AIT-QA: Question Answering Dataset over Complex Tables in the Airline Industry

Yannis Katsis[1], **Saneem Chemmengath**[1], **Vishwajeet Kumar**[1], **Samarth Bharadwaj**[1], **Mustafa Canim**[1], **Michael Glass**[1], **Alfio Gliozzo**[1], **Feifei Pan**[2], **Jaydeep Sen**[1], **Karthik Sankaranarayanan**[1], **Soumen Chakrabarti**[3]
[1]IBM Research [2]Rensselaer Polytechnic Institute [3]IIT Bombay
yannis.katsis@ibm.com, saneem.cg@in.ibm.com, vishk024@in.ibm.com, samarth.b@in.ibm.com, mustafa@us.ibm.com, mrglass@us.ibm.com, gliozzo@us.ibm.com, panf2@rpi.edu, jaydesen@in.ibm.com, kartsank@in.ibm.com, soumen.chakrabarti@gmail.com

Abstract

Recent advances in transformers have enabled Table Question Answering (Table QA) systems to achieve high accuracy and SOTA results on open domain datasets like WikiTableQuestions and WikiSQL. Such transformers are frequently pre-trained on open-domain content such as Wikipedia, where they effectively encode questions and corresponding tables from Wikipedia as seen in Table QA dataset. However, web tables in Wikipedia are notably *flat* in their layout, with the first row as the sole column header. The layout lends to a relational view of tables where each row is a tuple. Whereas, tables in domain-specific business or scientific documents often have a much more *complex* layout, including hierarchical row and column headers, in addition to having specialized vocabulary terms from that domain.

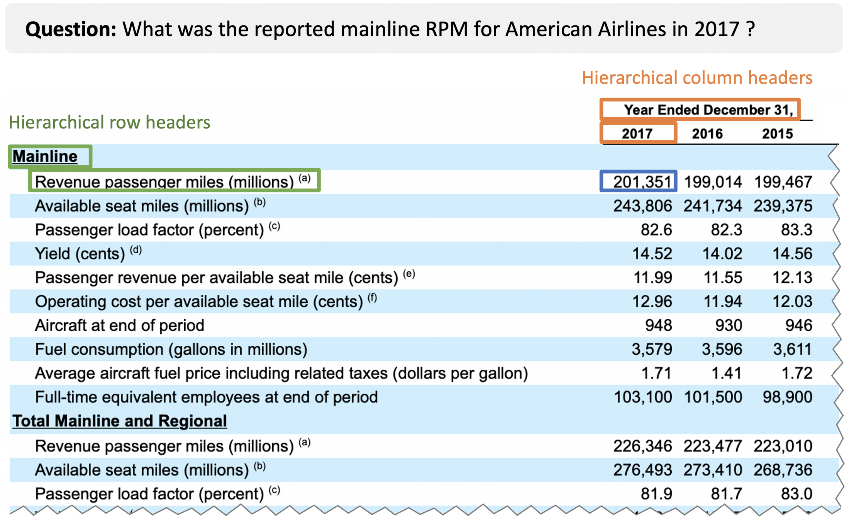
To address this problem, we introduce the domain-specific Table QA dataset AIT-QA (Airline Industry Table QA). The dataset consists of 515 questions authored by human annotators on 116 tables extracted from public U.S. SEC filings[1] of major airline companies for the fiscal years 2017-2019. We also provide annotations pertaining to the nature of questions, marking those that require hierarchical headers, domain-specific terminology, and paraphrased forms. Our zero-shot baseline evaluation of three transformer-based SOTA Table QA methods - TaPAS (end-to-end), TaBERT (semantic parsing-based), and RCI (row-column encoding-based) - clearly exposes the limitation of these methods in this practical setting, with the best accuracy at just 51.8% (RCI). We also present pragmatic table pre-processing steps used to pivot and project these complex tables into a layout suitable for the SOTA Table QA models.

1 Introduction

The tabular data format is commonly used in digital documents such as PDFs and HTMLs to store semi-structured information [5; 37; 26]. Due to the rich content found in tables, many studies have been carried out on extracting information out of the tables [2] and leveraging it for various NLP tasks, such as answering questions over tables [3; 35; 30; 31]. The quality of answers depends on first, high quality extraction of tables out of documents (aka *table extraction*); second, retrieval of relevant tables for a given natural language question or query keyword (aka *table retrieval*); and finally, identification of the relevant cells over the retrieved tables (aka *table QA*). Most recently,

[1]SEC Filings publicly available at: https://www.sec.gov/edgar.shtml

Submitted to the 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks. Do not distribute.



