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

Gilbert Badaro and **Mohammed Saeed** and **Paolo Papotti**
EURECOM, France
gilbert.badaro; mohammed.saeed; paolo.papotti@eurecom.fr

Abstract

In the last few years, the natural language processing community witnessed the advances in neural representations of free texts with transformer-based language models (LMs). Given the importance of knowledge available in relational tables, recent research efforts extend LMs by developing neural representations for tabular data. In this work, we present the first survey that analyzes these efforts. We first categorize the downstream tasks where the models are successfully utilized. The alternative solutions are then characterized and compared in terms of training data, input pre-processing, model training, and output representation. Finally, we provide insights on potential future work directions.

1 Introduction

Several efforts are researching how to represent tabular data with neural models for traditional and new natural language processing (NLP) applications. These models enable effective data-driven solutions that go beyond the limits of traditional declarative specifications built around first order logic and SQL. Examples include answering queries expressed in natural language (Katsogiannis-Meimarakis and Koutrika, 2021; Herzig et al., 2020; Thorne et al., 2021), performing natural language inference such as fact-checking (Chen et al., 2020a; Yang and Zhu, 2021), doing semantic parsing (Yin et al., 2020; Yu et al., 2021), retrieving relevant tables (Pan et al., 2021; Kostić et al., 2021; Wang et al., 2021a), understanding table metadata (Suhara et al., 2021; Deng et al., 2020; 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 models for representing tables accurately. Given the success of transformer-based models in developing pre-trained language models (LMs) (Devlin et al., 2019; Liu et al., 2019), we focus our analysis on the extension to this architecture for developing representations of relational tables. Indeed, attention-based approaches are successful also on visual (Dosovitskiy et al., 2020; Khan et al., 2021), audio (Gong et al., 2021) and time series data (Cholakov and Kolev, 2021), and they start gaining popularity for developing representations for tabular data.

While all solutions have contributions to the neural representation of tabular data, with alternative processes to develop and consume the encoded data, there is still no clear definition of the problem nor a systematic method to compare those representations given the different assumptions and target applications. In this work, we aim at bringing clarity in this space and at providing dimensions for a classification that highlights the main trends and enables future work to clearly position new results. Our contributions are threefold:

1. We formalize the problem by providing general definitions that are applicable to all procedures and that are agnostic to their assumptions and final application (Section 2).
2. We identify and describe the characteristics of each dimension based on the referenced works in terms of downstream tasks (Section 3), datasets (Section 4), data preprocessing (Section 5), transformer architecture (Section 6), output characteristics and usage (Section 7).
3. We discuss limitations of existing works and propose future work directions (Section 8).

Our work is the first to study methods for neural representations of tabular data with transformers. It is different from recent surveys and experi-

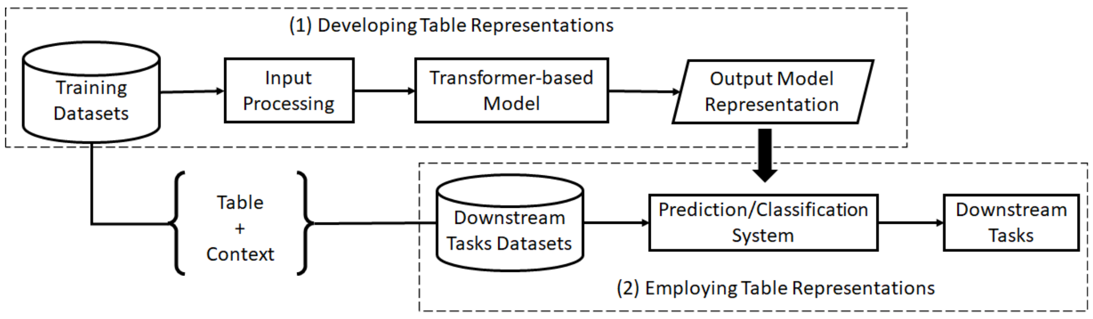
(1) Developing Table Representations
Training Datasets
Input Processing
Transformer-based Model
Output Model Representation
Table + Context
Downstream Tasks Datasets
Prediction/Classification System
Downstream Tasks
(2) Employing Table Representations

Figure 1: The generic framework for developing and consuming neural representations for tabular data.

mental analysis that cover the use of deep learning for classification, regression, and generation tasks with tabular data (Borisov et al., 2021; Gorishniy et al., 2021; Shwartz-Ziv and Armon, 2021). Differently from these efforts, we focus on the core problem of rendering the transformer architecture 'data structure-aware' and relate design choices and contributions to a large set of downstream tasks in the NLP and database communities.

