A large-scale classification of english verbs. *Language Resources and Evaluation* 42(1):21–40.

Kumar, A.; Aurisano, J.; Di Eugenio, B.; Johnson, A.; Gonzalez, A.; and Leigh, J. 2016. Towards a dialogue system that supports rich visualizations of data. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 304–309.

Kumar, A.; Di Eugenio, B.; Aurisano, J.; Johnson, A.; Alsaiari, A.; Flowers, N.; Gonzalez, A.; and Leigh, J. 2017. Towards multimodal coreference resolution for exploratory data visualization dialogue: Context-based annotation and gesture identification. *SEMDIAL 2017 SaarDial* 48.

Lowe, R.; Pow, N.; Serban, I.; and Pineau, J. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 285–294. Prague, Czech Republic: Association for Computational Linguistics.

Palmer, M.; Gildea, D.; and Kingsbury, P. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics* 31(1):71–105.

Reithinger, N.; Fedeler, D.; Kumar, A.; Lauer, C.; Pecourt, E.; and Romary, L. 2005. Miamm—a multimodal dialogue system using haptics. In *Advances in Natural Multimodal Dialogue Systems*. Springer. 307–332.

Setlur, V.; Battersby, S. E.; Tory, M.; Gossweiler, R.; and Chang, A. X. 2016. Eviza: A natural language interface for visual analysis. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 365–377. ACM.

Sun, Y.; Leigh, J.; Johnson, A.; and Lee, S. 2010. Articulate: A semi-automated model for translating natural language queries into meaningful visualizations. In *International Symposium on Smart Graphics*, 184–195. Springer.

Wen, T.-H.; Vandyke, D.; Mrkšić, N.; Gašić, M.; Rojas-Barahona, L. M.; Su, P.-H.; Ultes, S.; and Young, S. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 438–449. Valencia, Spain: Association for Computational Linguistics.

Williams, J.; Raux, A.; Ramachandran, D.; and Black, A. 2013. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, 404–413.

Yu, B., and Silva, C. T. 2019. Flowsense: A natural language interface for visual data exploration within a dataflow system. *IEEE transactions on Visualization and Computer Graphics*.