	Year Ended December 31,		
	2017	2016	2015
Mainline			
Revenue passenger miles (millions) [(a)]	201,351	199,014	199,467
Available seat miles (millions) [(b)]	243,806	241,734	239,375
Passenger load factor (percent) [(c)]	82.6	82.3	83.3
Yield (cents) [(d)]	14.52	14.02	14.56
Passenger revenue per available seat mile (cents) [(e)]	11.99	11.55	12.13
Operating cost per available seat mile (cents) [(f)]	12.96	11.94	12.03
Aircraft at end of period	948	930	946
Fuel consumption (gallons in millions)	3,579	3,596	3,611
Average aircraft fuel price including related taxes (dollars per gallon)	1.71	1.41	1.72
Full-time equivalent employees at end of period	103,100	101,500	98,900
Total Mainline and Regional			
Revenue passenger miles (millions) [(a)]	226,346	223,477	223,010
Available seat miles (millions) [(b)]	276,493	273,410	268,736
Passenger load factor (percent) [(c)]	81.9	81.7	83.0

Figure 1: A question-table pair in AIT-QA, showcasing the complex structure of tables in the dataset. The table cell containing the answer is highlighted in blue and its hierarchical column and row headers are highlighted in orange and green, accordingly.

transformer-based pre-trained architectures such as TaBERT [36], TaPas [14], and RCI [12] have been proposed to tackle the table QA task by identifying table cells containing the answer to a given question. These advanced models have been shown to perform well in answering questions over tables. However, most of these studies claim high accuracy in Table QA by evaluating the proposed techniques on open in-domain datasets, such as WikiTableQuestions [27] and WikiSQL dataset [39]; both built on top of Wikipedia tables.

However, based on our prior experience in table processing [2], open domain web tables typically exhibit a much simpler structure than tables found in scientific or business documents. For instance, consider the sample question-table pair from our proposed airline industry dataset shown in Figure 1. The table contains both column headers and row headers (i.e., descriptors of the contents of columns and rows, respectively) and both of them are hierarchical in nature. Moreover, answering the question often requires reasoning on these complex column and row header hierarchies. For instance, finding the requested mainline Revenue Passenger Miles (which are contained in the cell highlighted in blue) requires understanding and leveraging the fact that the cell has two hierarchical row headers "Mainline" and "Revenue passenger miles" (shown in green). Ignoring the row headers or not reasoning on the entire row header hierarchy may lead to the wrong result. For instance, if we simply searched for cells with a flat row header containing "Revenue Passenger Miles", we may mistakenly return the value 226,346 appearing further down the table. This value indeed corresponds to Revenue Passenger Miles (RPMs) but these are the RPMs of the entire *mainline and regional operations* of the airline, instead of only the *mainline* operations requested by the question. In contrast, web tables appearing in open domain Table QA datasets, such as WikiTableQuestions or WikiSQL, exhibit significantly simpler structures. Such tables do not contain any row headers at all and only have a single column header, closely resembling relational database tables.

Moreover, while open domain datasets capture common entities, such as locations, person names, etc, which often appear in Wikipedia articles, they often lack domain-specific vocabulary that one encounters in scientific or business documents (such as the "Revenue Passenger Miles" above).

Our experiments (reported in Section 4) show that even the most advanced transformer-based pre-trained models struggle to understand the layout of these domain specific tables and find the right answer of natural language questions. We argue that the lack of a domain-specific datasets for Table QA is partially responsible for this incompetency, as these models are all evaluated on open domain datasets, where tables contain limited specialized vocabulary and adopt a simple column/row layout.

To address this gap in the Table QA literature and assist the community in improving the performance of Table QA approaches for domain specific use cases, in this paper we propose a domain specific dataset, where tables are extracted from financial documents in the airline industry. The majority of the tables exhibit a complex structure, including hierarchical row and column headers, large amounts of numerical values, as well as technical vocabulary terms specific to the airline industry. To the best of our knowledge, this is the first dataset tailored for the Table QA task that includes and explicitly encodes such complex table layouts, domain-specific table contents, as well as questions manually created by human annotators to test Table QA algorithms.

This work makes the following contributions:

- **A complex and domain specific Table QA dataset** called *AIT-QA (Airline Industry Table QA)*, created by human annotators based on 10-K financial reports of major airline companies. The questions are created based on the content of tables appearing in the 10-K reports, as well as KPIs (Key Performance Indicators); i.e., important metrics commonly used by analysts in the airline industry.
- **Experimental evaluation of state-of-the-art Table QA models** applied on AIT-QA, demonstrating that high performance of open domain datasets with simple table structures does not guarantee similar performance on domain-specific datasets containing complex tables, further motivating the need for a domain-specific TableQA dataset.
- **A novel data pre-processing technique for existing Table QA models**, which improves their performance on datasets with complex table structures. This is accomplished by translating complex table structures (including hierarchical row and column headers) to simpler table structures resembling the structure of the tables on which such approaches have been trained.

