DT2VIS: A Focus+Context Answer Generation System to Facilitate Visual Exploration of Tabular Data

Qi Jiang, Guodao Sun, Yue Dong, and Ronghua Liang, *Zhejiang University of Technology, Hangzhou, Zhejiang, 310014, China*

The visual analysis dialog system utilizing natural language interface is emerging as a promising data analysis tool. However, previous work mostly focused on accurately understanding the query intention of a user but not on generating answers and inducing explorations. A focus+context *answer generation approach, which allows users to obtain insight and contextual information simultaneously, is proposed in this work to address the incomplete user query (i.e., input query cannot reflect all possible intentions of the user). A query recommendation algorithm, which applies the historical query information of a user to recommend a follow-up query, is also designed and implemented to provide an in-depth exploration. These ideas are implemented in a system called DT2VIS. Specific cases of utilizing DT2VIS are also provided to analyze data. Finally, the results show that DT2VIS could help users easily and efficiently reach their analysis goals in a comparative study.*

Data analysis has become an important decision-making proof for decision makers with its distinctive ability to mine the value of data. Especially with the rapid development of hardware interfaces and the emergence of massive amounts of data, visual analytics has been widely applied in many applications.[9–11] Visualization facilitates data analysis through vivid graphics and allows users to explore additional data simultaneously. Therefore, an increasing number of researchers and commercial organizations have developed certain visual data analysis tools. However, these tools and systems mostly have complex interfaces and may bring steep learning curves to novices.

The application of natural language interfaces (NLIs) to visualization may help these novices in exploring data, allowing them to ask high-level questions in natural language (NL) directly without frequent interactions with system interface. Therefore, numerous researchers have been applying NLIs into their work.[2,4,6,12] Ambiguity and underspecification in NL are prevalent in any form of communication, which make cooperating NLIs into data visualization a challenging task. How to capture users' query intentions as accurately as possible is a common challenge with respect to the combination of natural language interface and data visualization (NLI4DV). Previous NLI4DV work mostly focused on addressing the ambiguities and underspecification of NL to get users' intentions in NL.[2,4] However, certain potential intentions cannot be obtained directly from NL. For example, if a user asks "What is the trend of consumption in the past year?," we can get an intention of trend task about consumption directly from NL, but potential intentions are not easy to discover (e.g., there is a high possibility that the user also wants to know the month with the largest consumption change). How to prevent users from performing additional queries and interactions to achieve these potential intentions remains a challenge for us. In this work, a *focus+context* answer generation approach is proposed to discover these potential intentions. The *focus+context* answers we generate provide more information and insights than the ones from previous work. In this work, the query tasks of users are divided into two categories according to their characteristics. One is the *focus* type task indicating that the intention of the user is to get some specific values (e.g., extremum, average, and sum). The other one is the *context* type task, which indicates that the intention of the user is to query for some data characteristics (e.g., data distribution, trend direction, and correlation between different attributes). When users ask only one *focus* or *context* type question,

0272-1716 © 2021 IEEE
Digital Object Identifier 10.1109/MCG.2021.3097326
Date of publication 14 July 2021; date of current version 8 September 2021.

the system will automatically take the other into account in answer generation, thereby allowing users to obtain insight and contextual details simultaneously.

In addition, users sometimes may perform multiple rounds of queries to achieve a complex goal. How to guide users to query and explore data is also a great challenge in NLI4DV work. As far as we know, no NLI4DV work pays attention to this issue. Logical connection may exist among users' queries, for example, users are likely to ask a *find value* question after asking a *distribution* question. Thus, how to determine this connection information and apply it to guide users to perform in-depth exploration remains questionable. At present, little work is focused on the aforementioned points.[3,7] Most work only predicted a follow-up query by associating attribute fields, thus failing to explore from the level of query semantics. In this work, a query recommendation algorithm is also proposed to guide users by modeling the relationship among nine different analysis tasks, and feedback from users on the recommendation results is utilized to refine the model.

A dialog analysis system, namely DT2VIS, is developed in this work to help users explore tabular data. This system mainly comprises a query parsing engine (QPE), an answer generation engine (AGE), and a query recommendation engine. Moreover, the system obtains query details by parsing the input NL and transmits the details to the AGE to generate a *focus +context* answer. The exploration of a user is guided by generated recommendations from the recommendation engine. This system also provides an exploration history interface to help users review the analysis process. Specific cases of utilizing DT2VIS to analyze the data are provided. Finally, we conducted a comparative user study against NL4DV, and recruited ten participants to evaluate DT2VIS. DT2VIS is compared with NL4DV[6] by setting the same analysis tasks. The participants indicated that DT2VIS could provide visual help in the fast completion of analysis tasks with additional details. In addition, DT2VIS is provided as an open source project,[a] which aims to advance the research of visual analysis dialog utilizing NLIs.

The following contributions are presented in accordance with the building and evaluation of the DT2VIS system.

- A *focus+context* answer generation approach with automatic text and visual answer for nine tasks to help explore data more accurately and efficiently.
- A recommendation algorithm that utilizes user's historical exploration information to suggest follow-up queries.
- A runnable visual analysis dialog system named DT2VIS to explore and analyze the tabular data.

[a] https://github.com/jiangqicd/DT2VIS

RELATED WORK

This section elaborates on the related work. The current research is grounded in three main areas: 1) NLIs for data visualization; 2) automatic query answering; and 3) future interactions prediction.

NLIs for Data Visualization

The NLI allows users to communicate with complex visualization systems flexibly. Narechania *et al.*[6] provided a high-level application programming interface, which takes tabular datasets and an NL query as input, and returns a list of Vega-Lite specifications. Gao *et al.*[2] modeled ambiguity and used algorithms for interactive disambiguation in the process of transforming NL query into visualization.

The above work mainly focused on the ambiguity and understandardization in NL, but did not consider the intention of the follow-up query. By contrast, there are some works that provided users with a contextual query environment. Hoque *et al.*[4] applied the principles of pragmatics to visual analysis and allowed users to perform a contextual query; he used current visualization as a starting point, which allows users to query in the context of a given visualization, thereby strengthening the ability of users to modify and update queries continuously.

NLI4DV is an emerging and promising research field, but it also faces some challenges. For example, how to obtain users' query intentions accurately. In fact, there are some intentions not reflected in query NL. This work is to capture as much potential intentions, which are outside of query NL as possible.

