IM-TQA: A Chinese Table Question Answering Dataset with Implicit and Multi-type Table Structures

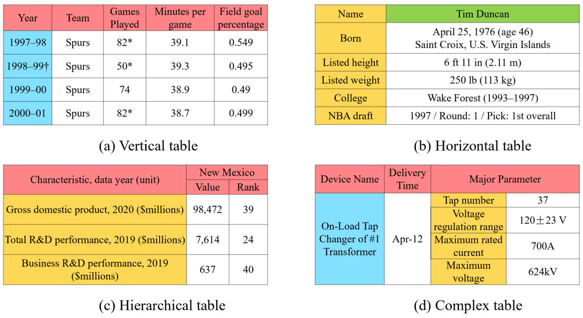
Mingyu Zheng[1,2*†], **Yang Hao**[3*], **Wenbin Jiang**[3*] , **Zheng Lin**[1,2‡],
Yajuan Lyu[3], **Qiaoqiao She**[3], **Weiping Wang**[1]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Baidu Inc, Beijing, China
{zhengmingyu,linzheng,wangweiping}@iie.ac.cn
{haoyang03,jiangwenbin,lvyajuan,sheqiaoqiao}@baidu.com

Abstract

Various datasets have been proposed to promote the development of Table Question Answering (TQA) technique. However, the problem setting of existing TQA benchmarks suffers from two limitations. First, they directly provide models with explicit table structures where row headers and column headers of the table are explicitly annotated and treated as model input during inference. Second, they only consider tables of limited types and ignore other tables especially complex tables with flexible header locations. Such simplified problem setting cannot cover practical scenarios where models need to process tables without header annotations in the inference phase or tables of different types. To address above issues, we construct a new TQA dataset with implicit and multi-type table structures, named IM-TQA, which not only requires the model to understand tables without directly available header annotations but also to handle multi-type tables including previously neglected complex tables. We investigate the performance of recent methods on our dataset and find that existing methods struggle in processing implicit and multi-type table structures. Correspondingly, we propose an RGCN-RCI framework outperforming recent baselines. We will release our dataset to facilitate future research.[1]

1 Introduction

To automatically gain valuable information from numerous tables, Table Question Answering (TQA) technique was developed and can answer natural language questions over tables (Zheng et al., 2022; Hui et al., 2022; Herzig et al., 2020). Correspondingly, researchers proposed various TQA datasets aiming at different application scenarios (Zhong et al., 2017; Iyyer et al., 2017; Chen et al., 2020). As the model performance continues to improve, TQA technique has been widely used in intelligent data analysis tools, e.g., Power BI[2], and Tableau[3]



Year	Team	Games Played	Minutes per game	Field goal percentage
1997–98	Spurs	82*	39.1	0.549
1998–99†	Spurs	50*	39.3	0.495
1999–00	Spurs	74	38.9	0.49
2000–01	Spurs	82*	38.7	0.499

(a) Vertical table

Name	Tim Duncan
Born	April 25, 1976 (age 46) Saint Croix, U.S. Virgin Islands
Listed height	6 ft 11 in (2.11 m)
Listed weight	250 lb (113 kg)
College	Wake Forest (1993–1997)
NBA draft	1997 / Round: 1 / Pick: 1st overall

(b) Horizontal table

Characteristic, data year (unit)	New Mexico	
	Value	Rank
Gross domestic product, 2020 ($millions)	98,472	39
Total R&D performance, 2019 ($millions)	7,614	24
Business R&D performance, 2019 ($millions)	637	40

(c) Hierarchical table

Device Name	Delivery Time	Major Parameter	
On-Load Tap Changer of #1 Transformer	Apr-12	Tap number	37
		Voltage regulation range	120±23 V
		Maximum rated current	700A
		Maximum voltage	624kV

(d) Complex table

Figure 1: Various types of tables in real world. Header cells are in different colors. Column Attribute (red), Column Index (green), Row Attribute (yellow), Row Index (blue). See Section 3.3 for details.

Though previous datasets have promoted the development of TQA technique, the problem setting of existing benchmarks suffers from two limitations. First, **existing benchmarks only evaluate the performance of model on tables with explicit table structures,** which means locations and directions of headers are explicitly annotated and treated as model input during inference. For example, Text2SQL benchmarks offer annotated column headers (Zhong et al., 2017; Yu et al., 2018) and recent hierarchical table datasets also contain hierarchical header annotations (Katsis et al., 2022; Cheng et al., 2022), which are available at inference time. This setting artificially lowers the difficulty of the task. Nevertheless, in practical scenarios, model may encounter plenty of tables without labeled headers. Manually annotating headers for these tables is prohibitively expensive and time-consuming. As a result, a benchmark is needed to evaluate the performance of TQA models on tables

[1] The code and data will be released at https://github.com/SpursGoZmy/IM-TQA

[*] Indicates equal contribution.

[†] This work was done during internship at Baidu Inc.

[‡] Corresponding author: Zheng Lin.

[2] https://powerbi.microsoft.com/en-us/

[3] https://www.tableau.com/

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 5074–5094
July 9-14, 2023 ©2023 Association for Computational Linguistics

with implicit table structures. Here, "implicit table structures" represents that header annotations of the table are not directly available during inference.

Second, **existing benchmarks only consider limited table types and ignore complex tables with flexible header locations.** Previous datasets mainly focus on vertical or hierarchical tables whose header cells only locate on the top or left side of the table (Pasupat and Liang, 2015; Wang et al., 2020b; Guo et al., 2021; Cheng et al., 2022), as shown in Figure 1 (a) and (c), but neglect the fact that model may require handling multi-type tables, especially complex tables whose header cells also appear at other positions in the table such as the bottom-right region, as illustrated in Figure 1 (d). Complex tables are prevalent in professional equipment specifications and record sheets, which are beyond the ability of current TQA systems.

Above analyses show that the problem setting of previous benchmarks restricts the application of TQA models. In this paper, we define a new problem, table question answering over implicit and multi-type table structures. In this problem setting, annotations of table headers are not available during inference and model needs to comprehend implicit and multi-type table structures to answer questions. To facilitate the study on this problem, we build the first Chinese **T**able **Q**uestion **A**nswering dataset with **I**mplicit and **M**ulti-type table structures, named **IM-TQA**, which consists of 1,200 tables and 5,000 questions (Section 3). Our dataset includes tables of four types from different domains, especially complex tables neglected by published studies. We annotate tables not only with various *Lookup* questions and their answer cells, but also with functional roles of table cells.

IM-TQA presents a strong challenge to existing models. Because of the inability to understand implicit and diverse table structures, the state-of-the-art baseline RCI (Row and Column Intersection) model (Glass et al., 2021) only achieves 49.6% overall accuracy and 23.8% accuracy on complex tables. To improve the performance of RCI, we propose an RGCN-RCI framework (Section 4). Specifically, this framework solves the new problem in two stages: (1) Table is modeled as graph and an RGCN (Schlichtkrull et al., 2017) is used to comprehend table structure and predict table header cells. (2) Based on predictions of RGCN, header information is merged into the text sequence representation of each row or column, which helps RCI model to predict whether a row or column contains the answer or not. Though our framework shows high effectiveness and improves the accuracy on all tables and complex tables by 3.8% (49.6% → 53.4%) and 8.2% (23.8% → 32.0%) respectively, it still lags far behind human performance, i.e., 95.1% and 94.1% on all and complex tables. We hope this work could raise attention to the implicit and multi-type table structures, facilitating the development of TQA models to address more complex and realistic table data.

To summarize, we conclude our contributions as follows:

- Considering the limitations of current TQA setting, we propose a new problem, table question answering over implicit and multi-type table structures, which is complementary to traditional TQA problem, i.e., TQA over explicit and limited (usually single-type) table structures.
- We construct and publicly release a new dataset, IM-TQA, to promote the research on this problem. Our dataset includes tables of four different types with implicit structures, especially complex tables ignored by former benchmarks.
- We investigate the performance of existing methods on our dataset and propose an RGCN-RCI framework which outperforms state-of-the-art baselines on all table types.

2 Problem Statement

In this section, we define the problem of table question answering over implicit and multi-type table structures. Consisting of multiple cells, a table t can be defined as $t \doteq \{\mathbb{P}, \mathbb{R}, \mathbb{V}\}$, where $\mathbb{P}$, $\mathbb{R}$ and $\mathbb{V}$ represents the set of position information, functional roles and values of cells in table t respectively. Table cells can be categorized into different types according to their functional roles, e.g., some cells are column headers and others are data cells.

Under the setting of traditional TQA problem, table structures are explicitly annotated (Zhong et al., 2017; Yu et al., 2018) or can be easily obtained by heuristic methods designed for specific table type (Cheng et al., 2022; Katsis et al., 2022). In this setting, functional roles $\mathbb{R}$ are directly provided for the model $f(\cdot)$, which outputs answer y given a natural language question q, i.e.,

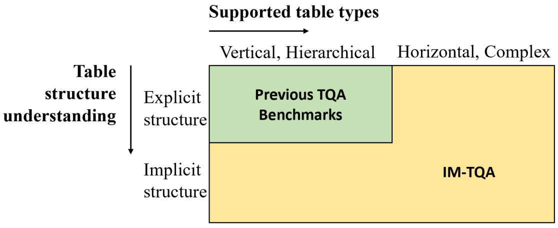


Figure 2: The improvements of IM-TQA compared with previous benchmarks.

0	1	2	2
3	4	5	6
3	4	7	8
3	4	9	10
3	4	11	12

(a) Cell ID Matrix

[Device Name, Delivery Time, Major Parameter, Record Time, On-Load Tap Changer of #1 Transformer, Apr-12, … ,]

(b) Cell Value List

Header Type	Cell ID List
Column Attribute	[0,1,2]
Row Attribute	[5,7,9,11]
Column Index	[]
Row Index	[3]

(c) Cell Type Annotation

Question	Answer Cells
What is the max rated current of the No.1 Transformer's on-load tap changer?	[10]

(d) QA Pair Annotation

Figure 3: An example of table storage format and annotations. Cell IDs constitute the matrix in (a) and are indices of cell value list in (b). Cells with same cell IDs stand for merged cells.

