# Question Answering on Tables, Other Tasks, and Future Directions

SIGIR 2019 tutorial - Part VI

Shuo Zhang **Krisztian Balog**

University of Stavanger

# Outline for this Part

1. QA using a single table
2. QA using multiple tables
3. Other tasks
4. Future directions

# Motivation for QA on Tables

- Facts/relations in tables can be used for answering questions
- It complements QA on other sources

| Year | City | Country | Nations |
|---|---|---|---|
| 1896 | Athens | Greece | 14 |
| 1900 | Paris | France | 24 |
| 1904 | St. Louis | USA | 12 |
| ... | ... | ... | ... |
| 2004 | Athens | Greece | 201 |
| 2008 | Beijing | China | 204 |
| 2012 | London | UK | 204 |

$x_1$: *"Greece held its last Summer Olympics in which year?"*
$y_1$: {2004}

$x_2$: *"In which city's the first time with at least 20 nations?"*
$y_2$: {Paris}

$x_3$: *"Which years have the most participating countries?"*
$y_3$: {2008, 2012}

Figure: Illustration from Pasupat and Liang (2015)

# QA using a Single Table

**Definition**

QA using a single table takes as input and seeks to answer the question based on that table (by treating it as a knowledge base).

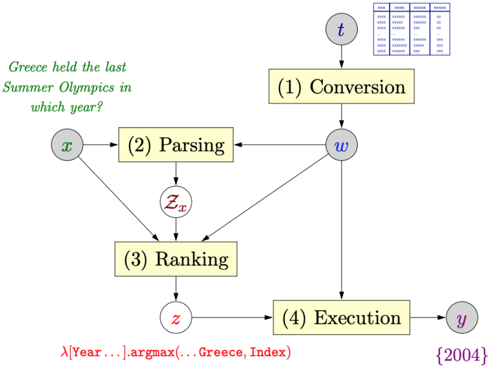
The only restriction on the input question is that a person must be able to answer it using just the table. Other than that, it can be of any type, ranging from a simple table lookup question to more complicated ones that involves various logical operations.

# Semantic Parsing

- Semantic parsing is often used in question answering, by generating logical expressions that are executable on knowledge bases
- Main challenges
  - Knowledge bases contain a canonicalized set of relations, while tabular data is much more noisy
  - Traditional semantic parsing sequentially parses natural language queries into logical forms and executes them against a knowledge base. To make them executable on tables, special logical forms are required
  - Semantic parsing and query execution become complicated for complex questions as they need carefully designed rules to parse them into logic forms

# Pasupat and Liang (2015)

Pasupat and Liang (2015) propose to answer complex questions, involving operation such as comparison, superlatives, aggregation, and arithmetics



# Pasupat and Liang (2015)

- The input table is converted into a knowledge graph by taking table rows as row nodes, strings as entity nodes, and columns as directed edges
- The column headings are used as predicates. Numbers and strings are normalized following a set of manual rules
- A traditional parser design strategy is followed, training a semantic parser on a set of question-answer pairs


Figure: Logical form for the question *"Greece held its last Summer Olympics in which year?"*.

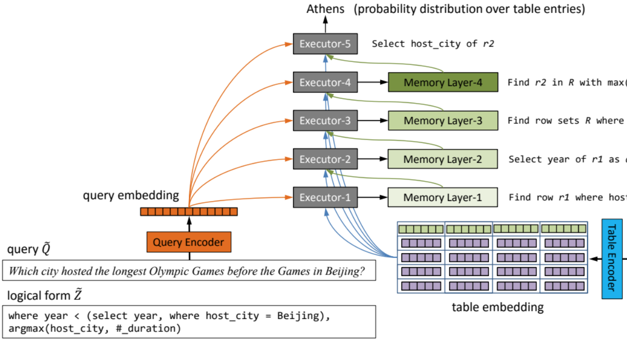
# Pasupat and Liang (2015)

- Given a table and a question, a set of candidate logical forms is generated by parsing the question
- Then, logic forms are ranked using a feature-based representation
- Finally, the highest ranked one is applied on the knowledge graph table representation to obtain the answer
- Resource: WikiTableQuestion dataset
  - Random sample of 2,100 tables from Wikipedia
  - 22,000 question-answer pairs

