State of the Art and Open Challenges in NL Interfaces to Data

Fatma Özcan, IBM Research – Almaden
Abdul Quamar, IBM Research – Almaden
Jaydeep Sen, IBM Research – India,
Chuan Lei, IBM Research – Almaden
Vasilis Efthymiou, IBM Research - Almaden
Problem

- Natural Language Querying of Complex Datasets

Democratize access to data!!

Easy Access for Business Users

User does need to know SQL or any other complex lang!!

Exact knowledge of underlying data is not required

Conversational interfaces
Challenges and Opportunities

- Understanding user intent (disambiguation)
- Converting the intent to target language

Recent advances in natural language understanding enable more applications
  - Glove, fastText, BERT, ...
- Conversational agents also gaining popularity
  - Watson Assistant, SIRI, Cortana, Google Assistant,
How does it work?

Natural language query/user utterance

Entity-based:
Interpretation in two steps

Intermediate representation, AST

ML/DL based:
Holistic, single step

NLU and interpretation

structured query generation

SQL Query
Historical Perspective

Early systems

1993 NAUDA
2002 Banks
2005 NALIX
2006 SQAK
2008 Precis
2009 QUICK
2012 Bela
2014 SODA
2015 TR Discover
2016 Athena
2017 Seq2SQL
2018 SQLNet
2019 DialSQL
2020 Athena++
2020 Duoqest

ML/DL Approaches

Entity-based
What we will cover in this tutorial?

Complexity of the generated queries
- Simple to complex
- Single table queries
- Queries with joins between multiple tables
- Complex queries with subqueries

Interpretation Approaches
- Entity based
- ML/DL based
- Hybrid

Extension to dialogue
- Opportunity for disambiguation via interaction with the user
Complexity
Why complexity?

- **Definition**: (target) Query complexity $\propto$ # of different SQL clauses needed to construct the complete query
  - Defining in terms of query clauses extends to other query languages as well
  - Why not define complexity for NL queries.
    - NL query often is highly ambiguous and NLU is still an AI-Hard problem [Yampolskiy, R.V. 2013]

- **Complexity is often Application Specific**
  - IR or Search on Single Table
    - What is the Capital of France
  - Analytic Question Answering Systems over Database Schema
    - What is the average income per state in France

- The set of challenges differ depending on what type of queries are being needed and supported.

- Complexity of queries influences solution paradigms used in NLIDB systems
Target roadmap in terms of complexity

**Single Table**
- Select-Project
- Select-Project-Aggregation
- Select-Project-Aggregation-GroupBy-OrderBy

**Multiple Tables**
- Select-Project-Join
- Select-Project-Join-Aggregation
- Select-Project-Join-Aggregation-GroupBy-OrderBy

**BI and Analytic Queries**
- Comparison
- Date
- Simple Aggregation
- Top-k Queries
- L2L Compare
- Nested Queries
- Window Aggregation
- OLAP, etc.
**Complexity**

**An example schema**

<table>
<thead>
<tr>
<th>id</th>
<th>amount</th>
<th>date</th>
<th>decision</th>
<th>for</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14500</td>
<td>01-04-2018</td>
<td>approved</td>
<td>123</td>
</tr>
<tr>
<td>2</td>
<td>25000</td>
<td>11-02-2019</td>
<td>disapproved</td>
<td>345</td>
</tr>
<tr>
<td>3</td>
<td>10000</td>
<td>20-12-2019</td>
<td>approved</td>
<td>789</td>
</tr>
</tbody>
</table>

- **Borrower**
  - ssncode: 123
  - name: Paul Smith
  - Credit score: 510
  - birth date: 02-20-1970
  - Lives in: 987

- **Borrower**
  - ssncode: 345
  - name: John Doe
  - Credit score: 760
  - birth date: 04-12-1982
  - Lives in: 654

- **Address**
  - id: 987
  - street: streetA
  - zipcode: 100101
  - state: California

  - id: 345
  - street: streetB
  - zipcode: 102102
  - state: Texas

  - ..
  - ..
Simple Select-project queries on single table

- Example questions
  - Show me the **ssncode** for **Paul Smith**

- Challenges
  - Domain understanding
    - What column names or/and data instances have been mentioned in the query
      => “ssncode”, Paul Smith (=borrower.name)
      ⚠️ [What if ?] Show me ssn for Mr. Smith.