$y = f(q, t) = f(q, \mathbb{P}, \mathbb{R}, \mathbb{V})$. However, in the setting of real applications where the model may need to process implicit and multi-type table structures, functional roles $\mathbb{R}$ are not available in advance and the model requires distinguishing useful headers from data cells. Thus, TQA over implicit and multi-type table structures can be formalized as: $y = f(q, t^{'}) = f(q, \mathbb{P}, \mathbb{V})$. This problem setting poses a great challenge to existing methods. Unfortunately, none of previous benchmarks served as testbed for this problem. To fill the gap, we build IM-TQA, which includes multi-type tables and requires understanding implicit table structures. Compared with previous datasets, the improvements of IM-TQA are demonstrated in Figure 2.

3 Dataset Construction

In this section, we first introduce considered table types, and then elaborate the dataset construction procedure. We also compare IM-TQA with other related datasets. We recruit 10 professional annotators and provide them with sufficient instructions. The dataset construction totally costs 1,200 working hours. The ethical considerations will be discussed in the Section 9.

3.1 Considered Table Types

As shown in Figure 1, we divide tables into 4 types according to their structure characteristics, which is in line with previous work (Wang et al., 2021; Ghasemi-Gol and Szekely, 2018) with complex table as an important complement. More table and question examples are shown in **App.** F.

Vertical Table Table data is arranged in the vertical direction, with the first row as column headers and other rows as data tuples.

Horizontal Table Table data is arranged in the horizontal direction, with the first column as row headers and other columns as data tuples.

Hierarchical Table Table data is arranged in both vertical and horizontal directions, with headers exhibiting a multi-level hierarchical structure.

Complex Table In tables of above 3 types, header cells only locate on the top or left side of the table. But in complex tables, headers also appear at other positions such as bottom-right region in the table and can be mixed with data cells, as depicted in Figure 1 (d). Such tabular structures with flexible header locations often appear in professional equipment specifications and record sheets, presenting a great challenge to existing methods.

3.2 Table Collection and Storage

To ensure the diversity of table data, we collect tables from open websites of different domains. Data sources include company annual reports from different industries[4] such as manufacturing, medicine and education; web pages of Baidu Encyclopedia[5] on science, professional equipment, etc. Tables were extracted to Excel files by annotators. We correct typos in the collected tables and filter tables without meaningful data. Finally, we preserve 300 tables for every table type. Figure 4 shows the distribution of domains.

In order to store various tables, we design a storage method which separately stores cell positions $\mathbb{P}$ and cell contents $\mathbb{V}$. To store cell positions, a *cell ID* is assigned to each table cell in the row-first order. For a table including m rows and n columns, its cell IDs constitute an $m \times n$ matrix representing cell locations. This matrix contains table layout information such as neighbouring relations between different cells. As for cell contents, every cell value is put into a list in the same row-first order. Figure

[4] http://eid.csrc.gov.cn/101111/index.html

[5] https://baike.baidu.com/

3 (a) and (b) demonstrate the storage format of the complex table in Figure 1 (d). The original table can be recovered based on cell ID matrix and cell value list. The storage method does not waste extra space for merged cells and is also convenient to annotate header cells and answer cells.

3.3 Cell Type Annotation

In our problem setting, models are required to recognize functional roles $\mathbb{R}$ of table cells, i.e., conducting cell type classification (CTC) task. There are different taxonomies of cell types which focus on hierarchical spreadsheet tables (Ghasemi Gol et al., 2019; Dong et al., 2019; Zhang et al., 2021). To model different table structures especially complex tables, we adopt a new taxonomy of cell types with the concentration on header cells that are useful for TQA models to locate correct answer cells. Specifically, we categorize table cells into 5 types based on their functional roles.

Row Attribute and Column Attribute Row attribute and column attribute are traditional table headers which describes other cells in the same row and in the same column respectively, e.g., yellow cells and red cells in Figure 1. Attribute cells only serve the purpose of describing other cells and they are not meaningful data.

Row Index and Column Index Row index and column index are individual cells that are used to index data records in the row or column orientation, e.g., blue cells and green cells in Figure 1. Index cells are also meaningful data. For instance, in vertical tables, data cells in the primary key column are unique identifiers of each row.

Pure Data Pure data cells are the core body of a table. They do not have the function of describing or indexing other cells and their meanings should be understood with the help of above header cells.

We instructed annotators in distinguishing 5 cell types and asked them to annotate cell ID lists of attribute and index cells, as shown in Figure 3 (c). Other table cells are deemed pure data cells.

3.4 QA Pairs Construction

After identifying header cells, we asked annotators to raise questions about data cells and label answer cell IDs, as illustrated in Figure 3 (d). To avoid annotation artifacts from the homogeneous patterns of questions, e.g., always using the same question expression, annotators were asked to use diverse language expressions to raise questions and answers' locations should change frequently. Annotators were also encouraged to paraphrase questions to increase difficulty.

Questions about table are broadly classified into two types: *Lookup* and *Aggregation* (Glass et al., 2021). *Lookup* questions require selecting table cells as answers whereas *Aggregation* questions are answered by performing arithmetic operations over a subset of cells, such as *Sum()*. In this work, our primary focus is on *Lookup* questions and we leave the annotation of more complex aggregation and numerical reasoning questions in the future. Besides single-cell answer, annotators were allowed to select one row or one column or arbitrary table cells as answer. Figure 4 shows the distribution of question types. Four questions were raised for each vertical and horizontal table, and five questions for every hierarchical and complex table.

3.5 Data Review

In order to guarantee annotation quality, before each annotation task began, all annotators conducted a trial annotation with 50 samples and the results were checked. Feedback and sufficient Q&As were given to corresponding annotators until they fully comprehended the annotation requirements. After all annotation tasks finished, we and two most experienced annotators performed the final review to fix labeling errors. We inspected annotations for the correctness of cell type annotations and answer cell annotations. We also inspected the grammar and wording and filtered questions with obvious mistakes. We checked for offensive content and identifiers, and replaced identifying information with mono-directional hashes. Finally, we preserve 1,200 tables and 5,000 questions. Inner-annotator agreement and annotation instruction screenshots are shown in **App.** A and E respectively.

3.6 Dataset Analysis and Comparison

Table 1 shows a comprehensive comparison of IM-TQA to related TQA datasets. The advantages of the proposed dataset are as follows: (1) It is the first TQA dataset that contains implicit and multi-type table structures, especially complex tables ignored by former datasets. Though AIT-QA (Katsis et al., 2022), HiTab (Cheng et al., 2022) and MultiHiertt (Zhao et al., 2022) include vertical and hierarchical tables, they ignore the other two types. (2) It is annotated with cell functional roles and QA pairs, which supports both CTC and TQA task. Diverse table structures in our dataset challenge ex-

Dataset	Language	Table source	#Tables	#Questions	Avg. Q len	Implicit	Table type				Task	
							Ver	Hor	Hie	Com	CTC	TQA
NL2SQL (Sun et al., 2020)	Chinese	Reports, Spreadsheets	6,029	64,891	11	-	✓	-	-	-	-	✓
Cspider (Min et al., 2019)	Chinese	Spider	876	9,691	11.9	-	✓	-	-	-	-	✓
DuSQL (Wang et al., 2020b)	Chinese	Baidu Baike, Reports, Forums	813	28,762	19.3	-	✓	-	-	-	-	✓
CHASE (Guo et al., 2021)	Chinese	DuSQL, SParC	1,280	17,940	13	-	✓	-	-	-	-	✓
WTQ (Pasupat and Liang, 2015)	English	Wikipedia	2,108	22,033	10	-	✓	-	-	-	-	✓
WikiSQL (Zhong et al., 2017)	English	Wikipedia	26,521	80,654	11.7	-	✓	-	-	-	-	✓
Spider (Yu et al., 2018)	English	College data,WikiSQL	1,020	10,181	13.2	-	✓	-	-	-	-	✓
ToTTo (Parikh et al., 2020)	English	Wikipedia	83,141	-	-	-	✓	-	✓	-	-	-
AIT-QA (Katsis et al., 2022)	English	Annual reports	116	515	12.9	-	✓	-	✓	-	-	✓
HiTab (Cheng et al., 2022)	English	Stat. reports, Wiki	3,597	10,672	16.5	-	✓	-	✓	-	-	✓
MultiHiertt (Zhao et al., 2022)	English	FinTabNet	9,775	10,440	16.8	-	✓	-	✓	-	-	✓
IM-TQA(ours)	**Chinese**	**Reports, Baidu Encyclopedia**	**1200**	**5000**	**13.1**	✓	✓	✓	✓	✓	✓	✓

Table 1: Comparison of IM-TQA to related TQA datasets. Implicit represents implicit table structures. Ver, Hor, Hie and Com denotes vertical, horizontal, hierarchical and complex table respectively.

	Train	Valid	Test	Total
# table structures	466	40	80	586
# tables	936	111	153	1,200
# questions	3,909	464	627	5,000
# vertical tables	224	31	45	300
# horizontal tables	230	34	36	300
# hierarchical tables	231	35	34	300
# complex tables	251	11	38	300

Table 2: Dataset split statistics

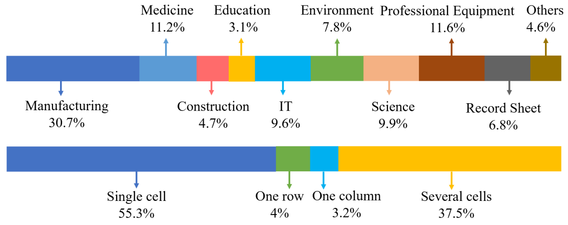


Figure 4: Distribution of domains and question types.

isting CTC and TQA models. (3) Compared with single-domain datasets like AIT-QA and WTQ, IM-TQA includes tables from various domains.

We split tables of each type based on table structures. If the location distributions of header cells in two tables are exactly same, i.e., same cell type at the same table position, we consider they share the same table structure. Table structures are randomly split into train (80%), valid (7%) and test set (13%). Tables of the same structure and questions of the same table are assigned to the same split. Table 2 shows the basic statistics of each split. More statistics are shown in **App.** A.

