Transformers for Tabular Data Representation: A Survey of Models and Applications

Gilbert Badaro Mohammed Saeed Paolo Papotti
EURECOM, France
{gilbert.badaro,mohammed.saeed,paolo.papotti}@eurecom.fr

Abstract

In the last few years, the natural language processing community has witnessed advances in neural representations of free texts with transformer-based language models (LMs). Given the importance of knowledge available in tabular data, recent research efforts extend LMs by developing neural representations for structured data. In this article, we present a survey that analyzes these efforts. We first abstract the different systems according to a traditional machine learning pipeline in terms of training data, input representation, model training, and supported downstream tasks. For each aspect, we characterize and compare the proposed solutions. Finally, we discuss future work directions.

1 Introduction

Many researchers are studying how to represent tabular data with neural models for traditional and new natural language processing (NLP) and data management tasks. These models enable effective data-driven systems that go beyond the limits of traditional declarative specifications built around first order logic and SQL. Examples of tasks include answering queries expressed in natural language (Katsogiannis-Meimarakis and Koutrika, 2021; Herzig et al., 2020; Liu et al., 2021a), performing fact-checking (Chen et al., 2020b; Yang and Zhu, 2021; Aly et al., 2021), doing semantic parsing (Yin et al., 2020; Yu et al., 2021), retrieving relevant tables (Pan et al., 2021; Kostić et al., 2021; Glass et al., 2021), understanding tables (Suhara et al., 2022; Du et al., 2021), and predicting table content (Deng et al., 2020; Iida et al., 2021). Indeed, tabular data contain an extensive amount of knowledge, necessary in a multitude of tasks, such as business (Chabot et al., 2021) and medical operations (Raghupathi and Raghupathi, 2014; Dash et al., 2019), hence, the importance of developing table representations. Given the success of transformers in developing pre-trained language models (LMs) (Devlin et al., 2019; Liu et al., 2019), we focus our analysis on the extension of this architecture for producing representations of tabular data. Such architectures, based on the attention mechanism, have proven to be successful as well on visual (Dosovitskiy et al., 2020; Khan et al., 2021), audio (Gong et al., 2021), and time series data (Cholakov and Kolev, 2021). Indeed, pre-trained LMs are very versatile, as demonstrated by the large number of tasks that practitioners solve by using these models with fine-tuning, such as improving Arabic opinion and emotion mining (Antoun et al., 2020; Badaro et al., 2014, 2018a,b,c, 2019, 2020). Recent work shows that a similar pre-training strategy leads to successful results when language models are developed for tabular data.[1]

As depicted in Figure 1, this survey covers both (1) the transformer-based encoder for pre-training neural representations of tabular data and (2) the target models that use the resulting LM to address downstream tasks. For (1), the training data consist of a large corpus of tables. Once the representation for this corpus has been learned, it can be used in (2) for a target task on a given (unseen at pre-training) table, such as the population table, along with its *context*, namely, relevant text information such as table header and captions. In most cases, the LM obtained from pre-training with large datasets in (1) is used in (2) by fine-tuning the LM with a labeled downstream dataset, for example, a table with cells annotated as the answer for the question in the context. Extensions on the typical transformer architecture are applied to account for the tabular structure, which is different and richer in some aspects than traditional free text.

[1]We refer to tabular ''language'' models with a slight abuse of the name, as the LM captures properties and relationships of the *structured data* rather than those of the *language*.

Transactions of the Association for Computational Linguistics, vol. 11, pp. 227–249, 2023. https://doi.org/10.1162/tacl_a_00544
Action Editor: Trevor Cohn. Submission batch: 5/2022; Revision batch: 9/2022; Published 3/2023.

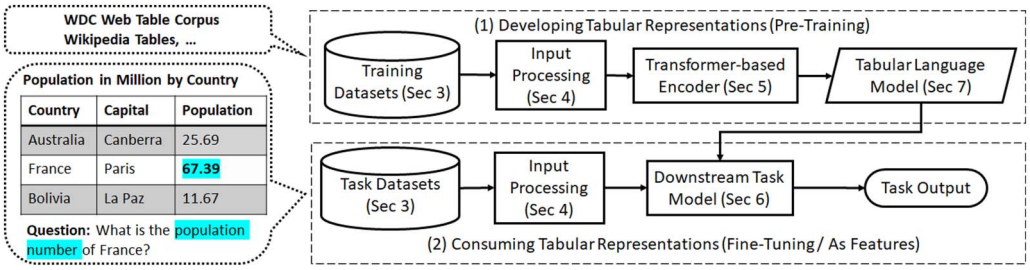

Figure 1: The overall framework for developing and consuming neural representations for tabular data. Wikipedia Tables or WDC Web Table Corpus are typically used in the pre-training step (1). In (2), a table along with its *context* (header, the additional question and the highlighted answer) are used for a Question Answering downstream task. Both processes combine the serialized table data with natural language text in the context.

While all proposed solutions make contributions to the neural representation of tabular data, there is still no systematic study to compare those representations given the different assumptions and target tasks. In this work, we aim to bring clarity in this space and to provide suitable axes for a classification that highlights the main trends and enables future work to clearly position new results. Our contributions are threefold:

1. We characterize the tasks of producing and consuming neural representations of tabular data in an abstract machine learning (ML) pipeline. This abstraction relies on five aspects that are applicable to all proposals and that are agnostic to methods' assumptions and final application (Section 2).
2. We describe and compare the relevant proposals according to the characteristics of the datasets (Section 3), the processing of the input (Section 4), and the adaptations of the transformer architecture to handle tabular data (Section 5).
3. To show the impact of the proposed solutions, we present six downstream tasks where they have achieved promising results (Section 6) and discuss how tabular language models are used by systems in practice (Section 7).

We conclude the survey with a discussion of the limitations of existing works and future research directions (Section 8).

Related surveys. Some recent surveys cover the use of deep learning on tabular data (Borisov et al., 2021; Gorishniy et al., 2021; Shwartz-Ziv and Armon, 2021). These surveys are different from ours in multiple aspects. First, they investigate how deep learning models (including non-transformer-based) compare against classical ML models (mainly gradient boosted trees) by providing empirical results with lack of details over the changes to the model, while our work taxonomizes transformer-based models in depth. Second, other works focus on standard classification and regression tasks where the term ''tabular data'' is used for labeled training data, representing data points with columns consisting of features. Every row is an input to the model together with a label (e.g., predicting house prices) (Somepalli et al., 2021). In contrast, we do not consider tabular data necessarily as a training dataset with features and labels, but rather as an input containing information needed for prediction by a target model, such as question answering on tables. For this reason, while they focus on data generation and classification, we relate design choices and contributions to a large set of downstream tasks. A recent survey (Dong et al., 2022) covers approaches for table pre-training and usage of LMs in downstream tasks with an overlap in terms of surveyed works. In contrast with their work, we provide a more detailed study of the extensions to the transformer architecture and analyze in detail the relevant datasets' properties and pre-processing steps.

2 Terminology and Overview

We focus on *tabular data* that come in the form of a *table*, as depicted in the two examples in Table 1. Table kinds differ in terms of horizontal/vertical orientation and with respect to the presence of hierarchies. A *relational table*, which is the most common kind, consists of *rows*, or records, and *columns* that together identify *cell values*, or cells. Columns represent *attributes* for a given table, and each row represents an instance having those attributes. This can be seen as a vertical orientation,

Player	Team	FG%
Carter	LA	56
Smith	SF	55

Player	Carter
Team	LA
FG%	55

Table 1: Examples of (vertical) *relational table* and (horizontal) *entity table*. A label 'Identity' on a top row spanning 'Player' and 'Team' would make the relational table a *spreadsheet*, similarly for the entity table with the same label spanning the first two rows.

where all cells in a column share the same atomic type, for example, the relational table in Table 1. An *entity table* has the same properties, but with a horizontal structure with only one entity whose properties are organized as rows. *Spreadsheets*, or matrix tables, are the most general kind with information that can be organized both horizontally and vertically, possibly with hierarchies in the header and formatting metadata, such as in financial tables.

Tables can have rich metadata, such as attribute types (e.g., DATE), domain constraints, functional dependencies across columns and integrity constraints such as primary keys. In the table sample in Figure 1, column *Country* is a primary key. Most systems focus on a single table, with or without metadata. However, a few systems, such as GTR (Wang et al., 2021a), GRAPPA (Yu et al., 2021), and DTR (Herzig et al., 2021), consume *databases*, which are collections of relational tables, possibly under referential constraints. We identify as input the table(s) and its *context*. The context is a text associated to the table. Depending on the dataset and task at hand, it varies from table metadata, the text surrounding the tables or their captions up to questions, expressed in natural language, that can be answered with the tabular data (Badaro and Papotti, 2022).