The paper is structured as follows: We start by reviewing existing datasets on answering questions over tables in Section 2. We then describe our proposed AIT-QA dataset and how it was constructed in Section 3. Section 4 describes experimental results of state-of-the-art models over AIT-QA and finally Section 5 concludes the paper.[2]

2 Related Work

Existing work on leveraging tables to answer questions has in general focused on two distinct tasks: (a) *Table retrieval*; i.e., given a corpus of tables, identifying the table that contain the answer to a question, and (b) *Table QA*; i.e., given a single table containing the answer to the question, finding the answer of that question. While our dataset is tailored for the Table QA task, we next summarize existing datasets proposed for either of the two aforementioned tasks.

2.1 Table Retrieval Datasets

WebTables [4] is one of the largest table corpora that is publicly available for table retrieval, with approximately 14.1 billion HTML tables crawled from the English text documents in the main index of Google. Most state-of-the-art models use keyword based queries [38; 29; 11]. For example, Zhang and Balog [38] evaluated their proposed model using *WikiTables* [1] as table corpus. WikiTables is a table dataset with about 1.6 million relational tables generated from 154 million Web tables. Zhang and Balog [38] also released a hand-crafted query dataset with 60 keyword queries, such as 'phases of the moon' and 'science discoveries', together with 3,120 annotated tables from WikiTables. This dataset was later used by many other researchers, including the current state-of-the-art model for table retrieval [31], published in 2020.

Shraga et al. [31] process the *Natural Questions (NQ)* corpus [20] and release a new table retrieval dataset, the *GNQtables* dataset, which contains 789 natural language questions over 74,224 tables. The NQ corpus is designed for general QA tasks, with more than 323,000 examples of real queries from the Google search engine. The answers to the questions in NQ are generated from Wikipedia articles. Except from the GNQtables dataset, a new derivative dataset extending NQ dataset has been contributed by Herzig et al. [13], which identifies 12,000 natural language questions with answers in tables.

Other published datasets for table retrieval include WebQueryTables [35], TableArXiv [11], Web Data Commons (WDC) table corpus [21], etc.

[2] All resources are available at `https://github.com/IBM/AITQA`. Datasheet and Neurips checklist are available as supplementary material.

Dataset	**Year**	**Table only**	**Wikipedia**	**Hierarchical Column Headers**	**Hierarchical Row Headers**
WikiTableQuestions [28]	2015	✓	✓	✗	✗
TabMCQ [17]	2016	✓	✗ (Science Exam)	✗	✗
WikiSQL [40]	2017	✓	✓	✗	✗
FeTaQA [24]	2021	✓	✓	✗	✗
HybridQA [7]	2020	✗	✓	✗	✗
OTT-QA [6]	2021	✗	✓	✗	✗
TAT-QA [41]	2021	✗	✗ (Finance)	✗	✗
AIT-QA (this work)	2021	✓	✗ (Airlines)	✓	✓

Table 1: Comparison of AIT-QA to other Table QA datasets

2.2 Table QA Datasets

In Table QA studies (which are the most relevant to our work), the most commonly used datasets include WikiTableQuestions, WikiSQL, and TabMCQ. *WikiTableQuestions* [28] is a dataset containing 22,033 question-table pairs over 2,108 Wikipedia tables. *WikiSQL* [40] is also a Wikipedia-based dataset with 80,654 hand-annotated natural language questions over a corpus of 24,241 Wikipedia tables. On the other hand, *TabMCQ* [17] does not build on Wikipedia, but instead contains 500 multiple choice questions over 70 manually-curated general knowledge tables created from the Regents 4th-grade exam. While this dataset is domain-specific, the included tables have a very peculiar structure, with table rows containing entire natural language sentences that have been appropriately split into columns. In our experience, the resulting tables capture a very special case and are not representative of tables appearing in most domains. More recently, Nan et al. [24] proposed FeTaQA, a new free-form Table QA dataset. Compared to prior datasets, the main novelty of FeTaQA lies in the structure of the answers, which are long free-form sentences (in contrast to the very short answers found in prior datasets). FeTaQA is also build on Wikipedia, containing 10,330 unique Wikipedia-based instances covering 16 different topics.

Finally, the last couple of years saw the introduction of three multi-hop QA datasets: *HybridQA* [7], *OTT-QA* [6], and *TAT-QA* [41]. These datasets are designed to accommodate a special scenario, where finding answers to natural language questions requires reasoning not only on tables but across both tables and associated text. Out of them, HybridQA and OTT-QA are both based on Wikipedia. On the other hand, TAT-QA, is based on data extracted from financial reports, making it the most similar dataset to our proposed AIT-QA.