4 Method

We propose the RGCN-RCI framework for IM-TQA, shown in Figure 5. RGCN-RCI consists of an RGCN-based cell type classification module and an RCI-based table question answering module. CTC module uses an RGCN to aggregate neighbour cell information and predicts cell's functional role. Based on the predictions of CTC module, TQA module adopts an improved RCI model to predict whether a row or column contains the answer or not. Final answer cells are selected from intersections of positive rows and positive columns.

4.1 Cell Type Classification Module

We convert the table into a graph, where nodes are table cells, and neighbouring relations between cells are edges. The resulted graph is processed by an RGCN to predict cell types.

Cell Features The initial node representation vector consists of two parts: hand-crafted features and semantic features extracted from the pre-trained language model. We select 24 available manual features from Koci et al. (2016) such as the cell text length (listed in **App.** B.3). The 24-dim integer vectors are transformed into 32-dim continuous numerical vectors by a trained auto encoder. We input cell text to the pre-trained BERT (Devlin et al., 2019) and take the 768-dim output vector of the special [CLS] token as semantic features of the whole cell. In the end, hand-crafted features and semantic features are concatenated to produce 800-dim initial node representation.

Edges We design four directed edges which point from each cell to its surrounding neighbour cells: *top to down*, *left to right*, *down to top*, *right to left*. We argue that neighbour cell information is important for predicting cell functional roles. For example, most data cells are surrounded by other data cells. As the table is converted to a heterogeneous graph, a relational graph convolutional network (RGCN) (Schlichtkrull et al., 2017) is used to aggregate neighbour cell information in different orientations and updates node representations. The updated node representations are input to the final linear layer to predict cell types.

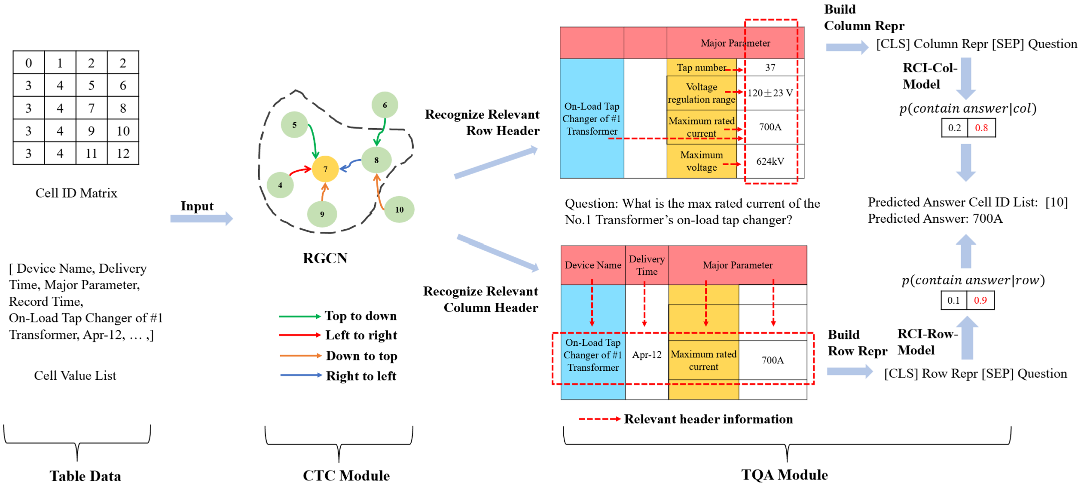


Figure 5: Overview of the proposed RGCN-RCI framework, with the table of Figure 1 (d) as a running example.

4.2 Table Question Answering Module

RCI (Glass et al., 2021) is a state-of-the-art TQA model for *Lookup* questions. It concatenates a text sequence representation of each row (or column) to the question text, and uses a pre-trained language model like ALBERT (Lan et al., 2020) to predict whether the row/column contains the answer or not. Cells on the intersection of positive rows and positive columns are final answers. When constructing the textual representation of each row (or column), RCI incorporates the structure of vertical table. The row textual representation is the concatenation of "column header : cell text", and the column textual representation is the concatenation of column header and all cell texts in this column.

However, this representation method does not fit other types of tables. Due to the inability to understand implicit and diverse table structures, this method may include irrelevant headers or miss useful header information. For example, when constructing the representation of the third row in Figure 1 (b), RCI treats all cells in the first row as column headers and gives wrong textual representation:

> Name : Listed height | Tim Duncan : 6 ft 11 in (2.11 m) |

where ":" is a delimiter token between column header and cell text, and "|" is a delimiter token between different cells. When building the representation of the fourth column in Figure 1 (d), it will lose relevant headers in the third column and output representation without necessary information:

> Major Parameter | 37 | 120±23 V | 700A | 624kV |

To overcome the defect of RCI's original textual representation method, we propose a new representation method with the help of predicted cell types from CTC module and cell ID matrix. Specifically, when constructing row textual representation, we locate the nearest column attribute and column index for every data cell in this row. These column headers are regarded as relevant headers and will be concatenated with the corresponding data cell. Similarly, when constructing column textual representation, the nearest row attribute and row index will be concatenated with the corresponding data cell in this column. Again, let's take the third row in Figure 1 (b) as an example. Based on CTC module's predicted result and cell ID matrix, "Tim Duncan" is the nearest column index to the data cell "6 ft 11 in (2.11 m)", and "Name" is a row attribute irrelevant to cells in the third row. The new row textual representation is:

> Listed height | Tim Duncan : 6 ft 11 in (2.11 m) |

For the fourth column in Figure 1 (d), its new column representation is:

> Major Parameter | On-Load Tap Changer of #1 Transformer - Tap number : 37 | On-Load Tap Changer of #1 Transformer - Voltage regulation range : 120±23 V | … |

where "-" is a delimiter token between index cell and attribute cell. Compared with original textual representation method, our proposed method helps RCI exclude irrelevant headers and include useful headers, which contributes to final predictions.

Model	All tables							Complex tables							Gap(%)↓
	Per-class F1(%)↑					Macro F1(%)↑	Macro F1-header(%)↑	Per-class F1(%)↑					Macro F1(%)↑	Macro F1-header(%)↑	
	PD	CA	RA	CI	RI			PD	CA	RA	CI	RI			
RF	95.0	80.4	69.4	78.6	65.2	77.7±0.2	73.4±0.3	92.0	67.2	64.4	29.8	53.6	61.4±1.4	53.8±1.6	16.3
MLP	95.8	79.8	76.2	78.3	72.3	80.5±0.4	76.7±0.7	91.8	64.8	80.0	43.8	54.3	66.9±0.5	60.7±0.8	13.6
CNN-BERT†	96.6	87.4	84.4	71.0	75.8	83.0±0.8	79.7±0.8	93.8	81.0	86.6	33.4	58.8	70.7±1.2	65.0±1.4	12.3
Bi-LSTM	97.2	91.4	87.4	79.0	79.4	86.9±0.3	84.3±0.5	93.8	86.6	90.2	60.4	72.2	80.6±0.7	77.4±0.8	6.30
RAT	96.0	85.3	82.5	82.0	78.3	84.8±0.3	82.0±0.4	92.3	73.0	85.0	69.5	61.0	76.2±0.5	72.1±0.5	8.60
RGCN (ours)	**98.8**	**92.4**	**89.6**	**85.6**	**85.4**	**90.4±0.5**	**88.3±0.7**	**97.2**	**89.4**	**93.0**	**69.6**	**79.4**	**85.7±0.6**	**82.9±0.9**	**4.70**

Table 3: CTC results on all tables and complex tables. PD, CA, RA, CI and RI are acronyms of five cell types, e.g., PD denotes Pure Data. † represents our implementation. RF stands for random forest. ± stands for standard deviation over 5 repeated experiments. More CTC results are shown in **App.** C.1 due to space limitation.

5 Experiments

5.1 Cell Type Classification

5.1.1 Experimental Setup

Considered baselines are as follows. **Random Forest** (Koci et al., 2016): A random forest classifier is used to conduct CTC task based on manual features. **CNN-BERT** (Dong et al., 2019): A method using BERT to extract semantic features and a CNN to learn spatial correlations between cells. **Bi-LSTM** (Ghasemi Gol et al., 2019): Two bidirectional LSTM are used to capture dependencies between different cells, one observing the sequence of cells in each row, and the other in each column. **MLP**: Directly applying a two-layer neural network to cell features to predict cell types without neighbour information. **RAT** (Shaw et al., 2018): The Relation-Aware-Transformer whose self-attention mechanism is extended to consider the edge label between each pair of nodes. In this setting, the table can be seen as a directed complete graph where each cell can communicate with all the other cells to update its node representations. Implementation details can be found in **App.** B.1.

5.1.2 Experimental Results

Table 3 shows CTC results on all tables and complex tables, averaged over 5 repeated experiments. Models are evaluated with F1 score. Macro F1-header represents the macro F1 on four header types. Gap represents the Macro F1 difference between all tables and complex tables.

From the results shown in Table 3, we can observe that: (1) RGCN achieves the best macro F1 on all tables and complex tables, beating the best-performing baseline Bi-LSTM by 3.5% and 5.1% respectively, which demonstrates the effectiveness of our proposed graph-based CTC model over various table types, especially complex table. (2) CNN-BERT, Bi-LSTM, RAT and RGCN aggregates information from different neighbour cells. The best performance of RGCN indicates that neighbour information in a local area is most important for CTC task. This is consistent with the intuition that human can determine cell types based on a small part of the table and do not need to read the whole table. (3) Performance on complex tables is worse than overall performance, showing that CTC models struggle in comprehending complex table structures with flexible header locations. More analyses are shown in **App.** C.1.

Model	Exact Match Acc(%)				
	All	Ver	Hor	Hie	Com
Ernie-Layout	11.6±0.5	11.5	4.10	5.66	22.6
Tapex	13.1±0.8	14.9	10.7	8.18	17.4
RAT†	18.5±0.9	34.5	33.6	5.03	4.07
TAPAS	33.2±0.8	58.0	31.1	26.4	15.7
RCI	47.2±0.3	68.4	45.1	56.0	19.2
RCI-AIT	49.6±0.4	69.5	43.4	60.4	23.8
RGCN-RCI (ours)	**53.4±0.4**	**70.7**	**45.9**	**62.9**	**32.0**
RGCN-RCI + Oracle headers	55.3±0.3	73.0	46.7	66.7	33.1
Human	95.1±0.6	96.6	95.1	94.3	94.1

Table 4: TQA results on the proposed dataset. † represents our implementation. Ver, Hor, Hie and Com denotes four table types, respectively. ± stands for standard deviation over 5 repeated experiments.

