Multi-Row, Multi-Span Distant Supervision For Table+Text Question Answering

Vishwajeet Kumar[*]
IBM Research
India

Yash Gupta[*]
IIT Bombay
India

Saneem Chemmengath
IBM Research†
India

Jaydeep Sen
IBM Research
India

Soumen Chakrabarti
IIT Bombay
India

Samarth Bharadwaj
IBM Research†
India

Feifei Pan
IBM Research†
US

Abstract

Question answering (QA) over tables and linked text, also called TextTableQA, has witnessed significant research in recent years, as tables are often found embedded in documents along with related text. HybridQA and OTT-QA are the two best-known TextTableQA datasets, with questions that are best answered by combining information from both table cells and linked text passages. A common challenge in both datasets, and TextTableQA in general, is that the training instances include just the question and answer, where the gold answer may match not only multiple table cells across table rows but also multiple text spans within the scope of a table row and its associated text. This leads to a noisy multi-instance training regime. We present MITQA, a transformer-based TextTableQA system that is explicitly designed to cope with distant supervision along both these axes, through a multi-instance loss objective, together with careful curriculum design. Our experiments show that the proposed multi-instance distant supervision approach helps MITQA get sate-of-the-art results beating the existing baselines for both HybridQA and OTT-QA, putting MITQA at the top of HybridQA leaderboard with best EM and F1 scores on a held out test set.

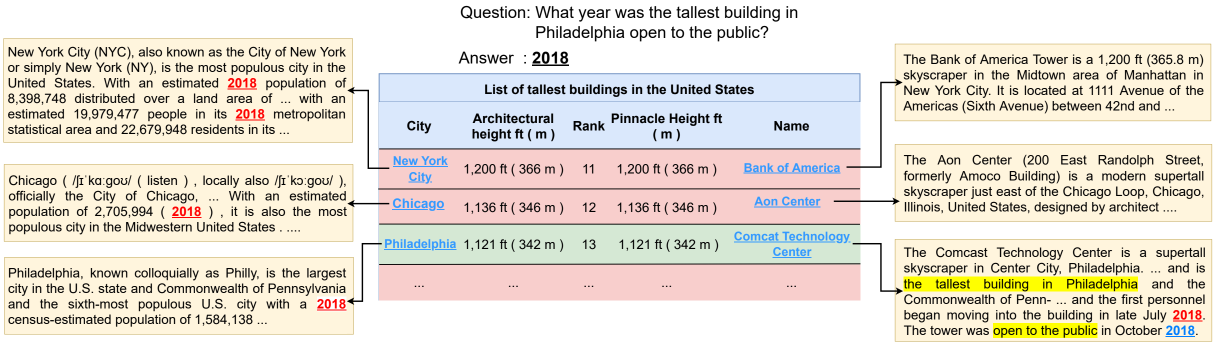
1 Introduction

Transformer-based question answering (QA) methods have evolved rapidly in recent years to handle open-domain, multi-hop reasoning over retrieved context paragraphs. Many existing QA datasets and benchmarks measure performance over homogeneous data sources, such as text (Rajpurkar et al., 2016; Chen et al., 2017a; Joshi et al., 2017; Dua et al., 2019) and more recently tables (Pasupat and Liang, 2015; Zhong et al., 2017; Liang et al., 2017; Herzig et al., 2020; Yin et al., 2020). Even though real-world documents often contain tables embedded in free form text, QA over such a hybrid corpus, i.e., a combination of tables and text — a.k.a. **TextTableQA** — remains relatively unexplored. As illustrated in Figure 1, even a relatively simple table from Wikipedia often references several entities, definitions or descriptions from the table elements. A question may be best answered by matching some parts of it to table elements and other parts to linked text spans. Existing Transformer-based QA solutions need significant modifications to score such heterogeneous corpus units. A key challenge is to reduce the cognitive burden of supervision to (question, answer) pairs, without humans having to identify the specific table cell or text span where the answer was mentioned. In TextTableQA, such 'distant' supervision is particularly challenging because it occurs along two distinct axes: (1) There could be multiple rows and associated passages that mention the answer string; and (2) Even for a specific table row with linked passages, the same candidate answer may occur in multiple text spans. Many of them may be spurious and detrimental to system training.