Automatic Query Answering

Automatic query answering has a wide range of applications in different fields (e.g., document collection and tabular data analysis). Tabular data are the most common data type in our daily life, hence our work aims at automatic query answering for the tabular data analysis. Currently, certain work on TableQA gave answers to the input query in plain text format. Pasupat *et al.*[8] performed compositional semantic analysis using question–answer pairs as supervision to answer the complex questions on semistructured tables. In addition, there are some other investigations that provided visual charts to answer the input questions and used text to assist in the explanation. Kim *et al.*[5] developed an automatic TableQA pipeline that extracts data and visual elements from the input Vega-Lite chart to generate a visual explanation for how the answer was obtained. The generated visual explanation is superior to manual explanation considering transparency, usefulness, and credibility.

However, previous question answering systems only gave direct answers to questions, such answers are difficult to convince users and aid users with follow-up exploration. These limitations can be improved under a *focus+context* scenario. For example, when a user asks a *focus* type question (*what is the average of consumption in the past year?*), the system can enhance the user's confidence in the average value by providing the distribution of consumption. When a user asks a *context* type question (*what is the trend of consumption in the past year?*), we can help the user exploration by adding some *focus* (e.g., the month with the biggest change in consumption).

Future Interactions Prediction

The interactions in the visualization system have different meanings, ranging from low-level interaction events (e.g., simple clicks) to complicated high-level tasks (e.g., query and filtering). Most predictive working models in visualization system are based on the visual attributes of the visualization or the data attributes. Gotz *et al.*[3] utilized the behavior of users to obtain their implicit intentions to provide effective recommendations. This method is called a behavior-driven visual recommendation. Ottley *et al.*[7] used hidden Markov models to infer users' attention and cognitive bias, respectively, resulting in anticipating future clicks and analyzing user exploration bias. The prediction based on visual attributes is substantially close to the design principles of the visualization system and can effectively understand the points of interest of users to make humanized recommendations.

In this work, we modeled the relationship between different analysis tasks and proposed a recommendation algorithm, which applies the historical query information of a user to recommend a follow-up query.

DT2VIS OVERVIEW

The design goals of DT2VIS are determined in this section. The workflow of the visual analysis dialog system and the user interface of DT2VIS are also introduced.

Design Goals

Four core design goals are introduced to build DT2VIS effectively.

G1 minimize learning curve. DT2VIS should enable the masses without experience in data analysis field to utilize the system to analyze data conveniently and efficiently. Regarding system design, this consideration is translated into providing high-level functions for interpreting NL query and automatically generating answers.

G2 support *focus* + *context* answer. Input query cannot reflect all possible intentions of the user. Hence, DT2VIS should generate the answer, which can cover the additional intentions of user as much as possible.

G3 support iterative user exploration. Users often need to have multiple rounds of dialog with the system to achieve an analysis goal. DT2VIS should provide certain guidance to promote user exploration. In this work, this consideration is transformed into a query recommendation algorithm.

G4 support interface for reviewing exploration history. Invalid analysis tasks may appear during the exploration process. Therefore, DT2VIS should provide a history interface, which can allow users to flexibly select a new starting point for exploration and review their exploration.

Analysis Workflow

Figure 2 shows an analysis workflow of the system. The QPE parses the relevant information (e.g., attributes, filter, and task) from the input query. Next, the system extracts specific data segments from the data space according to the parsed attributes and filter information, and answer generation templates corresponding to the task are also collected. These data segments and templates will be transported to the AGE to generate answers. Simultaneously, the parsed task and attributes information are also sent to the query recommendation engine. Then, the system updates the transition matrix based on user feedback on the previous recommended query and computes the recommended queries from the new task transition matrix. Finally, these generated answers and recommended queries are presented to the user together.

User Interface

The user interface of DT2VIS is shown in Figure 1. The user can select analysis dataset in the data detail view [see Figure 1(A)] and enter the query in the input box [see Figure 1(B)] freely. An input autocomplete function is designed to improve user experience [see Figure 3]. The answers of the system are presented in the answer view [see Figure 1(C1), (C2)]. In addition, the user can select the recommended query in the recommended view [see Figure 1(D)] and review their exploration in the query history view [see Figure 1(E)].

DT2VIS DESIGN AND IMPLEMENTATION

This section provides details of the design and implementation of DT2VIS, and highlights key functions. Query interpretation, *focus+context* answer generation, and query recommendation algorithm are mainly presented here. Due to lack of relevant training dataset publicly available and the hard labor of manual annotation, we

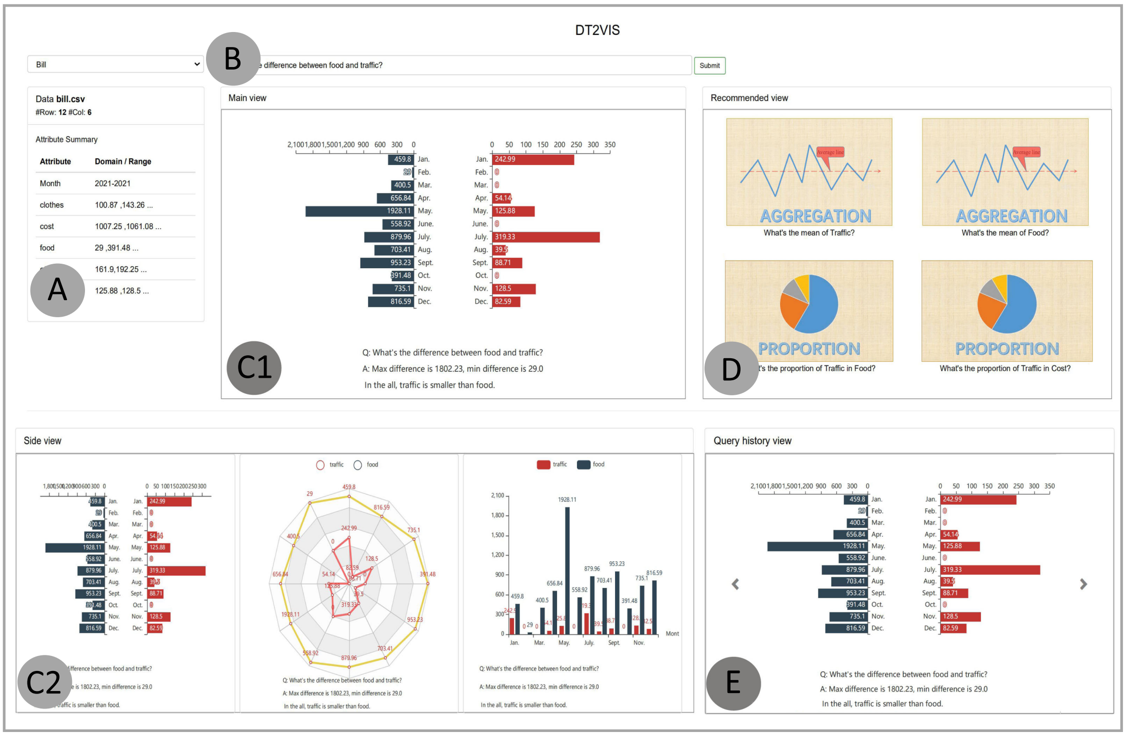

FIGURE 1. User interface of DT2VIS system includes five views. (A) User can select analysis dataset and overview the dataset in a data detail view. (B) Query sentences are typed in a query input view. (C1) is a main answer view, where user can view the details of the answer, (C2) is side answer view which provides other alternative answers. (D) is recommendation view for the follow-up query recommended, and (E) is query history view which shows the exploration history of the user.