- NLU
  - What column is supposed to be used for Projection, Filter
    => SELECT(borrower.ssncode)
    ⚠️ [What if ?] For Paul Smith, tell me the ssncode?
Select-project-aggregation queries on a single table

- Example NLQs:
  - *What is the average amount of loans approved by year*

- Challenges
  - NLU
    - Is there an aggregation? What is the argument of aggregation?
      \[ \text{AVG(amount)} \]
    - [What if ?] on average what is the amount of loans approved for each year
    - Is there a group by/order by? What are the arguments for each?
      - \[ \text{Group by(year(date))} \]

- Domain understanding
  - Are the arguments of Aggregation/Group by/Order By etc. semantically valid?
    - [What if ?] What is the average approved amount of loans by address
Complexity

Business intelligence queries

Comparison-based filters

SIMPLE comparison: show me people with credit score more than 500
TIME dimension: show me average loans in Q1 2019
AGGREGATE comparison: show me people with total loans more than 50,000

Top-k queries: show me top 5 zip codes in terms of maximum loan approved

Like-to-Like comparison: how does the total amount of approved loans in Q1 this year compare to last year

Window Aggregation: what is the moving average of approved loan amount in every consecutive 3 months of last year

OLAP: which of the zipcodes had an increase in average amount of loan approved by more than 20%
Select-project-aggregation-join on multiple tables

- Example NLQs
  - What is the average *amount* of *loans* approved by *zipcode*

- Challenges
  - NLU
  - What are all the tables mentioned i.e., candidates for FROM clause?
    =>$\text{LOANS ADDRESS}$ (any implicit mention of intermediate tables?)

- Domain understanding
  - How to join the tables needed to answer a user query?
    =>$\text{LOANS INNER JOIN BORROWER INNER JOIN ADDRESS}$
  - Needs to know the complete domain schema with relations
  - For large schema, finding the right join path is the main challenge
Complexity

Business intelligence queries - challenges

- **Complex computations**
  - Which of the zipcodes had an *increase in average amount* of loan approved by *more than 20%*

- **Implicit intents/arguments**
  - **Time dimension**
    
  *Show me average loans in Q1 2019 => (loan date) in Q1 2019*
  
  *Domain reasoning*
  
  *Who are the top 10 borrowers in zipcode 12345 => Top 10 borrowers (in terms of total amount of loans)*

- Solving these challenges require a combination of rich NL understanding as well as domain reasoning
  
  - NLU: how to detect the primary intent or/and mention of computations in the query
  
  - Domain reasoning: How to to infer the implicit arguments and/or resolve ambiguity in the NL utterance
Examples:
1. Applying NOT operation to obtain complement set
2. Numeric Comparison between subqueries.
3. Enforcing Equality/inequality between subqueries
   (.. and more categories....)

Subqueries:
- subquery formation: how to segregate the NL query into subquery parts ?
- Find all borrowers with more loans in this year than last year ?

NLU
- hard to detect : Many reasons why a NL query may require a nested query.
- Domain Understanding
- subquery formation: Needs to reason over domain semantics and query context.
Complexity

Key takeaways

- **Complexity is correlated with the application need**
  - QA systems aimed for analytic queries over DB schema needs complex queries

- **Multiple table queries needs the knowledge of full domain schema**

- **More complex queries in general need**
  - Deeper domain understanding
  - Ability to reason over query intent and domain semantics.

- **Implications**
  - ML models → adequate domain specific training examples
  - Entity based models → domain abstraction (ontology graph, schema graph, etc.)
NLQ Interpretation: Entity-based Approaches
Entity-based Approaches

Recognize entities and relationships between the entities in a query

Example: “Show me Amazon customers who are also from Seattle”

Internal/external representation of the underlying data using:

- an index structure (e.g., inverted index over tables and columns)
- a taxonomy of terms and their synonyms (e.g., WordNet)
- an ontology (i.e., a rich semantic data model allowing complex query interpretation)
Using Index Structures or Taxonomies

Common approach:

- Parsing of the NLQ to machine-readable format
- Identify slots in NLQ that correspond to entities
- Look up entity slots in NLQ in an inverted index of labels

Example: “Show me Amazon customers who are also from Seattle”

Précis [Koutrika et al., ICDE 06] [Simitsis et al., VLDBJ 08]
QUICK [Zenz et al, J. Web Semantics 09]
DuoQuest [Baik et al., CIDR 20] [Baik et al., SIGMOD 20]
NaLIR [Li et al., SIGMOD 14][Li et al., VLDB 14][Li et al., SIGMOD Rec. 16]
NLQ Interpretation: Entity-based Approaches

Précis [Koutrika et al., ICDE 06] [Simitsis et al., VLDBJ 08]