5.2 Table Question Answering

5.2.1 Experimental Setup

We include following TQA baselines. **RCI** (Glass et al., 2021): A state-of-the-art model with its original textual representation method. **RCI-AIT** (Katsis et al., 2022): A variant of RCI for AIT-QA dataset with the textual representation method designed for hierarchical tables. **TAPAS** (Herzig et al., 2020): A classic table pre-training model which takes the linearization of question and table cells as input and outputs token representations for answer cell selection. **Tapex** (Liu et al., 2022): A recent table pre-training model based on BART (Lewis et al., 2020) which can directly generate answer text given the flattened table and the question. **Ernie-Layout** (Peng et al., 2022): A visually-rich document understanding pre-training model which conducts TQA via prompting. **RAT**: Table cells and question are converted to a graph and cell representations are updated by an RAT model. We follow Mueller et al. (2019) to use cell-level relations such as cell to column header. **Human**: We evaluate the average performance of five annotators with bachelor degree. Implementation details are shown in **App.** B.2.

5.2.2 Experimental Results

Table 4 shows exact match accuracy on the IM-TQA test set, averaged over 5 repeated experiments. "RGCN-RCI" denotes our proposed framework. "RGCN-RCI + Oracle headers" represents using new representation with annotated headers.

From the results shown in Table 4, we can find that: (1) Existing TQA models struggle on our dataset. The best baseline RCI-AIT achieves an overall accuracy of 49.6%, and its accuracy on complex tables is only 23.8%. This shows that models designed for specific table types cannot achieve a great generalization on multi-type tables, especially previously ignored complex tables. (2) Compared with baselines which lack the ability to comprehend implicit and diverse table structures, RGCN-RCI framework achieves better performance and improves the accuracy on all tables and complex tables by 3.8% and 8.2% respectively, which demonstrates the effectiveness of our proposed textual representation method. RGCN-RCI + Oracle headers can achieve more performance boost, which further proves that correctly understanding table structure is beneficial to question answering over implicit and multi-type tables. A case study is shown in **App.** C.3. (3) Table pre-training models like TAPAS and Tapex still cannot generalize well on various table types. This is probably because they were originally designed for and only pre-trained on vertical tables with explicit structures available. Considering more diverse table types in pre-training stage may improve their performance on multi-type tables. (4) Interestingly, Ernie-Layout performs better on complex tables than other tables. An explanation might be that it is biased by layout information from document where answer text is often next to the keywords in reference text, which makes it more difficult for the model to locate answers that are far from their header cells. More discussions are given in **App.** D. (5) The performance of RAT is poor because it treats input tables as vertical tables to build the graph. When handling other tables, the resulted graph will contain wrong relations between different cells which hinder the model from locating correct answers. (6) Compared with human baseline, even with an oracle providing correct headers, RGCN-RCI only achieves 33.1% accuracy on complex tables. Therefore there is a long way to go for TQA models to accurately find correct answers in complicated tables. More analyses are shown in **App.** C.2.

5.2.3 Error Analysis

We randomly selected 100 error cases of RGCN-RCI framework to conduct error analysis. Error cases fall into four categories: (1) Row mistakes(40%): model fails to select the correct row which contains answer. (2) Column mistakes(28%): model fails to select the correct column. This shows that predicting the correct column is usually easier than predicting the correct row. (3) Row and column mistakes(14%): model selects wrong row and wrong column at the same time, which is common in complex tables(57%). (4) Missing rows or columns(18%): model mistakenly predicts that none of rows or columns contains answer, which often results from the paraphrased question expression such as synonyms. Model may need external knowledge to perform better on these questions.

We also analyse the performance of RGCN-RCI on different question types. The exact match score (51.2%) on the questions whose answers are arbitrary several cells is lower than that on the other questions whose answers are single cell (54.9%), cells in one column (55.6%) and cells in one row (55.0%). This shows that it is more difficult for the model to find answer cells with flexible and discontinuous locations.

5.2.4 Ablation Study of Header Cells

To analyse the contribution of different header cells to the final TQA prediction, we conduct ablation study by removing different headers when constructing new textual representation for each row and column. We use different textual representation to train and evaluate RCI models. As we can see from the results shown in Table 5, header cells have different contributions to the final TQA results. The attribute header cells are more important than index cells as they are most significant for understanding simple data cells. And the row attribute cells seem to be more important than column ones.

If we remove all header cells and simply concatenate every cell to build row/column representation, the model performance would suffer a dramatic decrease to 46.3% which is worse than original RCI method designed for vertical tables. This further validates that header cells are beneficial for RCI model to locate the correct answer cells.

Model	Exact Match Acc(%)				
	All	Ver	Hor	Hie	Com
RGCN-RCI + Oracle CTC	**55.3**	**73.0**	46.7	**66.7**	**33.1**
w/o Column Attribute	51.5	64.9	**47.5**	62.9	30.2
w/o Row Attribute	49.3	70.7	40.2	58.5	25.6
w/o Column Index	53.3	70.1	44.2	64.8	31.9
w/o Row Index	52.6	68.9	42.6	65.4	31.4
w/o Attribute Cells	47.7	69.5	44.3	52.2	23.8
w/o Index Cells	48.6	59.1	39.3	63.5	30.8
w/o All Headers	46.3	62.6	41.8	61.0	19.2

Table 5: TQA results based on different textual representation methods by removing different header cells.

6 Related Work

Table QA methods can be categorized into two types: semantic-parsing-based methods which transform the question into an executable logical form such as SQL (Wang et al., 2020a; Hui et al., 2022; Guo et al., 2019), and non-semantic-parsing methods which directly output final answers without generating logical forms (Yang et al., 2022; Liu et al., 2022). Researchers also proposed various TQA datasets (Iyyer et al., 2017; Chen et al., 2020). Most of previous datasets consist of vertical tables with regular structure except HiTab (Cheng et al., 2022), AIT-QA (Katsis et al., 2022) and MultiHiertt (Zhao et al., 2022), which also consider hierarchical tables but ignore other table types especially complex tables. By contrast, IM-TQA is the first dataset that include tables with implicit and multi-type structures.

Cell type classification is crucial for table structure understanding, which aims at recognizing table cells' functional roles. Previous work proposed different taxonomies of cell types, which mainly focused on spreadsheet tables (Dong et al., 2019; Zhang et al., 2021; Koci et al., 2016). Coarse-grained taxonomies classify table cells according to their roles in basic table structure (Sun et al., 2021), e.g., header, metadata, data. Fine-grained taxonomies will subdivide header cells or data cells according to more specific functions (Zhang et al., 2021). The proposed taxonomy belongs to coarse-grained taxonomies and we focus on headers that are helpful to locating answers.

Document Understanding is another interesting and related research direction, which requires model to answer questions over a document image that may also contain tables (Borchmann et al., 2021; Zhu et al., 2022; Huang et al., 2022). Compared with TQA models which usually consider table as semi-structured text data, document understanding models consider table as a part of document picture and try to understand table structures based on visual information. These actually represents two different perspectives to tackle table data. We give more discussion about document understanding in **App.** D.

7 Conclusion

We propose a new problem, table question answering over implicit and multi-type table structures, and construct a corresponding dataset named IM-TQA by collecting tables from various domains. Tables in our dataset range from the traditional simple table with fixed headers to the complex table with flexible headers, which poses a new challenge to previous methods. Besides QA pairs, we also annotate functional roles of table cells to promote the understanding of implicit table structures. In experiments, we benchmark recent methods on CTC and TQA tasks on IM-TQA, and propose a two-stage RGCN-RCI framework that outperforms existing methods. Experimental results show that IM-TQA can provide a challenging and valuable testbed for future research.

8 Limitations

Our proposed dataset is in Chinese and focuses on single tables. Though we translate the dataset from Chinese into English, we think it is better to directly construct a corresponding large-scale English TQA dataset in consideration of data quality. To build such a dataset with limited resource, one can fully utilize abundant tables in existing English TQA datasets. As an exploration of table question answering over implicit and multi-type structures, this work focuses on *Lookup* questions and we leave annotations of *Aggregation* and other numerical reasoning questions in the future work.

The proposed RGCN-RCI framework is not an end-to-end model. We look forward to seeing more end-to-end models which can simultaneously learn table structures and question answering with the help of our dataset. In addition, it takes more time to convert very large tables into graph structures which are used to recognize cell functional roles.

9 Ethical Considerations

Our proposed benchmark is a free and open resource for the community to study table question answering over implicit and multi-type tables. We follow existing works AITQA (Katsis et al., 2022) and DuSQL (Wang et al., 2020b) to select data sources when building our dataset. Tables are collected from the public organization China Securities Regulatory Commission and Baidu Encyclopedia of Baidu Company. We got the reply and permission from these two organizations which allow us to share and redistribute data for non-commercial use. Tables in these annual reports and web pages are open to public so there is no privacy risk.

In the annotation process, we asked annotators to check if there exist any offensive content such as insulting or discriminatory speech. They did not find any such content in our benchmark. We also checked for identifiers and replaced identifying information with mono-directional hashes. We recruit 10 professional annotators with bachelor degree (6 males and 4 females) and pay them at a price of 6.5 dollars per hour (above the average local payment of similar jobs). The total time cost for annotation is 1,200 working hours. Annotators were informed that these labeled data would be used as a table question answering dataset. The data collection protocol was approved by an ethics review board of an IT company. Main experiments in this paper can be run on a single NVIDIA GeForce RTX 3090 GPU. Our dataset follows the Computational Use of Data Agreement v1.0[6].

Acknowledgements

This work was supported by National Natural Science Foundation of China (No. 61976207).

References

Łukasz Borchmann, Michał Pietruszka, Tomasz Stanislawek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Graliński. 2021. Due: End-to-end document understanding benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. 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.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. 2022. HiTab: A hierarchical table dataset for question answering and natural language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1094–1110, Dublin, Ireland. Association for Computational Linguistics.