adopt a pure template-based approach in the query interpretation and answer generation. In addition, an example (see Figure 4) about covid19-state-data[b] is provided to help understand how each function works.

Query Interpretation

In the query interpretation section, QPE applies a series of NLP technologies to parse the input query sentence and extract the information (e.g., task, value, attribute, and filter) required by the subsequent engines. The query interpretation process consists of the following five steps. 1) Word segmentation and POS tags identification. QPE performs word segmentation on the inputted query and makes identification of the POS tags for each token using Stanford's CoreNLP (Step 1 in Figure 5). 2) Stop words removal and N-grams generation. QPE removes all stop words (except prepositions and conjunctions) and generates N-grams, which involve all possible phrases from the trimmed query words (Step 2 in Figure 5). 3) Entity detection from N-grams. In this step, QPE needs to identify those N-grams that have some relevance to the dataset attribute and task definition by computing the semantic similarity between N-grams and the lexicon of attributes, values, and task (Step 3 in Figure 5). A combination of cosine syntax similarity and Wu-Palmer semantic similarity is created to determine the absence or presence of special meaning in the phrases. Notably, *Simcos(i, j)* represents the cosine similarity between N-gram *i* and tokenized lexical entity *j*, and *Simwup(i, j)* represents the Wu–Palmer semantic similarity between N-gram *i* and tokenized lexical entity *j*. The final similarity between the N-gram and lexicon entries is defined as follows:

Sim(i, j)=MAX(*Simcos(i, j)*, *Simwup(i, j)*).

A minimum similarity threshold $\tau = 0.8$ is set, i.e., when the similarity between N-gram *i* and a tokenized lexical entity *j* is larger than 0.8, N-gram *i* is considered to be the preferred phrase and corresponds to the tokenized lexical entity *j*. 4) Dependence parsing. As discussed above, the meaningful attributes, values, and tasks obtained are insufficient to infer the intention of users. Understanding the relationship between the word segmentations is necessary to identify the

[b] https://www.kaggle.com/nightranger77/covid19-state-data

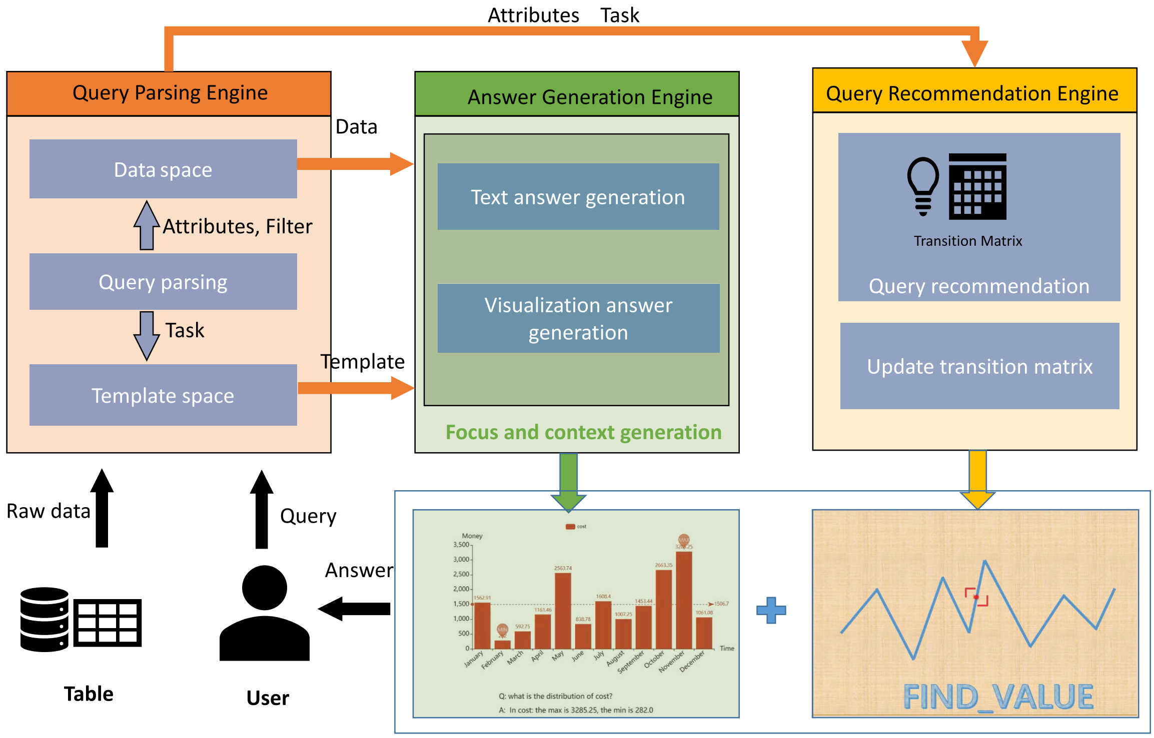

FIGURE 2. The architecture of DT2VIS. The arrows indicate the flow of information among different engines. Query Parsing Engine parses the input query sentence and extracts the information (e.g., task, value, attribute, and filter) required by the subsequent engines. Answer Generation Engine generates both the text and visualization answers under a *focus+context* scenario and Query Recommendation Engine generates recommendations of queries through a task transition matrix based on user behavior.

intention of the user accurately. Thus, QPE creates a dependence tree using the dependence parser of Core NLP (Step 4 in Figure 5). 5) Query intention identification. We designed some specific extraction rules for each task. Finally, QPE retrieves details of user intention from the dependence tree by these predefined templates (Step 5 in Figure 5), which were defined based on the query patterns and examples from prior NLIs,[2,4] and the questions were collected by Amar *et al.*[1]

