SpeakQL: Towards Speech-driven Multimodal Querying

Vraj Shah
University of California, San Diego
vps002@eng.ucsd.edu

1 INTRODUCTION

Speech-based inputs have become popular in many applications on constrained device environments such as smartphones and tablets, and even personal conversational assistants such as Siri, Alexa, and Cortana. Inspired by this recent success of speech-driven interfaces, in this work, we consider an important fundamental question: *How should one design a speech-driven system to query structured data?*

Recent works have studied new querying modalities like visual [5, 9], touch-based [4, 8], and natural language interfaces (NLIs) [6, 7], especially for constrained querying environments such as tablets, smartphones, and conversational assistants. The commands given by the user are then translated to the Structured Query Language (SQL). But conspicuous by its absence is a speech-driven interface for regular SQL or other structured querying. One might wonder: Why dictate structured queries and not just use NLIs or visual interfaces? From a practical standpoint, many users, including in the C-suite, enterprise, Web, and other domains are already familiar with SQL (even if only a subset of it) and use it routinely. A spoken SQL interface could help them speed up query specification, especially in constrained settings such as smartphones and tablets, where typing SQL would be painful. More fundamentally, there is a trade-off inherent in any query interface, as illustrated in Figure 1(A) [2]. SpeakQL aims for almost full SQL sophistication, while improving ease of use using both speech and touch. While NLIs are improving, they increasingly rely on more keywords and structured interactions to mitigate the AI-hard natural language understanding problem [3, 6], as shown by Figure 1(B). They are moving to the right towards the "cliff", where on the right hand side of the cliff, we have languages with unambiguous context-free grammars. SQL, however, is already a structured English query language. Thus, instead of forcing all users to only use NLIs, we study how to exploit ASR for SQL and make spoken querying more "natural"

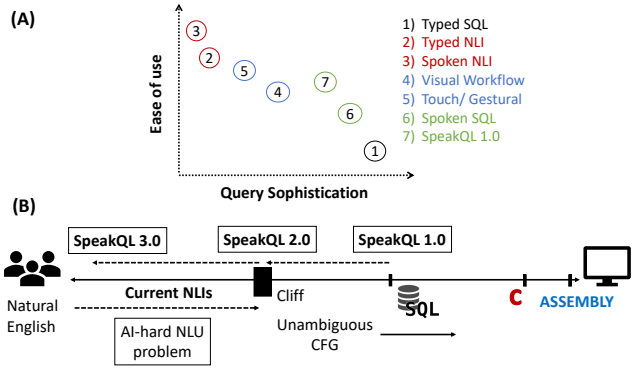

Figure 1: (A) Qualitative comparison of trade-off between ease of use and currently feasible query sophistication. (B) Contrasting SpeakQL's goals and current NLIs in terms of the "naturalness" of the query language.

without losing the sophistication enabled by unambiguous CFGs.

Desiderata. We aim to build an open-domain (support for queries in any application domain over any database schema), speech-driven, and multimodal querying system wherein the user can dictate the query and perform interactive correction using touch and/or speech, with the query results displayed on a screen. We plan to build different variants of our system, as shown in Figure 1 (B): SpeakQL 1.0 for a regular SQL language, SpeakQL 2.0 for a new speech-friendly dialect of SQL and SpeakQL 3.0 for a natural language. In the current work, the focus is only on regular SQL.

Technical Challenge. Non-vocabulary tokens (from an ASR perspective) are far more likely in SQL due to the infinite variety of database instances across domains. For instance, it is unlikely that any ASR engine can exactly recognize a literal like `CUSTID_1729A`. We call this the unbounded vocabulary problem; addressing this problem is a core technical challenge for SpeakQL.

2 APPROACH

We present the complete four-component system in Figure 2. The ASR Wrapper records the spoken SQL query, invokes a modern ASR tool, and obtains the transcription output. The Structure Determination component post-processes the ASR output(s) to obtain a syntactically correct SQL statement with placeholders for literals, where keywords and SplChars are fixed. It makes use of SQL grammar to generate all possible ground truth structures. The closest matching structure is retrieved by doing a similarity search based on edit distances

SIGMOD '19, June 30-July 5, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5643-5/19/06.
https://doi.org/10.1145/3299869.3300093