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*.

Majid Ghasemi Gol, Jay Pujara, and Pedro Szekely. 2019. Tabular cell classification using pre-trained cell embeddings. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 230–239.

Majid Ghasemi-Gol and Pedro Szekely. 2018. Tabvec: Table vectors for classification of web tables.

Michael Glass, Mustafa Canim, Alfio Gliozzo, Saneem Chemmengath, Vishwajeet Kumar, Rishav Chakravarti, Avi Sil, Feifei Pan, Samarth Bharadwaj, and Nicolas Rodolfo Fauceglia. 2021. Capturing row and column semantics in transformer based question answering over tables.

Jiaqi Guo, Ziliang Si, Yu Wang, Qian Liu, Ming Fan, Jian-Guang Lou, Zijiang Yang, and Ting Liu. 2021. Chase: A large-scale and pragmatic Chinese dataset for cross-database context-dependent text-to-SQL. 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 2316–2331, Online. Association for Computational Linguistics.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy. Association for Computational Linguistics.

[6] https://cdla.dev/computational-use-of-data-agreement-v1-0/

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, 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, Online. Association for Computational Linguistics.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking.

Binyuan Hui, Ruiying Geng, Lihan Wang, Bowen Qin, Yanyang Li, Bowen Li, Jian Sun, and Yongbin Li. 2022. S^2SQL: Injecting syntax to question-schema interaction graph encoder for text-to-SQL parsers. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1254–1262, Dublin, Ireland. Association for Computational Linguistics.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831, Vancouver, Canada. Association for Computational Linguistics.

Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. 2022. AIT-QA: Question answering dataset over complex tables in the airline industry. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 305–314, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Elvis Koci, Maik Thiele, Oscar Romero, and Wolfgang Lehner. 2016. A machine learning approach for layout inference in spreadsheets. IC3K 2016, page 77–88, Setubal, PRT. SCITEPRESS - Science and Technology Publications, Lda.

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 *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022. TAPEX: Table pre-training via learning a neural SQL executor. In *International Conference on Learning Representations*.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. 2020. Docvqa: A dataset for vqa on document images.

Qingkai Min, Yuefeng Shi, and Yue Zhang. 2019. A pilot study for chinese SQL semantic parsing. *CoRR*, abs/1909.13293.

Thomas Mueller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering conversational questions on structured data without logical forms. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5902–5910, Hong Kong, China. Association for Computational Linguistics.

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, 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.

Qiming Peng, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Teng Hu, Weichong Yin, Yongfeng Chen, Yin Zhang, Shikun Feng, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2022. Ernie-layout: Layout knowledge enhanced pre-training for visually-rich document understanding.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. Modeling relational data with graph convolutional networks.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Kexuan Sun, Harsha Rayudu, and Jay Pujara. 2021. A hybrid probabilistic approach for table understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4366–4374.

Ningyuan Sun, Xuefeng Yang, and Yunfeng Liu. 2020. Tableqa: a large-scale chinese text-to-sql dataset for table-aware sql generation.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Lijie Wang, Ao Zhang, Kun Wu, Ke Sun, Zhenghua Li, Hua Wu, Min Zhang, and Haifeng Wang. 2020b. DuSQL: A large-scale and pragmatic Chinese text-to-SQL dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6923–6935, Online. Association for Computational Linguistics.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021. Tuta: Tree-based transformers for generally structured table pre-training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery amp; Data Mining*, KDD '21, page 1780–1790, New York, NY, USA. Association for Computing Machinery.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2020a. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020b. Layoutlm: Pre-training of text and layout for document image understanding. KDD '20, page 1192–1200, New York, NY, USA. Association for Computing Machinery.

Jingfeng Yang, Aditya Gupta, Shyam Upadhyay, Luheng He, Rahul Goel, and Shachi Paul. 2022. Tableformer: Robust transformer modeling for table-text encoding.

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.

Yakun Zhang, Xiao Lv, Haoyu Dong, Wensheng Dou, Shi Han, Dongmei Zhang, Jun Wei, and Dan Ye. 2021. Semantic table structure identification in spreadsheets. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2021, page 283–295, New York, NY, USA. Association for Computing Machinery.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6588–6600, Dublin, Ireland. Association for Computational Linguistics.

Yanzhao Zheng, Haibin Wang, Baohua Dong, Xingjun Wang, and Changshan Li. 2022. HIE-SQL: History information enhanced network for context-dependent text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2997–3007, Dublin, Ireland. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

Fengbin Zhu, Wenqiang Lei, Fuli Feng, Chao Wang, Haozhou Zhang, and Tat-Seng Chua. 2022. Towards complex document understanding by discrete reasoning. In *Proceedings of the 30th ACM International Conference on Multimedia*. ACM.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. *CoRR*, abs/2105.07624.

A More Information about Dataset Construction and Statistics

Table 6 shows IM-TQA's basic statistical information such as average question number, row number, column number per table. We also count the number of distinct table structures. We think that table structure is represented by the distribution of locations of header cells. If header cells of two tables are distributed in the same way, i.e., same cell type at the same table position, we consider they share the same table structure. As expected, due to much more flexible header locations, structure number of complex tables (241) and hierarchical tables (153) are larger than that of vertical tables (103) and horizontal tables (89). Table 7 shows number of different table headers.

In our formal annotation process, every annotator handled annotation of different samples to meet the annotation schedule. To further quantitatively study the inner-annotator agreement, we randomly selected three annotators and asked them to annotate the same 100 samples in two annotation tasks. The resulted Fleiss' Kappa of cell type annotation task and answer cell annotation task are 0.86 and 0.88 respectively, which can be interpreted as 'Almost perfect agreement' (Landis and Koch, 1977).

Characteristic	Value
Avg. question number per table	4.1
Row number per table (median/mean)	6/7.3
Column number per table (median/mean)	5/5.2
Cell number per table (median/mean)	23/31.6
Header cell number per table (median/mean)	10/12.7
Avg. answer num per question	1.68

Table 6: Dataset basic statistics

	Ver	Hor	Hie	Com	Total
# table structures	103	89	153	241	586
# Column Attribute	1,158	99	2,185	1,434	4,876
# Row Attribute	145	1,445	831	3,649	6,070
# Column Index	0	1,092	16	268	1,376
# Row Index	1,355	0	872	781	3,008

Table 7: Different table structure and header number.

B Implementation Details

B.1 Implementation Details for CTC Experiments

In this paper, we use PaddlePaddle[7] and PGL[8] to implement our model and build graphs. We use valid set for model selection and hyper-parameter tuning, and then evaluate the best model on test set. We use bert-base-chinese to extract cell semantic features. For the RGCN model, the graph neural network (GNN) layer number is 4, the dimension of hidden layers is 800, and the ReLu activation is used between adjacent GNN layers. We train RGCN for 20 epochs. The Adam optimizer is adopted with learning rate of 1e-4. To mitigate the class-imbalance in the CTC task, we follow Ghasemi Gol et al. (2019) and use a weighted *Cross Entropy Loss* as our loss function. We set the weight w_k of cell type *k* to be inversely proportional to the number of cells with class type *k* in train set n_k^{train}, i.e., $w_k = 1 - \frac{n_k^{train}}{\sum_{k'=1}^{5} n_{k'}^{train}}$.

For RAT baseline, we follow the implementation in Wang et al. (2020a). The RAT layer number and the attention head number is 8. The probability for dropout layers is set to 0.1. The AdamW optimizer is adopted with learning rate of 2e-5, warmup fraction of 0.1 and weight decay of 0.01. We train RAT for 50 epochs with batch size of 16. For MLP baseline, we adopt a multilayer perceptron of one hidden layer with hidden size 800 and ReLu activation. For CNN-BERT baseline, we adopt a text CNN (Kim, 2014) on cell feature matrix along the row direction. For other baselines, we follow the original experimental setup in their papers. PSL^{RF} (Sun et al., 2021) is a very recent CTC model, but the code has not been publicly available. Thus, we do not compare with this method.

B.2 Implementation Details for TQA Experiments

We follow the original paper to train RCI model with our proposed textual representation method. We use bert-base-chinese as the sequence-pair classifier, which is trained for 3 epochs with batch size of 64. The AdamW optimizer is adopted with learning rate of 2e-5, warmup fraction of 0.1 and weight decay of 0.01. The probability of dropout layers is set to 0.1. The max grad norm is set to 1.0. For RAT baseline, to achieve better performance in multi-turn TQA task, Mueller et al. (2019) also introduce a node in the graph representing the answer of last question. We do not use these unnecessary designs. For TAPAS baseline, We pre-train a TAPAS-base model on about 6 million Chinese tables. The pre-training was run on four Tesla V100 GPU for about one week. The table corpus used for pre-training was collected from Baidu Encyclopedia. In fine-

[7] https://www.paddlepaddle.org.cn/

[8] https://github.com/PaddlePaddle/PGL

Model	Number of Parameters	Training Time
Pretrained BERT	110M	-
Auto Encoder	<0.01M	10.5 minutes
Random Forest	n_estimators=100	2 minutes
MLP	6M	1.5 minutes
CNN-BERT	<0.1M	3 minutes
Bi-LSTM	1.5M	10 minutes
RAT	<80M	1 hour
RGCN	30M	7 minutes
TAPAS	110 M	pre-training: 1 week finetune: 1 hour
Tapex	121M	1 hour
RCI	110 M×2	1 hour
RCI-AIT	110 M×2	1 hour

Table 8: Model parameter number and training time.

tuning stage, we concatenate output representations of the first and the last character in the cell text as cell representation, which is used to predict the probability of selecting the cell as answer. Note that out of resource limitation, we only pretrained TAPAS for once. For Tapex baseline, we are unable to pre-train a new Tapex model for Chinese due to the lack of synthetic Chinese SQL corpus. As an alternative, we adopt a commercial machine translation model[9] to translate tables and questions in IM-TQA from Chinese into English and finetune a pre-trained Tapex-base model with the official fine-tuning script. But it should noted that the performance of the model maybe influenced by the translation quality. For Ernie-Layout baseline, we transform tables into document images and adopt the official released model to evaluate its performance on our dataset. For RCI-AIT baseline, we use the same transformations from the original paper to transform all tables into vertical tables. Transformations include taking row headers as table cells in a new column and flatting hierarchical headers by concatenating parent header text with children text. Main model parameter number and training time are listed in Table 8. Main experiments in this paper can be run on a single NVIDIA GeForce RTX 3090 GPU.