In our example (Figure 4-2), intention (Task: Distribution; Attribute: Deaths) can be parsed from the question (*What is the distribution of Deaths?*)

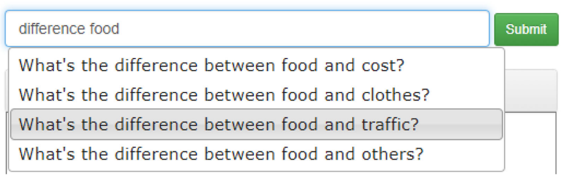

FIGURE 3. Autocompletion function, users only need to enter a few keywords, the system can automatically complete the query.

through above five steps. In this work, nine types of analysis tasks, including *Extreme*, *Correlation*, *Comparison*, *Trend*, *Find value*, *Rank*, *Distribution*, *Proportion*, and *Aggregation* are adopted by referring to the classification of Amar *et al.*[1] These selected analysis tasks are commonly used and well-established ones, and their own application scenarios are shown in Table 1 (Column of Derived Value).

Focus and Context Answer Generation

In this section, AGE generates *focus* and *context* answers based on the information extracted by QPE. Previous question answering systems generally utilized a single text or single visual form in response to queries. Considering that text and visualization have their own advantages, we adopt a combination of text and visualization as our answer form. Thereby, the system can not only provide a direct and unambiguous textual answer, but also connect the low-level data with users by a visual channel.

First, AGE extracts specific data segments from the low-level data based on the attributes and filter provided by QPE. As shown in Figure 4-3, system extracts Deaths data based on the keywords "Deaths." Second,

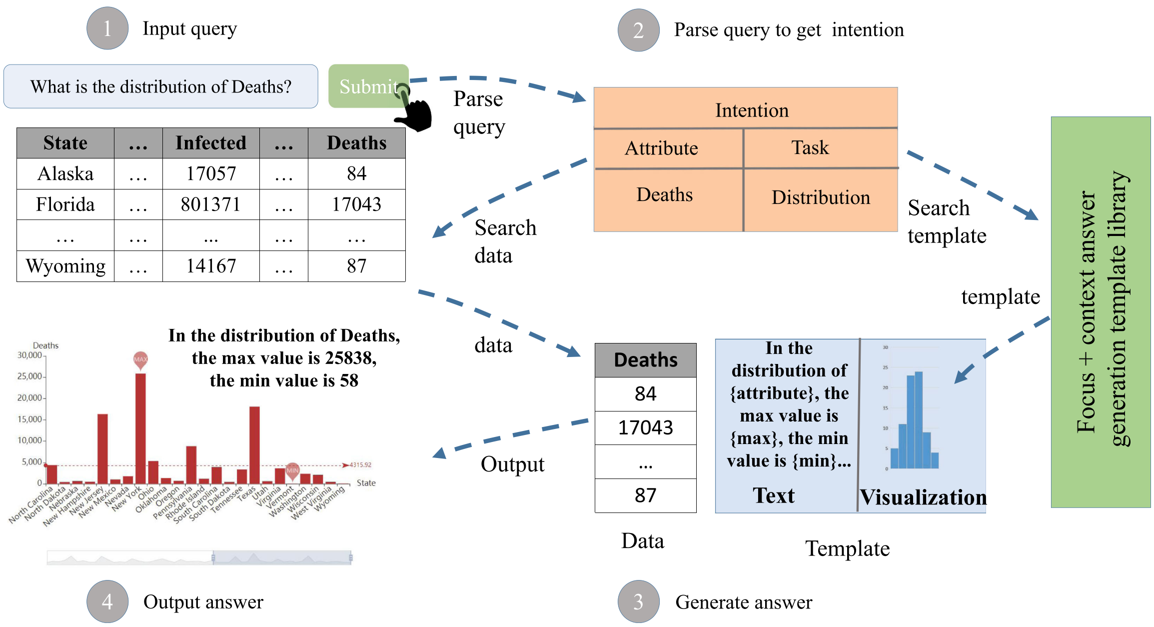

FIGURE 4. An example about the number of deaths, infections, and tests in the U.S. states during Covid-19.

these data will be transmitted to the textual answer generator and the visual answer generator simultaneously. Different from previous NLI4DV work, in order to capture the potential intentions outside of NL as much as possible, we propose a *focus+context* answer generation approach, whose design space is shown in Table 1.

Nine analysis tasks are divided into two categories (*focus* or *context*) according to their characteristics in Table 1 (Column of Task type). We take context information into consideration to answers generation for focus type tasks by adding contextual information (e.g., overview of tabular data, data distribution). Similarly,

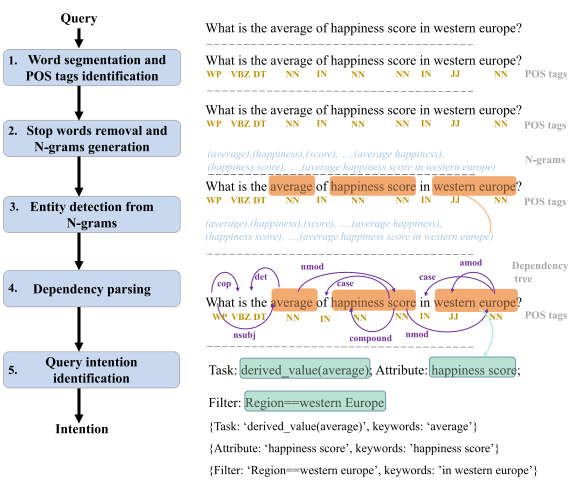

FIGURE 5. Query interpretation workflow.

we take the focus information into consideration to answers generation for context type tasks by adding certain insight information (e.g., extremum and average), which will be displayed through common auxiliary tools (e.g., text annotation, auxiliary line, and color highlight). This is why we name it a focus+context answer generation approach. We embed these design ideas into predefined templates (Focus+content answer generation template library in Figure 4). After referring to some prior work on NLI4DV,[2,4] we incorporated seven visualization forms into our visualization answer templates in Table 1 (column of visualization) and predefined text answer templates (Table 2). Our system can deal with queries involving temporal dimension. When the time span is small (e.g., one week), we display the full information. When the time span is large (e.g., one month), partial information is displayed. We provide a zoom slider to allow users to freely select time intervals of interest. After generator obtains the data, it will search the corresponding template from template library, then process the data and fill results in the template to generate answer (Figure 4-4).