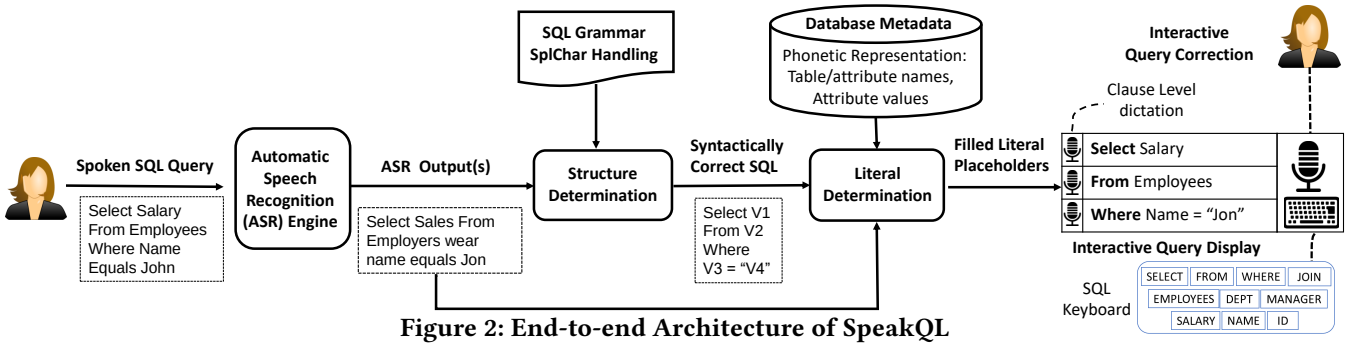

Figure 2: End-to-end Architecture of SpeakQL

with the ground truth structures. The Literal Determination component "fills in the blanks" for the literal placeholders using both the raw ASR outputs and pre-computed phonetic representation of the dataset being queried. This decoupling of structure and literal determination is a crucial design decision that helps us attack the unbounded vocabulary problem. Finally, there is an Interactive Display component that displays a single SQL statement that is the best possible transcription generated by our system. Even with our query determination algorithms, it might turn out that some of the tokens in the transcription are incorrect, especially for non-dictionary literals. Thus, we support user-in-the-loop query correction and provides speech or touch/click-based mechanisms for an interactive query correction. The users can either choose to dictate queries at the clause level or make use of a novel SQL keyboard.

3 EVALUATION

Data. Since there are no publicly available datasets for spoken SQL, we create our own dataset using the publicly available database schema: Employees Sample Database [1]. Firstly, we generate a dataset of 1250 SQL textual queries (750 for training and 500 for testing). Then, we use Amazon Polly to generate spoken SQL queries from these queries in text.

ASR. We use Azure's Speech Services that allows us to customize the vocabulary of the speech recognizer.

Results. Figure 3 shows the top1 results on the test data. We can see that the recall rates are already high for keywords and SplChar using just the ASR. For literals, however the recall rate is quite low (mean of 0.53). With SpeakQL, we achieve almost maximum possible precision and recall (mean of 0.98) for keywords and SplChars. While, even for literals the accuracy improves significantly. Figure 4 (A) shows the cumulative distribution (CDF) of the effort required from the end of the user in order to get to the desired query. The units of effort is the number of insertions, deletions or substitutions required at the token level in order to obtain the correct query, while using SpeakQL interface. For example, to correct 75% of the queries in test set, less than 4 units of effort is required. Figure 4 (B) plots the CDF for the running time of SpeakQL. We notice that almost 90% of the queries in the test set can be finished well within 2 seconds.

	Keyword Recall Rate	SplChar Recall Rate	Literal Recall Rate	Keyword Precision Rate	SplChr Precision Rate	Literal Precision Rate
ASR	0.92	0.96	0.53	0.84	0.87	0.49
SpeakQL	0.97	0.98	0.80	0.98	0.98	0.85

Figure 3: Mean Error Metrics: Precision and Recall

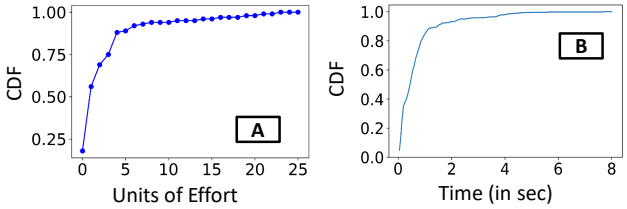

Figure 4: (A) Cumulative distribution of units of effort (B) Cumulative distribution of SpeakQL running time

User Study. 15 participants who were familiar with SQL were recruited through a short quiz. Each participant composed 12 queries on a tablet device, and for each query they performed two different task. In the first task, the participant had access to SpeakQL interface which allowed them to dictate a SQL query and perform interactive correction using both touch and speech. In the second task, the participant typed the SQL query from scratch with no access to our interface. The queries were divided into two segments: *simple* and *complex*. The simple queries have less than 20 tokens, while the queries that have 20 tokens or larger are the complex ones. Thus, composing a complex query would require a higher cognitive load relative to a simple query. We record the end to end time taken to perform both the tasks and evaluate our system using 180 data points (15 participants, 12 queries). We noticed a speedup of 2.4x on average (geomean) for the simple queries and a speedup of 2.9x on average (geomean) for the complex ones, when using SpeakQL in comparison with raw typing.

4 CONCLUSION

While the current space of natural language interfaces is grappling with the AI-hard problem of NLU, our work exploit structured information in the underlying language to make spoken querying more powerful and usable in practice. Thus, this work sets the stage in the direction of making a speech-first query language.

REFERENCES

[1] [n. d.]. Employees Dataset. dev.mysql.com/doc/employee/en

[2] Dharmil Chandarana, Vraj Shah, Arun Kumar, and Lawrence Saul. 2017. SpeakQL: Towards Speech-driven Multi-modal Querying. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*. ACM, 11.

[3] Alexa commands. 2018. https://www.cnet.com/how-to/amazon-echo-the-complete-list-of-alexa-commands

[4] Andrew Crotty et al. 2014. Vizdom: Interactive Analytics through Pen and Touch. In *VLDB Demo*.

[5] Oracle SQL Developer. 2008. oracle.com/technetwork/issue-archive/2008/08-mar/o28sql-100636.html

[6] Fei Li and HV Jagadish. 2014. Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment* 8, 1 (2014), 73–84.

[7] Gabriel Lyons et al. 2016. Making the Case for Query-by-Voice with EchoQuery. In *SIGMOD Demo*.

[8] Arnab Nandi et al. 2014. Gestural Query Specification. In *VLDB*.

[9] Moshé M. Zloof. 1975. Query by Example. In *National Computer Conference and Exposition*.