2 Transformers for Tabular Data

We start by introducing the terminology used in the article. With *tabular data*, we refer to a **relational table**. A relational table consists of **rows**, or records, and **columns** that together identify **cell values**, or cell. Columns consist of attributes for a given table, and each row represents an instance of those attributes. All cells in a single column share the same type. For example, a relational table about countries can include as column labels: *Country*, *Capital*, and *Total Population*. Exemplary rows for such table would contain cell values "France", "Paris", "67.39m" and "Bolivia", 'La Paz", "11.67m". Notice how a cell value, such as "La Paz", can contain multiple tokens.

Relational tables can go beyond the mandatory elements of tabular data and have richer metadata, such as attribute types (e.g., DATE), domain constraints, functional dependencies across columns and integrity constraints such as primary keys. In the example, column *Country* is a primary key. Most works focus on a single table, with or without metadata, however there are some exceptions. A few systems consume *databases*, which are collections of relational tables, possibly under referential constraints (Herzig et al., 2021; Wang et al., 2021a; Yu et al., 2021). Other works go beyond relational tables and handle **spreadsheets** as input. Those are tabular data with richer schema metadata and structure, such as multiple labels associated to the same column and nesting of cells (Wang et al., 2021b; Du et al., 2021).

As shown in Figure 1, we distinguish two problems. Our main focus is on the development of pre-trained representations for tables using transformer-based deep neural network (1). Given a relational table, the goal is to learn a representation of the structured data (cell values, rows, attributes) in a continuous vector space with a small number of dimensions. As the final application plays a key role in the design decisions in the task, we also discuss the use of those representations in the downstream tasks (2).

For both settings, we identify as input the relational table(s), as discussed above, and its *context*. The context is a text associated to the table. Depending on the dataset and application 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.

Figure 1 shows the general framework that we propose to model existing solutions.

- **Training Datasets**: the relational datasets used for training and fine-tuning the models.
- **Input Processing**: steps to prepare the data for the model learning the representation.
- **Transformer-based Model**: training objectives and customization of the typical transformer-based deep learning architecture.
- **Output Representation**: the final representations at the token, row, column, and table level.
- **Prediction/Classification System**: solutions consuming the representations or fine-tuning them to tackle downstream tasks.
- **Downstream Tasks**: tasks where the models have been utilized and evaluated.

As several insights for the first five dimensions depend on the downstream task, we start by describing the latter. We then discuss the remaining dimensions according to their framework order.

3 Downstream Tasks

Using neural representations for relational tables show improvements in performance in several downstream tasks. In this section, we describe the tasks and define their input and output. However, while they all consume tables as input, settings can be quite heterogeneous, with different solutions exploiting different information, even for the same task. We detail here the mandatory input elements and discuss the different contexts in Section 4.

Natural Language Inference (NLI): NLI relates mainly to two different tasks: fact-checking and text refusal/entailment. Several works harvested neural representations for tabular data to tackle these tasks. In this setting, fact-checking consists in verifying if a textual input claim is true or false against a trusted relational database (Liu et al., 2021; Yang and Zhu, 2021; Nakov et al., 2021), also provided as input. Similarly, table-based textual refusal/entailment consists of checking whether a given input relational table can imply or not the given input text (Dagan et al., 2013; Korman et al., 2018; Chen et al., 2020a). In addition to the typical binary output (entailed/refused), a third class for "not enough information" is sometimes used as output. Some systems also output the cells as evidence (Aly et al., 2021).

Question Answering (QA): While in free text setting, QA aims at retrieving passages that include the answer to a given question, in relational table setting it consists of finding 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, while a more complex QA task involves aggregation operations and requires support for numerical reasoning. Most of the systems proposing neural embeddings for relational data aim at improving accuracy in QA (Glass et al., 2021; Pan et al., 2021; Thorne et al., 2021; Eisenschlos et al., 2021; Liu et al., 2021; Herzig et al., 2021, 2020).

Semantic Parsing (SP): Given a question and a table as input, SP generates a declarative query, expressed in SQL or SPARQL, that retrieves the answer to the question over the table. While in QA the interest is in directly getting the answer, SP produces the (interpretable) query to obtain it (Yin et al., 2020; Yu et al., 2021; Liu et al., 2021).

Table Retrieval (TR): Given a question and a set of tables as inputs, TR identifies the table that can be used to answer the input question. TR is usually helpful when trying to reduce the search space for a QA task. It is a challenging task given the limited input size of transformers due to their inability of handling sequences of more than 512 tokens (Wang et al., 2021a; Herzig et al., 2021; Kostić et al., 2021).

Table Metadata (TM): Given an input table with corrupted or missing metadata, the TM objective is to predict inter-table metadata, such as column types and headers, and intra-tables relationships, such as equivalence between columns and entity linking/resolution. Relevant efforts focus both on spreadsheets (Wang et al., 2021b; Du et al., 2021) and relational tables (Deng et al., 2020; Suhara et al., 2021).

Table Content Population (TCP): Unlike TM 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 output the respective values (Tang et al., 2021; Iida et al., 2021; Deng et al., 2020).

