Transformers for Tabular Data Representation: Models and Applications

Gilbert Badaro
Paolo Papotti
Presenters

• Gilbert Badaro
  • Principal Data Scientist at Amadeus
    • Post Doc Fellow at EURECOM (May 2022)
    • Ph.D. from American University of Beirut, Lebanon, 2020
    • Data Warehouse Architect and Developer, CMS, CERN
  • Focus on NLP and ML

• Paolo Papotti
  • Associate Professor at EURECOM
    • Ph.D. from Roma Tre University, Italy, 2007
    • Research positions at QCRI and Arizona State University
  • Focus on data management and information quality
Tutorial Outline

• Motivation
  • Natural Language and Data-centric Applications

• Language Models and Transformers [q&a]

• Developing & Consuming Tabular Data Representation
  • TrainingDatasets
  • Input Processing [q&a]
  • Model Training & Architecture [q&a]
  • Tabular Language Model
  • Consuming Tabular LMs [q&a]

• Open Challenges [q&a]
Text and tabular data

- Several applications use both text and tabular data

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Capital</th>
<th>Population</th>
<th>Size</th>
<th>President</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paris</td>
<td>67.39M</td>
<td>644K Km2</td>
<td>Emmanuel Macron</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Club</th>
<th>Season</th>
<th>League</th>
<th>Division</th>
<th>Apps</th>
<th>Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cannes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1988-89</td>
<td>Ligue 3</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989-90</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1990-91</td>
<td></td>
<td>28</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1991-92</td>
<td></td>
<td>31</td>
<td>5 (1)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table-based Fact-Checking (TFC)

- **Fact-checking** (tabular setting): verify if an input claim, expressed in natural language (NL) is true/false against some trusted *structured data*

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

**Input claim**: France has a population of 67.39 million.
**Output**: True

**Input claim**: Bolivia has more citizens than France.
**Output**: False

**Input Text**: France has a more than double population of Australia.
**Output**: Entail

**Input Text**: France has a higher population density than Bolivia.
**Output**: Does not entail/Not Enough Information

*(Aly et al, 2022; Karagiannis et al, 2020)*

- **Text Entailment**: check whether an input relational table implies or not a given NL claim

*(Eisenschlos et al, 2020)*
Demo

• https://coronacheck.eurecom.fr/en
Question Answering (QA)

• Find the cell(s) that answer a given input NL question
• Complexity ranges from simple lookup queries to complex ones involving aggregations and numerical reasoning

Population in Million by Country

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td><strong>67.39</strong></td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

**Question:** What is the population number of France?  
**Output:** 67.39

Population in Million by Country

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td><strong>67.39</strong></td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

**Question:** What is the total population in France and Bolivia?  
**Answer:** 79.06  
(Herzig et al, 2020)
Demo

• google/tapas-base-finetuned-wtq · Hugging Face
Semantic Parsing (SP): Text-2-SQL

- Given a question in NL and a table, generate a declarative query expressed in SQL (or SPARQL)

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

**Population in Million by Country (PMC)**

**NL text:** Find the capital of Australia.
**Output:** Select Capital from PMC where Country = “Australia”;

**NL text:** What is the average population?
**Output:** Select AVG(Population) from PMC;

(Yu et al, 2021; Gkini et al, 2021)
Table Retrieval (TR)

• Given a question in NL and a set of tables, identify the tables that can answer the question

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>GDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>Berlin</td>
<td>3.806</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>2.603</td>
</tr>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>1.331</td>
</tr>
</tbody>
</table>

**Population in Millions by Country**

**GDP by Country in Trillions USD**

**Statistics for France**

**Question:** What is the GDP of Germany?
**Table:** GDP by Country in Trillions USD
**Answer:** 3.806

(Wang et al, 2021; Pan et al, 2021)
Why are they challenging?