In response, we present **MITQA** — a TextTableQA system specifically engineered to address the above challenges. MITQA defines each table row, together with linked passages, as the fundamental *retrieval unit*. To adapt to memory-hungry Transformer networks, constrained by the number of input tokens they can efficiently process, MITQA uses a novel query-informed passage filter to prepare a contextual representation of each retrieval unit. MITQA then uses an early interaction (cross attention) Transformer network to score retrieval units. While training MITQA, its most salient features are multi-instance loss functions and data engineering curricula to tackle distant supervision, along both the multi-row and multi-span axes. Many of the above challenges are not faced by homogeneous text-only or table-

[*] Equal Contribution
[†] Work done while at IBM Research

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 8080–8094
July 9-14, 2023 ©2023 Association for Computational Linguistics



Question: What year was the tallest building in Philadelphia open to the public?

Answer : **2018**

New York City (NYC), also known as the City of New York or simply New York (NY), is the most populous city in the United States. With an estimated **2018** population of 8,398,748 distributed over a land area of ... with an estimated 19,979,477 people in its **2018** metropolitan statistical area and 22,679,948 residents in its ...

Chicago (/ʃɪˈkɑːgoʊ/ (listen) , locally also /ʃɪˈkɔːgoʊ/), officially the City of Chicago, ... With an estimated population of 2,705,994 (**2018**) , it is also the most populous city in the Midwestern United States . ....

Philadelphia, known colloquially as Philly, is the largest city in the U.S. state and Commonwealth of Pennsylvania and the sixth-most populous U.S. city with a **2018** census-estimated population of 1,584,138 ...

List of tallest buildings in the United States

City	Architectural height ft (m)	Rank	Pinnacle Height ft (m)	Name
New York City	1,200 ft (366 m)	11	1,200 ft (366 m)	Bank of America
Chicago	1,136 ft (346 m)	12	1,136 ft (346 m)	Aon Center
Philadelphia	1,121 ft (342 m)	13	1,121 ft (342 m)	Comcat Technology Center
...	...	...	...	...

The Bank of America Tower is a 1,200 ft (365.8 m) skyscraper in the Midtown area of Manhattan in New York City. It is located at 1111 Avenue of the Americas (Sixth Avenue) between 42nd and ...

The Aon Center (200 East Randolph Street, formerly Amoco Building) is a modern supertall skyscraper just east of the Chicago Loop, Chicago, Illinois, United States, designed by architect ....

The Comcast Technology Center is a supertall skyscraper in Center City, Philadelphia. ... and is the tallest building in Philadelphia and the Commonwealth of Penn- ... and the first personnel began moving into the building in late July **2018**. The tower was open to the public in October **2018**.

Figure 1: An instance of question answering over hybrid context of table and text (from HybridQA). Gold answer in correct context is highlighted in blue and gold answer appearing in irrelevant context is highlighted in red. The context used to arrive at the answer in the correct passage is shaded in yellow. The relevant row to be retrieved is shaded green and irrelevant rows are shaded red.

only QA systems. We report results on extensive experiments on two recent TextTableQA challenge data sets, HybridQA and OTT-QA, where our system outperforms baselines and is currently at the top of HybridQA[1] leaderboard. Source code is available at https://github.com/primeqa/primeqa.

2 Related Work

TableQA has gained much popularity in recent years, resulting in diverse approaches including semantic parsing-based (Pasupat and Liang, 2015; Zhong et al., 2017; Liang et al., 2017; Krishnamurthy et al., 2017; Dasigi et al., 2019) and more recently BERT-based (Devlin et al., 2018) systems for table encoding by, inter alia, Herzig et al. (2020); Yin et al. (2020); Glass et al. (2021a). A more realistic application scenario is "TextTableQA" where tables are often embedded in documents and a natural language query needs to combine information from a table as well as its correlated textual context to find an answer.

HybridQA (Chen et al., 2020) pioneered a TextTableQA benchmark, with Wikipedia tables linked to relevant free-form text passages (e.g., Wikipedia entity definition pages). They curated questions which need information from both tables and text to answer correctly. They also proposed HYBRIDER as the first system in TextTableQA with an F1 score of 50%, leaving much scope for improvement. The OTT-QA (Chen et al., 2021) benchmark extended HybridQA to an open domain setting where a system needs to retrieve a relevant set of tables and passages first before trying to answer a question. Moreover, the links from table and passage are not provided explicitly. To our knowledge, no existing TextTableQA system (Chen et al., 2020, 2021; Zhong et al., 2022) attempts to handle the challenge of multiple candidate instances arising from distant supervision during system training, owing to multiple matching table rows and multiple matching spans within a row and its linked text. Our experiments with HybridQA and OTT-QA show that superior handling of multi-instance matches by MITQA improves QA accuracy.