We observe that most tasks can be seen as traditional NLP tasks involving structured data as a new modality, replacing the free text. NLI applied on tabular data involves retrieving cells that entail or refute a given statement whereas on free text the objective is to select sentences as evidence. TR on tabular data corresponds to passage retrieval on free text. TCP is analogous to predicting missing words or values in a sentence. SP and TM are more specific to relational tables.

4 Training Datasets

We present in this section the datasets that are commonly utilized for the downstream tasks that we have introduced. We identify several characteristics for these datasets including the context provided in addition to the tables.

Table 2 summarizes our findings about the most commonly used datasets. Column "Task Categories" includes the tasks for which the dataset has been used for model training. It is important to note that the top five datasets are mostly used for

Task ID	Task Label	Tasks Coverage	Input	Output	Representative Examples
NLI	Natural Language Inference	Fact-Checking Text Refusal/Entailment	Table + Claim	True/False Refused/Entailed (Data Evidence)	(Yang and Zhu, 2021) (Chen et al., 2020a)
QA	Question Answering	Retrieving the Cells for the Answer	Table + Question	Answer Cells	(Herzig et al., 2020) (Eisenschlos et al., 2021) (Herzig et al., 2021)
SP	Semantic Parsing	Text-to-SQL	Table + NL Query	Formal QL	(Yin et al., 2020) (Yu et al., 2021) (Liu et al., 2021)
TR	Table Retrieval	Retrieving Table that Contains the Answer	Tables + Question	Relevant Table(s)	(Wang et al., 2021a; Pan et al., 2021)
TM	Table Metadata	Column Type Prediction Header Detection Cell Role Classification Column Relation Annotation Column Name Prediction	Table	Column Types Header Row Cell Role Relation between Two Columns Column Name	(Suhara et al., 2021) (Du et al., 2021) (Deng et al., 2020)
TCP	Table Content Population	Cell Content Population	Table with Corrupted Cell Values	Table with Complete Cell Values	(Deng et al., 2020; Iida et al., 2021)

Table 1: List of tasks utilizing neural representations for tables.

pre-training, while the others can be used for fine-tuning as well since they include annotations for the ultimate application, e.g., questions in natural language for QA. The column "Data Reduction" is a metric to assess the average size of tables in the datasets, where ✔ and ✘ indicate whether or not some pre-processing is needed to filter out table content to meet the limits of LMs (512 input tokens in most cases). Some works apply filtering in any case to reduce noisy input (Yin et al., 2020; Pan et al., 2021). Finally, the "Context" column describes the additional texts that come with the tables. This can be text describing the table, such as a caption, a title or a description 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.

Combination of these datasets are used by different systems. For instance, Yin et al. (2020) use Wikipedia and WDC, Wang et al. (2021b) use Wikitables, WDC and spreadsheets (Dong et al., 2019), Yu et al. (2021) use Wikitables, SPIDER and WikiSQL, and Kostić et al. (2021) use NQ-Tables, WikiSQL and OTT-QA.

We observe that TCP and TM downstream tasks use minimal context, i.e., only table metadata, while NLI, SP, QA, and TR need additional text information whether in the form of a question, claim, table caption or description.

5 Input Pre-processing

In addition to the typical tokenization executed before feeding the tokens to the neural network (Lan et al., 2020), some pre-processing steps are performed mainly to reduce and filter the table content, to reshape the filtered table content, i.e., table serialization, and to combine the serialized table with the context information.

5.1 Data Retrieval and Filtering

While some models objective is to retrieve tables that contain the answer to a given question (Wang et al., 2021a; Herzig et al., 2021), others (Pan et al., 2021) use a ranking function, such as BM25 (Robertson et al., 1995), to retrieve relevant tables prior to training the model or to generate negative examples (Kostić et al., 2021).

Filtering methods on the table content are applied to stay within the size limits of LMs, to improve the training time of the model, and to eliminate potential noise in the neural representations. Yin et al. (2020) use 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. In addition to keeping tables with number of columns below a fixed threshold, Wang et al. (2021b) and Du et al. (2021) split large tables into smaller ones (horizontal partition). Liu et al. (2021) randomly select rows at each iteration and Glass et al. (2021) down-sample rows that are not relevant using term frequency inverse document frequency (TF-IDF) score. Thorne et al. (2021) handle rows as sentences. For this task, they train a neural support set generator using T5 (Raffel et al., 2020) to select relevant rows that are transformed into facts in natural language text.