Example (keyword search in DNF): “Clint Eastwood” AND “thriller”

- Interpretations:
  - thrillers directed by Clint Eastwood
  - thrillers in which Clint Eastwood is acting
  - thrillers directed by Clint Eastwood, in which Clint Eastwood is also acting

- Interpretations ranked based on join importance
NLQ Interpretation: Entity-based Approaches

QUICK [Zenz et al, J. Web Semantics 09]

Example (keyword search): “Wright London”

- User interaction to determine which interpretation is correct
**NLQ Interpretation: Entity-based Approaches**

**NaLIR** [Li et al., SIGMOD 14][Li et al., VLDB 14][Li et al., SIGMOD Rec. 16]

Example: “show all authors who have more papers than H. V. Jagadish in VLDB after 2005”

- **Initial parse tree from Stanford NLP**
  - ROOT
    - return
      - author
        - paper
          - more
          - VLDB
        - Jag
        - after
          - 2005

- **Refined parse tree**
  - ROOT
    - return
    - more
    - author
    - paper
    - Jag
    - after
      - 2005

- **User interaction to disambiguate**

- **May refer to WordNet terms:**
  - VLDB conference, PVLDB, and VLDB Journal

- **User may further edit the refined parse tree (e.g., add new nodes)**
Using an Ontology

Common approach:

• Look up entity slots in NLQ in an ontology

• Identify possible join paths based on the underlying ontology relationships

BELA [Walter et al., ISWC 12]
SODA [Blunschi et al., VLDB 12]
USI Answers [Waltinger et al., IAAI 13]
TR Discover [Song et al., ISWC 15]
ATHENA [Saha et al., VLDB 16][Lei et al., IEEE Data Eng. Bull. 18]
ATHENA++ [Sen et al., SIGMOD 19]
NLQ Interpretation: Entity-based Approaches

Example: “What is the currency of the Czech Republic?”

- Query templates:
  SELECT ?y WHERE {?x ?p ?y}
  slots: (?x: Czech Republic), (?p: currency)

- Inverted index lookup, built from DBpedia labels:
  - lookup result for ?x: http://dbpedia.org/resource/Czech_Republic
  - lookup result for ?p: http://dbpedia.org/ontology/currency
    - if no exact match, the closest property of Czech Republic to “currency” is returned

- Interpretation:
  - fill slots with lookup results:
    SELECT ?y WHERE {dbr:Czech_Republic ?dbonto:currency ?y}
NLQ Interpretation: Entity-based Approaches

**SODA** [Blunschi et al., VLDB 12]

- Looks up each query keyword in two indices:
  - one for the data in the database
  - one for the meta-data in ontologies (so-called *metadata warehouse*)
    - including synonyms and homonyms extracted from DBpedia

- Multiple interpretations generated
  - ontology hierarchies and relationships help in disambiguation
  - Ranking of interpretations based on lookup scores aggregations
  - Top-10 interpretations executed, and snippets are shown to user to select
Natural Language Query Interpretation -> Entity-based Approaches

**USI Answers** [Waltinger et al., IAAI 13]

- Parse query using Stanford Core NLP and ClearTK

- Dictionary- and regex-based look-ups to generate candidates

- Distinguish between concepts, instances, relationships, and identify time mentions
  - a dedicated annotator is used for each of the above components

- Data stored in a relational DB, meta-data represented in an ontology
  - allows relationship extraction between ontology concepts
NLQ Interpretation: Entity-based Approaches

**TR Discover** [Song et al., ISWC 15]

- Provides query auto-completion
- Suggestions based on nodes centrality in RDF graph

<table>
<thead>
<tr>
<th>d</th>
<th>drugs</th>
<th>drugs manufactured by</th>
</tr>
</thead>
<tbody>
<tr>
<td>NL</td>
<td>using having a secondary indication of</td>
<td>NL</td>
</tr>
<tr>
<td>drugs</td>
<td>developed by</td>
<td>companies</td>
</tr>
<tr>
<td>drugs using</td>
<td></td>
<td>company</td>
</tr>
<tr>
<td>drugs having a secondary indication of</td>
<td></td>
<td>Pfizer Inc</td>
</tr>
<tr>
<td>drugs having a primary indication of</td>
<td>manufactured by</td>
<td>National Institutes of Health</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GlaxoSmithKline plc</td>
</tr>
</tbody>
</table>

“d” is typed

“drugs” is selected and suggestions are provided
- Properties having “Drugs” as subject in RDF graph

“manufactured by” is selected and “Pfizer Inc” can be chosen to complete the query
NLQ Interpretation: Entity-based Approaches

ATHENA [Saha et al., VLDB 16][Lei et al., IEEE Data Eng. Bull. 18]

- Two-phase approach (physical-logical independence):
  - Phase 1: Query interpretation against a domain ontology
  - Phase 2: Structured query generation

Example: “How many people bought IBM stocks in the last 5 years?”