However, while the TAT-QA paper mentions the existence of complex table structures (including row headers) in financial tables, the resulting dataset does not include explicit annotations of row and column headers (not to mention hierarchies thereof). Without explicit annotations of such headers, not only is it hard to understand the complexity of the included tables (for instance the results of a manual table inspection included in the Appendix of [41] points to the absence of column header hierarchies in TAT-QA), but it also makes it harder to understand the effect of table structure complexity on the performance of TableQA algorithms. Instead, our proposed AIT-QA treats hierarchical column and row headers as first-class citizens and is to the best of our knowledge the first Table QA dataset that contains explicit annotations of such complex table structures. This provides three advantages: First, it is the first dataset with guaranteed coverage of such complex structures. Second, it enables a principled analysis of the effect of complex table structures on Table QA algorithms. Finally, it opens up the path of a principled separation of the column/header identification task from the table QA task, thus leveraging prior work on table header identification [25; 10; 8; 18].

Table 1 summarizes the aforementioned Table QA datasets. As discussed above, apart from our work, all existing TableQA datasets with the exception of TabMCQ and TAT-QA are based on Wikipedia tables. Moreover, only our work contains and explicitly encodes hierarchical row and column headers. While TAT-QA may contain a subset of such complex structures, they are not encoded as such. Moreover, all other datasets do not consider row headers at all and consider only flat column headers (i.e., cases where the single top row of the table contains column headers).

3 Dataset

We next explain the process we followed to generate our dataset, starting from data acquisition and data preparation and continuing to question annotation and table header identification.

Data Acquisition. AIT-QA is based on 10-K forms; comprehensive annual reports that publicly traded companies file with the U.S. Securities and Exchange Commission (SEC). For this dataset, we focused on the airline industry and retrieved recent 10-K forms of all 5 airlines included in the Standard & Poor's 500 (S&P 500) stock market index (Wikipedia). The covered airlines include (stock ticker symbols shown in parenthesis): Alaska Air Group (ALK), American Airlines Group (AAL), Delta Air Lines Inc. (DAL), Southwest Airlines (LUV), and United Airlines Holdings (UAL). The 10-K forms were downloaded through the SEC EDGAR online system (U.S. Securities and Exchange Commission) in HTML form. The dataset files are available at `https://github.com/IBM/AITQA`.

Data Preparation and Cleaning. While the downloaded 10-K forms encode tables using standard HTML tags, the tabular structures are designed with human consumption in mind. As such, table rows/columns/cells are used to allow for the table to be neatly rendered on the screen and/or paper and they do not always correspond to the table's logical structure. In particular, we found that tables in the downloaded 10-K forms contain extraneous rows/columns (introduced to allow for more space between table elements). Moreover, the contents of a single logical cell are often split into multiple physical cells, to allow for better vertical alignment of the information within a table. For instance, cells containing a currency symbol and negative monetary amounts such as $\$(1,234)$, are often split into three physical cells $\boxed{\$}\boxed{(1,234}\boxed{)}$ so that the currency symbols and numbers align with other similar contents across table rows. To separate these formatting decisions from the logical structure of the table, we post-processed the downloaded HTML files to remove extraneous rows and columns and merge back together components of logical cells that were split into multiple cells. Processing was done through a combination of scripts and manual error correction.

Question Annotation. The cleaned 10-K forms were given to 8 co-authors of this paper to generate question-answer pairs over tables appearing on the forms. To capture questions of particular interest to domain experts in the domain, while ensuring a diversity of question topics, we asked annotators to provide two types of questions:

- *KPI-driven questions:* These are questions that inquiry about Key Performance Indicators (KPIs), which are metrics of particular interest to analysts in the airline industry. While creating these questions, annotators were provided with a list of KPIs along with common synonyms to ensure that the questions capture not only the topic of interest but also use the respective vocabulary. To generate these questions, annotators were instructed to search the document for mentions of KPIs appearing within tables and create corresponding questions. Thirteen KPIs were used in total, with each of them having three variants, depending on whether it referred to the airlines' mainline operations, its regional operations, or the combination thereof.
- *Table-driven questions:* While KPI-driven questions capture the common metrics inquired by analysts, they can be limiting for two main reasons: First, there is a limited number of KPIs and second, given their importance in the domain, they often appear within a small set of tables. As a result, limiting ourselves only to such questions would lead to a non-diverse dataset. To avoid this issue, annotators were asked to also provide questions that inquired about other concepts appearing within the input tables. To create such questions, annotators had to browse through the tables in the input documents and write questions that could be answered by them.