The main advantage of the transformer architecture is its ability to generate a LM, a large neural network, with self-supervised pre-training. This pre-trained LM is then usually followed by supervised fine-tuning to adapt it to the target task with a small amount of training data. While transformers have proven to be effective in modeling textual content, tables have a rich structure that comes with its own relationships, such as those across values in the rows and attributes. New solutions are therefore needed to jointly model the characteristics of the table, its text content, and the text in the table context. As shown in Figure 1, we distinguish two main phases to spell out these contributions. First, we focus on the development of tabular LMs by using transformer-based deep neural networks (1). Given a table and its context, the goal is to learn a pre-trained representation of the structured data (cell values, rows, attributes) in a continuous vector space. We then discuss the use of those representations in the downstream tasks (2).

Figure 1 shows the reference pipeline and the aspects that we propose to model existing systems.

- **Training Datasets** (Sec. 3): the datasets used for pre-training and fine-tuning the models toward specific tasks; datasets for the latter case usually come with annotations and/or labels.
- **Input Processing** (Sec. 4): the steps to prepare the data for the model processing, such as the transformation from the two dimensional tabular space to one dimensional input.
- **Transformer-based Encoder** (Sec. 5): the pre-training objectives and customization of the typical transformer-based deep learning architecture.
- **Downstream Task Model** (Sec. 6): the models consuming the representations or fine-tuning them to tackle downstream tasks.
- **Tabular Language Model** (Sec. 7): the output representations, including at the token, row, column, table level, and their usage.

For the tasks consuming the models, we report on Table based Fact-Checking (TFC), Question Answering (QA), Semantic Parsing (SP), Table Retrieval (TR), Table Metadata Prediction (TMP), and Table Content Population (TCP).

3 Training Datasets

We present both the datasets used for pre-training and for fine-tuning in the downstream tasks. Pre-training tables are not annotated, in some cases scraped from the web, while data used for fine-tuning have task-dependent annotation labels. The datasets consist of tables and their context, such as table metadata, surrounding texts, claims or questions. To construct large pre-training

datasets and in an attempt to reduce bias, multiple sources can be used, independently of the target task at hand. For instance, unlike TAPAS (Herzig et al., 2020), which only uses Wikipedia Tables for QA, and TABULARNET (Du et al., 2021), which only uses Spreadsheets for TMP, TABERT (Yin et al., 2020) uses Wikipedia Tables and WDC for SP; GRAPPA uses Wikipedia Tables, Spider, and WikiSQL for SP; MMR (Kostić et al., 2021) uses NQ, OTT-QA, and WikiSQL for TR; MATE (Eisenschlos et al., 2021) uses Wikipedia Tables and HybridQA for QA; and TUTA (Wang et al., 2021b) uses Wikipedia Tables, WDC, and spreadsheets for TMP. In general, it is recommended to utilize different data sources for pre-training to ensure covering different kinds and content, and thus, improve the scope of representations. For instance, Wikipedia tables has a large number of relational tables (Bhagavatula et al., 2015), while WDC and Spreadsheets include also entity tables and spreadsheets with complex structure.

Table 2 summarizes the main characteristics for the most common datasets. We mark the tasks for which the dataset has been used by ✔ under the column ''Task''. We note that the top four datasets are mostly used for pre-training, while the others can be used for fine-tuning as well since they include annotations for the target task, for example, questions/answers for QA. The column ''Large Tables'' is a binary indicator, where ✔ and ✘ indicate whether or not the tabular corpus include large tables and hence whether or not some pre-processing is needed to reduce table content to meet the limits of the transformer architecture (512 input tokens in most cases). Some works, such as TABERT, TAPEX (Liu et al., 2021a), and CLTR (Pan et al., 2021), apply filtering in any case to reduce noisy input. Finally, the ''Context'' column describes additional text that come with the tables. This can be text describing the table, such as caption or title of the document containing the table; table metadata, such as table orientation, header row, and keys; or questions and claims that can be addressed with the table.

4 Input Processing

As for the original text setting, transformers for tabular data require as input a sequence of tokens. However, in addition to the typical tokenization executed before feeding the text to the neural network (Lan et al., 2020), tabular data requires some steps to be processed correctly. Some requirements come from the nature of the transformer, such as the limitation on the input data size (Section 4.1). Other requirements are due to the nature of the tabular data, with structural information expressed in two dimensions that need to be converted into a one dimensional space (Section 4.2). Finally, given a table, its data and its context must be jointly fed to the transformer (Section 4.3).

4.1 Data Retrieval and Filtering

Filtering methods on the table content are applied to stay within the size limits of the transformer architecture, to reduce the model training time, and to eliminate potential noise in the representation.

TABERT uses content snapshot to keep the top-k most relevant rows in the table. Such content is identified with the tuples with highest *n*-gram overlap with respect to the given context (question or utterance). TAPEX and the retrieval model in FEVEROUS (Aly et al., 2021) randomly select rows to limit the input size, while RCI (Glass et al., 2021) down-samples rows using term frequency inverse document frequency (TF-IDF) scores; the frequency can also be used to summarize cells with long text (Li et al., 2020). In addition to keeping tables with number of columns below a fixed threshold, TUTA and TABULARNET split large tables into non-overlapping horizontal partitions. Every partition contains the same header row and it is processed separately by the model. While effective, splitting is more demanding in terms of computation cost.

In terms of table selection, whereas for systems like GTR and DTR the objective is to retrieve tables that contain the answer to a given question, others, such as CLTR, use a ranking function, such as BM25 (Robertson et al., 1995), to retrieve relevant tables prior to training, or to generate negative examples, as in MMR. Regardless of the downstream task, most systems filter and reduce the size of the input data to meet the limits of transformers technology. However, frequency-driven sampling, such as in RCI, is more effective than random as it reduces noise as well in data representations.

To summarize, one can group the different content selection strategies based on the targeted downstream task. For TFC, QA, and SP, a selection strategy relying on *n*-gram overlap such as content snapshot or TF-IDF is recommended,

Dataset	Reference	Used for Task						Number of Tables	Large Tables	Context	Application Example
		TFC	QA	SP	TR	TMP	TCP				
Wikipedia Tables	Wikipedia		✔	✔	✔	✔	✔	3.2M	✔	**Surrounding Text:** table caption, page title, page description, segment title, text of the segment. **Table Metadata:** statistics about number of headings, rows, columns, data rows.	TAPAS
WDC Web Table Corpus	(Lehmberg et al., 2016)		✔	✔		✔		233M	✔	**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.	TABERT
VizNet	(Hu et al., 2019)					✔	✔	1M	✘	**Table Metadata:** Column Types.	TABBIE
Spreadsheets	(Dong et al., 2019)					✔		3,410	✘	**Table Metadata:** Cell Roles (Index, Index Name, Value Name, Aggregation and Others).	TABULARNET
NQ-Tables	(Herzig et al., 2021)		✔		✔			169,898	✔	**Questions:** 12K.	DTR
TabFact	(Chen et al., 2020b)	✔						16K	✘	**Textual Claims:** 118K.	DECO
WikiSQL	(Zhong et al., 2017)		✔	✔	✔			24,241	✘	**Questions:** 80,654.	MMR
TabMCQ	(Jauhar et al., 2016)		✔		✔			68	✘	**Questions:** 9,092.	CLTR
Spider	(Yu et al., 2018)			✔				200 databases	✘	**Questions:** 10,181 **Queries:** 5,693.	GRAPPA
WikiTable Question (WikiTQ)	(Pasupat and Liang, 2015)		✔	✔				2,108	✘	**Questions:** 22,033.	TAPEX
Natural Questions (NQ)	(Kwiatkowski et al., 2019)			✔		✔		169, 898*	✔	**Questions:** 320K.	MMR
OTT-QA	(Chen et al., 2021)		✔		✔			400K	✔	**Surrounding Text:** page title, section title, section text limited to 12 first sentences. **Questions:** 45,841.	MMR
Web Query Table (WQT)	(Sun et al., 2019)				✔			273,816	✘	**Surrounding Text:** captions. **Queries:** 21,113.	GTR
HybridQA	(Chen et al., 2020c)		✔					13K	✘	**Questions:** 72K. **Surrounding Text:** first 12 sentences per hyperlink in the table.	MATE
FEVEROUS	(Aly et al., 2021)	✔						28.8K	✔	**Textual Claims:** 87K. **Surrounding Text:** article title. **Table Metadata:** row and column headers.	FEVEROUS

Table 2: Datasets for the development and evaluation of neural representation models of tabular data. The top four datasets are mostly used for pre-training models, the rest of the datasets also come with a context and labels that are used in the downstream task for training and evaluation. For Large Tables, ✔/✘ denotes whether or not the dataset includes large tables and thus requiring pre-processing to meet transformers' limits. Application example refers to sample systems using the respective dataset. For the task, we use the acronyms as follows. TFC: Table based Fact-Checking, QA: Question Answering, SP: Semantic Parsing, TR: Table Retrieval, TMP: Table Metadata Prediction, TCP: Table Content Population. *: the number of tables is derived from Herzig et al. (2021).

while for TR, BM25 is endorsed. In the cases of TMP and TCP, where having the full content of the table is required, such as relation extraction or cell filling, splitting the tabular data without any selection is adopted.