Dataset	Reference	Task Categories						Number of Tables	Data Reduction	Context	Application Example
		NLI	QA	SP	TR	TM	TCP				
Wikipedia Tables	Wikipedia		✔	✔			✔	-	✔	**Surrounding Text:** table caption, page title, page description, segment title, text of the segment.	(Herzig et al., 2020)
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.	(Yin et al., 2020)
WikiTables	(Bhagavatula et al., 2015)		✔	✔		✔	✔	1.6M	✔	**Surrounding Text:** caption, page title, section title. **Table Metadata:** statistics about number of headings, rows, columns, data rows.	(Herzig et al., 2021)
VizNet	(Hu et al., 2019)						✔	1M	✘	**Table Metadata:** Column Types.	(Iida et al., 2021)
Spreadsheets	(Dong et al., 2019)					✔		3,410	✘	**Table Metadata:** Cell Roles (Index, Index Name, Value Name, Aggregation and Others).	(Du et al., 2021)
NQ-Tables	(Herzig et al., 2021)		✔					169,898	✔	**Questions:** 12K.	(Herzig et al., 2021)
TABFACT	(Chen et al., 2020a)	✔						16K	✘	**Textual Claims:** 118K claims.	(Yang and Zhu, 2021)
WikiSQL	(Zhong et al., 2017)		✔	✔	✔			24,241	✘	**Questions:** 80,654.	(Kostić et al., 2021)
TabMCQ	(Jauhar et al., 2016)		✔		✔			68	✘	**Questions:** 9,092.	(Glass et al., 2021)
SPIDER	(Yu et al., 2018)			✔				200 databases	✘	**Questions:** 10,181 Queries: 5,693.	(Yu et al., 2021)
WikiTable Question (WikiTQ)	(Pasupat and Liang, 2015)		✔	✔				2,108	✘	**Questions:** 22,033.	(Liu et al., 2021)
Natural Questions (NQ)	(Kwiatkowski et al., 2019)			✔				169,898	✔	**Questions:** 320K.	(Kostić et al., 2021)
OTT-QA	(Chen et al., 2021)				✔			400K	✔	**Surrounding Text:** page title, section title, section text limited to 12 first sentences. **Questions:** 45,841.	(Kostić et al., 2021)
Web Query Table	(Sun et al., 2019)				✔			273,816	✘	**Surrounding Text:** captions. **Queries:** 21,113.	(Wang et al., 2021a)
HybridQA	(Chen et al., 2020b)		✔					13K	✘	**Questions:** 72K. **Surrounding Text:** first 12 sentences surrounding the table.	(Eisenschlos et al., 2021)

Table 2: Datasets for the development and evaluation of neural representation models of tabular data. ✔/✘ denotes that pre-processing is either needed/not needed to filter out table content to meet the limits of LMs. The top five datasets are mostly used for pre-training models with unsupervised tasks. The rest of the datasets also come with a context that is used in the downstream task for training and evaluation. Application example refers to samples of the referenced work where they utilized the respective dataset.

Context and Table parsed by row:
[CLS] *Population of Countries* [CLS] Country | Capital | Population [SEP] France | Paris | 67.39m ... [SEP] Italy | Rome | 59.55m

Context and Table parsed by column:
[CLS] *Population of Countries* [CLS] Country | France | ... | Italy | ... [SEP] Capital | France | ... | Rome | ... [SEP] Population | 67.39m | ...

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

Regardless of the eventual downstream application, most works filter and reduce the size of the input data to meet the limits of transformers technology. However, methodical sampling is more efficient than random one since it reduces noise in data representations.

5.2 Table Serialization

The methods for table serialization can be grouped into three main types. The first one consists of scanning the table by row: this can be simply a flattened table (Herzig et al., 2020, 2021; Eisenschlos et al., 2021; Kostić et al., 2021; Yang and Zhu, 2021; Deng et al., 2020), a flattened table with special token separator to indicate the beginning of new row, new cell, or header row (Liu et al., 2021; Wang et al., 2021b); a flattened table where each cell is represented as concatenation of column name, column type, and cell value (Yin et al., 2020); or just the row of column headers (Yu et al., 2021).

The second linearization is done by scanning the table by column again either by simple concatenation of column values or by using a special token as separator (Suhara et al., 2021).

The third technique consists of combining the output from both types of serialization (Glass et al., 2021; Du et al., 2021; Pan et al., 2021; Iida et al., 2021). Representative row and column serializations for the country population example in Section 2 are reported in Figure 2.

It is not clear whether row encoding is better than column encoding or vice versa. Chen et al. (2020a) experiment with different settings and there is no significant difference in performance. Multiple efforts incorporate both aspects either by appending column headers to cell content, combining row and column encoding, or by adding structure aware indicators as we discuss in Section 6.

5.3 Context and Table Concatenation

In most systems, the table content is concatenated with the available context information detailed in Table 2. The context can be combined by concatenating it in the serialization before the table content (Yin et al., 2020; Herzig et al., 2020, 2021; Yu et al., 2021; Yang and Zhu, 2021), or appended to it (Liu et al., 2021). Chen et al. (2020a) test both strategies and report minor improvements in performance when context is appended to table data. In some cases, the table and the context are encoded separately and then are combined at a later stage in the model (Kostić et al., 2021; Pan et al., 2021; Glass et al., 2021).