3 Preliminaries

3.1 Notation

T denotes a set of tables, each table being denoted as t. Title, caption, and other available metadata of table t is accessed as $t.\mathsf{meta}$. Table t has $t.\mathsf{rows}$ rows and $t.\mathsf{cols}$ columns. Its column headers are denoted $t.\mathsf{hdr}$. (Row headers may also assume a similar salient role, but we limit notation to column headers for simplicity of exposition.) $[N]$ denotes the set of indices $\{1, \ldots, N\}$. For $r \in [t.\mathsf{rows}]$, the rth row is denoted $t[r, \star]$. For $c \in [t.\mathsf{cols}]$, the cell at position (r, c) is written as $t[r, c]$. The cth column header cell is denoted $t.\mathsf{hdr}[c]$. The set of passages linked with the row r of table t is denoted by $t[r, \star].\mathsf{psg}$. A passage p is a sequence of tokens. The set of all token spans in p is denoted by $\mathsf{spans}(p)$. One token span is denoted $\sigma \in \mathsf{spans}(p)$. A set of such spans is denoted Σ.

3.2 Task Definition

Given a question q (modeled as a sequence of tokens) and a table t together with linked text, the task is to find a relevant row r, and then an answer text a, which can be a cell from $t[r, \star]$, or a span

[1] https://competitions.codalab.org/competitions/24420#results

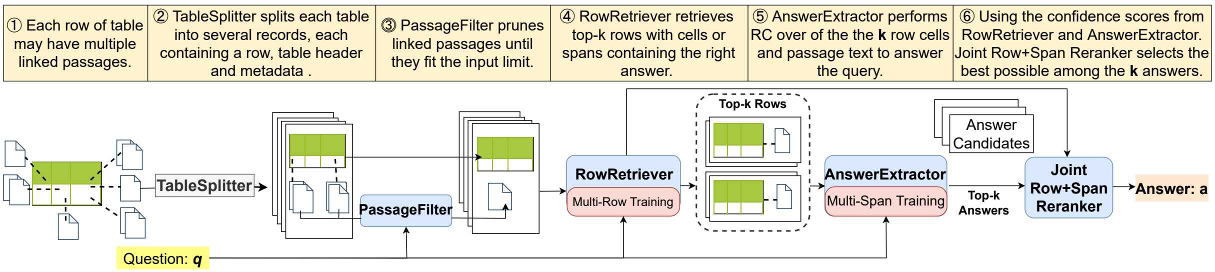

① Each row of table may have multiple linked passages.
② TableSplitter splits each table into several records, each containing a row, table header and metadata .
③ PassageFilter prunes linked passages until they fit the input limit.
④ RowRetriever retrieves top-k rows with cells or spans containing the right answer.
⑤ AnswerExtractor performs RC over of the the **k** row cells and passage text to answer the query.
⑥ Using the confidence scores from RowRetriever and AnswerExtractor. Joint Row+Span Reranker selects the best possible among the **k** answers.
TableSplitter
PassageFilter
RowRetriever
Multi-Row Training
Top-k Rows
AnswerExtractor
Multi-Span Training
Answer Candidates
Top-k Answers
Joint Row+Span Reranker
Answer: a
Question: q

Figure 2: MITQA system sketch. TableRetriever and RowPassageLinker are not shown.

from $\mathsf{spans}(t[r,\star].\mathsf{psg})$. In HybridQA, the table t and associated linked passages are provided along with the question q. In contrast, for OTT-QA, the correct table t and linked passages need to retrieved from a corpus of tables and initially unconnected passages—a more challenging setting.