B.3 Complete Manual Features

We list all manual features in Table 9. We consider 15 content features and 9 spatial features. We follow Ghasemi Gol et al. (2019) and train an auto encoder to transform 24-dim integer vectors into 32-dim continuous numerical vectors. We train the auto encoder by a vector reconstruction task. The auto encoder tries to reconstruct the input integer vector at its output, and generates continuous vector representations at the output of the encoder layer. Mean square error is used as loss function. At test time, we feed the 24-dim integer manual vectors into the trained auto encoder and gain the 32-dim continuous vector from the encoder output.

Manual Cell Features
LENGTH#(character-level)
NUM OF TOKENS#(word-level)
LEADING SPACES#
IS NUMERIC?
STARTS WITH NUMBER?
STARTS WITH SPECIAL?
IS CAPITALIZED?
IS UPPER CASE?
IS ALPHABETIC?
CONTAINS SPECIAL CHARS?
CONTAINS PUNCTUATIONS?
CONTAINS COLON?
WORDS LIKE TOTAL?
WORDS LIKE TABLE?
IN YEAR RANGE?
ROW NUMBER#
COL NUMBER#
NEIGHBOUR CELL NUM#
HAS 0 NEIGHBORS?
HAS 1 NEIGHBORS?
HAS 2 NEIGHBORS?
HAS 3 NEIGHBORS?
HAS 4 NEIGHBORS?
IS MERGED CELL?

Table 9: Manual Feature List

C More Experimental Analyses

C.1 More CTC Experimental Analyses

(4) The performance of MLP baseline is the worst among methods based on neural networks, which proves that neighbour information is essential for distinguishing divergent types of cells. (5) F1 scores on the column index (CI) and row index (RI) are lower than that on the column attribute (CA) and row attribute (RA). This indicates that, compared with attribute cells that are not data, it is more difficult for CTC models to distinguish index cells from pure data cells. (6) CTC results on vertical, horizontal and hierarchical tables are shown in Table 10, Table 11 and Table 12, which demonstrates that our graph-based method achieves a better generalization on tables of different types.

Model	Vertical tables						
	per-class F1(%)					Macro F1(%)	Macro F1 -Header(%)
	PD	CA	RA	CI	RI		
RF	93.5	92.3	35.2	-	78.7	74.9	68.7
MLP	98.5	87.5	40.5	-	86.3	78.2	71.4
CNN-BERT	98.0	93.8	56.6	-	87.8	84.1	79.4
Bi-LSTM	98.8	**98.4**	61.8	-	88.8	87.0	83.0
RAT	97.1	95.2	55.3	-	87.4	83.8	79.3
RGCN	**99.3**	96.3	**62.7**	-	**94.6**	**88.2**	**84.5**

Table 10: CTC results on vertical tables.

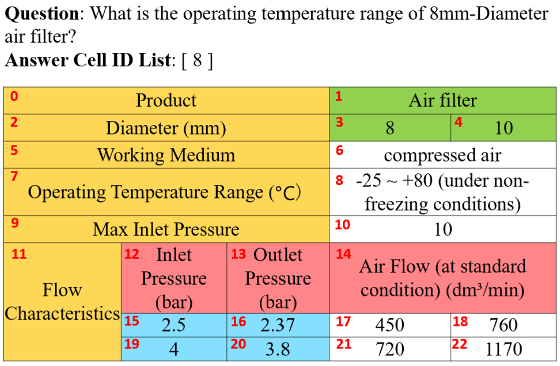
[9] https://cloud.baidu.com/product/mt/text_trans

Model	Horizontal tables						
	per-class F1(%)					Macro F1(%)	Macro F1 -Header(%)
	PD	CA	RA	CI	RI		
RF	94.0	8.20	79.3	90.1	-	67.9	59.2
MLP	95.0	16.0	88.5	81.5	-	70.3	62.0
CNN-BERT	97.2	18.0	94.8	81.4	-	72.9	64.7
Bi-LSTM	98.6	18.8	94.2	79.6	-	72.8	64.2
RAT	96.3	15.0	90.4	88.2	-	72.5	64.5
RGCN	**99.4**	**20.4**	**95.6**	**92.4**	-	**77.0**	**69.5**

Table 11: CTC results on horizontal tables.

Model	Hierarchical tables						
	per-class F1(%)					Macro F1(%)	Macro F1 -Header(%)
	PD	CA	RA	CI	RI		
RF	93.3	92.6	50.7	-	64.4	75.3	69.2
MLP	98.6	95.1	51.6	-	73.0	79.6	73.2
CNN-BERT	98.8	95.8	55.6	-	75.6	81.4	75.7
Bi-LSTM	98.4	**98.4**	67.6	-	75.2	84.9	80.4
RAT	99.2	98.1	64.5	-	**83.7**	86.4	82.1
RGCN	**99.6**	98.0	**68.4**	-	82.0	**87.0**	**82.8**

Table 12: CTC results on hierarchical tables.



Question: What is the operating temperature range of 8mm-Diameter air filter?
Answer Cell ID List: [8]

<table>
<tr><td colspan="3">0 Product</td><td colspan="2">1 Air filter</td></tr>
<tr><td colspan="3">2 Diameter (mm)</td><td>3 8</td><td>4 10</td></tr>
<tr><td colspan="3">5 Working Medium</td><td colspan="2">6 compressed air</td></tr>
<tr><td colspan="3">7 Operating Temperature Range (°C)</td><td colspan="2">8 -25 ~ +80 (under non-freezing conditions)</td></tr>
<tr><td colspan="3">9 Max Inlet Pressure</td><td colspan="2">10 10</td></tr>
<tr><td rowspan="3">11 Flow Characteristics</td><td>12 Inlet Pressure (bar)</td><td>13 Outlet Pressure (bar)</td><td colspan="2">14 Air Flow (at standard condition) (dm³/min)</td></tr>
<tr><td>15 2.5</td><td>16 2.37</td><td>17 450</td><td>18 760</td></tr>
<tr><td>19 4</td><td>20 3.8</td><td>21 720</td><td>22 1170</td></tr>
</table>

Figure 6: A real case (translated to English) where RCI fails but RGCN-RCI gives the correct answer.

C.2 More TQA Experimental Analyses

(7) As a non-table-pre-training method, RCI beats the TAPAS on all model types, which is consistent with the results in Glass et al. (2021). This shows that selecting answer cells based on column and row information maybe better than directly predicting whether a cell is the answer or not. This is more close to the way that human locates answer cells. (8) RCI-AIT achieves better performance than RCI especially on hierarchical tables, which shows the effectiveness of its special textual representation method. But it cannot perform well on all table types as its special representation method aims at hierarchical table and may include irrelevant information when applied to other tables. (9) Compared with TAPAS, the recent Tapex model achieves worse performance. We suppose the reason is that TAPAS contains extra position embeddings such as column/row ID embedding, which may benefit the model to understand diverse table structures to some extent. By contrast, Tapex is totally based on BART (Lewis et al., 2020) architecture and does not include such special designs, which increases difficulty for the model to understand various table structures except vertical tables. In addition, Tapex autoregressively outputs answer(s) separated by commas. We think it is also difficult for this fashion to generate answer text in multiple answer cells.

C.3 TQA Case Study

Figure 6 shows a QA sample on a complex table containing parameters of an air filter. In this case, original RCI cannot correctly determine which column contains the answer as its representation of the fourth column lacks necessary row header information, such as "Operating Temperature Range". By contrast, RGCN-RCI with our new textual representation method could incorporate useful row header information into its column representation and finally locates the correct answer.

D Discussion about Document Understanding and LayoutLM

Document understanding aims at converting a document into meaningful information, and it involves several subtasks including answering questions over tables in a document. Document understanding benchmarks (Borchmann et al., 2021; Zhu et al., 2022; Mathew et al., 2020) emphasize that no structured representation of the underlying document text is provided, such as a table structure given in advance, and it has to be learned by models from the input document file. From this perspective, document understanding requires a more general "implicit" structure understanding ability which needs to handle more data structure types, such as tables, graphs and lists and so on.

By contrast, IM-TQA mainly focuses on implicit structure understanding over different tables. However, IM-TQA considers more diverse table types and table structures than document understanding benchmarks like DUE (Borchmann et al., 2021) and TAT-DQA (Zhu et al., 2022). For instance, DUE mainly considers vertical tables from the WTQ (Pasupat and Liang, 2015) dataset and TAT-DQA considers vertical or hierarchical tables from the TAT-QA (Zhu et al., 2021) dataset. We believe that it is meaningful for document understanding benchmarks to include more table types and we hope IM-TQA could also serve as a useful resource to achieve this target.

LayoutLMs (Xu et al., 2020b,a; Huang et al., 2022; Peng et al., 2022) are important and recent methods for the Document Understanding task, which shares some similarity with our motivation of understanding implicit table structures. LayoutLM jointly models interactions between text and layout information across scanned document images and it supports several downstream document understanding tasks including answering questions over tables in a document. We converted tables into document images and investigated the performance of a state-of-the-art Chinese LayoutLM Ernie-Layout (Peng et al., 2022).

From the results shown in Table 4, we find that the Ernie-Layout model struggles in dealing with diverse table structures. Interestingly, its performance on complex table is better than that on other table types. We suppose the reason is that the LayoutLM is biased towards excessively utilizing layout information from the scanned document images. For instance, in a registration form of personal information, given a key (e.g., "ID number"), its corresponding value is much more likely on its right or below rather than on the left or above. The LayoutLM tends to use such layout information to answer questions about document but such biased information may distract the model from locating the correct answer cells in the table. As a result, the model performs better on complex tables where lots of answer cells are next to their header cells, but it struggles on other tables where answer cells are relatively farther from their header cells.

E Instructions for Annotators

Screenshots of original instructions for annotators have been saved and shown in this section. The instruction for CTC annotation task is shown in Figure 12. And the instruction for TQA annotation task is shown in Figure 13. A real CTC annotation case of complex table is shown in Figure 14. Besides instructions, we also provided annotators with sufficient QAs to ensure that they fully comprehended the annotation requirements.