4.2 Table Serialization

A crucial step is the transformation from the two dimensions of a table to its serialized version consumable by the transformer. The methods for table serialization can be grouped into four main types. The first type consists of horizontally scanning the table by row. Most systems achieve this task with a flattened table with value separators, for example, DTR, MMR, TURL (Deng et al., 2020), TAPAS, MATE, and DECO (Yang and Zhu, 2021). For the table in Figure 1, it corresponds to a sequence such as [CLS] Population in Million by Country | Country | Capital | Population | Australia | Canberra | 25.69 . . . Bolivia | La Paz | 11.67.

Context and Table parsed by row:
[CLS] *Population in Million by Country* [CLS] Country | Capital | Population [SEP] Australia | Canberra | 25.69 [SEP] France | Paris | 67.39 [SEP] ...

Context and Table parsed by column:
[CLS] *Population of Countries* [CLS] Country | Australia | ... | Bolivia | ... [SEP] Capital | Canberra | ... | La Paz | ... [SEP] Population | 25.69 | ...

Figure 2: Examples of context (table caption, in italic) concatenated with row and column data linearization.

Another option is a flattened table with special token separators to indicate the beginning of a new row or cell (TAPEX, TUTA, [FORTAP (Cheng et al., 2022)]), as in the first example in Figure 2. Finally, few methods flatten the table representing each cell as a concatenation of column name, column type, and cell value (TABERT), for example, Country | String | Australia [SEP]; or just the row of column headers (GRAPPA).

The second linearization type scans the table by column, again either by simple concatenation of column values or by using special tokens as separators, as in DODUO (Suhara et al., 2022). A column serialization for the country population table is the second example in Figure 2.

The third linearization type consists of combining the output from both types of serialization by using element-wise product (RCI, CLTR), average pooling and concatenation (TABULARNET), or the average of row and column embeddings (TABBIE (Iida et al., 2021)). In a context outside of our framework, which focuses on transformers, it has also been proposed to transform the input relation table in a graph and perform random walks on the latter to ultimately produce node embeddings (Cappuzzo et al., 2020).

The fourth type consists of using a text template to represent the tabular data as sentences (Chen et al., 2020b; Suadaa et al., 2021; Chen et al., 2020a). Instead of using predefined templates, natural sentences are generated out of the tabular data, by fine-tuning sequence to sequence language models such as T5 (Raffel et al., 2020) or GPT2 (Radford et al., 2019), as in DRT (Thorne et al., 2021; Neeraja et al., 2021). The generation can rely on models for this *table-to-text* task, such as TOTTO (Parikh et al., 2020), PYTHIA (Veltri et al., 2022; 2023), TABLEGPT (Gong et al., 2020), LOGIC2TEXT (Chen et al., 2020d), UNIFIEDSKG (Xie et al., 2022), or other efforts (Suadaa et al., 2021; Chen et al., 2020a,e). However, most of these methods show limitations for tables with prevalence of numerical attributes and with missing table context. Within the efforts for table-to-text, graph traversal algorithms have been explored for linearization such as relation-biased breadth first search (Li et al., 2021). While table-to-text generation is an important and challenging task, most methods rely on fine-tuning existing pre-trained language models.

While it is still not clear which table serialization method should be used for a given task, few papers performed ablation studies. For instance, TABFACT (Chen et al., 2020b) does not report any significant difference in performance when comparing row and column encoding. TABERT reports that both (i) adding type information for cell values and (ii) phrasing the input as a sentence improve results. The most promising approach is to incorporate several aspects by appending column headers to cell content, combining row and column encoding, or by adding structure-aware indicators as the positional embeddings discussed in Section 5. Finally, there is evidence that textual templates to represent table content is a valid solution when one directly fine-tunes existing pre-trained language models, without performing pre-training on tabular data (Suadaa et al., 2021).

4.3 Context and Table Concatenation

When available, the table context is concatenated with the table content. Most systems (TABERT, TAPAS, DTR, GRAPPA, and DECO) combine the context by concatenating it in the serialization before the table data, while TAPEX appends it. The authors of UNIFIEDSKG found that placing context before the tabular knowledge yields to better results in their setting. In some cases, including CLTR, MMR, and RCI, the table and the context are encoded separately and then are combined at a later stage in the system.

Some works do not include context in their pre-training input besides the column headers (TABBIE, TABULARNET, DODUO, GRAPPA). Typically, the decision is based on the target downstream task. A richer context is used when the tasks are closer to the corresponding NLP task applied on free text. For instance, all models for QA use table captions or descriptions as context. To summarize, including context is crucial for TFC, QA, SP, TR, while it can be excluded for TMP and TCP. In case the context is needed,

prepending or appending it to the table content does not modify the performance of the model.

5 Adaptation of Transformers

To account for structured tables in the input, several pre-trained transformer-based LM and systems have been developed. Vanilla LMs are customized to make the model more ''data structure-aware'', thus rendering a modified transformer-based encoder to be utilized on other tasks. These encoders, depicted in part (1) of Figure 1, capture both structure and semantic information. To exploit such resources and build applications, as in part (2) of Figure 1, several systems build on top of the encoder, usually with more modules and fine-tuning. In a different approach, other systems use the encoder as part of a bigger architecture in a more task-oriented fashion rather than encoder-oriented. We first briefly revise the vanilla transformer architecture, then discuss customizations to LMs.

5.1 Vanilla Transformer

The vanilla transformer (Vaswani et al., 2017) is a seq2seq model (Sutskever et al., 2014) consisting of an encoder and a decoder, each of which is a stack of N identical modules. The encoder block is composed of a multi-head self-attention module and a position-wise feed-forward network. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Residual connections and layer-normalization modules are also used. Decoder blocks consist of cross-attention modules between the multi-head self-attention modules and the position-wise feed-forward networks, where masking is used to prevent each position from attending to subsequent positions.

The transformer architecture can be used as an encoder-decoder (Vaswani et al., 2017; Raffel et al., 2020), an encoder-only (Devlin et al., 2019; Liu et al., 2019), or decoder-only (Radford et al., 2019; Brown et al., 2020) model. The choice of the architecture depends on the final task. Encoder-only models are mainly used for classification and are the most popular choice for extensions for tabular data. In this case, pre-training is done with a masked language modeling (MLM) task, whose goal is to predict masked token(s) of an altered input. The encoder-decoder architecture is used for models that focus on sequence generation tasks (RPT, TAPEX).

5.2 LM Extensions for Tabular Data

To properly model the structure of data in tables, the vanilla transformers are extended and updated by modifying components at the (i) input, (ii) internal, (iii) output, and (iv) training-procedure levels. We discuss each of them in the following. A summary of the extensions is provided as a taxonomy in Figure 3. We note that since the encoder and decoder modules of a transformer share similar structure, most of these modifications can be applied to both.

5.2.1 Input Level

Modifications on the input level are usually designated with **additional positional embeddings** to explicitly model the table structure. TABERT and TAPAS show how such embeddings improve performance on structure-related tasks. For example, embeddings that represent the position of the cell, indicated by its row and column IDs, are common for relational tables, for example, in TABERT, TAPAS, and TABBIE. TABLEFORMER (Yang et al., 2022) drops row and column ids to avoid any potential row and column biases and they instead use per cell positional embeddings similar to MATE. For tables without a relational structure, such as entity tables and spreadsheets, including complex financial tables, TUTA introduces tree-based positional embeddings to encode the position of a cell using top and left embeddings of a bi-dimensional coordinate tree. Other supplementary embeddings include those that provide relative positional information for a token within a caption/header (TURL) or a cell (TUTA). For tasks such as QA, segment embeddings are used to differentiate between the different input types, question and table, for example, in RPT (Tang et al., 2021) and TAPAS. Finally, TUTA introduces embeddings for numbers when discrete features are used.