3.3 System Overview

Figure 2 shows the overall architecture of MITQA. In some workloads (e.g., HybridQA), a question comes already associated with a table and its linked text. In other "open domain" workloads (e.g., OTT-QA), tables and linked passages must be retrieved from a large corpus by a **TableRetriever**. The **TableSplitter** segments the table t into *retrieval units*, each comprising one row r (i.e., all cells in $t[r,\star]$) and its linked passages $t[r,\star].\mathsf{psg}$. For data sets (like OTTQA) which are not provided pre-linked, the **RowPassageLinker** module links spans in table cells to corpus passages to prepare the retrieval units. To score retrieval units, we will use an early interaction (cross-attention) Transformer network, to which we will feed the question and a retrieval unit, suitably encoded into text. Rather than naive truncation, or expensive hierarchical encodings, we use a question-sensitive **PassageFilter** to select a subset of passages $\mathrm{PassageFilter}(t,r,q) \subseteq t[r,\star].\mathsf{psg}$ to retain with each candidate row. The **RowRetriever** can then identify the most relevant retrieval units. Next, an **AnswerExtractor** module selects the answer span as a cell from $t[r,\star]$ or as a token span from a passage $p \in t[r,\star].\mathsf{psg}$ linked to the row $t[r,\star]$.

Distant supervision (as described above), and the consequent need for multi-instance learning, are handled by three modules: RowRetriever, AnswerExtractor, and a final **RowSpanReranker**. RowRetriever employs a special loss function that can handle spurious matches of the gold answer in multiple rows and associated retrieval units (Dietterich et al., 1997; Andrews et al., 2003). AnswerExtractor employs a data programming (Ratner et al., 2016) curriculum to a similar end. The final Reranker module refines the score for each answer candidate, based on a learned weighted combination of RowRetriever and AnswerExtractor confidence scores. We describe the most important components of MITQA in Section 4 and defer the rest to Appendix A.

4 MITQA System Architecture

In this section we first describe the modules shown in Figure 2, that are shared for closed-domain (table and linked text provided, as in HybridQA) and open-domain (OTT-QA) applications. After that we describe TableRetriever and RowPassageLinker that are needed for open-domain scenarios.

4.1 PassageFilter

The total tokens in passages linked to a row can be large, exceeding the input capacity of BERT-like models. Efforts (Beltagy et al., 2020; Zaheer et al., 2020) have recently been made to remove these capacity limits, but at the cost of additional complexity, unsuited for our fine-grained application to table rows. In any case, the query has a critical role in determining the utility of each passage linked to a row. Our PassageFilter module orders the linked passages such that the prefix that fits within the input capacity of a BERT-like model is likely to be the most valuable for judging the relevance of a row. More details are in Appendix A.3.

4.2 RowRetriever

Given question q and table t, the task of RowRetriever is to identify the correct row r from which the answer can be obtained, either as a cell $t[r,c]$ from the cth column, or a span from a passage in $t[r,\star].\mathsf{psg}$. We implement RowRetriever by training a BERT-based sequence classification model (Devlin et al., 2018) on a binary classification task with correct rows to be labelled as 1s and the rest as 0s. Suppose the columns of t are indexed left-to-right using index c. Then $t.\mathsf{hdr}[c]$ and $t[r,c]$ are

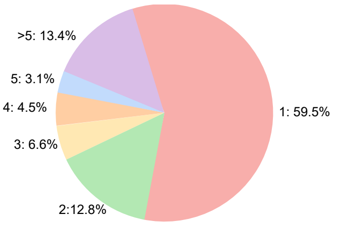


Figure 3: Distribution of number of rows containing the answer-text in the training set of HybridQA. “2: 12.8%” in the chart means that 12.8% instances of training set has exactly 2 rows with answer-text appearing in them.

the header and cell in column c. The input $\boldsymbol{x}$ to the BERT encoder is fashioned as:

$$\texttt{[CLS]}\ q\ \texttt{[SEP]} \mathop{\Big\|}_{c\in[t.\mathsf{cols}]} t.\mathsf{hdr}[c]\ \mathsf{is}\ t[r,c]\ \texttt{[DOT]}$$

$$\texttt{[SEP]}\ t.\mathsf{meta}\ \texttt{[DOT]} \mathop{\Big\|}_{p\in\text{PassageFilter}(t,r,q)} p\ \texttt{[DOT]} \qquad (1)$$

where ‘$\|$’ is the concatenation operator and ‘is’ is literally the word ‘is’. `[DOT]` and `[SEP]` are separator tokens. In words, we concatenate: (1) the question q; (2) phrases of the form “header is cell–value”, over all columns; (3) table metadata (title etc.); and (4) passages linked to the given row, that survive through PassageFilter; before passing into a BERT-Large encoder in a specific format to get a suitable latent states. The `[CLS]` embedding output by BERT is sent to a feed-forward neural network to make the label prediction. During inference, all {question, row} pairs are passed through this sequence classifier. The row with the largest score for class 1 is identified as the chosen row.