Query Recommendation Algorithm

Users sometimes will perform multiple rounds of queries to achieve a complex goal. How to guide users to query and explore the data is the challenge to be addressed in this section. With regards to this, we propose a query recommendation algorithm (Algorithm 1), which applies the historical query information of a user to recommend follow-up query

TABLE 1. Focus + context answer generation design space, which includes Analysis task, Derived value, Task type, Visualization selected, Focus + context generation approach, and Auxiliary tools used.

Analysis task	Derived value	Task type		Visualization							Focus+context generation	Auxiliary tools
		Focus	Context	Pie	Bar	Line	Radar	Box plot	Scatter	Parallel		
Find value	lookup value	✓			✓					✓	Add contextual information (e.g., Overview of tabular data)	Details
Extreme	maximum minimum	✓				✓		✓			Add contextual information (e.g., Data distribution)	Details
Aggregation	-	✓			✓			✓			Add contextual information (e.g., Deviation from the average)	Details
Proportion	percentage		✓	✓	✓						Add maximum and minimum proportion	Text Graphic
Comparison	difference value		✓		✓		✓				Add maximum and minimum difference	Text Graphic
Distribution	-		✓		✓						Add some statistics (e.g., max, min, mean.)	Text,Line Graphic
Trend	increasing decreasing		✓		✓	✓		✓			Add maximum increase and decrease	Text,Line Graphic
Rank	sort		✓		✓						Add maximum and minimum	Text Graphic Color
Correlation	correlation coefficient		✓						✓		Add maximum and minimum correlation coefficient	Text Graphic Color

TABLE 2. Text answer generation template. Notably, *filter* represents the filter requirements (e.g., before or after 2015) of user, *attribute* represents the attributes involved in the analysis task, *trend* indicates trends (e.g., increase or decrease).

Analysis task	Default Template
Find value	"The value of *attribute* in *filter* is *value*, and the max of *attribute* is *max_value*, the min of *attribute* is *min_value*"
Extreme	"In *filter*, the max/min of *attribute* is *max_value/min_value*"
Aggregation	"The mean/sum of *attribute* in *filter* is *mean/sum*, and the max of *attribute* is *max_value*, the min of *attribute* is *min_value*."
Proportion	"In the proportion of *attribute_1* in *attribute_2* in *filter*, the max proportion is *max_proportion*, the min proportion is *min_proportion*.The overall trend of proportion is *trend*."
Comparison	"In the comparison between *attribute_1* and *attribute_2*, the max difference is *max_difference*, the min difference is *min_difference*.In the all, *attribute_1* is greater or less than *attribute_2*."
Distribution	"In the distribution of *attribute* in *filter*, the max value is *max_value*, the min value is *min_value*, the overall trend of *attribute* is *trend*."
Trend	"In the trend of *attribute* in *filter*, the count of increase is *up_count*, the count of decrease is *down_count*, the overall trend is *trend*."
Rank	"In the rank of *attribute* in *filter*, and the max of *attribute* is *max_value*, the min of *attribute* is *min_value*."
Correlation	"In the correlation of (*attribute_1*,*attribute_2*...) in *filter*, the most relevant is *attribute_i* and *attribute_j*, The least relevant is *attribute_m* and *attribute_n*."

$$TTM = \begin{bmatrix} P(t_0,t_0) & P(t_0,t_1) & \cdots & P(t_0,t_8) \\ P(t_1,t_0) & P(t_1,t_1) & \cdots & P(t_1,t_8) \\ \vdots & \vdots & \ddots & \vdots \\ P(t_8,t_0) & P(t_8,t_1) & \cdots & P(t_8,t_8) \end{bmatrix} \quad (1)$$

$$\forall\, t_j \in Task, \sum_{i=0}^{8} P(t_j,t_i) = 1 \quad (2)$$

$$P(task_n, task_{adopt}) = P(task_n, task_{adopt}) + \zeta \quad (3)$$

$$P(task_n, task_i) = P(task_n, task_i) - \beta_i * \zeta. \quad (4)$$

A 9 × 9 *Task Transition Matrix* [*TTM* as shown in (1)] is created in the task recommendation engine to model the relationship among different tasks. Notably, *P(i, j)* represents the probability of task *j* following task *i* and the sum of all the cells in each row of the *TTM* is 1, as shown in (2). Our algorithm includes computing recommended query and updating TTM. In the

Algorithm 1. Framework of query recommendation.

Input: Current query, Q_n, Q_n includes the current query task T_n and the current query attributes A_n; Feedback of user to recommended query, $Feedback$; $Feedback$ consists of response status(adopt or not adopt) and $Q_{adopt}(T_{adopt}, A_{adopt})$ adopted;
Output: The follow up recommended query, $Q_{n+1}(T_{n+1}, A_{n+1})$;
// computing recommended query.
Initialize matrix TTM
Given T_n, compute T_{n+1}
Given A_n, compute A_{n+1}
Compose T_{n+1} and A_{n+1} to generate Q_{n+1}
// updating TTM, T_{n-1} is last query.
for all $Feedback$ **do**
    if response status is $adopt$ **then**
        Update the $P(T_{n-1}, T_{adopt})$ based on Eq. (3)
        for all T_{other} **do**
            Update the $P(T_{n-1}, T_{other})$ based on Eq. (4)
        end for
    else
        Perform the reverse operation in status of $adopt$
    end if
end for
return Q_{n+1}

computing recommended query section, algorithm will first select the top two tasks in the ranking of the probability between the current query task and other tasks as the recommended tasks. In other words, the two tasks that are closest to the current query task are selected as recommended tasks. Similarly, the attribute most relevant to the current query attribute is selected as the recommended attribute. The Pearson coefficient is used to measure the correlation between different attributes. Note that the current query attribute may still be the user's concern, so it will also be used as a recommended attribute. These tasks and their corresponding attributes are the keywords in the generated queries. We reuse the autocompletion function, which defines the rules for generating queries, and these keywords are filled into these predefined rule templates to generate recommended queries. In addition, certain user query data ($\sim$300 records) are collected to initialize the *Task Transition Matrix* to avoid a cold boot. Due to cognitive differences among different users and their individualized exploration habits, there is no guarantee that the transition matrix can be suitable for all users. Hence, we added a feedback mechanism to update the transition matrix in real time through the user feedback on our recommendation. If users adopt the query recommended, the system will reward the probability between the previous query and the accepted query task. For example, if current query task is a type of *trend* and the user adopts the recommended task *find value*, then the algorithm increases the transition probability of the task from *trend* to *find value*. Equations (3) and (4) show that the increased probability value ζ is siphoned from other tasks, and the system will allocate reduced probability value of each task according to β. β is related to the transition probability, a high transition probability indicates a low β. Similarly, when users do not adopt the recommended query, the algorithm performs the opposite operation. That is to punish the probability between the previous query and the recommended query task. By this way, the *Task Transition Matrix* can gradually fit the users based on the long-term exploration.