F More Table and Question Examples

Figure 7, Figure 8, Figure 9, Figure 10 and Figure 11 depict more table examples and question examples. We translated them into English for reading convenience. Different header cells are represented with the same color as Figure 1. As demonstrated in these examples, header cells of complex table are more flexible than those of traditional table. They may appear at arbitrary positions and can be mixed with other data cells. Such table structures have not been thoroughly investigated and challenge existing TQA methods.

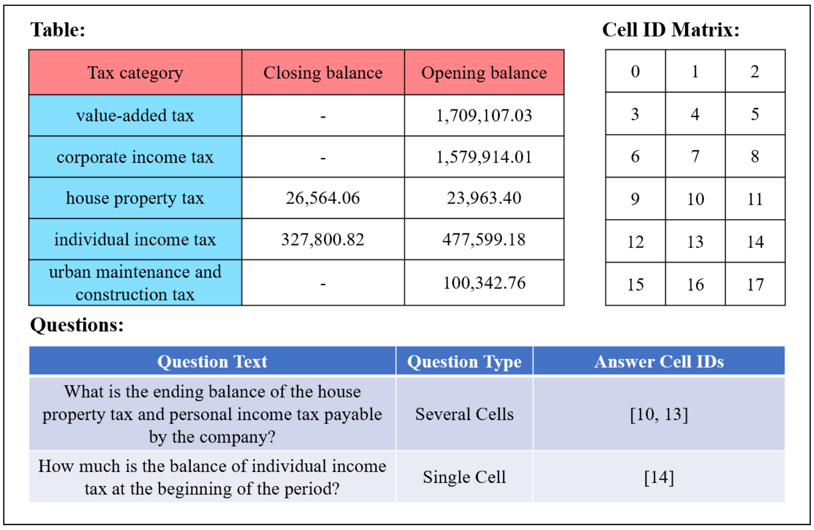


Table:

Tax category	Closing balance	Opening balance
value-added tax	-	1,709,107.03
corporate income tax	-	1,579,914.01
house property tax	26,564.06	23,963.40
individual income tax	327,800.82	477,599.18
urban maintenance and construction tax	-	100,342.76

Cell ID Matrix:

0	1	2
3	4	5
6	7	8
9	10	11
12	13	14
15	16	17

Questions:

Question Text	Question Type	Answer Cell IDs
What is the ending balance of the house property tax and personal income tax payable by the company?	Several Cells	[10, 13]
How much is the balance of individual income tax at the beginning of the period?	Single Cell	[14]

Figure 7: A vertical table example and question examples.

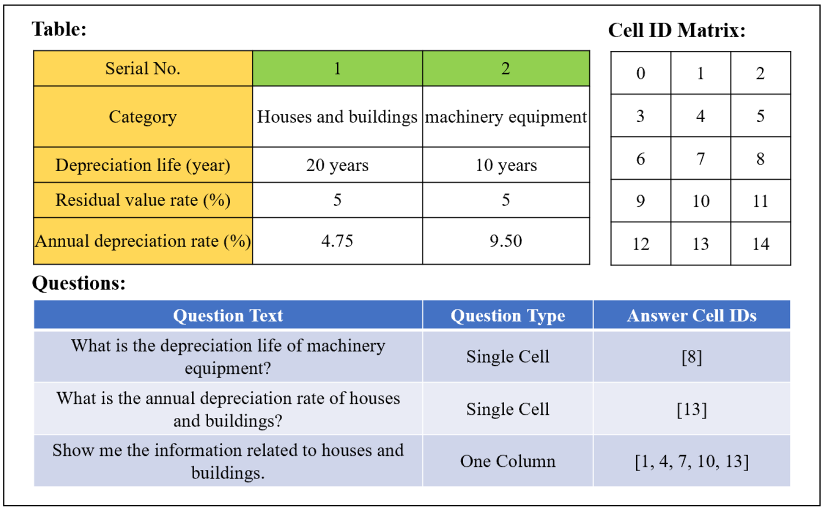


Table:

Serial No.	1	2
Category	Houses and buildings	machinery equipment
Depreciation life (year)	20 years	10 years
Residual value rate (%)	5	5
Annual depreciation rate (%)	4.75	9.50

Cell ID Matrix:

0	1	2
3	4	5
6	7	8
9	10	11
12	13	14

Questions:

Question Text	Question Type	Answer Cell IDs
What is the depreciation life of machinery equipment?	Single Cell	[8]
What is the annual depreciation rate of houses and buildings?	Single Cell	[13]
Show me the information related to houses and buildings.	One Column	[1, 4, 7, 10, 13]

Figure 8: A horizontal table example and question examples.

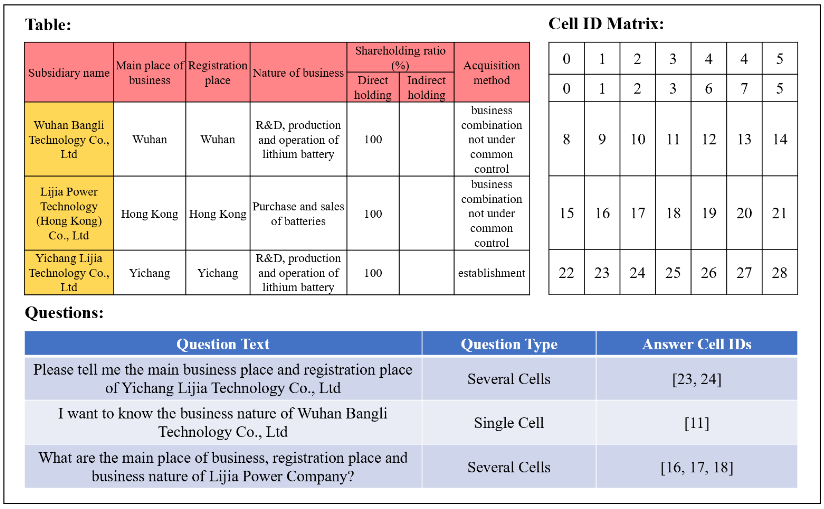


Table:

Subsidiary name	Main place of business	Registration place	Nature of business	Shareholding ratio (%)		Acquisition method
				Direct holding	Indirect holding	
Wuhan Bangli Technology Co., Ltd	Wuhan	Wuhan	R&D, production and operation of lithium battery	100		business combination not under common control
Lijia Power Technology (Hong Kong) Co., Ltd	Hong Kong	Hong Kong	Purchase and sales of batteries	100		business combination not under common control
Yichang Lijia Technology Co., Ltd	Yichang	Yichang	R&D, production and operation of lithium battery	100		establishment

Cell ID Matrix:

0	1	2	3	4	4	5
0	1	2	3	6	7	5
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

Questions:

Question Text	Question Type	Answer Cell IDs
Please tell me the main business place and registration place of Yichang Lijia Technology Co., Ltd	Several Cells	[23, 24]
I want to know the business nature of Wuhan Bangli Technology Co., Ltd	Single Cell	[11]
What are the main place of business, registration place and business nature of Lijia Power Company?	Several Cells	[16, 17, 18]

Figure 9: A hierarchical table example and question examples.

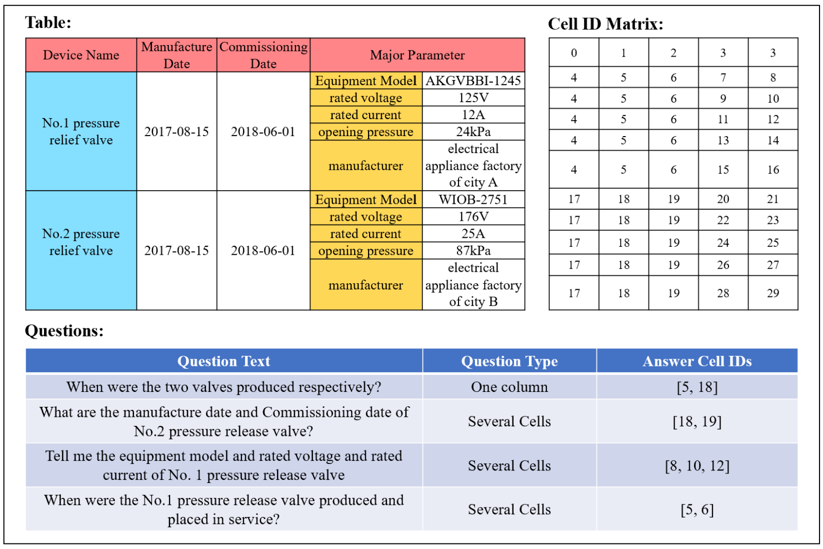


Table:

Device Name	Manufacture Date	Commissioning Date	Major Parameter	
No.1 pressure relief valve	2017-08-15	2018-06-01	Equipment Model	AKGVBBI-1245
			rated voltage	125V
			rated current	12A
			opening pressure	24kPa
			manufacturer	electrical appliance factory of city A
No.2 pressure relief valve	2017-08-15	2018-06-01	Equipment Model	WIOB-2751
			rated voltage	176V
			rated current	25A
			opening pressure	87kPa
			manufacturer	electrical appliance factory of city B

Cell ID Matrix:

0	1	2	3	3
4	5	6	7	8
4	5	6	9	10
4	5	6	11	12
4	5	6	13	14
4	5	6	15	16
17	18	19	20	21
17	18	19	22	23
17	18	19	24	25
17	18	19	26	27
17	18	19	28	29

Questions:

Question Text	Question Type	Answer Cell IDs
When were the two valves produced respectively?	One column	[5, 18]
What are the manufacture date and Commissioning date of No.2 pressure release valve?	Several Cells	[18, 19]
Tell me the equipment model and rated voltage and rated current of No. 1 pressure release valve	Several Cells	[8, 10, 12]
When were the No.1 pressure release valve produced and placed in service?	Several Cells	[5, 6]

Figure 10: A complex table about pressure relief valves, where many row attributes are mixed with data cells.