After an initial set of question-answer pairs was collected, annotators were also asked to generate paraphrases. While creating the paraphrased questions, annotators were given access to the set of question-answer pairs collected in earlier stages and asked to pick a subset of questions to paraphrase. This leads to the second major dimension along which questions in our dataset can be classified:

- *Original questions:* Questions collected during the initial stages of the annotation.
- *Paraphrased questions:* Questions generated as paraphases of original questions.

Finally, in all stages of the annotation process, annotators were also asked to keep track of additional metadata indicating whether a question relied on the hierarchy of row headers to be answered. A

Question Type	Count (%)
KPI-driven questions	145 (28%)
Table-driven questions	370 (72%)
Original questions	441 (86%)
Paraphrased questions	76 (15%)
Row header hierarchy questions	146 (28%)
No row header hierarchy questions	369 (72%)

(a) Breakdown of questions across 3 dimensions

Property	Value
Documents	13
Companies	5
Date range	2017-19
Questions	515
Tables	116

(b) Other dataset properties

Table 2: Dataset Statistics

question relies on the hierarchy of row headers when in order to be unambiguously answered, one has to not only see the row header that appears on the same row as the answer, but also on the higher levels of the hierarchy. For instance, the question in Figure 1 depends on the row header hierarchy, as ignoring the hierarchy may lead to an incorrect answer, as explained in the introduction. Based on these metadata, questions in the dataset can be differentiated across a third dimension into:

- *Row header hierarchy questions:* Questions whose answer relies on the row header hierarchy.
- *No row header hierarchy questions:* Questions whose answer does not rely on the row header hierarchy.

For each question-answer pair, annotators provided the question, the table cell where the answer appears, as well as metadata indicating the classification of the question along the three aforementioned dimensions. For the first version of the dataset, we focus on *lookup* questions - i.e., questions where the answer appears within table cells and does not require aggregate operations (such as min/max/sum/count) to be returned [12], leaving the expansion of the dataset with aggregate questions as future work. Annotation was carried out using a custom-built Table QA annotation tool (screenshot in Supplementary material). Finally, the collected question-answer pairs and associated metadata were subsequently reviewed by other annotators to verify their validity and correct minor issues, such as typos or associated metadata.

Hierarchical Column/Row Header Identification. To identify column and row headers of tables, we leveraged Table Understanding technology incorporated in IBM Watson Discovery [15]. Table Understanding allows among others identifying for each body (i.e., non-header cell), the set of column headers and row headers that describe the cell [16]. Table Understanding allows identifying both column and row header hierarchies, as described above. The identified header hierarchies are included as part of the dataset so that they can be leveraged by Table QA models.

Dataset Statistics. Statistics of the resulting dataset are shown in Table 2. Table 2a shows the breakdown of questions along the three aforementioned dimensions, while Table 2b lists other properties of the dataset, including number of 10-K forms, companies, and tables.

4 Experimental Evaluation

To analyze the effect of AIT-QA's domain-specific complex tables to existing Table QA approaches, we next provide a comprehensive evaluation of state-of-the-art Table QA models when applied on it.

4.1 Baseline methods

We evaluate three Table QA systems - **RCI** [12], **TaBERT** [36], and **TaPaS** [14] - selected as being representative of most of the existing Table QA approaches.

TaBERT is a table and question encoder specifically designed for Table QA. The encoder is used to predict a logical form in an encoder-decoder approach [1; 23; 22; 19]. This logical form is then executed over the table and provides the final answer to the original question. TaBERT employs a BERT [9] encoder and an LSTM decoder (which generates the logical forms [22]) and is trained using reinforcement learning [23]. A content snapshot heuristic is used to handle tables that are too large for the BERT encoder.

On the other hand, both TaPaS [14] and RCI [12] treat Table QA as classification problem and skip generating logical forms. In the case of TaPaS, during the pre-training phase, it jointly learns representations for natural language sentences and structured tables, which makes it is suitable for table question answering. Then, during the fine tuning phase, it follows a two step procedure, where it first selects relevant cell/cells and then optionally applies additional operators to the selected cells. RCI splits tables into rows and columns and carries out inference on them separately. To this end, it employs a row predictor identifying the row containing the answer and a column predictor identifying the corresponding column. Two separate BERT models are used for generating column/row representations and query representations. In all three systems, tables and questions are encoded using transformers [33].

4.2 Experimental Setup

We attempt to test whether high Table QA performance reported on open domain tables translates to AIT-QA dataset. All three Table QA models are pre-trained on the larger WikiSQL [40] train split and tested on AIT-QA without any hyper-parameter tuning. We use the original source code released by the respective authors, pretrained weights along with details in their papers, for setting up all baseline models. In particular, for TaBERT [36], we use the pre-trained BERT released on the official GitHub repository[3] with semantic parser MAPO[4]. TaBERT was trained for 10 epochs on 4 Nvidia Tesla v100s with batchsize of 10, number of explore samples as 10 and all other hyperparameters kept exactly the same as [36]. For RCI [12], we use the code released with the paper[5] to train the model for 2 epochs on 2 Tesla v100s, with learning rate 2.5e-5 and batch size 128. For TaPaS [14], we use the model[6] trained on the WikiSQL dataset from the official GitHub repository[7]. On WikiSQL dev set TaBERT gives an accuracy of 70.5%, TaPas 89.2%, and RCI 89.8%.