While row/column positional embeddings can better map context and table content, such as in QA or TFC tasks, it cannot overcome the challenge of empty cells, nested row headers, or descriptive cells in complex spreadsheets such as financial tables. In this case, tree-based positional embeddings are required. Aside from new positional embeddings encoding the structure of a token in a table, original, vanilla positional embeddings can also be modified for a better representation of

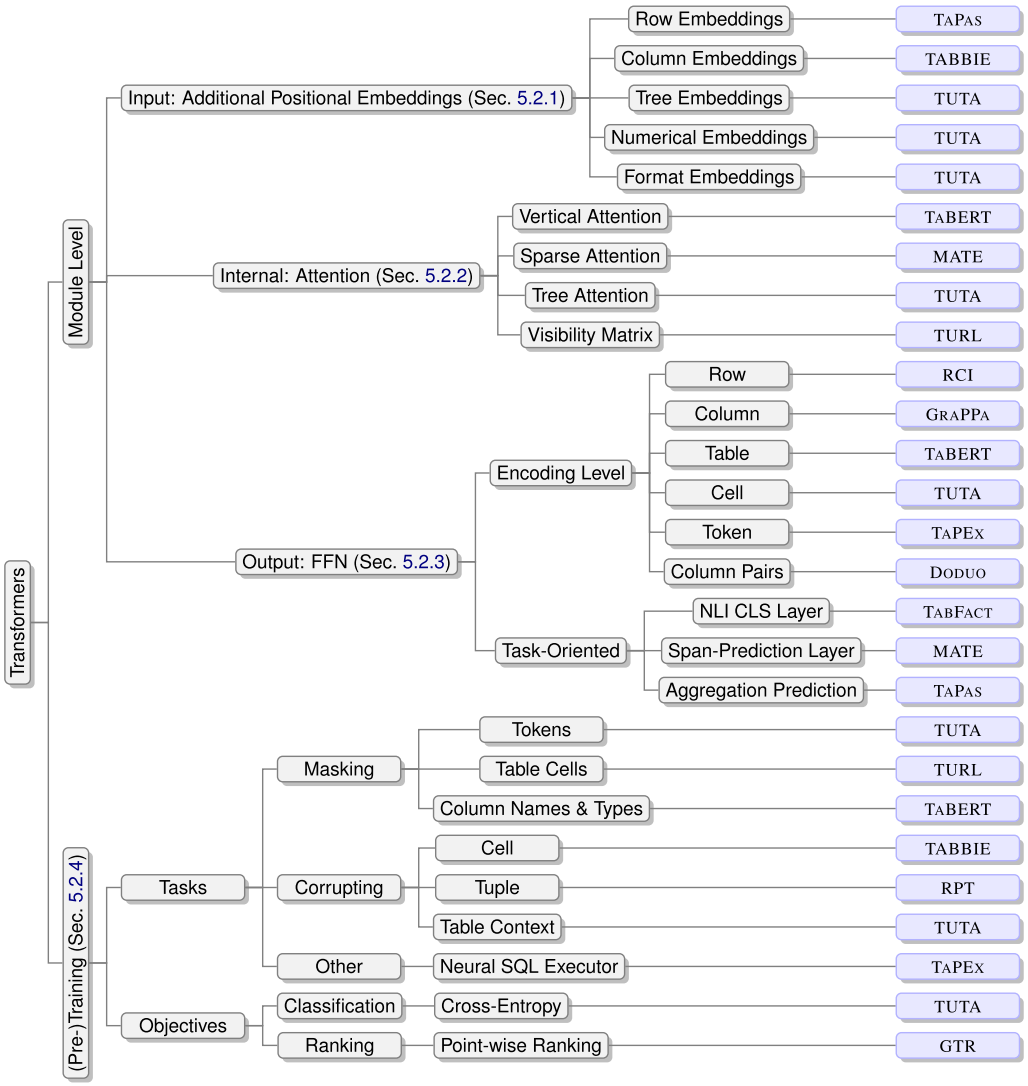

Figure 3: Taxonomy of extensions to transformer-based LMs for handling tabular data. We report as leaves a representative system for every extension.

tokens within table cells. For example, TABERT employs a pre-training task that explicitly uses positional embeddings, alongside a cell representation, for the goal of recovering a cell value. In this case, the original positional embeddings help better model multiple tokens in a cell.

5.2.2 Internal Level

Most of the modifications on the internal level are applied to make the system more ‘‘structure-aware’’. Specifically, the **attention** module is updated to integrate the structure of the input table. For example, in TABERT vertical self-attention layers are produced to capture cross-row dependencies on cell values by performing the attention module in a vertical fashion. Empirical results in TUTA show that employing row-wise and column-wise attention, instead of having additional positional embeddings for rows and columns, hurts model performance for cell-type classification tasks, but it is not the case for table-type classification tasks.

Other systems, such as TURL, employ a masked self-attention module, which attends to structurally related elements such as those in the same row or column, thus ignoring the other elements, unlike the traditional transformer where each element attends to all other elements in the sequence. Moreover, for categorical data, named entities, such as city names, can be identified in the cell values. In TURL and TUTA, masking such entities helps the models capture the factual knowledge embedded in the table content as well as the associations between table metadata and table content. Ablation studies showed the positive impact of these two modifications on model performance.

Other modifications address the input size constraint of attention modules, where large tables are often neglected. Sparse attention methods are proposed to cope with this issue (Tay et al., 2020). For instance, MATE sparsifies the attention matrix to allow transformer heads to efficiently attend to either rows or columns.

Modifications to the input level, by additional input embeddings to encode cell positions (TaPas), help in tasks requiring information at the cell level, such as QA and cell type classification. However, tasks requiring understanding of table structure, such as table type classification, benefit more from modifying the attention module (TaBERT). Systems modifying both, such as TUTA, seem to be the best option. However, modifying the internal level is more effective, as removing such modification leads to the larger decrease in performance.

5.2.3 Output Level

Additional layers can be added on top of the feed-forward networks (**FFNs**) of the LM depending on the task at hand. Tasks such as cell selection (TaPas), TFC (TabFact), TMP (Doduo), and QA (TaPas) require to train one or more additional layers. Classification layers for aggregation operations and cell-selection are used to support aggregating cell values (GraPPa, MATE, Doduo, Deco, RCI). In TaPEx, aggregation operations are also ''learned'' end-to-end in a seq2seq task.

5.2.4 Training-Procedure Level

Modifications on the training-procedure level can be attributed to the pre-training task and objective.

Pre-training Tasks. Systems are either trained end-to-end or fine-tuned after some pre-training procedure. Almost all pre-training tasks fall under the category of reconstruction tasks, where the idea is to reconstruct the correct input from a corrupted one. Rather than applying traditional token-level MLM on the serialized tabular data, effectively treating it as natural language sequences, most pre-training tasks are designed to consider the information about the table structure.

Typically, traditional masking of the tokens is used for both, the textual context surrounding the table and the table cells (e.g., TaPas 15%, TURL 20%) with some novelties for table cells such as masked columns (TaBERT) and masked entities (TURL). Specifically, TaPas follows the strategy to mask the whole cell table whenever a token in that cell is randomly masked. Inspired by TaPas, TUTA masks 15% of the cells by randomly selecting those that consist mainly of text, rather than numerical values. Among these selected cells, 30% are masked completely, while for 70% only a single token within the cell is masked. Masking column names and data types for relational tables encourages the model to recover the metadata, while masking tokens and whole cell values ensure that data information is retained across the different vertical self-attention layers. Indeed, masking at the entity level enables the model to integrate the factual knowledge embedded in the table content and its context.

In TABBIE and RPT, the pre-training tasks detect if a cell/tuple is corrupted or not. While corruption has been shown to outperform MLM for standard textual LMs (Clark et al., 2020), it is not clear if the benefit generalizes to tabular setting. In TUTA, text headers of tables are used to learn representations of tables in a self-supervised manner, i.e., a binary classification task where a positive example is the table and its associated header, and a negative example is any other header.

While most systems pre-train for the purpose of learning initial tabular representations, performing aggregations and numerical operations is usually adapted by fine-tuning a classifier for predicting cells and operations, e.g., in TaPas. However, systems such as GraPPa and TaPEx pre-train with SQL queries. In GraPPa, the pre-training task enables the discovery of suitable fragments for SP by pre-training on a large number of queries, with SQL logic grounding to each table, which might help generalize to other similar queries. Empirical results show a decrease in performance without this pre-training task. TaPEx enforces the LM to emulate a SQL engine by pre-training on sampled tables and synthesized queries. This pre-training task gives (i) the LM a deeper understanding of the tables, as the system encounters operations such as selection and aggregation rendering its representations 'aware' of such operations, and (ii) improves the performance in QA.