- Annotate each token with possible ontology elements (e.g., Company.name or ListedSecurity.legalName for “IBM” token)
- Selecting all combinations of candidate elements per token gives different interpretations
- Model every possible interpretation as an Interpretation Tree (ITree)
- Pick a single element for each token in a holistic way (Steiner Tree-based)
NLQ Interpretation: Entity-based Approaches

ATHENA++ [Sen et al., SIGMOD 19]

- Extends ATHENA to cover complex nested queries (commonly found in BI queries)

Example: “Show me everyone who bought stocks in 2019 that have gone up in value”

Evidence Set $ES_1$
- everyone
- stocks
- in 2019

Evidence Set $ES_2$
- stocks
- value

Nested Query Token
- {gone up}

Interpretation Tree ($ITree_1$)
- Transaction
- ListedSecurity
- date
- Account
- MonetaryAmount
- value

Operator: ‘>’
Pros and Cons of Entity-based Approaches

Handling complex input queries and generating complex structured queries

Easier to incorporate domain knowledge

Usually don’t require labelled training data

Highly sensitive to variations in the user query
NLQ Interpretation: Machine Learning-based Approaches
Machine learning-based approaches

– General idea

• Apply supervised machine learning techniques (RNNs) on a set of question/answer pairs
  – Questions: natural language queries
  – Answers: respective SQL statements
Machine learning-based approaches: progression

- **Seq2Seq (RNN)**
  - Seq2SQL
  - SQLNet
  - TypeSQL
  - IncSQL
  - SQLova
  - X-SQL
  - HydraNet

- **Domain Adaptation**
  - SyntaxSQLNet
  - IRNet
  - RYANSQL
  - Adversarial method
  - RAT-SQL

- **Training Data Generation**
  - DBPal
Seq2SQL [Zhong et al, arXiv 2017]

- Key ideas

  - A deep neural network leverages SQL structure to prune generated query space
  - Policy-based reinforcement learning (RL) to generate query conditions
  - A mixed object (cross entropy losses + RL rewards from in-the-loop query execution)
Machine Learning-based Approach

Seq2SQL cont.

– Aggregation operation
  • An MLP over aggregated hidden representations of the inputs
  • 4 possible outputs: COUNT, MIN, MAX, or NONE

– Select column
  • A list of column representations using LSTM + a question representation (similar to aggregation operation)
  • Combine two representations as input for an MLP

– Where clause
  • Augmented pointer network and RL
Machine Learning-based Approach

SQLNet [Xu et al. arXiv 2017]

– Key ideas

• Sketch-based approach to avoid RL and “order-matters” issue

• Sequence-to-set prediction using column attention (WHERE clause)
  – An MLP with one layer over the embeddings computed by 2 LSTMs (one for the question, one for the column names)

SQL sketch

```
SELECT $AGG $COLUMN
WHERE $COLUMN $OP $VALUE
(AND $COLUMN $OP $VALUE) *
```
Machine Learning-based Approach

**TypeSQL** [Yu et al. NAACL 2018]

– Key ideas

- Sketch-based approach to fill query slots
- Utilize types extracted from either knowledge graph or table content to help model better understand entities and numbers in the question
  - Two bi-directional LSTMs to encode words in the question with their types and the column names separately
  - The output hidden states of LSTMs are then used to predict the values for the slots in the SQL sketch
Machine Learning-based Approach

SyntaxSQLNet [Yu et al. EMNLP 2018]

– Key ideas

• SQL path history and table-aware column attention encoders
  – Attention mechanism to encode question representation as well as SQL path history

• SQL specific syntax tree-based decoder with SQL path history
  – Determine a specific module to invoke and predict the next SQL token to generate based on the current SQL token and the tokens gone over to reach the current token
Adversarial method for domain adaptation [Wang et al. ICDE 2020]

– Key ideas

• Separate out data-specific components and focus on the latent semantic structure
• Domain-specific knowledge will NOT be a strong signal for prediction

– Example

What is the height $c_0$ of LeBron James $v_1$?
SELECT $c_0$ WHERE $c_1 = v_1$
SELECT height WHERE name = 'LeBron James'

What is the population $c_0$ of NYC $v_1$?
SELECT $c_0$ WHERE $c_1 = v_1$
SELECT population WHERE city = 'NYC'
Adversarial method for domain adaptation cont.