4.3 Transforming Table Structures

Existing table QA models are based on open domain web tables. So they assume that the input tables contain flat column headers (i.e., a single row of column headers) and no row headers. Therefore, none of the existing baselines are built for handling complex column or row header hierarchies seen in AIT-QA. So, we experiment with transformation operations on the table to maximize these baselines' performance on AIT-QA.

Base transformations are first performed on AIT-QA tables to render the tables compatible to the models as follows:

- *Row headers as Table cells in a new column:* Row headers are added as the first column of the table as regular body cells. We use a dummy text *header* as the column header for this new column.
- *Header hierarchies as flat headers*: Header hierarchies are flattened by concatenating parent header text with children text.

Note that these base transformations are designed to help the baseline models perform better than if we ran them on the raw table. For instance, when converting the table of Figure 1, the cell on the left of the blue cell will contain the concatenated row header hierarchy (i.e., 'Mainline passenger revenue miles (millions)'). This should help the models (which are not built to recognize row header hierarchies) perform better on AIT-QA.

Transposing tables. However, after running the models, we observed that there was further room for improvement. In particular, we observed that in many tables in AIT-QA, row headers contain more information than column headers. For example, in Figure 1, row headers contain the metric names,

[3] `https://github.com/facebookresearch/TaBERT`

[4] Source code available at `https://github.com/pcyin/pytorch_neural_symbolic_machines`

[5] `https://github.com/IBM/row-column-intersection/`

[6] `https://storage.googleapis.com/tapas_models/2020_08_05/tapas_wikisql_sqa_masklm_large_reset.zip`

[7] `https://github.com/google-research/tapas`

Version	TaBERT	TaPaS	RCI
Base	33.20	49.32	40.58
All T	33.39	43.88	48.54
Partial T	33.98	46.80	51.84

(a) Accuracy of Table QA on different transformations of the tables in AIT-QA (**Base** = No transpose, **All T** = All transpose, **Partial T** = Partial Transpose).

Data subset	TaBERT	TaPaS	RCI
Overall accuracy	33.98	49.32	51.84
KPI-driven	41.37	48.26	60.00
Table-driven	31.08	50.0	48.64
Row header hierarchy	21.92	47.26	45.89
No row header hierarchy	38.75	50.39	54.20

(b) Accuracy of Table QA models on slices of AIT-QA

Table 3: Accuracy of Table QA models on AIT-QA

which are much more descriptive than the column headers containing just the year information. Based on this intuition, we experimented with transposing the headers, so that row headers become column headers (which the models are trained to pay more attention to) and vice versa (and body cells are appropriately transposed as well). This led to three versions of AIT-QA data: (1) *Base:* without transposing tables, (2) *All transpose:* With all tables transposed, and (3) *Partial transpose:* Transposing tables that have more characters in row headers than column headers. Table 3a depicts the accuracy of baseline models on each dataset version. Interestingly, RCI and TaBERT benefit from transposing tables, while the performance of TaPas declines. For our subsequent analysis, we pick for each model the version of the data that provides the highest performance for it.

4.4 Analyzing Baseline Performance on AIT-QA's Dimensions

To gain further insights on how domain vocabulary, table structure, and question phrasing affect the performance of Table QA models, we next evaluate them on each of the three dimensions of our dataset: (1) KPI-driven vs Table-driven, (2) Row header hierarchy vs No Row header Hierarchy and (3) Original vs Paraphrased questions. Table 3b shows the results for the the first two, while Table 4 provides the results for the third dimension. We next discuss each dimension in detail:

Overall Accuracy. Unlike existing datasets, such as WikiSQL and WikiTableQuestions with flat column headers and no row headers, AIT-QA with its rich row and column header hierarchies, poses an additional challenge for state-of-the-art Table QA approaches. RCI framework is designed to give attention to individual rows and columns to extract the right intersection. Therefore, as expected, it performs best on AIT-QA with 52% accuracy. TaPas relies on an attention model as well, but considers the entire table as a whole, including its rows and columns. Therefore, TaPas performs comparable to RCI and with 49% accuracy. On the other hand, TaBERT takes a two step approach and is designed to produce intermediate logical forms to capture the complex intents from the question. Unlike RCI or TaPas, TaBERT does not have an end-to-end differentiable model for identifying right row(s) and column(s); therefore it performs worst on AIT-QA with 34% accuracy.