CASE STUDY

This section introduces two specific cases of using DT2VIS. The first case uses DT2VIS to help users verify the answers given by the system, and the other one uses the proposed system to help users explore information from data.

Verify the System's Answer

First, DT2VIS is utilized to help users verify the answers given by the system. Certain outliers are found in the analyzed data due to many human and natural factors. If the analyst examines data directly without realizing these outliers in advance, then such an approach may inevitably affect the final analysis conclusion. In this section, let us take following scenario as an example. When a data typist enters the income situation of a certain type of customer group, the typist accidentally enters a wrong value of \$89,400 rather than the right one (\$8940). We invited one of our colleagues as an analyst to analyze this data and recorded his exploration process. Some previous dialog analysis systems usually only give users conclusive answers and cannot help avoid such dilemmas. In our system, although the conclusive text answer provided was "*Parker has the highest income: \$89,400*," the visual answer interface (see Figure 6) provided by the system reveals that the income value of *Parker* \$89,400 is an outlier (**G2**). The analyst may still suffer certain errors regarding the answer of the system after removing this outlier and continuing the analysis task. Figure 7 shows that the analyst re-enters the same question as before and after removing the outlier. Two customers (*Abbott* and *Sebastian*) have the same highest income, but the system ignores the latter. However, the visual answer interface shows that the analyst can still observe the presence of another customer *Sebastian* who also has the highest income. The *focus+context* answer can still enhance the credibility of the system answer and inspire the exploration of users in the absence of

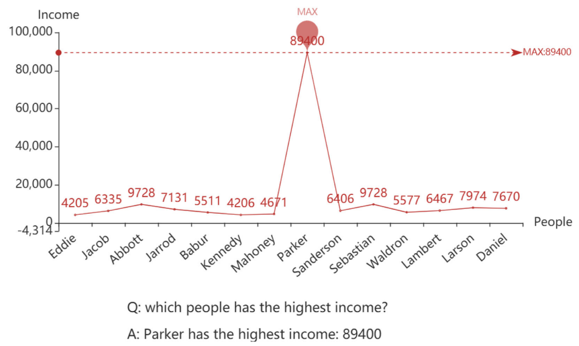

FIGURE 6. "The result of system's response to user input when there is an outlier in the data."

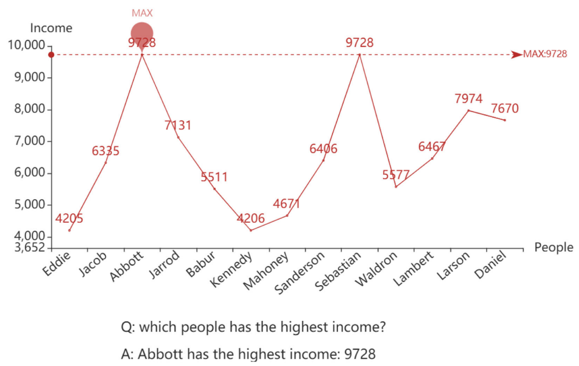

FIGURE 7. "The result of the system's response to the user's query when the system's answer is error."

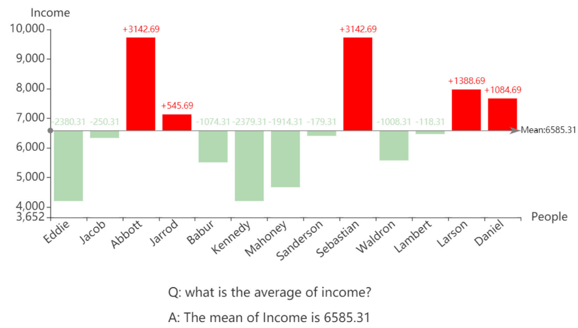

FIGURE 8. "The result of the *focus+context* answer respond to the user's query."

problems with the system answer and the presence of clean data. Figure 8 shows the user input, "*what is the average of income?*" Thus, the mean of income and the deviation between each customer income and the average value are also provided.

Data Story Mining

The second experiment is conducted on a consumption bill data of Wang to validate the method and visual analysis workflow. The dataset mainly includes monthly total consumption cost (*cost*), meal expenses (*food*), clothing expenses (*clothes*), transportation expenses (*traffic*), and other expenses (*others*). Wang found that he had spent more than expected in the past year. Therefore, he wanted to determine the reason for his high spending. First, Wang checked the cost distribution of each month in the past year (Figure 9-A) (**G1**). Finding that the value of cost in November was the highest is easy by using the graphic element annotation. He then supplemented the recommended query to investigate the value of all consumptions in November according to the analysis task Find value (*What is the value of cost, food?*) recommended by the system (Figure 9-B) (**G3**). Wang found that other expenses (*others*) played an important role in all consumptions. Then, he recollected that he bought some expensive electronic products during the shopping festival in November, which indicated that November is the month with the highest consumption level. Moreover, he noticed that he spent a considerable amount of money on meal expenses. He further looked at the trend of food throughout the year to verify its normality. The trend chart (see Figure 9-D) revealed that food reached the valley in February because Wang went home on winter vacation during that time. In addition, the value of food and the magnitude of ups and downs reached a peak in May. Wang quickly responded that his birthday was in May, and he held a birthday party outside with his friends. Finally, Wang reviewed the query history view and summarized the findings of this exploration (**G4**). The *focus+context* answers and the recommended analysis task can help users explore data efficiently and conveniently.

USER STUDY

In this section, we conducted a formal user study to evaluate the effectiveness of DT2VIS. We designed an experiment that introduces DT2VIS to the participants and assigns analytical tasks to be solved in the system.