– Phrase that mentions “to launch on” should be the most influential to the prediction using Fast Gradient Method

1. Classifier predicts if domain-specific keyword is mentioned

2. Identify terms by searching for a continuous span that changes the prediction the most
Machine Learning-based Approach

RAT-SQL [Wang et al. ACL 2020]

– Key ideas

• Relation-aware self-attention
  – Schema entities and question words
  – Predefined schema relations

• Represent database schema and the question-contextualized schema as graph
  – Schema linking
    » Name-based and value-based linking
    » Memory-schema alignment matrix
Machine Learning-based Approach

TAPAS [Herzig et al. ACL 2020]

Key ideas

- Extend BERT’s architecture to pre-train the model over tables and related text segments
  - Additional positional embeddings used to encode tabular structure
- Weak supervision reasons over tables without generating logical forms
  - Predict the denotation by selecting table cells
  - Optionally apply aggregation operator to such selection

Based on material from: Herzig et al. 2020. TAPAS: Weakly Supervised Table Parsing via Pre-training. ACL.
Machine Learning-based Approach

DBPal [Weir et al. SIGMOD 2020]

– Key idea – generates synthetic training data
  
  Improve overall translation accuracy
  Increase robustness to linguistic variations
  Specialize the model for the target database

• Training phase
  – Provide large corpora of synthesized training data

• Runtime phase
  – Replace the constants in the input NL query with placeholders to make the translation model independent from the actual database
Machine Learning-based Approach

DBPal cont.

– Training phase

• Data instantiation
  – Each SQL template <-> one or more NL templates (slot filling)
    SQL template – `Select {Attribute}(s) From {Table} Where {Filter}`
    NL template – `{SelectPhrase} {Attribute}(s) {FromPhrase} {Table}(s) {WherePhrase} {Filter}`

• Data augmentation
  – Automatic paraphrasing (using PPDB)
  – Missing information (dropping words and subphrases)

• Optimization procedure
Machine learning-based approach takeaways

– Pros
  • Robust to natural language variations
  • Easy instantiation

– Cons
  • Limited capability of handling complex queries
  • Require large amounts of training data
Extension to Dialog
Extension to Dialog

Dialog as an extension to one-shot Q&A

• Next natural step in NLID is a dialog
  • Ability to understand, respond and clarify ambiguity using a two-way conversation
  • Persistent context across turns of conversation
  • Interactive experience for data exploration

Show me the cost incurred on claims for the female population over the age of 55 in the North America region

Here are the results for claims for the female population over 55 in North America

What about males in the same age range?

Here are the results for the male population
Extension to Dialog

Taxonomy of Conversation Systems

Pre-Built

Apple Siri
Microsoft Cortana
Google Home
Amazon Alexa

Custom Conversation-as-a-Service

IBM Watson Assistant
Google Dialog Flow
Microsoft Bot Framework
Facebook Wit.ai

Based on material from J. Gao, M. Galley, and L. Li. Neural approaches to conversational AI. CoRR, abs/1809.08267, 2018.
Components of a Conversation System

- **Intents:** Express the purpose or goal expressed in the user query/input.
- **Entities:** Represent real world objects relevant in the context of a user query.
- **Dialog:** Uses discovered intents, entities and context from the application to provide an interactive conversational experience to the user.
- **External Data Sources:** Interaction with external data sources needs to be orchestrated to respond to user/application queries.
Extension to Dialog

Intent Identification

- Intent Specification
  - Need for up-front specification of a fixed set of intents based on
    - What the users might want to ask (Expected Workload)
    - What the chatbot is designed to handle/support

- Approaches for Intent Classification
  - ML Classifiers
  - Deep Learning Techniques
    - Seq2Seq Networks
      - Translate a natural language query into SQL
        - [SEQ2SQL, SQLNet, …]
    - Utilize user feedback
      - Dial SQL [Gur et al. ACL 2018]
      - Echo Query [Gabriel Lyons et al. SIGMOD 2016]
    - Utilize conversational context
      - Editing based SQL Query generation [Zhang et al. EMNLP 2019]
  - Utilize user feedback and context
    - A-BI, [Francia et al., EDBT 2019]
Follow a two-step approach
1. Classify user utterances into a set of predefined intents
2. Structured Query Generation
   - A structured query generated corresponding to each identified intent
   - Template based query generation a common approach
   - One template corresponding to each intent
   - Templates populated using the entities identified in the user utterance to generate structured query