<table>
<thead>
<tr>
<th>Task ID</th>
<th>Task Label</th>
<th>Tasks Coverage</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>TFC</td>
<td>Table-based Fact-Checking or Entailment</td>
<td>Fact-Checking Text Refusal/Entailment</td>
<td>Table - Claim</td>
<td>True/False Refused/Entailed (Data Evidence)</td>
</tr>
<tr>
<td>QA</td>
<td>Question Answering</td>
<td>Retrieving the Cells for the Answer</td>
<td>Table - Question</td>
<td>Answer Cells</td>
</tr>
<tr>
<td>SP</td>
<td>Semantic Parsing</td>
<td>Text-to-SQL</td>
<td>Table - NL Query</td>
<td>Formal QL</td>
</tr>
<tr>
<td>TR</td>
<td>Table Retrieval</td>
<td>Retrieving Table that Contains the Answer</td>
<td>Tables - Question</td>
<td>Relevant Table(s)</td>
</tr>
<tr>
<td>TMP</td>
<td>Table Metadata Prediction</td>
<td>Column Type Prediction Table Type Classification</td>
<td>Table</td>
<td>Column Types Table Types Header Row Cell Role Relation between Two Cols Column Name</td>
</tr>
<tr>
<td>DI</td>
<td>Data Imputation</td>
<td>Cell Content Population</td>
<td>Table with Corrupted Cell Values</td>
<td>Table with Complete Cell Values</td>
</tr>
</tbody>
</table>
Table Metadata Prediction (TMP)

- Given an input table with corrupted or missing metadata, predict
  - column types and headers, and
  - intra-tables relationships
    - equivalence between columns, entity linking/resolution

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

Predict that the missing column header is **Country**

Predict that the table type is a **relational** table

(Cappuzzo et al, 2020; Deng et al. 2020; Li, Yuliang et al 2020)
Data Imputation (DI)

- Given a table with corrupted/missing values, populate the missing cell data

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

Population in Millions by Country

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

(Deng et al. 2020; Tang et al, 2021)
Text and tabular data

• Several applications use both
  • Table-based Fact-Checking/Text Entailment (TFC)
  • Question Answering (QA)
  • Semantic Parsing / Text-to-SQL (SP)
  • Table Retrieval (TR)
  • Table Metadata Prediction (TMP)
    • detecting column types, table types, relations, header cells,
    • entity resolution and linking; column name prediction
  • Data imputation (DI)

How can we exploit NL understanding in building such applications?
Tutorial Outline

• Motivation
  • Data-centric Applications and Natural Language

• Language Models and Transformers

• Developing & Consuming Tabular Data Representation
  • Training Datasets
  • Input Processing
  • Model Training & Architecture
  • Tabular Language Model
  • Consuming Tabular LMs

• Open Challenges
Deep learning can help with NL text

• A language model (LM) is a probability distribution over sequences of words
  • Given a sequence of words, it
    • assigns a probability to the sequence
    • predicts the most probable next word in the sequence

\[ p_{LM}(\text{the house is small}) > p_{LM}(\text{small the is house}) \]

• Modern LMs are obtained by (unsupervised) pre-training on large text corpora

• Pre-trained LMs enable state-of-the-art results in downstream NLP tasks, even in cases with limited amount of annotated training data
What can we do with Language Models?

Sydney is the capital city of the state of New South Wales, and the most populous city in Australia and Oceania. Located on Australia's east coast, the metropolis surrounds Port Jackson and extends about 70 km (43.5 mi) on its periphery [...]. Sydney is made up of 658 suburbs, spread across 33 local government areas. Residents of the city are known as "Sydneysiders". As of June 2020, Sydney's estimated metropolitan population was 5,361,466, meaning the city is home to approximately 66% of the state's population. Nicknames of the city include the 'Emerald City' and the 'Harbour City'.

Fact-checking (text): Sydney’s population as of June 2020 is less than 2 millions. False

Question Answering: What is an example of a nickname for Sydney? Emerald City / Harbour City

Sentiment Analysis: Neutral

Document Classification: Geography

Translation to French: Sydney est la capitale de l’État de la Nouvelle-Galles du Sud et la ville la plus peuplée d'Australie et d'Océanie.