# Neural Enquirer Yin et al. (2016)

- Motivation: For queries that involve complex semantic constraints and logic, semantic parsing and query execution become extremely complex
  - E.g., *"Which city hosted the longest Olympic Games before the Games in Beijing?"*
  - Classical semantic parsing approaches which require a predefined set of all possible logical operations
- Idea: Learn the representations of queries and the KB table as well as of the query execution logic via end-to-end training using query-answer pairs

# Neural Enquirer Yin et al. (2016)




| year | host_city | #_duration | #_medals |
|---|---|---|---|
| 2000 | Sydney | 20 | 2,000 |
| 2004 | Athens | 35 | 1,500 |
| 2008 | Beijing | 30 | 2,500 |
| 2012 | London | 40 | 2,300 |

# Neural Enquirer Yin et al. (2016)

Architecture:

- The query and table are encoded into distributed representations
- Then, they are sent to a cascaded pipeline of Executors
  - Each executor models a specific type of operation conditioned on the query
  - The executors output annotations that encode intermediate execution results, and can be accessed by executors at the next level
  - By stacking several executors, the model is able to answer complex queries that involve multiple steps of computation

# QA using Multiple Tables

**Definition**

QA on tables seeks to answer questions using a collection of tables.

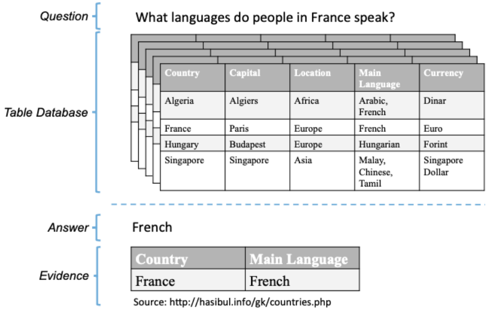



Figure: Example from Sun et al. (2016)

# Sun et al. (2016)

- Table cells are decomposed into relational chains, where each relational chain is a two-node graph connecting two entities. Any pair of cells in the same row form a directional relational chain
- The input query is also represented as a two-node graph question chain, by identifying the entities using an entity linking method
- The task then boils down to finding the relational chains that best match the question chain
- This matching is performed using deep neural networks, to overcome the vocabulary gap limitation of bag-of-words models
- The combination of deep features with some shallow features (like term-level similarity between query and table chains) was found to achieve the best performance

# Take-away Points for QA on Tables

- Web tables complement knowledge bases, providing rich knowledge missing from existing KBs
- Often, tables represent relations in a more straightforward way than KBs
- Coverage issue still persists

# Outline for this Part

1. ~~QA using a single table~~
2. ~~QA using multiple tables~~
3. **Other tasks**
4. Future directions

# Other Tasks

- **Table generation** (Zhang and Balog, 2018b)
- Title generation

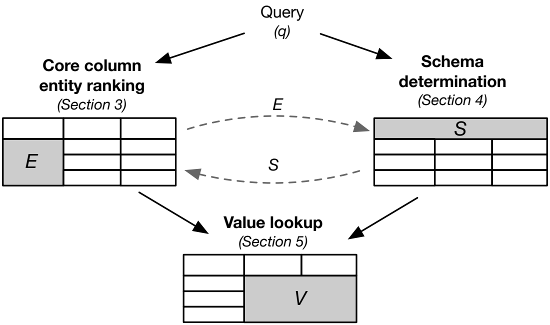
# Zhang and Balog (2018b)

**Definition**

On-the-fly table generation: given a query, generate a relational table that contains relevant entities (as rows) along with their key properties (as columns).

# Zhang and Balog (2018b)

Key idea: core column entity ranking and schema determination could potentially mutually reinforce each other.