Study Design

Due to the impact of COVID-19, we ended up recruiting 10 participants (six males, four females, aged between 20–30) who studied in the field of computer science. First, we conducted about 25 min training for each participant to introduce our study design. We selected *World Happiness Report*[c] as the test dataset. Our work is partially inspired by NL4DV's work, DT2VIS and NL4DV apply NLIs for data exploration. As far as

[c] https://www.kaggle.com/unsdsn/world-happiness

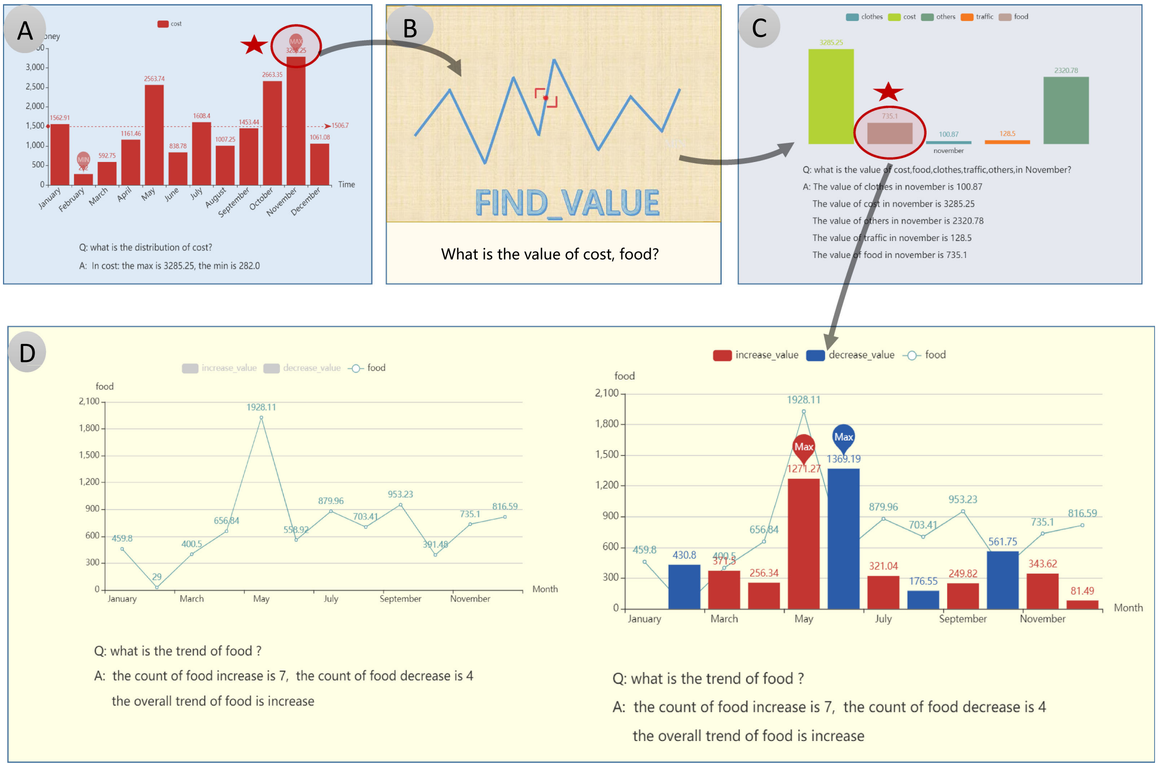

FIGURE 9. "(A) The journey of mining stories from data. The answer in response to query (*what is the distribution of cost?*). (B) Queries recommended by the system. (C) The answer in response to query (*what is the value of cost, food, clothes, traffic, and others in November?*). (D) The answer in response to query (*what is the trend of food?*)."

we know, NL4DV is the most recent work in the field of NLI4DV and it has open source code and web interface. Therefore, we chose NL4DV as the comparison of our work and required participants to utilize DT2VIS and NL4DV to answer seven questions, which are common in the data analysis. After the experiment, we collected feedback from participants on the recommendation query as materials for evaluating the recommendation algorithm. Participants were required to score recommended query in the range of 0–5. In addition, we communicated with six participants to collect more feedback. We also counted answer accuracy and time cost, which will be used as materials for our analysis of study results. The analysis tasks we designed are as follows: **(T1)** *What is the average of Happiness Score in Western Europe?* **(T2)** *Which regions of Western Europe and North America have a higher happiness score?* **(T3)** *Which two countries have the biggest difference in happiness scores? How much is the difference?* **(T4)** *What is the health of the country ranked 15th in happiness score?* **(T5)** *In Southeastern Asia, which country has the highest proportion of economy in happiness score?* **(T6)** *Are the distribution of happiness scores in Southeastern Asia fluctuating?* **(T7)** *Which health and economy have more influence on happiness score?*

Result

Figure 10(a) shows the distribution of answer accuracy, Figure 10(b) shows the distribution of the completion time of each task in the user study, and Figure 10(c) shows users ratings for recommended query. It can be observed that most participants utilizing DT2VIS will have the higher answer accuracy and less time cost than the ones utilizing NL4DV, and participants generally hold a positive attitude toward the recommended query results. The answer accuracy and time cost between different analysis tasks are also different. We found that users have poor performance in compound tasks (e.g., T4 and T7). After reading the comments in the feedback, we believe this may be due to the fact that many participants are first-time users of these visual dialog analysis systems with NLI, and cannot accurately and effectively convert complex analysis tasks into simple subsets which the system can handle.

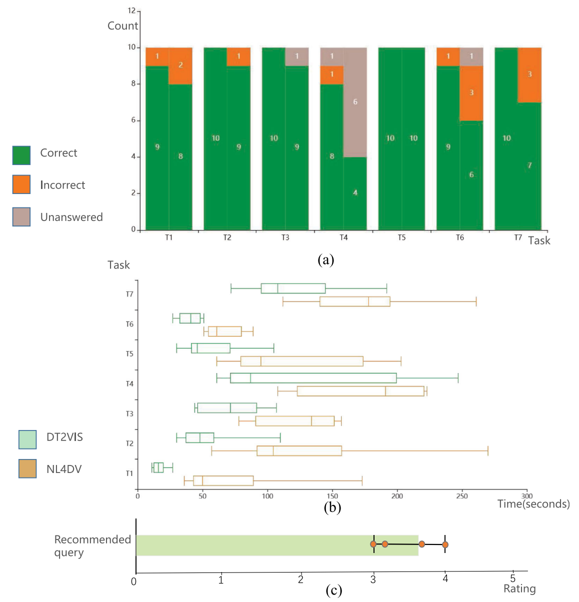

FIGURE 10. (a) Distribution of participant answers to each of the user study tasks, DT2VIS is on the left, and NL4DV is on the right. (b) Box plot of completion time for each of the user study tasks. (c) Users' feedback on recommended queries.