Some works do not include context in their pretraining input besides the column headers (Iida et al., 2021; Du et al., 2021; Suhara et al., 2021). A richer context is used when the downstream tasks are similar to the corresponding NLP task applied on free text. For instance, all models for QA use table captions or descriptions as context.

6 Model Characteristics

To account for structured tables in the input, several pre-trained transformer-based language models (LMs) 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 deliver the table representation, as depicted in part (1) of Figure 1. To exploit such representation 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. Other systems instead use the encoder as part of a bigger architecture in a more task-oriented fashion rather than encoder-oriented. We first briefly revise the transformer architecture, then discuss customizations to LMs.

6.1 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. Residual connections and layer-normalization modules are also utilized. On the other hand, 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.

Attention Modules. The transformer relies on the "Scaled Dot-Product Attention" mechanism, where three matrices are used: the query $\mathbf{Q} \in \mathbb{R}^{O\times d_k}$, the key $\mathbf{K} \in \mathbb{R}^{P\times d_k}$, and the value $\mathbf{V} \in \mathbb{R}^{P\times d_v}$ matrices, where O and P denote respectively the lengths of queries and keys/values, and d_k and d_v denote respectively the dimensions of keys/queries and values. The scaled dot-product attention is given by:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

The matrix $softmax(\frac{QK^T}{\sqrt{d_k}})$ is often called the attention matrix, as it portrays how much each element attends to all elements of the sequence. For input with large size, computing the attention matrix is infeasible and sparse methods are utilized (Tay et al., 2020). Instead of relying on a single head, transformers use multi-head attention, where the original queries, keys, and values with dimension d_m are projected into d_k, d_k, and d_v dimensions, respectively, with h attention heads each with a different set of learned projections. Each head output is computed with Eq. (1), and all outputs are concatenated and projected back to a d_m-dimensional representation.

$$\begin{aligned} MultiHeadAttn(Q, K, V) &= \\ Concat(head_1, \ldots, head_H)W^O \end{aligned} \quad (2)$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

Feed-forward network. The position-wise feed-forward network is a fully connected feed-forward module operating separately on each position:

$$FFN(x) = ReLU(xW_1 + b_1)W_2 + b_2 \quad (4)$$

Positional Encoding. Since transformers do not incur any form of recurrence, positional information is designated by absolute sinusoidal position encodings added to each token embedding of the transformer (Vaswani et al., 2017). Such embeddings can also be learned (Devlin et al., 2019).

Transformer Usage. 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. Encoder-only models are mainly used for classification where pre-training is done with a task called masked language modeling (MLM) whose goal is to predict masked token(s) of the altered input (Devlin et al., 2019).

6.2 LM Extensions for Tabular Data

To properly model the structure of data in tables, the vanilla models 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.

6.2.1 Input Level

Modifications on the input level are usually designated with **additional positional embeddings** to explicitly model the table structure. For example, embeddings that represent the position of the cell, indicated by its row and column IDs, are common for relational tables (Wang et al., 2021b; Herzig et al., 2020; Iida et al., 2021). For tables without a relational structure (such as entity and matrix tables), tree-based positional embeddings have been proposed that encode the position of a cell using top and left embeddings of a bi-dimensional coordinate tree (Wang et al., 2021b). Other supplementary embeddings include those that provide relative positional information for a token within a caption/header (Deng et al., 2020) or a cell (Wang et al., 2021b). For tasks such as QA, segment embeddings are used to differentiate between the different input types, question and table (Tang et al., 2021; Herzig et al., 2020). Embeddings for numbers are introduced where discrete features are used (Wang et al., 2021b).

6.2.2 Internal Level

Most modifications on the internal level are to render the system more "structure-aware". Specifically, the **attention** module is updated to integrate the structure of the input table. For example, vertical self-attention layers were produced where the aim is to capture cross-row dependencies on cell values by performing the attention module in a vertical fashion (Yin et al., 2020). Other systems employ a masked self-attention module which attends to structurally related elements such as elements in a row or a column, unlike the traditional transformer where each element attends to