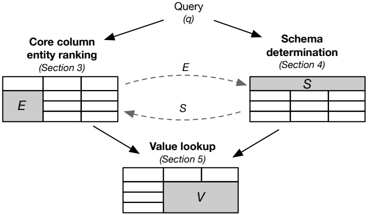


# Algorithm

**Algorithm 1:** Iterative Table Generation

**Data:** $q$, a keyword query
**Result:** $T = (E, S, V)$, a result table

```
begin
    E^0 ← rankEntites(q, {});
    S^0 ← rankLabels(q, {});
    t ← 0 ;
    while ¬terminate do
        t ← t + 1 ;
        E^t ← rankEntites(q, S^{t−1});
        S^t ← rankLabels(q, E^{t−1});
    end
    V ← lookupValues(E^t, S^t);
    return (E^t, S^t, V)
end
```



# Evaluation

- WikiTables corpus: 1.6M tables extracted from Wikipedia
- DBpedia (2015-10): 4.6M entities with an English abstract
- Two query sets (112 list queries and 600 complex entity-relationship queries)
- Resources: https://github.com/iai-group/sigir2018-table

# Example

| | Names | County | Country | Peter count de salis |
|---|---|---|---|---|
| Cork City and Suburbs | | | | |
| Belturbet | | | | |
| Kildare | | | | |
| Portarlington,_County_Laois | | | | |
| List_of_settlements_on_... | | | | |

Round #0

| | Population | Other counties | County | Notes |
|---|---|---|---|---|
| Cork City and Suburbs | | | | |
| Belturbet | | | | |
| Kildare | | | | |
| Roscommon | | | | |
| Athy | | | | |

Round #1

| | Population | County | Other counties | Population 2011 |
|---|---|---|---|---|
| Cork City and Suburbs | | | | |
| Belturbet | | | | |
| Thomastown | | | | |
| Roscommon | | | | |
| Kildare | | | | |

Round #2

| | County | Country | Population | Notes |
|---|---|---|---|---|
| Cork City and Suburbs | Cork | Ireland | 190,384 | Arms shown are those of Cork City |
| Thomastown | Kilkenny | Ireland | 1,837 | |
| Belturbet | Cavan | Ireland | 1,395 | |
| Kildare | Kildare | Ireland | 7,538 | |
| Roscommon | Roscommon | Ireland | 5,017 | Also administrative |

Round #3

# Other Tasks

- ~~Table generation~~
- **Title generation** (Hancock et al., 2019)

# Title generation (Hancock et al., 2019)

- Generating a descriptive title for tables (to help understand a table's relevance to the search query)
- Challenges:
  - The title should *relevant* (neither too vague nor too specific)
  - The title should be *readable* (sound natural to a human reader)
  - Table semantics tends to be distributed among a variery of elements on a web page
- Approach:
  - Sequence-to-sequence neural network model with both a copy mechanism and a generation mechanism

# Title generation (Hancock et al., 2019)

- The ideal title is often composed from multiple table elements, rather than selected from among them
- Table elements considered
  - Page title
  - Section headings
  - Table captions
  - Column headers
  - Text preceding/following the table
  - Table rows

# Title generation (Hancock et al., 2019)

| | |
|---|---|
| Page Title: | 1936–37 NHL season |
| Section Heading: | Regular Season |
| Section Heading: | Final Standings |
| Caption: | American Division |
| Title: | 1936-37 NHL Regular Season American Division Final Standings |

| | |
|---|---|
| Page Title: | The Beach at Anse Canot |
| Section Heading: | Anse Canot |
| Section Heading: | What's Nearby |
| Caption: | Attractions |
| Title: | The Beach at Anse Canot Nearby Attractions |

Figure: Illustration of composing a title from multiple table elements.

# Title generation (Hancock et al., 2019)

- Crowdsourced dataset
  - 10k web tables scraped from the tables returned as featured snippets on Google
  - 3 trained crowdworkers were asked to provide a descriptive title
    - Also mark whether that title occurred verbatim anywhere on the page or was composed (most informative and relevant title was composed 83% of the time)
    - If two or more titles were identical, accept that; otherwise select the longest title
  - Majority of the tables (72.6%) come from Wikipedia

# Outline for this Part

1. ~~QA using a single table~~
2. ~~QA using multiple tables~~
3. ~~Other tasks~~
4. **Future directions**

# Table Extraction

- In the early years, research was mainly focused on detecting, identifying, and extracting tables from web pages, and classifying them according to some type taxonomy
- Gradually, spreadsheet documents were also considered for table extraction, and type taxonomies became more fine-grained
- With the advancement of table extraction and classification methods, large-scale table corpora were constructed, which became available as resources to be utilized in other tasks
- One open issue is that the available table corpora are all a result of a one-off extraction effort; as such, these collections get quickly outdated