**What Drugs treat Psoriasis?**

Intent: Treatment

Structured Query Template

```
SELECT oDrug.name FROM Drug oDrug INNER JOIN Condition oCondition WHERE oDrug.treats=oCondition.ConditionID AND oCondition.name = '<@Condition>'
```

Structured Query (SQL)

```
SELECT oDrug.name FROM Drug oDrug INNER JOIN Condition oCondition WHERE oDrug.treats=oCondition.ConditionID AND oCondition.name = 'Psoriasis'
```
Extension to Dialog

Intent Identification: Utilizing Feedback

- Iterative query generation-Dial SQL [Gur et al. ACL 2018 ]:
  - Iterative dialog-based query generation framework using user feedback
  - Ask users for validating via simple multi-choice questions interface
  - User feedback is then leveraged to revise the query

---

Natural Language Question : What are the countries that joined the NATO before 2004?
Ground Truth SQL Query : SELECT country WHERE date of join < 2004
Initial SQL Query : SELECT count ( country ) WHERE date of join = 2004

<table>
<thead>
<tr>
<th>Turn Side</th>
<th>NL Expression using Rule-Based Templates</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>System:</td>
<td>Could you please confirm if you are asking total number of the results?</td>
<td>validate agg (1, 2)</td>
</tr>
<tr>
<td>User:</td>
<td>No, that’s not true.</td>
<td>negate()</td>
</tr>
<tr>
<td>System:</td>
<td>Could you please select one of the following options :</td>
<td>offer agg (average, no agg)</td>
</tr>
<tr>
<td>User:</td>
<td>(i) average number of results, (ii) all the results</td>
<td>select(2)</td>
</tr>
<tr>
<td>System:</td>
<td>Could you please confirm if date of join is 2004 ?</td>
<td>validate where change(6, 11)</td>
</tr>
<tr>
<td>User:</td>
<td>That’s wrong.</td>
<td>negate()</td>
</tr>
<tr>
<td>System:</td>
<td>Could you please select one of the following options :</td>
<td>offer where (date of join &lt; 2004, date of invitation &lt; 2004)</td>
</tr>
<tr>
<td>User:</td>
<td>(i) date of join is before 2004, (ii) date of invitation is before 2004</td>
<td>select(1)</td>
</tr>
</tbody>
</table>

- Multi-Layer RNN network to encode dialogue history and provide candidate query choices to users
  - First layer encodes dialog history
  - Second layer decodes error span
  - Third layer decodes list of choices to offer to user
Extension to Dialog

Intent Identification: Utilizing Feedback

Echo Query [Gabriel Lyons et al. SIGMOD’16]:
• User feedback for query clarification
• Vocabulary personalization through user interactions
• Focuses on NL to SQL translation for simple SPJ Queries with filters and group bys
• Hands Free Voice Dialogue based interaction
• Built using the Amazon Alexa Voice Service
Extension to Dialog

Intent Identification: Utilizing Conversational Context

- Editing based SQL Query generation [Zhang et al. EMNLP 2019]:
  - An encode-decoder architecture with attention mechanisms
  - Use neural networks (Bi-LSTMs) to capture semantic understanding of user utterances, table schema and the mapping between the two
    - Utterance encoder uses bi-LSTM to generate utterance token embedding with attention to the column header embeddings and context from previous utterances
    - Table encoder uses bi-LSTM with attentions to encode the internal structure of the table schema as well as the relationship between utterance and the table schema
Extension to Dialog

Intent Identification: Utilizing User Feedback and Context

- Augmented Business Intelligence: (A-BI, Francia et al., EDBT 2019)
  - Takes the situational context of the user into account (Device, Role, location, date, etc.)
  - Incorporates user feedback on queries
  - Uses collaborative filtering for better user experience (Recommendations)
Extension to Dialog

Intent Identification

• Open Challenges
  • Requirement of substantial amount of training data
  • Incorporation of domain specific understanding
    • Understanding workload patterns and their mapping to the domain schema
  • Ability to handle unseen user utterances
    • Need for Hybrid approaches
    • Intent classification for a fixed set of pre-defined intents
    • Dynamic learning:
      • Rule based interpretation[Athena] / Other NN approaches required to respond to new/unseen utterances
      • Allows learning of new intents dynamically
Extension to Dialog