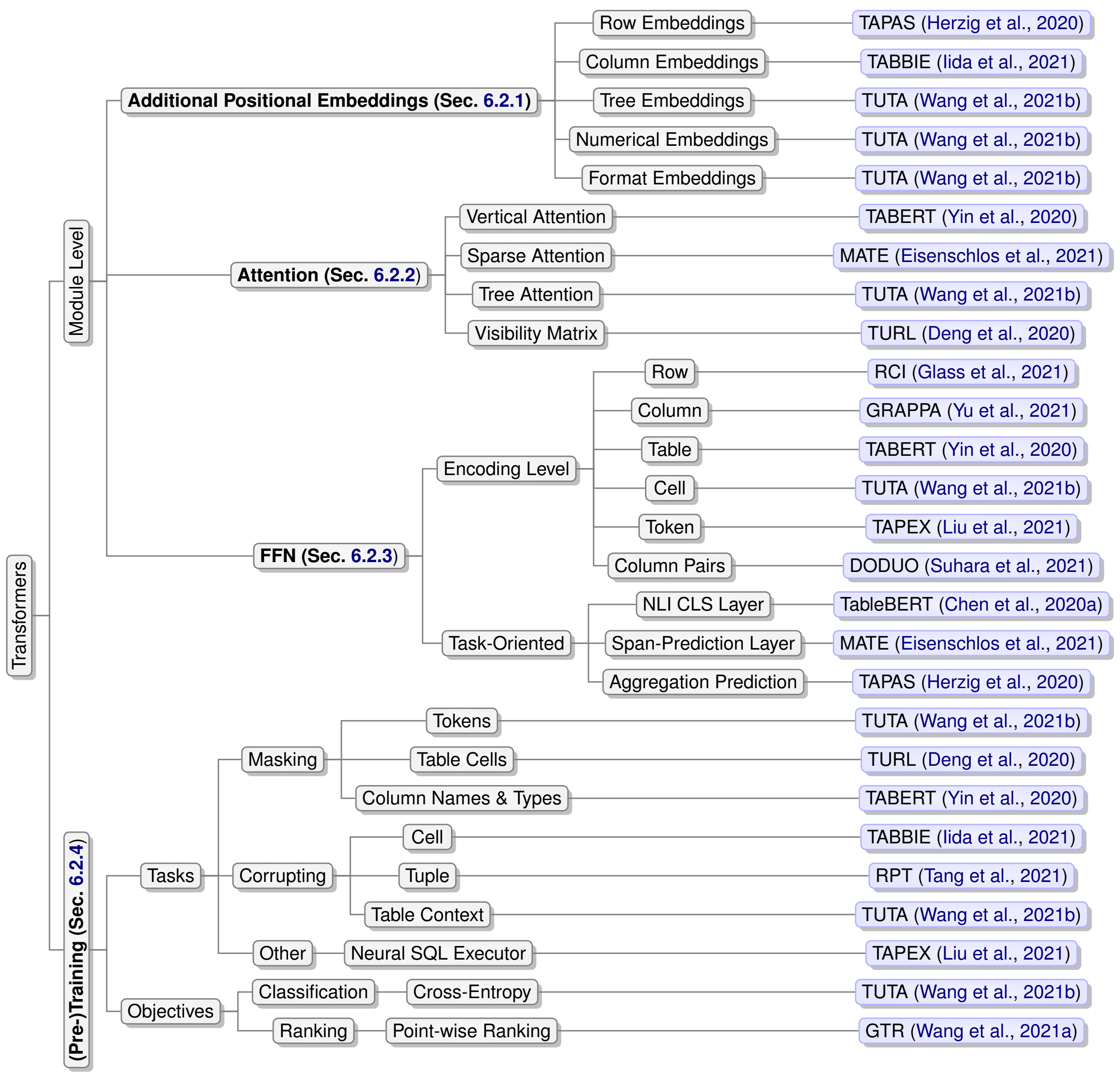

Figure 3: Taxonomy of extensions to transformer-based LMs for handling tabular data.

all other elements in the sequence (Deng et al., 2020; Wang et al., 2021b).

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 ordeal (Tay et al., 2020). For instance, Eisenschlos et al. (2021) sparsified the attention matrix to allow transformer heads to efficiently attend to either rows or columns.

6.2.3 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 (Herzig et al., 2020), NLI (Chen et al., 2020a), TM (Suhara et al., 2021), and QA (Herzig et al., 2020) require to train one or more additional layers. Classification layers for aggregation operations and cell-selection are used to support aggregating cell values (Herzig et al., 2020; Eisenschlos et al., 2021; Glass et al., 2021; Yang and Zhu, 2021; Yu et al., 2021). Aggregation operations could be also "learned" end-to-end in a seq2seq task (Liu et al., 2021).

6.2.4 Training-Procedure Level

Modifications on the training-procedure level can be attributed to the pre-training task or the pre-training 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. Aside from MLM, other pre-training tasks are employed. Such methods rely on masking tokens of table cells (Wang et al., 2021b) or masking the whole cell (Yin et al., 2020; Wang et al., 2021b), column names and types (Yin et al., 2020). Other pre-training tasks detect if a cell/tuple is corrupted or not (Iida et al., 2021; Tang et al., 2021). Text headers of tables are utilized to learn representations of tables in a self-supervised manner (Wang et al., 2021b). One pre-training task adopts an SQL engine to train the model to act as a neural SQL executor, thus enabling it to mimic the SQL semantics with relational tables (Liu et al., 2021).

Pre-training Objectives. The majority of the systems minimize cross-entropy loss. One system utilizes a point-wise ranking objective for end-to-end training after pre-training (Wang et al., 2021a).

Overall, a more data structure-aware 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.

7 Output Model Representation

We first describe the final data representations and then how those are used in target systems.