# Table Interpretation

- The problem of uncovering table semantics, including but not limited to identifying table column types, linking entities in tables, and extracting relational data from tables, still represents an active research area
- While there exist methods for high-precision extraction, there is plenty of room for improvement in terms of recall, as most existing methods can only interpret a small portion of tables
  - For example, Ritze et al. (2016) find that only 2.85% of web tables can be matched to DBpedia
- Most of the emphasis has been on relational tables; other table types (e.g., entity tables) bring about a different set of challenges
- Another line of future research concerns the development of user interfaces and tools for facilitating and visualizing the annotations (Mazumdar and Zhang, 2019)

# Knowledge Base Augmentation

- Shortcomings of current approaches include
  1. the lack of consideration of temporal information
  2. identifying entities at the right level of granularity (e.g., location may be given as a city or as a state or country) (Ritze et al., 2016)
- The former is especially important, as it may promote further utilization of tables to help keep KBs up-to-date

# Table Search

- Table search is a core task from the early days and remains to be an active research topic ever since
- One limitation of existing work is that it often makes assumptions about underlying query intent and the preferred answer table types
  - For example, Zhang and Balog (2018a) assume that queries follow a class-property pattern, which can be successfully answered by relational tables. As a result, relational tables with this pattern are preferred, which might therefore result in lower coverage
  - TableNet (Fetahu et al., 2019), a recent study on the interlinking of tables with has-A and is-A relations, can provide a better understanding of table patterns
- In the future, it would be desirable if an automatic query intent classifier were to identify the type of result table sought, which does not need to be limited to relational tables

# Table Augmentation

- There are at least two issues that remain:
  1. Tapping into the large volumes of unstructured sources (e.g., web pages)
  2. Combining data from multiple sources, which brings about a need for techniques to draw users' attention to conflicting information and help them to deal with those cases
- Normalizing cell values, without hand-crafting rules, is also an open problem

# Question Answering

- Works that address QA on a single table all take a carefully selected table (which is to be treated as a knowledge base) for granted; locating that table is a challenging table search task that remains to be addressed
- There seems to be a lack of understanding of when tables can actually aid QA
  - Even though QA on tables suffers from low coverage, it can complement QA on text
  - Yet, there has not been any systematic study on understanding what are the types of questions where tables can help or what is the scope of facts or relations where web tables have sufficient coverage
  - The heterogeneity of web tables limits the applicability of current methods to a small portion of tables

# Novel tasks

- Table generation on-the-fly
- Result presentation for tables
  - Generating snippets and/or natural language descriptions
- ...
- [Your proposal here]

# The End

**Questions?**

Slides and resources:
https://iai-group.github.io/webtables-tutorial/

# Bibliography I


Besnik Fetahu, Avishek Anand, and Maria Koutraki. Tablenet: An approach for determining fine-grained relations for wikipedia tables. In *Proc. of WWW '19*, pages 2736–2742, 2019.

Braden Hancock, Hongrae Lee, and Cong Yu. Generating titles for web tables. In *Proc. of WWW '19*, pages 638–647, 2019.

Suvodeep Mazumdar and Ziqi Zhang. A tool for creating and visualizing semantic annotations on relational tables. In *Proc. of ESWC '19*, pages 1447–1447, 2019.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, ACL '15, pages 1470–1480, 2015.

Dominique Ritze, Oliver Lehmberg, Yaser Oulabi, and Christian Bizer. Profiling the potential of web tables for augmenting cross-domain knowledge bases. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 251–261, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

Huan Sun, Hao Ma, Xiaodong He, Wen-tau Yih, Yu Su, and Xifeng Yan. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 771–782, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

# Bibliography II


Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. Neural enquirer: Learning to query tables in natural language. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 2308–2314. AAAI Press, 2016.

Shuo Zhang and Krisztian Balog. Ad hoc table retrieval using semantic similarity. In *Proceedings of The Web Conference*, WWW '18, pages 1553–1562, 2018a.

Shuo Zhang and Krisztian Balog. On-the-fly table generation. In *Proceedings of 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, 2018b.