Entity Specification

- Entities are a critical part of deep domain understanding
- Constitute the domain vocabulary for the conversation system
  - Can refer to both meta data and data instances (Company: IBM, Drug: Aspirin)
- Synonyms
  - Provide flexibility in understanding user utterances
  - General purpose synonyms may be provided using external sources such as WordNet

<table>
<thead>
<tr>
<th>Entities:</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concepts:</td>
<td>Drug, Precautions, Dosage, Indication</td>
</tr>
<tr>
<td>Risk:</td>
<td>Contra-Indication, Black Box Warning</td>
</tr>
<tr>
<td>Drug Interaction:</td>
<td>DrugFood Interaction, DrugLab Interaction</td>
</tr>
<tr>
<td>Drug:</td>
<td>Aspirin, Ibuprofen, Citicoline, Pancreatin</td>
</tr>
<tr>
<td>Indication:</td>
<td>Fever, Headache, Bronchitis, Diabetes</td>
</tr>
<tr>
<td>Contra-Indication:</td>
<td>Cardiovascular disease, Breast carcinoma</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Entity</th>
<th>Synonyms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adverse Effect:</td>
<td>Side effect, adverse reaction, adverse event, AE</td>
</tr>
<tr>
<td>Condition:</td>
<td>disease, finding, disorder</td>
</tr>
<tr>
<td>Drug:</td>
<td>medicine, meds, medication, substance</td>
</tr>
<tr>
<td>Precaution:</td>
<td>caution, safe to give</td>
</tr>
<tr>
<td>Dosage:</td>
<td>dosing</td>
</tr>
<tr>
<td>Dose adjustment:</td>
<td>dose modification, dosing modification, dose reduction</td>
</tr>
</tbody>
</table>
Entity Specification

• Open Challenges
  • Amount of state that needs to be built for entity recognition can become quite large
  • Deep domain understanding
    • Domain specific synonyms (Kidney Disease, Renal Failure)
    • Hierarchical relationships
      • Taxonomies
      • External ontologies
      • Query Relaxation: Incorporating information from external KBs [Chuan et.al EDBT 2020]
Building the dialog

**The Dialog Tree**
- Defines the space of user utterances the system can recognize and respond to
- Responses conditioned on a combination of intents and entities identified in the user utterance
- Context captured from previous utterances

User input matches intent2 but does not contain entity2 so Agent produces an elicitation of entity2.

Next user input contains entity2, which is added to the context, so Agent produces the response to intent2.

**Open Challenges:**
- Designing dialog to support expected interaction patterns
  - Static specification common but laborious
  - Learning dialog from prior user experience (Agent based systems [Miner et al. JAMA Internal Medicine 2016])
- Need to handle both domain specific requests and general conversation management [IBM’s Alma, Conversational UX Design, Moore et al., ACM 2019]
Extension to Dialog

Training Examples

• Intent identification relies on training samples for identifying intents from user utterances

• The distribution and number of generated training examples for different intents, and the methodology for training the classifier model have a direct impact on its accuracy.

• Domain specific understanding required to generate appropriate training samples

• Most manual methods do not scale well

• Need for incorporating domain specific knowledge

• Substantial manual effort required to build a domain specific conversation system
  • Automatic generation of intents and training examples for domain specific applications
  • Ontologies provide a way to capture and utilize domain specific information

Training examples for dosage for a drug

Show me the Dose Adjustment for Aspirin?
Find Dose Adjustment for Aspirin?
Give me the increased dosage for Aspirin?
How do I perform a Dose Adjustment for Aspirin?
I want to see the modifications to dosing for Aspirin?
Ontology-based approach for building conversational systems

Quamar et.al SIGMOD 2020, C. Lei et.al IEEE Engr Bulletin 2018

**Domain Schema**
- Ontologies capture the semantics of the domain schema in terms of entities, relationship providing deep domain specialization

**Intents:**
- Workload patterns mapped onto the domain schema and identified as intents

**Entities:**
- Build the domain vocabulary of the system
- Ontology concepts, instances, synonyms

**Dialog:**
- Supports the desired interaction for the application conditioned on identified intent and entities

**Knowledge Base data:**
- Interaction with external data source through structured queries to respond to user/application queries
Open Challenges

- Complex workloads with complex queries
- Hybrid approaches for interpretations: Combine the strength of ML and entity-based solutions
- Domain adaptation and use in the enterprise
- Extensions to conversation
- Benchmarks
Benchmarks