Using a small labeled dataset, we customize the same pre-trained LM for several tasks
How does it work? Big Picture

1- Develop LM through *pre-training* using large unlabeled text corpora

2- *Fine-tune* LM using (relatively small) labeled training data for target application

3- Given a new paragraph, predict sentiment
Embeddings

- Focus on **neural** language models
- Instead of using probabilities, each word is mapped to the distributed representation encoded in the networks' hidden layers
  - one word $\rightarrow$ one vector
- Use continuous representations based on n-dimensional real-valued **word** (token) **embeddings**
  - words closer in the vector space are expected to be similar in meaning

(Mikolov et al, 2013)
Transformers 1/3

- Many ways to obtain a LM
- **Transformers** introduced *parallelism* (GPU/TPU) and enabled *larger models*
  - **Encoder-decoder architecture**
  - (Self) Attention mechanism to understand relationships between all words in a sentence, regardless of their respective position

(Vaswani et al, 2017)
Transformers 2/3

• BERT (encoder only) got SOTA in most NLP task with
  • New pre-training (masking, next sentence)
  • Left and right context from the word
• The LM learns relationships among tokens at multiple levels
  • Grammar/Syntax
  • Semantic
Transformers 3/3

• Token embeddings are complemented with more information

• Position is key as a transformer is not a RNN
  • sequential nature of RNNs precludes parallelization within training examples

More on Transformers from Immanuel Trummer in VLDB Tutorial 9 (Thu 8th 10:30 - 12:00)
From BERT to GPT-3 Codex: Harnessing the Potential of Very Large Language Models for Data Management
How does it work for Tabular Data?

• LMs are state-of-the-art for NL but tabular data has different forms (relational tables, spreadsheets, entity tables, ...) and different relationships
  • E.g., Position, co-occurrence vs same-row, same-column

• Problem: develop LMs that model tabular data
  • How to change the transformer architecture to account for the 2D characteristics of tables and its relationships?
Questions?
Tutorial Outline

• Motivation
  • Data-centric Applications and Natural Language

• Transformers and Language Models

• Developing & Consuming Tabular Data Representation
  • Training Datasets
  • Input Processing
  • Model Training & Architecture
  • Tabular Language Model
  • Consuming Tabular LMs

• Open Challenges
Characterization Study

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td><strong>67.39</strong></td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

**WikiTables**

**WDC Web Table Corpus**

**Population in Million by Country**

**Question:** What is the population of France?
Dimensions

1. Training Datasets
2. Input Processing
   - Data retrieval and filtering
   - Table serialization
   - Context and table concatenation
3. Model Architecture and Training
4. Output Model Representation: Tabular Language Model
5. Fine-tuning Representation for Downstream Tasks
Training Datasets
Training Datasets

• Large number of tables along with their context are used for pre-training
  • Better representation, less bias

• Context represents additional textual data that comes with tables
  • Text describing the table: caption, title or document surrounding the table
  • Table metadata: table orientation, header, keys
  • Question and claims addressed by the table

• Two types of datasets:
  • Unlabeled, such as Wikipedia Tables, are used exclusively for pre-training
  • Labeled, such as SPIDER (Yu et al., 2018), can also be used for fine-tuning
### GDP by Country in Trillions USD

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>GDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>Berlin</td>
<td>3.806</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>2.603</td>
</tr>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>1.331</td>
</tr>
</tbody>
</table>