According to (Chang et al., 2020), a pre-training task should be cost-efficient, ideally not requiring additional human supervision. As manually annotating queries with tables is costly, using surrounding information, such as the table header or caption, as a query acts as a self-supervised signal, helping the system learn better representations for the tables. It is the case for TR systems, such

System	Module Level																			(Pre-)Training									
	Input:					Internal:				Output: FFN										Task							Objective		
	Addition. Pos. Emb.					Attention				Encod. Level						Task-Orien				Mask			Corrupt			Oth	Cls	Rnk	
	Ro	Co	Tr	Nu	Fr	Vrt	Sp	Tr	Vi	Ro	Co	Tb	Ce	To	CP	Nli	Sp	Ag	To	Ce	Co	TC	Tu	Ce	Nse	CE	PR		
TUTA			X	X	X			X				X	X						X			X				X		
TURL									X				X	X					X	X						X		
TABERT						X						X	X	X						X	X			X		X		
TABBIE	X	X								X	X		X											X		X		
MATE	X	X					X						X	X			X		X	X		X		X		X		
RCI										X	X																	
GRAPPA											X								X		X							
TAPEX										X			X	X											X			
DODUO											X				X											X		
TABFACT														X		X			X							X		
RPT													X	X					X	X				X		X		
TAPAS	X	X											X	X				X	X	X						X		
GTR												X										X						X

Table 3: Transformer customizations for every system. The acronyms from left to right follow the same order of the leaves of the taxonomy tree in Figure 3 from top to bottom.

as DTR and GTR, where performance improves when using a pre-training task rather than when training from scratch.

To summarize, token-level masking is the base for all the systems and it can be sufficient for TFC and TR tasks, while cell or entity masking are also recommended for QA, TMP, such as cell role classification, and TCP, such as cell filling. Column masking is additionally suggested for SP since it helps in identifying the columns to formulate the logical queries.

Pre-training Objectives. The objective of the majority of the systems is to minimize cross-entropy loss for a certain classification task. GTR utilizes a point-wise ranking objective for end-to-end training after pre-training, where multiple tables are ranked according to a relevance score given a certain query.

Table 3 reports a summary of the transformer customizations adopted by every system. Overall, a tabular LM requires modifications to the input level, through additional embeddings, and to the internal level, through adjustments of the attention module. Pre-training on table-related tasks, such as masking or corrupting table cells, also enhances encoding capabilities for structured data on fine-tuned tasks. Modifications on the output level are more task specific and have less impact on making the LM ‘‘understand’’ the data structure.

6 Downstream Tasks

Using neural representations for tabular data show, improvements in performance in several downstream tasks. In this section, we describe the tasks and define their input and output. While they all consume tables, settings can be quite heterogeneous, with systems exploiting different information, even for the same task. A summary of the covered tasks along their input, output, and some representative systems addressing them is shown in Table 4. We detail next the mandatory input elements and the different contexts.

Table-based Fact-Checking (TFC): Similar to text-based textual entailment (Dagan et al., 2013; Korman et al., 2018), checking facts with tables consists of verifying if a textual input claim is true or false against a trusted database (TAPEX, DECO, TABFACT), also provided as input. Some fact-checking systems, such as FEVEROUS, also output the cells used for verification as evidence (Nakov et al., 2021; Karagiannis et al., 2020).

Question Answering (QA): In the free text setting, QA aims at retrieving passages that include the answer to a given question. In the tabular data setting, it consists of returning as output the cells that answer an input consisting of a question and a table. One can distinguish two levels of complexity. Simple QA involves lookup queries on tables (DTR, CLTR), while a more complex QA

Task ID	Task Label	Tasks Coverage	Input	Output	System Examples
TFC	Table-based Fact-Checking	Fact-Checking Text Refusal/Entailment	Table + Claim	True/False Refused/Entailed (Data Evidence)	DECO TABFACT TAPEX
QA	Question Answering	Retrieving the Cells for the Answer	Table + Question	Answer Cells	TAPAS MATE DTR
SP	Semantic Parsing	Text-to-SQL	Table + NL Query	Formal QL	TABERT GRAPPA TAPEX
TR	Table Retrieval	Retrieving Table that Contains the Answer	Tables + Question	Relevant Table(s)	GTR CLTR
TMP	Table Metadata Prediction	Column Type Prediction Table Type Classification Header Detection Cell Role Classification Column Relation Annotation Column Name Prediction	Table	Column Types Table Types Header Row Cell Role Relation between Two Cols Column Name	DODUO TABULARNET TURL TUTA
TCP	Table Content Population	Cell Content Population	Table with Corrupted Cell Values	Table with Complete Cell Values	TURL TABBIE RPT

Table 4: List of tasks utilizing neural representations for tables.

task involves aggregation operations and numerical reasoning (TAPAS, MATE, RCI, DRT, TAPEX). Most of the systems in this survey aim at improving accuracy in QA with respect to hand-crafted embeddings.

Semantic Parsing (SP): In the tabular data setting, given a question and a table as input, SP generates a declarative query in SQL over the table's schema to retrieve the answer to the question. While in QA the interest is in directly getting the answer, SP produces the (interpretable) query to obtain it (TABERT, GRAPPA, TAPEX).

Table Retrieval (TR): Given a question and a set of tables as inputs, TR identifies the table that can be used to answer the question. TR is helpful when trying to reduce the search space for a QA task (GTR, DTR, MMR). It is a challenging task given the limited input size of transformers, that is, their constraint to sequences of 512 tokens.

Table Metadata Prediction (TMP): Given an input table with corrupted or missing metadata, the TMP objective is to predict *inter*-table metadata, such as column types and headers, cell types, table types, and *intra*-tables relationships, such as equivalence between columns and entity linking/resolution. Relevant efforts focus both on spreadsheets (TUTA, TABULARNET) and relational tables (TURL, DODUO).

Table Content Population (TCP): Unlike TMP, where the table metadata is noisy or missing, TCP deals with corrupted cell content. Given an input table with missing cell values, the objective is to impute the respective values (RPT, TABBIE, TURL).

We observe that most tasks can be seen as traditional NLP problems where structured data replace free text, such as the case of QA where answers are located in tabular data instead of documents (Gupta and Gupta, 2012). TFC involves retrieving cells that entail or refute a given statement, whereas on free text the corresponding objective is to select sentences as evidence (Thorne et al., 2018). SP is the task of converting natural language utterance into a logical form (Berant and Liang, 2014), which in this setting is expressed as a declarative query over a relational table. TR on tabular data corresponds to passage retrieval on free text (Kaszkiel and Zobel, 1997). TCP is analogous to predicting missing words or values in a sentence (Devlin et al., 2019). Finally, TMP can be related to syntactic parsing in NLP (Van Gompel and Pickering, 2007), where relationships between different tokens are depicted.

We conclude this section with an analysis of the performance of the systems over the different downstream tasks. For every task, we selected datasets for which at least two systems have reported results. All datasets reported in Table 5 are described in Table 2, with the exception of WCC

System (size)	TFC	QA			SP	TR	TMP		TCP
	TabFact accu.	HybridQA accu./F1	WikiTQ accu.	WikiSQL accu.	Spider MAP	WQT MAP	WCC F1	VizNet F1	EntiTab MAP
TABERT (367M)			53.5*	70.9*	65.2	63.4	83.6*	**97.2***	33.1*
TAPAS (340M)	81.2*	62.7/70.0	48.8	86.4					
MATE (340M)	81.4	**62.8/70.2**	51.5						
TAPEX (406M)	**84.2**		**57.5**	89.5					
TABBIE (170M)								96.9	**37.9**
RCI (235M)				**89.8**					
GRAPPA (355M)			52.1*		**69.6**				
DODUO (110M)								94.3	
DECO (1454 M)	82.7								
TUTA (134M)							**87.6**		
GTR (117M)						**73.7**			

Table 5: Performance table of results reported from the original system papers for every downstream task and different datasets. Metrics (accuracy, F1 measure, MAP) change across datasets. Results reproduced by follow-up papers are denoted with *. Reported sizes are for the largest model of every system.