This relative performance trend, as witnessed in overall accuracy, holds also true when evaluating baselines on slices of the dataset along various dimensions. RCI and TaPas exhibit the best performance, while TaBERT trails further behind.

KPI-driven vs Table-driven. Table 3b shows the performance of the baselines on KPI-driven vs Table-driven questions. As shown, accuracy is always higher for the former than the latter. The result can be correlated with the definitions of KPI-driven and Table-driven questions as mentioned in section 3 and indicates two key observations:

(1) There is a limited number of Airline KPIs, which most frequently exist as is in 10K documents submitted by an airline company. Therefore, even if a KPI name is an airline industry-specific term, the uniqueness of it helps each of the baselines to correctly identify the right answer.

(2) KPI-driven questions are formed by having a KPI in mind and searching for the corresponding term in the document. As a result, KPI-driven questions may be more natural than table-driven questions, which were formed by looking at a table and trying to form a question. As a result, it is much more common to find distorted utterances of row/column headers and/or cell values in table-driven questions, making it harder for the baseline to identify the correct answer from the question utterance. This observation has been also made in previous table-driven question annotation

Paraphrase	TaBERT	TaPaS	RCI
All correct	25.00	38.88	33.33
Any correct	29.17	30.55	34.72
All wrong	45.83	30.55	31.94

Table 4: Percentage of paraphrased question sets that are (a) all correctly answered, (b) at least one correctly and another one incorrectly answered, and (c) all incorrectly answered by each baseline

datasets, such as WikiTableQuestions. By containing both types of questions, AIT-QA combines the more natural KPI-driven questions with the more challenging Table-driven questions.

For the above reasons, KPI-driven questions tend to have better results than Table-driven questions. The gap is highest in RCI at 12%, which is expected, as RCI can uniquely identify individual rows/columns using the KPI name.

Row Header Hierarchy vs No Row Header Hierarchy. One of the key challenges associated with AIT-QA are row/column header hierarchies. While we tried to help the baselines (which have not built with complex header structures in mind) deal with hierarchies through the table transformations described in Section 4.3, this implicit treatment of headers has two important limitations: (1) the explicit hierarchical information is lost and (2) in some cases, transformations may add noise into a row/column. Therefore, it is not surprising that questions that depend on row header hierarchies negatively affect the performance of all baselines and cause an average drop of 13%.

Paraphrase Handling. Paraphrasing is an important aspect of AIT-QA, because it allows us to see the effect of domain shift in the natural language understanding capability of Table QA systems. AIT-QA contains 76 paraphrased questions which can be grouped into 72 paraphrased question sets (i.e., sets that include the original questions and all its paraphrases). Table 4 shows the effect of paraphrasing by breaking down the predictions of the baselines on the paraphrased question sets into three categories:

1. **All Correct:** When all questions in the set are answered correctly.
2. **Any Correct:** When at least one question in the set is answered correctly and at least another question is answered incorrectly.
3. **All Wrong:** When all questions in the set are answered incorrectly.

The 2nd category (**Any correct**) is important to analyze from the point of view of a Table QA system's NLU capability. It essentially means that there exists a way of asking a question that leads to the right answer, implying that the Table QA system can handle such questions. Whereas, phrasing that same question in a different way leads to wrong results. This could be attributed to the domain shift. Since baselines are trained on the open-domain WikiSQL dataset, when tested on the domain-specific AIT-QA, they can only handle certain ways of phrasing the questions, with almost 30% of the questions being supported when phrased in one way but not in a different way. This points to an important observation for practical domain shift scenarios: Table QA systems become sensitive to question phrasing in a significant way (in almost 30% of cases).

5 Conclusion

Table QA systems form a special type of QA systems that leverage information within tables to provide answers to natural language questions. Recent transformer-based Table QA approaches mainly innovate at the decoder stage to ensure that the tabular format is understood and leveraged. As a result, they provide high performance on existing Wikipedia-based datasets with simple tables. In this work, we present the first Table QA dataset that explicitly captures tables with complex structure, including column and row header hierarchies. Note, that while our dataset focuses on financial documents of the airline industry, such tabular structures are common in many other scientific and business documents. Our benchmarking of state-of-the-art Table QA methods show the deficiency of end-to-end, weakly-supervised and row-column encoding methods. We hope to encourage the community to consider new Table QA approaches that can support such complexity, so that Table QA methods can more effectively support a wider range of scientific and business use cases.

References

[1] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Douglas C Downey. 2015. TabEL: Entity linking in web tables. In *The Semantic Web – ISWC 2015 - 14th International Semantic Web Conference, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 425–441. Springer Verlag.