**Question:** What is the GDP of Germany?

**Table:** GDP by Country in Trillions USD

(Answer: 3.806)
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Reference</th>
<th>Task Categories</th>
<th>Number of Tables</th>
<th>Large Tables</th>
<th>Context</th>
<th>Application Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>WDC Web Table Corpus</td>
<td>(Lehmberg et al., 2016)</td>
<td>✓ ✓</td>
<td>✓</td>
<td>✓</td>
<td>Table Metadata: Table orientation, header row, key column, timestamp before and after table. Surrounding Text: table caption, text before and after table, title of HTML page.</td>
<td>TABERT</td>
</tr>
<tr>
<td>VizNet</td>
<td>(Hu et al., 2019)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Table Metadata: Column Types.</td>
<td>TABBIE</td>
</tr>
<tr>
<td>Spreadsheets</td>
<td>(Dong et al., 2019)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Table Metadata: Cell Roles (Index, Index Name, Value Name, Aggregation and Others).</td>
<td>TABULARUNET</td>
</tr>
</tbody>
</table>
## Mostly Fine-Tuning Datasets (1/2)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Reference</th>
<th>Task Categories</th>
<th>Number of Tables</th>
<th>Large Tables</th>
<th>Context</th>
<th>Application Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>NQ-Tables</td>
<td>(Herzig et al., 2021)</td>
<td>✔</td>
<td>169,898</td>
<td>✔</td>
<td>Questions: 12K.</td>
<td>DTR</td>
</tr>
<tr>
<td>TABFACT</td>
<td>(Chen et al., 2020a)</td>
<td>✔</td>
<td>16K</td>
<td>✗</td>
<td>Textual Claims: 118K Claims</td>
<td>DECO</td>
</tr>
<tr>
<td>WikiSQL</td>
<td>(Zhong et al., 2017)</td>
<td>✔ ✔ ✔</td>
<td>24,241</td>
<td>✗</td>
<td>Questions: 80,654.</td>
<td>MMR</td>
</tr>
<tr>
<td>TabMCQ</td>
<td>(Jauhar et al., 2016)</td>
<td>✔ ✔</td>
<td>68</td>
<td>✗</td>
<td>Questions: 9,092.</td>
<td>RCI</td>
</tr>
<tr>
<td>SPIDER</td>
<td>(Yu et al., 2018)</td>
<td>✔</td>
<td>200 databases</td>
<td>✗</td>
<td>Questions: 10,181 Queries: 5,693.</td>
<td>GRAPPA</td>
</tr>
<tr>
<td>WikiTable Question (WikiTQ)</td>
<td>(Pasupat and Liang, 2015)</td>
<td>✔ ✔</td>
<td>2,108</td>
<td>✗</td>
<td>Questions: 22,033.</td>
<td>TAPEX</td>
</tr>
</tbody>
</table>
Mostly Fine-Tuning Datasets (2/2)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Reference</th>
<th>Task Categories</th>
<th>Number of Tables</th>
<th>Large Tables</th>
<th>Context</th>
<th>Application Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural Questions (NQ)</td>
<td>(Kwiatkowski et al., 2019)</td>
<td>✓</td>
<td>169,898</td>
<td>✓</td>
<td>Questions: 320K.</td>
<td>MMR</td>
</tr>
<tr>
<td>OTT-QA</td>
<td>(Chen et al., 2021)</td>
<td>✓</td>
<td>400K</td>
<td>✓</td>
<td>Surrounding Text: page title, section title, section text limited to 12 first sentences. Questions: 45,841.</td>
<td>MMR</td>
</tr>
<tr>
<td>Web Query Table</td>
<td>(Sun et al., 2019)</td>
<td>✓</td>
<td>273,816</td>
<td>✓</td>
<td>Surrounding Text: captions. Queries: 21,113.</td>
<td>GTR</td>
</tr>
<tr>
<td>HybridQA</td>
<td>(Chen et al., 2020b)</td>
<td>✓</td>
<td>13K</td>
<td>✓</td>
<td>Questions: 72K. Surrounding Text: first 12 sentences surrounding the table.</td>
<td>MATE</td>
</tr>
<tr>
<td>FEVEROUS</td>
<td>(Aly et al., 2021)</td>
<td>✓</td>
<td>28.8K</td>
<td>✓</td>
<td>Claims: 87K. Surrounding Text: article title. Table Metadata: row and column headers.</td>
<td>MATE</td>
</tr>
</tbody>
</table>
Input Processing
Data Retrieval and Filtering
Table Serialization: Reshaping 2D tabular structure to 1D
Context and Table Concatenation
Data Retrieval and Filtering

• Why do we need it?
  • Meet the limit (typically of 512 tokens) of Transformers
    • Transformers architecture theoretically has no limits on the input size
    • However, practically it is not the case: limit derived from positional embeddings, fixed attention size and computational complexity
  • Improve training time
  • Eliminate potential noise in output representations

(Devlin et al., 2019; Yin et al., 2020; Liu et al. 2021a)
Data Retrieval and Filtering

• How?
  • Can be downstream task by itself, Table Retrieval
  • Using a ranking function like BM25 (Robertson et al., 1995)
  • Using content snapshot (TABERT (Yin et al., 2020))
  • Term Frequency Inverse Document Frequency (TFIDF) (RCI (Glass et al., 2021))
  • Setting a threshold to limit the number of columns/rows allowed (DRT (Thorne et al., 2021))
  • Splitting Tables into smaller chunks (TUTA (Wang et al., 2021b), TabularNet (Du et al., 2021))
### Keeping 2 columns

<table>
<thead>
<tr>
<th>Country</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>11.67</td>
</tr>
</tbody>
</table>