(Ghasemi-Gol and Szekely, 2018), which contains web tables annotated with their type (relational, entity, matrix, list, and non-data), and EntiTab (Zhang and Balog, 2017), which contains web tables annotated with possible header labels for the column population task. Table 5 also contains the size, expressed as number of parameters, of the largest model used by every system. As some systems are not comparable on any shared datasets, we report here their size: TABFACT (110M), MMR (87M), TURL (314M), RPT (139M), and CLTR (235M). Larger models do not correlate with better performance across different systems for the same task, but a larger model always brings higher scores for the same system, as expected. Execution times for training and testing depend on the size of the model and the computing architecture.

The results in the table show that some tasks, such as TFC and TMP, can already be handled successfully by the systems, while some tasks, such as TCP, TR, and SP, are harder. QA is the task supported by most systems and the quality of the results vary depending on the dataset at hand. Differences in performance can be explained with different improvements across the systems. For example, MATE has better performance with respect to TAPAS in two tasks because of its mechanism to deal with larger input tables. Similarly, TUTA improves over TABERT because it handles tabular data beyond relational tables. Finally, TABERT and TAPAS are the systems that show most coverage in terms of tasks, with multiple papers using them as baselines in their experiments. For the systems that are not reported in Table 5, we notice that TURL obtains similar F1 results for column type prediction, but on a dataset different from VizNet; TURL, TABBIE, and TABERT also report comparable MAP results for the row population task (not in Table 5) over different datasets.

7 Using the Language Model

The initial neural representations of tabular data are the result of the pre-training. Systems use the output LM in different ways. It can be fine-tuned or used ‘‘as-is’’—for example, by using its representations as features in traditional ML algorithms (Section 7.1). However, as pre-trained LMs can act as encoders of the input, they are also used as a module in bigger systems (Section 7.2).

7.1 Encoder Output and its Usage

The alternative systems expose different granularity of the output table representation as embeddings. Almost all systems provide token and cell output embeddings. Most of them also expose column and row embeddings, while few provide table and pairwise column (DODUO) or pairwise table (DECO) embeddings. When a special separator token separates the context and the

table content, a representation of the context is also provided (TABFACT, CLTR, DTR, MMR).

These representations can be used in an arbitrary ML program simply as features or directly by the systems creating them to tackle a given task (Section 6). Indeed, as we discussed in Section 5.2.3, most systems use the pre-trained embeddings for further fine-tuning to tackle the tasks. We observe a relationship between the granularity of the output representations and the target downstream tasks. For instance, table representations are used for the TR task, while column-pairs representations are used for the TMP task to support columns relation annotation. Cell representations are used for the QA task, since the cells including the answer should be returned. Finally, column representations are used for the SP task, as columns are needed to formulate the output query.

Most pre-trained models, such as TAPAS, are usually available out-of-the-box, while in some cases, such as in TURL, MATE, and CLTR, the users have to retrain the models. For such systems, the code and the dataset are available, but the users need to run the training on their side to generate the representations (no checkpoints available).

7.2 Encoder as a Module

Several systems, such as TUTA and TURL, add layers on top of the LMs, which are then fine-tuned for a task. While this is a common use of pre-trained LMs, other works employ LMs as components in a larger system. In these cases, the system not only needs to learn how to encode properly, but also to adjust its representations to a certain task by training end-to-end together multiple components, such as the ones that (individually) generate the embeddings for tables and text with the one for scoring similarity of textual and tabular embeddings.

Most of these larger systems focus on the retrieval of tables from an input natural language query. DTR answers a natural language question from a corpus of tables in two steps. First, it retrieves a small set of candidate tables (TR), where encoding of questions and tables are learned through similarity learning. The similarity score is obtained through an inner product between questions and table embeddings. Then, it performs the standard answer prediction (QA) with each candidate table in the input. A recent study (Wang et al., 2022) claims that table-specific models may not be needed for accurate table retrieval and that fine-tuning a general text-based retriever leads to superior results compared to DTR. MMR studies a multi-modal version of DTR using both tables and text passages by proposing several bi-encoders and tri-encoders. Similarly, CLTR introduces an end-to-end system for QA over a table corpus, where the retrieval of candidate tables is performed by a coarse-grained BM25 module, followed by a transformer-based model that concatenates the question with the row/column and classifies whether the associated row/column contains the answer. Other systems, as GTR, support retrieval of tables, where tables are represented by graphs. In this setting, stacked layers of a variant of Graph Transformers (Koncel-Kedziorski et al., 2019) are employed for obtaining node features that are combined with query embeddings. These combined embeddings are then aggregated with the BERT embeddings of the table context and query, and a relevance score is finally obtained.

8 Future Directions

Tabular LMs effectively address some of the challenges that arise with classical ML models (Borisov et al., 2021), such as transfer learning and self-supervision. However, several challenges remain unaddressed.

Interpretability. Only a few systems expose a justification of their model output, for example, TAPAS, CLTR, and MATE, thus model usage remains a black box. One direction is to use the attention mechanism to derive interpretations (Serrano and Smith, 2019; Dong et al., 2021). Looking at self-attention weights of particular layers and layer-wise propagation with respect to the input tokens, we can capture the influence of each cell value/tuple on the output through back-propagation (Huang et al., 2019). For instance, in TFC, providing explanation is crucial when the decision is derived from aggregating several cells, such as sum or average operation. In this case, a basic explanation would be to show all the cells that led to the final true/false decision.

Error Analysis. Most studies focus on the downstream evaluation scores rather than going through manual evaluation of errors. In the downstream task level, this analysis could trace back misclassified examples to get evidence of issues in the tabular data representation. For example,

in a QA task with wrong answers, there could be a pattern that explains how these errors are all due to the system confusing two columns with similar meaning (e.g., *max* value and *last* value for a stock), thus returning the wrong cell value in several cases.

Complex Queries and Rich Tables. Several systems, such as TAPEX and MATE, handle queries with aggregations by adding classification layers. However, such methods fail short with queries that join tables. As most works assume a single table as input, a much needed direction is to develop models that handle multiple tables, for example, with classification layers predicting when a join is required. Also, as tables might contain heterogeneous types and content, such as non-uniform units (e.g., *kg* and *lbs*), systems should be able to handle such differences, which are abundant in practice. Moreover, an interesting direction is to conduct a study to show where tabular LMs can be successfully applied and where they fail in querying data.

Model Efficiency. Transformer-based approaches are computationally expensive and some approaches try to approximate the costly attention mechanism by using locality-sensitive hashing to replace it (Kitaev et al., 2020), approximating it by a low-rank matrix (Wang et al., 2020), or applying kernels to avoid its computational complexity (Katharopoulos et al., 2020; Choromanski et al., 2020). While there exist methods to make transformers more efficient for long context (Tay et al., 2020), all optimizations consider an unstructured textual input. We believe more traction is needed for efficient transformers on structured data, with ideas from the textual counterpart, such as limiting attention heads to rows/columns, which can be obtained naturally from the structured input (Eisenschlos et al., 2021), or exploring prompt learning and delta tuning (Ding et al., 2022).

Benchmarking Data Representations. There are no common benchmark datasets where researchers can assess and compare the quality of their data representations in a level playing field. Current evaluation is *extrinsic*, that is, at the downstream task level, where each work has its own assumptions. *Intrinsic* methods to evaluate the quality of the representations, such as those for word embeddings (Bakarov, 2018), can include predicting table caption given table representation or identifying functional dependencies. A set of precise tests can be designed to assess data-specific properties, such as the ability of the transformer-based models, designed to model sequences, to capture that row and attributes are sets in tables. Also, it is not clear whether the model representations are consistent with the table structure, effectively capturing their relationships. For example, given two cell values in the same row/column, are their embeddings closer than values coming from different rows/columns? For this, following the lines of CheckList for text LMs (Ribeiro et al., 2020), basic tests should be designed to measure the consistency of the data representation.

Data Bias. It is recognized that LMs incorporate bias in the model parameters in terms of stereotypes, race, and gender (Nadeem et al., 2021; Vig et al., 2020). The bias is implicitly derived from the training corpora used to develop LMs. Therefore, there is a need to develop methods to overcome this drawback by, for instance, pre-filtering the training data or by correcting the tabular LMs, similarly to the ongoing efforts for text LMs (Liu et al., 2021b; Bordia and Bowman, 2019).

Green LMs. The use of large-scale transformers for learning LMs requires considerable computation, which contributes to global warming (Strubell et al., 2020; Schwartz et al., 2020). Therefore, it is important to consider enhancements or potentially new techniques that limit the carbon footprint of tabular language models without a significant decrease in the performance in the downstream tasks. One enhancement can be at the level of the size of the training data by removing redundant, or less informative, tuples and tables. How to identify such data is a key challenge.