7.1 Output Characteristics

The output characteristics of the neural representations of relational data are the result of the model pre-training. The output can be fine-tuned or utilized as-is, as features with standard supervised machine learning algorithms. The systems expose different granularity of the output table representation. Almost all systems provide token and cell output representations. The majority also expose column and row representations and few works provide table and pairwise column (Suhara et al., 2021) or pairwise table representations (Yang and Zhu, 2021). The pre-trained models are usually available out-of-the-box, while in some cases the users have to retrain the models. In case of special separator token preceding the context appended to the table content, a representation of that context is also provided (Herzig et al., 2021; Chen et al., 2020a; Pan et al., 2021; Kostić et al., 2021).

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 for the TM 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.

7.2 Prediction/Classification Systems

As pre-trained transformer-based LMs act as encoders of the input, they are used in many settings as a building block to process the input. Novel layers are added on top of the LM and the entire model is then fine-tuned for a specific task (Deng et al., 2020; Wang et al., 2021b). While this is a common use of pre-trained LMs, other works employ LMs as components in a larger system, where encoding the input is not a first-class citizen, and rather the aim is to develop an end-to-end trained system oriented towards a certain task.

Most of these larger systems focus on the retrieval of tables from an input natural-language query. Herzig et al. (2021) answer a natural-language question from a corpus of tables in two steps. First, their approach 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. Then, it performs the standard answer prediction (QA) with each candidate table in the input. Kostić et al. (2021) study a multi-modal version of this system using both tables and text passages by proposing several bi-encoders and tri-encoders. Similarly, Pan et al. (2021) introduce 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 an RCI-interaction model (Glass et al., 2021) that concatenates the question with the row/column and classifies whether the associated row/column contains the answer. Other systems support retrieval of non-relational tables where tables are represented by graphs (Wang et al., 2021a). 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, through a multi-layer perceptron, with BERT embeddings of the table context and query, and a relevance score is then obtained.

8 Future Directions

While there has been progress in representing tabular data, several challenges remain unaddressed.

Interpretability. Some systems expose a justification of their model output (Yang and Zhu, 2021; Pan et al., 2021; Thorne et al., 2021; Eisenschlos et al., 2021; Herzig et al., 2020), but the majority does not, and model usage remains a black box. One direction might be 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 w.r.t. the input tokens, we can capture the influence of each input on the output through back-propagation (Huang et al., 2019).

Error Analysis. Besides developing explainable models, error analysis can help understand the model behavior. Most works 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 or mispredicted examples to get evidence of issues in the tabular data representation.

Complex Queries. Several works handle queries with aggregations by adding classification layers (Thorne et al., 2021; Eisenschlos et al., 2021; Liu et al., 2021). However, such methods fail short with queries that join tables. Indeed, most works assume a single table as input. A natural direction is to develop models able to handle multiple tables, for example with classification layers predicting when a join is needed.

Model Efficiency. Transformer-based approaches suffer from the upper bound on the supported input size (typically 512 tokens). This is a concern with large tables. In this direction, several architectures have made improvements around computational and memory efficiency by using locality-sensitive hashing to replace the attention mechanism (Kitaev et al., 2020), approximating the self-attention mechanism by a low-rank matrix (Wang et al., 2020), and applying kernels to avoid the computational complexity of the attention mechanism (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 of them consider the input to be unstructured. We believe more traction is needed for efficient transformers on structured data, where ideas such as limiting attention heads to rows/columns (obtained naturally from the structured input) can be utilized for efficiency (Eisenschlos et al., 2021).

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*, i.e., 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 relational tables. Also, it is not clear whether the model representations are consistent or not with the table structure. 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 (Ribeiro et al., 2020), basic tests should be designed to measure the consistency of the data representation.

9 Conclusion

In this article, 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 system to perform an experimental study based on our suggested problem formulation. The first part of the system would develop tabular data representations with alternative design choices, while the second part would evaluate them in different downstream tasks. This work would help identifying the impact of choices, made to develop those representations, on the performance on the final applications.

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)*.

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

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

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:2110.01889*.

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.

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*.

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, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020a. 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. 2020b. 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.

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

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

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.

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.

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.

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.

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. *ArXiv*, abs/2103.03404.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 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.

Julian Martin Eisenschlos, Maharshi Gor, Thomas Müller, and William W Cohen. 2021. MATE: Multi-view attention for table transformer efficiency. *arXiv preprint arXiv:2109.04312*.

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.

Yuan Gong, Yu-An Chung, and James Glass. 2021. AST: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. 2021. Revisiting deep learning models for tabular data. *arXiv preprint arXiv:2106.11959*.

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.

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.

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:1904.05342*.

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.

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

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.

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.

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.

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.

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.

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

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.

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.

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.

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al.

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.

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.

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

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.

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)*.

Yoshihiko Suhara, Jinfeng Li, Yuliang Li, Dan Zhang, Çağatay Demiralp, Chen Chen, and Wang-Chiew Tan. 2021. Annotating columns with pre-trained language models. *arXiv preprint arXiv:2104.01785*.