### Keeping 2 rows

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Canberra</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>
Table Serialization

• Four possible ways:
  1- Horizontal scanning of the table row by row
    • Flattened table with value separators
      • Country | Capital | Population | Australia | Canberra | 25.69 ... Bolivia | La Paz | 11.67
    • Flattened table with **special token separator** to indicate beginning of a new row, new cell, new header (TAPEX (Liu et al. 2021a), TUTA (Wang et al. 2021b), ForTaP (Cheng et al., 2021))
    • Flattened table where each cell is represented as a concatenation of the **column name**, **column type** and cell value (TABERT)
    • Flattened **column headers** only (GRAPPA (Yu et al., 2021))
      • Country|Capital|Population
Table Serialization (continued)

2- Vertical scanning of the table column by column
   • Simple concatenation of column values or by using special separator tokens (DODUO (Suhara et al., 2021))

3- Combining the output from both horizontal and vertical serialization
   • element-wise product (RCI (Glass et al., 2021)), CLTR (Pan et al., 2021),
   • average pooling and concatenation (TabularNet),
   • average of row and column embeddings (TABBIE (Iida et al., 2021)).

4- Transforming data to text
   • using meaningful sentences generated out of the tabular data (DRT (Thorne et al., 2021))
   • using table-to-text systems such as Totto (Parikh et al., 2020).
<table>
<thead>
<tr>
<th>Name</th>
<th>Profession</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nicholas</td>
<td>Doctor</td>
<td>Washington D.C.</td>
</tr>
<tr>
<td>Sarah</td>
<td>Doctor</td>
<td>NY</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Birth City</th>
<th>Birth Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sheryl</td>
<td>...</td>
<td>1978</td>
</tr>
<tr>
<td>Sarah</td>
<td>Chicago</td>
<td>1982</td>
</tr>
<tr>
<td>Teuvo</td>
<td>Ruskala</td>
<td>1912</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Husband Name</th>
<th>Wife Name</th>
<th>Marriage Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nicholas</td>
<td>Sheryl</td>
<td>...</td>
</tr>
<tr>
<td>John</td>
<td>Sarah</td>
<td>2010</td>
</tr>
</tbody>
</table>

- Nicholas lives in Washington D.C. with his wife.
- Sheryl is Nicholas’s wife.
- Teuvo was born in 1912 in Ruskala.
- Sheryl’s mother gave birth to her in 1978.
- Nicholas is a doctor.
- Sarah was born in Chicago in 1982.
- Sarah married John in 2010.
- Sarah works in a hospital in NY as a doctor.

DRT (Thorne et al., 2021)
Table Serialization: Which method to choose?

• Most of the systems do not compare the different approaches:
  • One approach is typically selected and followed

• TABERT reports experiments with different table linearization strategies:
  • adding type information and cell values
  • phrasing the input as a sentence such as in TABFACT (Chen et al., 2020a)
=> Improvement in results

• (Veltri et al., 2022) in a table to text generation task experimented row vs column serialization:
  • Row performed better
Context and Table Concatenation

- **Context** is either ** prepended** or **appended** to the serialized table.
- Common case is to be prepended to the serialized table.
- TabFACT tested both strategies:
  - no significant difference in performance
- Type of context added usually depends on target downstream application
  - QA: a question is prepended to the serialized table.
- Some works like RCI (Glass et al., 2021) encode the context and the serialized table separately.

- Some works, like TABBIE (Iida et al., 2021), Doduo, TabularNet, do not include context.
  - Due to nature of downstream tasks specifically **TMP** and **DI**

---

**Context and Table parsed by row:**

```plaintext
[CLS] Population in Million by Country [CLS]
```

**Context and Table parsed by column:**

```plaintext
[CLS] Population of Countries [CLS] Country | France | ... | Italy | ... [SEP] Capital | France | ... | Rome | ... [SEP] Population | 67.39 | ...
```
TabFact experimented different serializations

![Diagram showing 12-Layer BERT-Base Model with Word, Position, and Type connections]