9 Conclusion

We conducted a survey on the efforts in developing transformer-based representations for tabular data. We introduced a high level framework to categorize those efforts and characterized each step in terms of solutions to model structured data, with special attention to the extensions to the transformer architecture. As future work, we envision a generic system to perform an experimental

study based on our framework. The first part of the system would develop tabular data representations with alternative design choices, while the second part would evaluate them in downstream tasks. This work would help identifying the impact of alternative techniques on the performance in the final applications.

Acknowledgments

We would like to thank the action editor and the reviewers for their valuable inputs that helped in further improving the content and the presentation of our article. This work has been partially supported by the ANR project ATTENTION (ANR-21-CE23-0037) and by gifts from Google.

References

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. FEVEROUS: Fact extraction and VERification over unstructured and structured information. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. https://doi.org/10.18653/v1/2021.fever-1.1

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15.

Gilbert Badaro, Ramy Baly, Hazem Hajj, Wassim El-Hajj, Khaled Bashir Shaban, Nizar Habash, Ahmad Al-Sallab, and Ali Hamdi. 2019. A survey of opinion mining in Arabic: A comprehensive system perspective covering challenges and advances in tools, resources, models, applications, and visualizations. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 18(3):1–52. https://doi.org/10.1145/3295662

Gilbert Badaro, Ramy Baly, Hazem Hajj, Nizar Habash, and Wassim El-Hajj. 2014. A large scale Arabic sentiment lexicon for arabic opinion mining. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 165–173. https://doi.org/10.3115/v1/W14-3623

Gilbert Badaro, Obeida El Jundi, Alaa Khaddaj, Alaa Maarouf, Raslan Kain, Hazem Hajj, and Wassim El-Hajj. 2018a. EMA at SemEval-2018 task 1: Emotion mining for Arabic. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 236–244. https://doi.org/10.18653/v1/S18-1036

Gilbert Badaro, Hazem Hajj, and Nizar Habash. 2020. A link prediction approach for accurately mapping a large-scale Arabic lexical resource to english wordnet. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(6):1–38. https://doi.org/10.1145/3404854

Gilbert Badaro, Hussein Jundi, Hazem Hajj, and Wassim El-Hajj. 2018b. EmoWordNet: Automatic expansion of emotion lexicon using English Wordnet. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 86–93. https://doi.org/10.18653/v1/S18-2009

Gilbert Badaro, Hussein Jundi, Hazem Hajj, Wassim El-Hajj, and Nizar Habash. 2018c. ArSEL: A large scale Arabic sentiment and emotion lexicon. *OSACT*, 326.

Gilbert Badaro and Paolo Papotti. 2022. Transformers for tabular data representation: A tutorial on models and applications. *Proceedings of the VLDB Endowment*, 15(12):3746–3749. https://doi.org/10.14778/3554821.3554890

Amir Bakarov. 2018. A survey of word embeddings evaluation methods. *arXiv preprint arXiv:1801.09536v1*.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425. https://doi.org/10.3115/v1/P14-1133

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2015. TabEL: Entity linking in web tables. In *International Semantic Web Conference*, pages 425–441. Springer.

https://doi.org/10.1007/978-3-319-25007-6_25

Shikha Bordia and Samuel R. Bowman. 2019. Identifying and reducing gender bias in word-level language models. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2019-Student Research Workshop, SRW 2019*, pages 7–15. Association for Computational Linguistics (ACL). https://doi.org/10.18653/v1/N19-3002

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. 2021. Deep neural networks and tabular data: A survey. *arXiv preprint arXiv:2110.01889v3*. https://doi.org/10.1109/TNNLS.2022.3229161

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Riccardo Cappuzzo, Paolo Papotti, and Saravanan Thirumuruganathan. 2020. Creating embeddings of heterogeneous relational datasets for data integration tasks. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, pages 1335–1349, New York, NY, USA. Association for Computing Machinery. https://doi.org/10.1145/3318464.3389742

Yoan Chabot, Pierre Monnin, Frédéric Deuzé, Viet-Phi Huynh, Thomas Labbé, Jixiong Liu, and Raphaël Troncy. 2021. A framework for automatically interpreting tabular data at orange. In *Proceedings of the 20th International Semantic Web Conference*.

Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations*.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. 2021. Open question answering over tables and text. In *International Conference on Learning Representations*.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942. https://doi.org/10.18653/v1/2020.acl-main.708

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020b. TabFact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020c. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.91

Zhiyu Chen, Wenhu Chen, Hanwen Zha, Xiyou Zhou, Yunkai Zhang, Sairam Sundaresan, and William Yang Wang. 2020d. Logic2Text: High-fidelity natural language generation from logical forms. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2096–2111. https://doi.org/10.18653/v1/2020.findings-emnlp.190

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020e. Few-shot nlg with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190. https://doi.org/10.18653/v1/2020.acl-main.18

Zhoujun Cheng, Haoyu Dong, Ran Jia, Pengfei Wu, Shi Han, Fan Cheng, and Dongmei Zhang. 2022. FORTAP: Using formulas for numerical-reasoning-aware table pretraining.

In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1166. https://doi.org/10.18653/v1/2022.acl-long.82

Radostin Cholakov and Todor Kolev. 2021. Transformers predicting the future. Applying attention in next-frame and time series forecasting. *arXiv preprint arXiv:2108.08224v1*.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, David Belanger, Lucy Colwell, and Adrian Weller. 2020. Masked language modeling for proteins via linearly scalable long-context transformers. *arXiv preprint arXiv:2006.03555v3*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020*. OpenReview.net.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool publishers. https://doi.org/10.1007/978-3-031-02151-0

Sabyasachi Dash, Sushil Kumar Shakyawar, Mohit Sharma, and Sandeep Kaushik. 2019. Big data in healthcare: Management, analysis and future prospects. *Journal of Big Data*, 6(1):1–25. https://doi.org/10.1186/s40537-019-0217-0

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. TURL: Table understanding through representation learning. *Proceedings of the VLDB Endowment*, 14(3):307–319. https://doi.org/10.14778/3430915.3430921

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904v2*. https://doi.org/10.21203/rs.3.rs-1553541/v1

Haoyu Dong, Zhoujun Cheng, Xinyi He, Mengyu Zhou, Anda Zhou, Fan Zhou, Ao Liu, Shi Han, and Dongmei Zhang. 2022. Table pre-training: A survey on model architectures, pre-training objectives, and downstream tasks. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5426–5435. International Joint Conferences on Artificial Intelligence Organization. Survey Track. https://doi.org/10.24963/ijcai.2022/761

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*.

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. 2021. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International Conference on Machine Learning*, pages 2793–2803. PMLR.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Lun Du, Fei Gao, Xu Chen, Ran Jia, Junshan Wang, Jiang Zhang, Shi Han, and Dongmei Zhang. 2021. TabularNet: A neural network architecture for understanding semantic structures of tabular data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 322–331.

https://doi.org/10.1145/3447548.3467228

Julian Eisenschlos, Maharshi Gor, Thomas Mueller, and William Cohen. 2021. MATE: Multi-view attention for table transformer efficiency. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7606–7619. https://doi.org/10.18653/v1/2021.emnlp-main.600

Majid Ghasemi-Gol and Pedro A. Szekely. 2018. TabVec: Table vectors for classification of web tables. *CoRR*, abs/1802.06290.

Michael Glass, Mustafa Canim, Alfio Gliozzo, Saneem Chemmengath, Vishwajeet Kumar, Rishav Chakravarti, Avirup Sil, Feifei Pan, Samarth Bharadwaj, and Nicolas Rodolfo Fauceglia. 2021. Capturing row and column semantics in transformer based question answering over tables. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1212–1224. https://doi.org/10.18653/v1/2021.naacl-main.96

Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. TableGPT: Few-shot table-to-text generation with table structure reconstruction and content matching. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1978–1988. https://doi.org/10.18653/v1/2020.coling-main.179

Yuan Gong, Yu-An Chung, and James Glass. 2021. AST: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778v3*. https://doi.org/10.21437/Interspeech.2021-698

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. 2021. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943.