Yibo Sun, Zhao Yan, Duyu Tang, Nan Duan, and Bing Qin. 2019. Content-based table retrieval for web queries. *Neurocomputing*, 349:183–189.

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, page 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. *Proc. VLDB Endow.*, 14(8):1254–1261.

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

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

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, page 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:2006.04768*.

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.

Xiaoyu Yang and Xiaodan Zhu. 2021. Exploring decomposition for table-based fact verification.

In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1045–1052, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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.

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

Customization Categories	**Details**	**Task Categories**						**Referenced Work**
		NLI	**QA**	**SP**	**TR**	**TM**	**TCP**	
Encoding	Row Encoding	✔	✔	✔	✔			(Liu et al., 2021)
	Column Encoding	✔				✔	✔	(Suhara et al., 2021)
	Combined Row & Column Encoding		✔		✔		✔	(Pan et al., 2021)
Table Structure Aware Representation	Positional Embeddings		✔					(Herzig et al., 2020)
	Visibility Matrix					✔	✔	(Deng et al., 2020)
	Tree-based Positional Embedding					✔		(Wang et al., 2021b)
Direction of Attention	Vertical Attention			✔				(Yin et al., 2020)
	Tree Attention						✔	(Wang et al., 2021b)
	Sparse Attention		✔					(Eisenschlos et al., 2021)
Addition of CLS Layers	Cell Representation		✔			✔	✔	(Herzig et al., 2020)
	Aggregation Operator Prediction		✔					(Thorne et al., 2021)
	Column/Row Representation		✔	✔		✔	✔	(Glass et al., 2021)
	Table Representation			✔				(Wang et al., 2021a)
	Column Pairs Representation					✔		(Suhara et al., 2021)
	Final Verdict	✔						(Chen et al., 2020a)
Pre-training Objectives	Masked Language Model	✔	✔	✔		✔	✔	(Eisenschlos et al., 2021)
	Masked Cell/Entity Recovery			✔		✔	✔	(Deng et al., 2020)
	Masked Column Prediction			✔				(Yin et al., 2020)
	Inverse Cloze Task		✔		✔			(Herzig et al., 2021)
	Table Context Retrieval					✔		(Wang et al., 2021b)
Fine-tuning for Supervised Tasks	Fine-tuning without Pre-training	✔			✔			(Kostić et al., 2021)
	Using Embeddings as Features					✔		(Du et al., 2021)
Fine-tuning Objectives	Cross-Entropy Loss		✔			✔		(Eisenschlos et al., 2021)
	Binary Cross-Entropy Loss	✔				✔	✔	(Suhara et al., 2021)
	Logistic Loss		✔					(Herzig et al., 2021)
Numerical Value Reasoning	**Yes:** Model Handles Aggregations		✔	✔	✔			(Herzig et al., 2020)
	No: Model Handles Value Selection only	✔	✔	✔	✔	✔	✔	(Iida et al., 2021)

Table 3: Summary of Model Customizations.

Article	**Direction of Attention**	**Addition of CLS Layers**	**Row & Column Encodings**	**Table Structure Aware Representation**	**Pretrain Objectives**	**Fine-tuning Objectives**	**Fully Supervised Approaches**	**Downstream Tasks**
(Yin et al., 2020)	Vertical Attention	Table Representation	Row+Column Name & Type		MLM; Masked Column; Masked Cell			SP
(Herzig et al., 2020)		Cell Selection; Aggregation Operator Prediction	Row	Positional Embeddings	MLM	Cross-Entropy Loss		QA
(Chen et al., 2020a)		Textual Entailment/Refusal	Row		MLM	Binary Cross-Entropy		NLI
(Yu et al., 2021)			Row (Column Headers Only)		MLM	SQL Semantic Loss		SP
(Deng et al., 2020)			Row	Visibility Matrix	MLM+; Masked Entity Recovery			TM, TCP
(Wang et al., 2021b)	Tree Attention	Cell Selection	Row	Tree-based Positional Embeddings	MLM; Masked Cell; Masked Context	Cross-Entropy Loss		TM
(Iida et al., 2021)		Cell Representation	Combined	Positional Embeddings	Modified MLM to dectect corrupted cells	Binary Cross-Entropy		TCP
(Du et al., 2021)							LM Embeddings as Features	TM
(Liu et al., 2021)			Row		Masked Result of SQL Executor			SP, NLI
(Eisenschlos et al., 2021)	Sparse Attention	Cell Selection	Row	Positional Embeddings	MLM	Cross-Entropy Loss		QA
(Kostić et al., 2021)			Combined			Similar embeddings between tables and question		TR
(Suhara et al., 2021)		Column Representation; Column-pair Representation	Column			Cross-Entropy Loss; Binary Cross-Entropy		TM

Table 4: A summary of model customization adopted by a sample of the surveyed works.