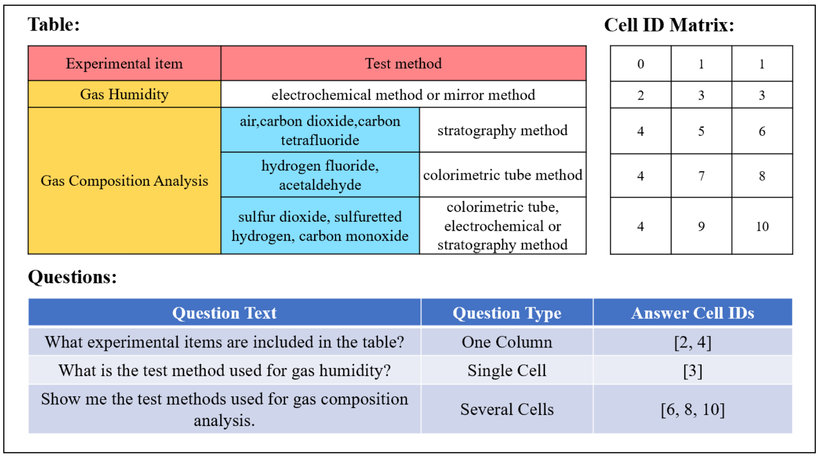


Table:

Experimental item	Test method	
Gas Humidity	electrochemical method or mirror method	
Gas Composition Analysis	air,carbon dioxide,carbon tetrafluoride	stratography method
	hydrogen fluoride, acetaldehyde	colorimetric tube method
	sulfur dioxide, sulfuretted hydrogen, carbon monoxide	colorimetric tube, electrochemical or stratography method

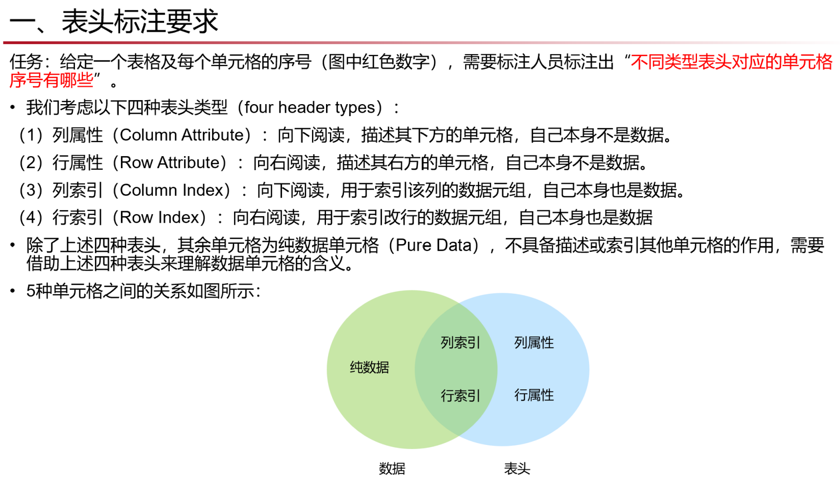
Cell ID Matrix:

0	1	1
2	3	3
4	5	6
4	7	8
4	9	10

Questions:

Question Text	Question Type	Answer Cell IDs
What experimental items are included in the table?	One Column	[2, 4]
What is the test method used for gas humidity?	Single Cell	[3]
Show me the test methods used for gas composition analysis.	Several Cells	[6, 8, 10]

Figure 11: A complex table about gas test requirements. A col attribute cell, a pure data cell, and three row indices appear in sequence in the second column. Such mixed data arrangement increases difficulty for TQA models to find the correct answer.



一、表头标注要求

任务：给定一个表格及每个单元格的序号（图中红色数字），需要标注人员标注出“不同类型表头对应的单元格序号有哪些”。

- 我们考虑以下四种表头类型（four header types）：

（1）列属性（Column Attribute）：向下阅读，描述其下方的单元格，自己本身不是数据。

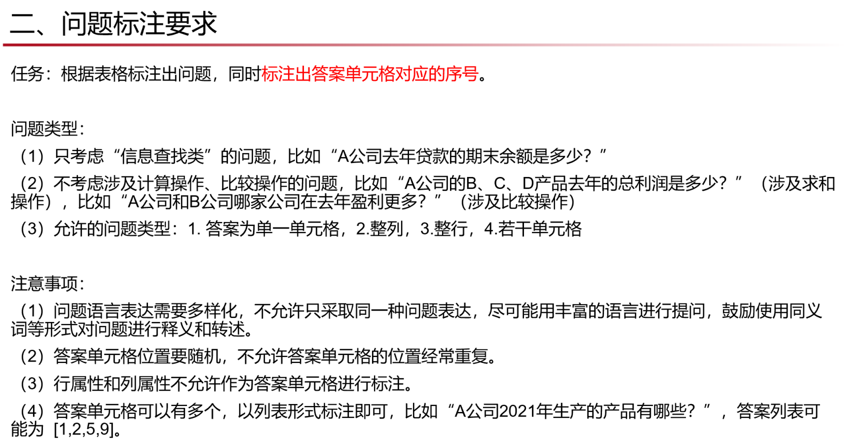
（2）行属性（Row Attribute）：向右阅读，描述其右方的单元格，自己本身不是数据。

（3）列索引（Column Index）：向下阅读，用于索引该列的数据元组，自己本身也是数据。

（4）行索引（Row Index）：向右阅读，用于索引改行的数据元组，自己本身也是数据

- 除了上述四种表头，其余单元格为纯数据单元格（Pure Data），不具备描述或索引其他单元格的作用，需要借助上述四种表头来理解数据单元格的含义。
- 5种单元格之间的关系如图所示：

Figure 12: Instructions(in screenshot) for CTC annotation tasks.



二、问题标注要求

任务：根据表格标注出问题，同时标注出答案单元格对应的序号。

问题类型：

（1）只考虑“信息查找类”的问题，比如“A公司去年贷款的期末余额是多少？”

（2）不考虑涉及计算操作、比较操作的问题，比如“A公司的B、C、D产品去年的总利润是多少？” （涉及求和操作），比如“A公司和B公司哪家公司在去年盈利更多？” （涉及比较操作）

（3）允许的问题类型：1. 答案为单一单元格，2.整列，3.整行，4.若干单元格

注意事项：

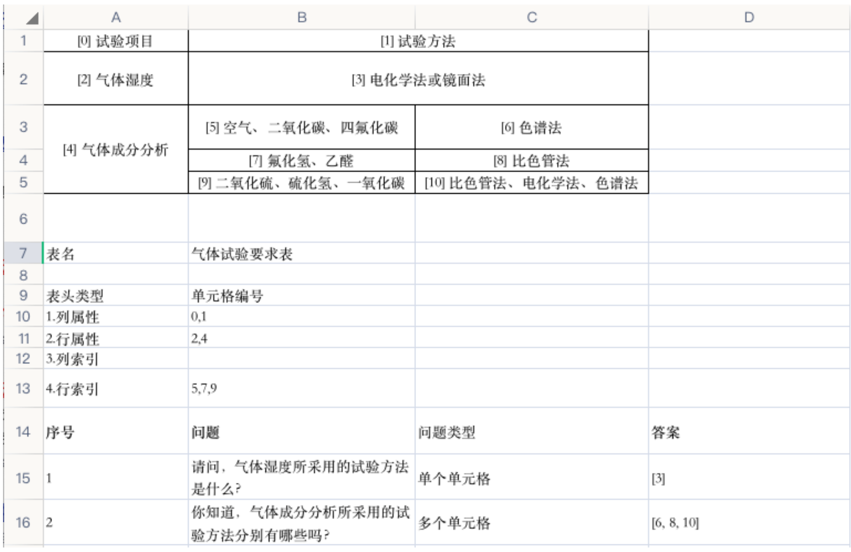
（1）问题语言表达需要多样化，不允许只采取同一种问题表达，尽可能用丰富的语言进行提问，鼓励使用同义词等形式对问题进行释义和转述。

（2）答案单元格位置要随机，不允许答案单元格的位置经常重复。

（3）行属性和列属性不允许作为答案单元格进行标注。

（4）答案单元格可以有多个，以列表形式标注即可，比如“A公司2021年生产的产品有哪些？”，答案列表可能为［1,2,5,9]。

Figure 13: Instructions(in screenshot) for TQA annotation tasks.



	A	B	C	D
1	[0] 试验项目	[1] 试验方法		
2	[2] 气体湿度	[3] 电化学法或镜面法		
3	[4] 气体成分分析	[5] 空气、二氧化碳、四氟化碳	[6] 色谱法	
4		[7] 氟化氢、乙醛	[8] 比色管法	
5		[9] 二氧化硫、硫化氢、一氧化碳	[10] 比色管法、电化学法、色谱法	
6				
7	表名	气体试验要求表		
8				
9	表头类型	单元格编号		
10	1.列属性	0,1		
11	2.行属性	2,4		
12	3.列索引			
13	4.行索引	5,7,9		
14	序号	问题	问题类型	答案
15	1	请问，气体湿度所采用的试验方法是什么?	单个单元格	[3]
16	2	你知道，气体成分分析所采用的试验方法分别有哪些吗?	多个单元格	[6, 8, 10]

Figure 14: A real annotation case(in screenshot).

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 8 Limitations.

☑ A2. Did you discuss any potential risks of your work?
Section 9 Ethical Considerations.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section 1 Introcution.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 3 Dataset Construction and Section 4 Method.

☑ B1. Did you cite the creators of artifacts you used?
Section 3 Dataset Construction and Section 4 Method.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section 9 Ethical Considerations.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 3 Dataset Construcution and Section 9 Ethical Considerations.

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Section 3 Dataset Construcution and Section 9 Ethical Considerations.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 3 Dataset Construcution and Appendix A.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 3 Dataset Construction.

C ☑ Did you run computational experiments?

Section 5 Experiments

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5 Experiments and Appendix B.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5.1.1, 5.2.1 and Appendix B.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5.1.2, 5.2.2 and Appendix C.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix B.

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Section 3 Dataset Construcution, Section 9 Ethical Considerations and Appendix E.

☑ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Section 9 Ethical Considerations and Appendix E.

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Section 9 Ethical Considerations.

☑ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Section 9 Ethical Considerations.

☑ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Section 9 Ethical Considerations.

☑ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Section 9 Ethical Considerations.