Poonam Gupta and Vishal Gupta. 2012. A survey of text question answering techniques. *International Journal of Computer Applications*, 53(4). https://doi.org/10.5120/8406-2030

Jonathan Herzig, Thomas Mueller, Syrine Krichene, and Julian Eisenschlos. 2021. Open domain question answering over tables via dense retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 512–519. https://doi.org/10.18653/v1/2021.naacl-main.43

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Mueller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333. https://doi.org/10.18653/v1/2020.acl-main.398

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. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. ClinicalBERT: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342v3*.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. 2021. TABBIE: Pretrained representations of tabular data. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3446–3456. https://doi.org/10.18653/v1/2021.naacl-main.270

Sujay Kumar Jauhar, Peter Turney, and Eduard Hovy. 2016. TabMCQ: A dataset of general knowledge tables and multiple-choice questions. *arXiv preprint arXiv:1602.03960v1*.

Georgios Karagiannis, Mohammed Saeed, Paolo Papotti, and Immanuel Trummer. 2020. Scrutinizer: A mixed-initiative approach to large-scale, data-driven claim verification. *Proceedings of the VLDB Endowment*, 13(11):2508–2521. https://doi.org/10.14778/3407790.3407841

Marcin Kaszkiel and Justin Zobel. 1997. Passage retrieval revisited. In *SIGIR*, pages 178–185. ACM. https://doi.org/10.1145/278459.258561

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR.

George Katsogiannis-Meimarakis and Georgia Koutrika. 2021. A deep dive into deep learning approaches for text-to-SQL systems. In *Proceedings of the 2021 International Conference on Management of Data*, pages 2846–2851. https://doi.org/10.1145/3448016.3457543

Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. 2021. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293.

Daniel Z. Korman, Eric Mack, Jacob Jett, and Allen H. Renear. 2018. Defining textual entailment. *Journal of the Association for Information Science and Technology*, 69(6):763–772. https://doi.org/10.1002/asi.24007

Bogdan Kostić, Julian Risch, and Timo Möller. 2021. Multi-modal retrieval of tables and texts using tri-encoder models. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 82–91, Punta Cana, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.mrqa-1.8

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466. https://doi.org/10.1162/tacl_a_00276

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. 2016. A large public corpus of web tables containing time and context metadata. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 75–76. https://doi.org/10.1145/2872518.2889386

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2021. Few-shot knowledge graph-to-text generation with pretrained language models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1558–1568.

Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. *Proceedings of the VLDB Endowment*, 14(1):50–60. https://doi.org/10.14778/3421424.3421431

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian-guang Lou. 2021a. TAPEX: Table pre-training via learning a neural SQL executor. *arXiv preprint arXiv:2107.07653v3*.

Ruibo Liu, Chenyan Jia, Jason Wei, Guangxuan Xu, Lili Wang, and Soroush Vosoughi. 2021b. Mitigating political bias in language models through reinforced calibration. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, pages 14857–14866. AAAI Press. https://doi.org/10.1609/aaai.v35i17.17744

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin

Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371. https://doi.org/10.18653/v1/2021.acl-long.416

Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated fact-checking for assisting human fact-checkers. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4551–4558. International Joint Conferences on Artificial Intelligence Organization. Survey Track. https://doi.org/10.24963/ijcai.2021/619

J. Neeraja, Vivek Gupta, and Vivek Srikumar. 2021. Incorporating external knowledge to enhance tabular reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2799–2809, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-main.224

Feifei Pan, Mustafa Canim, Michael Glass, Alfio Gliozzo, and Peter Fox. 2021. CLTR: An end-to-end, transformer-based system for cell-level table retrieval and table question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 202–209, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-demo.24

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186. https://doi.org/10.18653/v1/2020.emnlp-main.89

Panupong Pasupat and Percy Liang. 2015. 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 (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics. https://doi.org/10.3115/v1/P15-1142

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Wullianallur Raghupathi and Viju Raghupathi. 2014. Big data analytics in healthcare: Promise and potential. *Health Information Science and Systems*, 2(1):1–10. https://doi.org/10.1186/2047-2501-2-3, PubMed: 25825667

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.442

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline M. Hancock-Beaulieu, and Mike Gatford. 1995. Okapi at TREC-3. *Nist Special Publication Sp*, 109:109.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2020. Green AI. *Communications of the ACM*, 63(12):54–63. https://doi.org/10.1145/3381831

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1282

Ravid Shwartz-Ziv and Amitai Armon. 2021. Tabular data: Deep learning is not all you need. In *8th ICML Workshop on Automated Machine Learning (AutoML)*. https://doi.org/10.1016/j.inffus.2021.11.011

Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C. Bayan Bruss, and Tom Goldstein. 2021. SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342v1*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2020. Energy and policy considerations for modern deep learning research. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 13693–13696. AAAI Press. https://doi.org/10.1609/aaai.v34i09.7123

Lya Hulliyyatus Suadaa, Hidetaka Kamigaito, Kotaro Funakoshi, Manabu Okumura, and Hiroya Takamura. 2021. Towards table-to-text generation with numerical reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1451–1465.

Yoshihiko Suhara, Jinfeng Li, Yuliang Li, Dan Zhang, Çağatay Demiralp, Chen Chen, and Wang-Chiew Tan. 2022. Annotating columns with pre-trained language models. In *Proceedings of the 2022 International Conference on Management of Data*, pages 1493–1503. https://doi.org/10.1145/3514221.3517906

Yibo Sun, Zhao Yan, Duyu Tang, Nan Duan, and Bing Qin. 2019. Content-based table retrieval for web queries. *Neurocomputing*, 349:183–189. https://doi.org/10.1016/j.neucom.2018.10.033

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA. MIT Press.

Nan Tang, Ju Fan, Fangyi Li, Jianhong Tu, Xiaoyong Du, Guoliang Li, Samuel Madden, and Mourad Ouzzani. 2021. RPT: Relational pre-trained transformer is almost all you need towards democratizing data preparation. *Proceedings of the VLDB Endowment*, 14(8):1254–1261. https://doi.org/10.14778/3457390.3457391

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *CoRR*, abs/2009.06732.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: A large-scale dataset for fact extraction and verification. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 809–819. ACL. https://doi.org/10.18653/v1/N18-1074

James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, and Alon Halevy. 2021. Database reasoning over text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3091–3104, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-long.241

Roger P. G. Van Gompel and Martin J. Pickering. 2007. Syntactic parsing. In *The Oxford handbook of psycholinguistics*, pages 289–307. https://doi.org/10.1093/oxfordhb/9780198568971.013.0017

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural*

Information Processing Systems, volume 30. Curran Associates, Inc.

Enzo Veltri, Gilbert Badaro, Mohammed Saeed, and Paolo Papotti. 2023. Data ambiguity profiling for the generation of training examples. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, California, USA, April 3–7, 2023*. IEEE.

Enzo Veltri, Donatello Santoro, Gilbert Badaro, Mohammed Saeed, and Paolo Papotti. 2022. Pythia: Unsupervised generation of ambiguous textual claims from relational data. In *Proceedings of the 2022 International Conference on Management of Data*, SIGMOD '22, pages 2409–2412, New York, NY, USA. Association for Computing Machinery. https://doi.org/10.1145/3514221.3520164

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in Neural Information Processing Systems*, 33:12388–12401.

Fei Wang, Kexuan Sun, Muhao Chen, Jay Pujara, and Pedro Szekely. 2021a. Retrieving complex tables with multi-granular graph representation learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, pages 1472–1482, New York, NY, USA. Association for Computing Machinery.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768v3*.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021b. TUTA: Tree-based transformers for generally structured table pre-training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1780–1790. https://doi.org/10.1145/3447548.3467434

Zhiruo Wang, Zhengbao Jiang, Eric Nyberg, and Graham Neubig. 2022. Table retrieval may not necessitate table-specific model design. *arXiv preprint arXiv:2205.09843v1*. https://doi.org/10.18653/v1/2022.suki-1.5

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. UnifiedSKG: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966v3*.

Jingfeng Yang, Aditya Gupta, Shyam Upadhyay, Luheng He, Rahul Goel, and Shachi Paul. 2022. TableFormer: Robust transformer modeling for table-text encoding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 528–537. https://doi.org/10.18653/v1/2022.acl-long.40

Xiaoyu Yang and Xiaodan Zhu. 2021. Exploring decomposition for table-based fact verification. In *EMNLP*, pages 1045–1052, Punta Cana, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.findings-emnlp.90

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, Bailin Wang, Yi Chern Tan, Xinyi Yang, Dragomir Radev, Richard Socher, and Caiming Xiong. 2021. GraPPa: Grammar-augmented pre-training for table semantic parsing. In *International Conference on Learning Representations*.

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 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1425

Shuo Zhang and Krisztian Balog. 2017. EntiTables: Smart assistance for entity-focused tables. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 255–264. ACM. https://doi.org/10.1145/3077136.3080796

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103v7*.