<table>
<thead>
<tr>
<th>Game</th>
<th>Date</th>
<th>Opponent</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>February 3, 2009</td>
<td>Florida</td>
<td>3-4</td>
</tr>
<tr>
<td>52</td>
<td>February 4, 2009</td>
<td>Buffalo</td>
<td>0-5</td>
</tr>
<tr>
<td>53</td>
<td>February 7, 2010</td>
<td>Montreal</td>
<td>5-2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Val</th>
<th>Test</th>
<th>Test (simple)</th>
<th>Test (complex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT classifier w/o Table</td>
<td>50.9</td>
<td>50.5</td>
<td>51.0</td>
<td>50.1</td>
</tr>
<tr>
<td>Table-BERT-Horizontal-F+T-Concatenate</td>
<td>50.7</td>
<td>50.4</td>
<td>50.8</td>
<td>50.0</td>
</tr>
<tr>
<td>Table-BERT-Vertical-F+T-Template</td>
<td>56.7</td>
<td>56.2</td>
<td>59.8</td>
<td>55.0</td>
</tr>
<tr>
<td>Table-BERT-Vertical-T+F-Template</td>
<td>56.7</td>
<td>57.0</td>
<td>60.6</td>
<td>54.3</td>
</tr>
<tr>
<td>Table-BERT-Horizontal-F+T-Template</td>
<td>66.0</td>
<td>65.1</td>
<td>79.0</td>
<td>58.1</td>
</tr>
<tr>
<td>Table-BERT-Horizontal-T+F-Template</td>
<td>66.1</td>
<td>65.1</td>
<td>79.1</td>
<td>58.2</td>
</tr>
</tbody>
</table>
Questions?
Model Training & Architecture

Customizations to account for tabular data structure

Extensions at the input/output level and/or on the internals of the architecture
Adaptations of Transformers’ Architecture

• Model with tabular data structure aware ➞ Customization to Vanilla transformer-based LMs

• Extensions are at different levels:
  • Input
  • Internal
  • Output
  • Training procedure
Input Level

• Additional positional embeddings to explicitly model the table structure
  • Typically for relational tables
  • Example position of the cell (row and column IDs), segment id: whether it is a context or a table entry, relative positional information of a token in cell/column header and rank id for sorting floats and dates

(TAPAS (Herzig et al., 2020)
Input Level (continued)

- Tree-based positional embeddings (TUTA (Wang et al, 2021))
  - Typically for entity tables or spreadsheets
  - Encode the position of a cell using top and left embeddings of a bi-dimensional coordinate tree.
Internal Level

• The **attention module** is the mostly concerned with updates at the internal level

• **Vertical self-attention** layers capture cross-row dependencies on cell values (TABERT)
Internal Level

- **Tree-based attention (TA):** row-wise or column-wise attention instead of additional positional embeddings (TUTA (Wang et al, 2021))
  - Tree-structure injected using a symmetric binary matrix to indicate visibility between tokens
  - Based on ablation study results:
    - **TA improved accuracy** for both cell-type classification and table-type classification compared to basic attention mechanism where all cells are visible to each other.
Internal Level

• **Masked self-attention** module attends to structurally related elements (TURL (Deng et al., 2021)):
  • Elements in one row or in one column (using a visibility matrix)
  • Different than vanilla transformer where each element attends to all other elements
  • Helps model to capture:
    • Factual knowledge embedded in the table
    • Associations between table metadata and table content
Internal Level

- **Sparse attention** method is used to address the limit of input size of transformers (512 tokens)
  - MATE sparsifies the attention matrix to attend to either rows and columns
Output Level

• Additional layers added on top of the feed-forward networks (FFNs) of the LM based on the targeted downstream task

• Question Answering (QA):
  • Additional classification layer for aggregations and cell selection (TAPAS)
Training Procedure Level

• Pretraining Task:
  • Prior to fine-tuning
  • Typically consist of reconstruction tasks, i.e., reconstruct correct input out of corrupted one
    • Usually using cross-entropy loss as objective function
  • Modifications on the typical MLM are applied to take into consideration the tabular structure:
    • Masking tokens from cells
    • Masking the whole cell regardless of the number of tokens it has
      • Enables the model to integrate the factual knowledge embedded in the table content and its context
      • Masking columns names and data types
  • Occasionally an SQL engine is used (TAPEX) to train the model to act as a neural SQL executor
    • Enable to mimic SQL semantics with relational tables
<table>
<thead>
<tr>
<th>Year</th>
<th>City</th>
<th>Country</th>
<th>Nations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1896</td>
<td>Athens</td>
<td>Greece</td>
<td>14</td>
</tr>
<tr>
<td>1900</td>
<td>Paris</td>
<td>France</td>
<td>24</td>
</tr>
<tr>
<td>1904</td>
<td>St. Louis</td>
<td>USA</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2004</td>
<td>Athens</td>
<td>Greece</td>
<td>201</td>
</tr>
<tr>
<td>2008</td>
<td>Beijing</td>
<td>China</td>
<td>204</td>
</tr>
</tbody>
</table>

```
[HEAD] Year | City | Country | Nations
[ROW] 1 1896 | Athens ...
SELECT City WHERE Country = France ORDER BY Year ASC LIMIT 1
```
Summary of customization