In our further communication with the six participants, they all preferred DT2VIS to NL4DV. The autocompletion function of DT2VIS facilitates them to enter natural language queries. Five of the six participants were interested in the *focus+context* answers in DT2VIS and believed that the visualizations of DT2VIS are more vivid than NL4DV. Although the queries recommended by DT2VIS may not be the most wanted component in our system, they still could find certain additional information through that. Finally, they deemed that DT2VIS and NL4DV can only query specific query sentences, which may affect their experience.

DISCUSSION AND FUTURE WORK

The participants in the user study indicated that applying NLIs to visual dialog analysis systems is a promising work. Such systems allow users to communicate with the system by natural language directly, which is convenient for nonprofessional users. Similar to other empirical investigations, current work also has the following three limitations: 1) parsing the input natural language, 2) providing some interactive widgets, and 3) predicting the follow-up query.

Parsing the Input Natural Language

The system allows users to type free text to conduct the visual exploration. However, this work currently designs specific parsing rules for nine low-level visual analysis tasks. Therefore, input queries that do not relate to the previous nine tasks may be excluded from the parsing scope of the system. We will conduct users to dissect complex problems into a combination of these nine low-level tasks and provide more hints to aid users to input specification queries in follow-up work. In addition, we utilize the method of defining templates to parse the query, the scalability of the system is limited. Machine learning technology may introduce novel solutions to this dilemma in the future.

Providing Interactive Widgets

The proposed *focus+context* answer generation approach allows users to obtain insight and contextual details simultaneously. However, helping users to explore additional points is disregarded. Although the query recommendation engine can recommend follow-up query, the recommended content mainly comprises tasks and attributes, which cannot meet the more detailed demands of users (e.g., narrow the scope of exploration by filtering). Besides, certain flexible interactive widgets, which can facilitate the user exploration are not realized, these designs will be implemented in the future work.

Predicting the Follow-Up Query

The interviews were conducted after user study indicated that not all queries recommended by the system could advance the analysis. However, some queries could provide information outside the main task and occasionally promote further exploration. Nevertheless, the query recommendation engine has certain limitations. Certain query records are counted to avoid a cold boot, and the relationship between current and follow-up queries is modeled on the basis of these statistical probabilities. However, the collected query records are insufficiently large, which may lead to inaccurate modeling. The modeling is also not customized for every user, which may cause particularly unfriendly recommended results to other individual users. In addition, the recommended query involves only tasks and attributes and ignores low-level elements (e.g., filtering). The correlation between attributes and tasks is disregarded in the attribute recommendation step. All the above limitations will be improved in the future work.

CONCLUSION

This article introduces DT2VIS, a visual analysis dialog system that helps users explore the tabular data efficiently. DT2VIS allows users to query by the natural language directly. A *focus+context* answer generation approach is proposed to allow users to obtain insight and contextual details simultaneously. The query process of the user is also modeled, and a query recommendation algorithm is proposed to help users perform in-depth exploration. Finally, specific cases of DT2VIS application and a user study are provided to evaluate the proposed system.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grants 61972356 and 62036009, and in part by the Fundamental Research Funds for the Provincial Universities of Zhejiang under Grant RF-A2020001.

REFERENCES

1. R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," in *Proc. IEEE Symp. Inf. Visualization*, 2005, pp. 111–117.
2. T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios, "Datatone: Managing ambiguity in natural language interfaces for data visualization," in *Proc. 28th Annu. ACM Symp. User Interface Softw. Technol.*, 2015, pp. 489–500.
3. D. Gotz and Z. Wen, "Behavior-driven visualization recommendation," in *Proc. 14th Int. Conf. Intell. User Interfaces*, 2009, pp. 315–324.
4. E. Hoque, V. Setlur, M. Tory, and I. Dykeman, "Applying pragmatics principles for interaction with visual analytics," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 309–318, Jan. 2018.
5. D. H. Kim, E. Hoque, and M. Agrawala, "Answering questions about charts and generating visual explanations," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–13.
6. A. Narechania, A.Srinivasan, and J. Stasko, "NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 369–379, Feb. 2021.
7. A. Ottley, R. Garnett, and R. Wan, "Follow the clicks: Learning and anticipating mouse interactions during exploratory data analysis," *Comput. Graph. Forum*, vol. 38, no. 3, pp. 41–52, 2019.
8. P. Pasupat and P. Liang, "Compositional semantic parsing on semi-structured tables," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process. Asian Federation Natural Lang. Process.*, 2015, pp. 1470–1480, doi: 10.3115/v1/p15-1142.
9. G.-D. Sun, Y.-C. Wu, R.-H. Liang, and S.-X. Liu, "A survey of visual analytics techniques and applications: State-of-the-art research and future challenges," *J. Comput. Sci. Technol.*, vol. 28, no. 5, pp. 852–867, 2013.
10. J. Wang, J. Wu, A. Cao, Z. Zhou, H. Zhang, and Y. Wu, "Tac-miner: Visual tactic mining for multiple table tennis matches," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 6, pp. 2770–2782, Jun. 2021.
11. D. Weng *et al.*, "Towards better bus networks: A visual analytics approach," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 817–827, Feb. 2021.
12. B. Yu and C. T. Silva, "Flowsense: A natural language interface for visual data exploration within a dataflow system," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 1, pp. 1–11, Jan. 2020.

QI JIANG is currently working toward the M.E. degree in control science and engineering from the Zhejiang University of Technology, Hangzhou, China. He received the B.E. degree in electrical engineering and automation from the Zhejiang University of Technology, in 2019. He is the corresponding author of this article. Contact him at jiangqi@zjut.edu.cn.

GUODAO SUN is an Associate Professor with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, China. He received the B.Sc. degree in computer science and technology from the Zhejiang University of Technology, and the Ph.D. degree in control science and engineering from the Zhejiang University of Technology. His main research interests include urban visualization and visual analytics of social media. See more information: http://godoorsun.org. He is the corresponding author of this article. Contact him at guodao@zjut.edu.cn.

YUE DONG is currently working toward the M.E. degree in electronic information from the Zhejiang University of Technology, Hangzhou, China. She received the B.E. degree in software engineering from the Zhejiang University of Technology, in 2020. Contact her at daisya99@163.com.

RONGHUA LIANG is currently a Professor of computer science and the Dean of the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China. He received the Ph.D. degree in computer science from Zhejiang University in 2003. He was a Research Fellow with the University of Bedfordshire, U.K., from April 2004 to July 2005 and as a Visiting Scholar with the University of California, Davis, from March 2010 to March 2011. His research interests include visual analytics, computer vision, and medical visualization. He is a member of IEEE. Contact him at rhliang@zjut.edu.cn.