[2] Doug Burdick, Marina Danilevsky, Alexandre V. Evfimievski, Yannis Katsis, and Nancy Xin Ru Wang. 2020. Table extraction and understanding for scientific and enterprise applications. *Proc. VLDB Endow.*, 13(12):3433–3436.

[3] Michael J Cafarella, Alon Halevy, and Nodira Khoussainova. 2009. Data integration for the relational web. *Proceedings of the VLDB Endowment*, 2(1):1090–1101.

[4] Michael J. Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. Webtables: Exploring the power of tables on the web. *Proc. VLDB Endow.*, 1(1):538–549.

[5] Mustafa Canim, Cristina Cornelio, Arun Iyengar, Ryan Musa, and Mariano Rodriguez-Muro. 2019. Schemaless queries over document tables with dependencies. *CoRR*.

[6] Wenhu Chen, Ming wei Chang, Eva Schlinger, William Wang, and William Cohen. 2021. Open question answering over tables and text. *Proceedings of ICLR 2021*.

[7] 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.

[8] Xilun Chen, Laura Chiticariu, Marina Danilevsky, Alexandre Evfimievski, and Prithviraj Sen. 2017. A rectangle mining method for understanding the semantics of financial tables. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 268–273.

[9] 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.

[10] Jing Fang, Prasenjit Mitra, Zhi Tang, and C. Lee Giles. 2012. Table header detection and classification. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, page 599–605. AAAI Press.

[11] Kyle Yingkai Gao and Jamie Callan. 2017. Scientific table search using keyword queries.

[12] 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.

[13] Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Martin Eisenschlos. 2021. Open domain question answering over tables via dense retrieval.

[14] 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.

[15] IBM. 2018. IBM Watson Discovery (Accessed: 2021-05-18). `https://www.ibm.com/cloud/watson-discovery`.

[16] IBM. 2018. IBM Watson Discovery: Understanding tables (Accessed: 2021-05-18). `https://cloud.ibm.com/docs/discovery-data?topic=discovery-data-understanding_tables`.

[17] Sujay Kumar Jauhar, Peter Turney, and Eduard Hovy. 2016. Tabmcq: A dataset of general knowledge tables and multiple-choice questions.

[18] Elvis Koci, Maik Thiele, Oscar Romero, and Wolfgang Lehner. 2016. Cell classification for layout recognition in spreadsheets. In *International Joint Conference on Knowledge Discovery, Knowledge Engineering, and Knowledge Management*, pages 78–100. Springer.

[19] Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526.

[20] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

[21] 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*, WWW '16 Companion, page 75–76, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

[22] Chen Liang, Jonathan Berant, Quoc Le, Kenneth D Forbus, and Ni Lao. 2017. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 23–33.

[23] Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis and semantic parsing. In *NeurIPS*.

[24] Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryscinski, Nick Schoelkopf, Riley Kong, Xiangru Tang, Murori Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, and Dragomir R. Radev. 2021. Fetaqa: Free-form table question answering. *CoRR*, abs/2104.00369.

[25] Kyosuke Nishida, Kugatsu Sadamitsu, Ryuichiro Higashinaka, and Yoshihiro Matsuo. 2017. Understanding the semantic structures of tables with a hybrid deep neural network architecture. *AAAI Conference on Artificial Intelligence*.

[26] Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of ACL-IJCNLP*.

[27] Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables.

[28] 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.

[29] Rakesh Pimplikar and Sunita Sarawagi. 2012. Answering table queries on the web using column keywords. *Proceedings of the VLDB Endowment*, 5(10):908–919.

[30] Roee Shraga, Haggai Roitman, Guy Feigenblat, and Mustafa Canim. 2020. Ad hoc table retrieval using intrinsic and extrinsic similarities. In *Proceedings of The Web Conference 2020*, pages 2479–2485.

[31] Roee Shraga, Haggai Roitman, Guy Feigenblat, and Mustafa Cannim. 2020. Web table retrieval using multimodal deep learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1399–1408, New York, NY, USA. Association for Computing Machinery.

[32] U.S. Securities and Exchange Commission. 2018. EDGAR (Accessed: 2021-05-18). https://www.sec.gov/edgar.shtml.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008.

[34] Wikipedia. 2018. List of S&P 500 companies (Accessed: 2021-05-18). https://en.wikipedia.org/wiki/List_of_S%26P_500_companies.

[35] Zhao Yan, Duyu Tang, Nan Duan, Junwei Bao, Yuanhua Lv, Ming Zhou, and Zhoujun Li. 2017. Content-based table retrieval for web queries.

[36] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pre-training 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.

[37] Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of the WWW'2018*.

[38] Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of the 2018 World Wide Web Conference*, pages 1553–1562.

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

[40] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

[41] 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.