- A more table structure aware LM requires modifications:
  - Input level through additional embeddings
  - Internal level through adjustment of attention module
  - Training procedure level through pre-training task and objective that are table related such as masking and reconstructing cells
  - Output level through additional classification layers that are task dependent
Questions?
Tabular Language Model

Output data representation and granularity
Tabular Language Model

• As a result of (1), pre-trained tabular language model is developed
• Two major ways to be utilized:
  • Build on top of the encoder with more modules and fine-tuning
  • Use the encoder as part of a bigger architecture in a more task-oriented fashion rather than encoder-oriented (as embeddings feature)
• This model can be fine-tuned to learn the specifics of a downstream task, or it can be used as is with standard supervised machine learning algorithms
• Output representations can be extracted at different granularities:
  • Token
  • Cell
  • Row
  • Column
  • Column pairs (Doduo)
  • Table
  • Table pairs (Deco)
• While token and cell are the most common, the granularity is highly dependent on the target task
  • E.g.: Table representation for TR task
Consuming Tabular Language Models

(2) In

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>Sydney</td>
<td>25.69</td>
</tr>
<tr>
<td>France</td>
<td>Paris</td>
<td>67.39</td>
</tr>
<tr>
<td>Bolivia</td>
<td>La Paz</td>
<td>11.67</td>
</tr>
</tbody>
</table>

Question: What is the population number of France?
Prediction/Classification Systems

• Pre-trained transformer-based LM act as encoders of the input and typically:
  • Used as building block in a bigger system
  • Additional layers are added on top and the entire model is fine-tuned for a specific downstream task
Prediction/Classification Systems

- Sometimes LM are:
  - Employed as components of bigger system whose aim is to develop end-to-end trained system towards a certain task
  - Examples:
    - DTR (Herzig et al., 2021) compute a similarity score between embeddings of question and embedding of table
    - CLTR classifies whether an associated row/column with a given question contains the answer
Tutorial Outline

• Motivation
  • Natural Language and Data-centric Applications
• Language Models and Transformers
• Developing & Consuming Tabular Data Representation
  • Training Datasets
  • Input Processing
  • Model Training & Architecture
  • Tabular Language Model
  • Consuming Tabular LMs
• Open Challenges
Complex Queries and Rich Tables

• Few systems support aggregation operations such as max, min, avg
• No support for joins
• No support for dependencies
• No support for heterogeneity
  • E.g., columns with different measurement units such as adding kgs and lbs
Model Efficiency

• Transformers suffer from the upper bound limit of 512 tokens
  • Problem for large tables
• Multiple techniques to improve computation and memory usage
  • Locality sensitive hashing to replace attention
  • Approximate self-attention by a low-rank matrix
• New methods to make transformers more efficient for long context
  • only done on free text and not tabular data

(Treviso et al, 2022; Zaheer et al, 2020)
Benchmarks and Data Representations

• No benchmark datasets to establish baselines for tabular language models

• Current evaluation is extrinsic
  • Only considers the performance of the language model on the downstream tasks

• There is a need for intrinsic evaluation to evaluate the quality of those tabular representations
  • Checklist: generation of general linguistic capabilities and test types
  • We can design tests that evaluate properties of rows/columns/dependencies

(Ribeiro et al, 2020; Cappuzzo et al 2020)
Green Tabular LMs

• Large-scale transformers with billion of parameters requires heavy computation: several days of GPUs/TPUs for training
  • Contributes to global warming

• Need for new techniques that limit carbon footprint of tabular LMs without decrease in performance of downstream tasks