- Benchmarks allow tracking progress, and great tool
- Many emerging benchmarks for NL to SQL
  - Early ones -
  - WikiSQL - [https://github.com/salesforce/WikiSQL](https://github.com/salesforce/WikiSQL)
  - Spider - [https://yale-lily.github.io/spider](https://yale-lily.github.io/spider)
  - FIBEN – IBM
  - SParC (multi-turn) - [https://yale-lily.github.io/sparc](https://yale-lily.github.io/sparc)
  - CoSQL (multi-turn) - [https://yale-lily.github.io/cosql](https://yale-lily.github.io/cosql)
• Crowd-source set of labeled dataset for NLQ over relational data
• About 80,000 hand-annotated example questions and corresponding SQL queries
• About 2400 tables from Wikipedia
• Single table queries with aggregation and selection
• Many systems that report results
• HydraNet (2020) at 92.2% test execution accuracy
• **PRO:** Largest labeled data set that covers tables from many domains
• **CON:** Simple query focus; systems risking overfitting to the data set
Benchmarks:

Spider

- Large scale complex and cross-domain benchmark
- Emphasis on testing cross domain robustness
- Has multiple schemas, each having multiple tables
  - 200 databases with multiple tables, covering 138 different domains
- Complex workload: 5,693 unique complex SQL queries with 10181 NLQ
  - Joins, and nested queries, aggregations
- **PRO:** Cross domain focus, more complex queries
- **CON:** Individual database are still simple, more reflexive of database supporting web pages
Benchmark:
FIBEN

- Benchmark from IBM [Athena++, SIGMOD 2019]
- Simulates a complex financial data mart
- Combines SEC data, and TPoX benchmark
- Final database conforms to a combined FIBO and FRO ontologies
- Complex query set: 300 complex BI queries with nesting, as well as joins, and aggregations
- **PRO:** Only workload that addresses a complex warehouse scenario
- **CON:** Single domain
Concluding Remarks

Very active area of research both from NLP and database communities
- Covered a subset of system that are representative, many more

Recent advancements in NLU major propellant

Yet, NLQ is not widely used in the enterprise

Conversational data exploration is the next wave
QUESTIONS?
References

- B. Bogin, M. Gardner, J. Berant. Representing Schema Structure with Graph Neural Networks for Text-to-SQL Parsing. ACL 2019: 4560–4565
References, cont.

- J. Gao, M. Galley, and L. Li. Neural approaches to conversational AI. CoRR, abs/1809.08267, 2018
- M. Golfarelli, S. Rizzi and M. Francia. Augmented Business Intelligence. DOLAP 2019
• E. Kaufmann and A. Bernstein. Evaluating the usability of natural language query languages and interfaces to Semantic Web knowledge bases. J. Web Semant. 8(4): 377–393, 2010
• G. Koutrika, A. Simitsis, and Y. E. Ioannidis. Précis: The Essence of a Query Answer. ICDE 2006: 69-78
• D. Küpper, M. Storbel, and D. Rösner. NAUDA: A Cooperative Natural Language Interface to Relational Databases. SIGMOD 1993: 529-533
• C. Lei, V. Efthymiou, R. Geis and F. Özcan: Expanding Query Answers on Medical Knowledge Bases. EDBT 2020: 567-578
• F. Li and H. V. Jagadish. Constructing an Interactive Natural Language Interface for Relational Databases. PVLDB 8(1): 73–84, 2014
• Y. Li and D. Rafiei. Natural Language Data Management and Interfaces: Recent Development and Open Challenges. SIGMOD 2017: 1765–1770
• Y. Li, H. Yang, and H. V. Jagadish. NaLIX: An Interactive Natural Language Interface for Querying XML. SIGMOD 2005: 900-902
References, cont.

- A. Simitsis, G. Koutrika, and Y. E. Ioannidis. Précis: from unstructured keywords as queries to structured databases as answers. The VLDB J. 17 (1), 117–149, 2008
- S. Tata and G. M. Lohman. SQAK: Doing More with Keywords. SIGMOD 2008: 889-902
References, cont.

- C. Wang, K. Tatwawadi, M. Brockschmidt, P. S. Huang, Y. Mao, O. Polozov, R. Singh. Robust Text-to-SQL Generation with Execution-Guided Decoding. CoRR, abs/1807.03100, 2018
- R. V. Yampolskiy.. Turing Test as a Defining Feature of AI-Completeness . In Artificial Intelligence, Evolutionary Computation and Metaheuristics (AIECM) --In the footsteps of Alan Turing. Xin-She Yang (Ed.). pp. 3-17. (Chapter 1). Springer. 2013