• One direction: reduce training data by removing redundant or less informative cells, tuples, tables
  • How to identify such data is a key challenge

(Yang et al 2009)
More general challenges

• Data bias
  • NLP LMs incorporate stereotypes + race, gender bias in the model parameters
    • Bias inherited from the dataset used for training the models
    • Reduce bias by preprocessing training dataset or postprocessing LMs

• Interpretability
  • How to justify the final output for a given task?
    • E.g., provide the cells that led to a given output (True/False)
    • Look at attention weights wrt input tokens to capture their influence on output

• Error Analysis
  • Most systems report only evaluation scores (p, r, accuracy)
  • no explanations for the cases where the model fails
    • for a QA task with a set of wrong answers, a pattern could explain misclassification
      • E.g., two column names having an overlap of more than 5 characters
Conclusions
### Representation learning for tabular data

<table>
<thead>
<tr>
<th>Task ID</th>
<th>Task Label</th>
<th>Tasks Coverage</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>TFC</td>
<td>Table-based Fact-Checking or Entailment</td>
<td>Fact-Checking Text Refusal/Entailment</td>
<td>Table - Claim</td>
<td>True/False Refusal/Entailed (Data Evidence)</td>
</tr>
<tr>
<td>QA</td>
<td>Question Answering</td>
<td>Retrieving the Cells for the Answer</td>
<td>Table - Question</td>
<td>Answer Cells</td>
</tr>
<tr>
<td>SP</td>
<td>Semantic Parsing</td>
<td>Text-to-SQL</td>
<td>Table - NL Query</td>
<td>Formal QL</td>
</tr>
<tr>
<td>TR</td>
<td>Table Retrieval</td>
<td>Retrieving Table that Contains the Answer</td>
<td>Table - Question</td>
<td>Relevant Table(s)</td>
</tr>
<tr>
<td>TMP</td>
<td>Table Metadata Prediction</td>
<td>Column Type Prediction, Table Type Classification, Header Detection, Cell Role Classification, Column Relation Annotation, Column Name Prediction</td>
<td>Table</td>
<td>Column Types, Table Types, Header Row, Cell Role, Relation between Two Cols, Column Name</td>
</tr>
<tr>
<td>DI</td>
<td>Data Imputation</td>
<td>Cell Content Population</td>
<td>Table with Corrupted Cell Values</td>
<td>Table with Complete Cell Values</td>
</tr>
</tbody>
</table>

#### Diagram:

- **Input**:
  - N× Positional Encoding
  - Input Embedding

- **Output**:
  - N× Positional Encoding
  - Output Embedding
  - Outputs (shifted right)
  - Add & Norm
  - Multi-Head Attention
  - Add & Norm
  - Masked Multi-Head Attention
  - Add & Norm
  - Feed Forward
  - Multi-Head Attention
  - Add & Norm
  - Fed Forward
  - Softmax

#### Additional Note:

- **DALL-E**
  - Collection: My collection
  - Edit the detailed description
  - muppet bert logo, scary, tough
Questions?
References


References


• (Gkini et al, 2021) Orest Gkini, Theofilos Belmpas, Georgia Koutrika, Yannis E. Ioannidis: An In-Depth Benchmarking of Text-to-SQL Systems. SIGMOD Conference 2021: 632-644


References


References


• (Wang et al., 2021b) Zhiruo Wang, Haoyu Dong, Ran Jia, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021. TUTA: Tree-based Transformers for Generally Structured Table Pre-training. In ACM SIGKDD. 1780–1790.


• (Yu et al., 2018) Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. In EMNLP. ACL, 3911–3921.


• (Hu et al., 2019) Kevin Hu, Snehalkumar’Neil’S Gaikwad, Madelon Hulsebos, Michiel A Bakker, Emanuel Zgraggen, César Hidalgo, Tim Kraska, Guoliang Li, Arvind Satyanarayan, and Çağatay Demiralp. 2019. VizNet: Towards a large-scale visualization learning and benchmarking repository. CHI
References

- (Dong et al., 2019) Haoyu Dong, Shijie Liu, Zhouyu Fu, Shi Han, and Dongmei Zhang. 2019. Semantic structure extraction for spreadsheet tables with a multi-task learning architecture. In Workshop on Document Intelligence at NeurIPS 2019