Distant supervision of RowRetriever: A row retrieval system that expects supervision in the form of gold rows exacts a high cognitive burden from annotator in preparing training instances. In the case of HybridQA and OTT-QA, we only have final answer-text as supervision, not relevant row/s, cell/s or text span/s. Given a table with connected passages and a question, we identify potential gold rows by exact string matching answer-text on rows (cells and linked texts).

As depicted in Figure 3, for HybridQA, ∼40% of the training instances have the problem of multiple rows containing the correct answer text. In training set of HybridQA dataset, for some instances, the gold answer appears in *19 rows*!

Multi-instance (-row) training: A naive way is to label all matches with label 1 and the rest with label 0 for training. This reduces the performance of the *RowRetriever* as a large chunk of training data gets incorrect labels. To address the issue of multiple potentially correct rows we map this problem into a multiple-instance learning setup (Dietterich et al., 1997; Andrews et al., 2003), with question-row pairs as instances and potential correct rows for a question forming a *bag*. We are given a question q and table t, with row subset $B \subseteq t.\mathsf{rows}$ labeled 1 (relevant) and the rest, $t.\mathsf{rows} \setminus B$ labeled 0 (irrelevant). RowRetriever applied to the retrieval unit of row r is modeled as a function $f(\boldsymbol{x}_r)$, where $\boldsymbol{x}_r$ is the text constructed in Eqn. (1) from row r. Let $\ell(y_r, f(\boldsymbol{x}_r))$ be the binary cross-entropy classification loss, where $y_r \in \{0,1\}$ is the gold label of instance $\boldsymbol{x}_i$. For a given table and a question, we define the row retriever loss as

$$\min_{r\in B} \ell(1, f(\boldsymbol{x}_r)) + \sum_{r'\notin B} \ell(0, f(\boldsymbol{x}_{r'})). \qquad (2)$$

The intuition is that RowRetriever can avoid a loss if it assigns a large score to *any one* of the rows in B, whereas it must assign small score to *all* rows not in B. Apart from this multi-instance loss function, we also deployed a form of curriculum learning (Bengio et al., 2009). In early epochs, we only use instances whose labels we are most confident about: negative rows, and questions with only one positive row. In later epochs, we increase the fraction of instances with multiple relevant rows.

4.3 AnswerExtractor

In TextQA, answer extraction is solved by a reading comprehension (RC) module (Baradaran et al., 2020). An RC module is usually trained with the query, the passage, and the start and end token positions of the span in the passage where the gold answer is found. In MITQA, neither start and end index of the span is available (when the answer is

Algorithm 1 Multi-span AnswerExtractor training.

```
Input: training instances $D=\{(q,t,r_\oplus,\Sigma[r_\oplus])\}$
$D_1 \leftarrow \{(q,t,r_\oplus,\Sigma[r_\oplus])\in D : \big|\Sigma[r_\oplus]\big| = 1\}$
AE$_{\text{init}}$ $\leftarrow$ train AnswerExtractor on $D_1$
                                ▷ initial model based on ‘easy’ cases
$D_{>1} \leftarrow \{(q,t,r_\oplus,\Sigma[r_\oplus])\in D : \big|\Sigma[r_\oplus]\big| > 1\}$
$\widehat{D} \leftarrow \varnothing$                   ▷ collects ‘denoised’ instances
for $(q,t,r_\oplus,\Sigma[r_\oplus]) \in D_{>1}$ do
    $\sigma^* \leftarrow \text{argmax}_{\sigma\in\Sigma[r_\oplus]}$
      AnswerExtractor$_{\text{AE1}}(q, t[r_\oplus,\star].\mathsf{psg}, \sigma)$
    ▷ $\sigma^*$ is the best span among $\Sigma[r_\oplus]$ as per
    initial model AE$_{init}$
    $\widehat{D} \leftarrow \widehat{D} \cup (q,t,r_\oplus,\{\sigma^*\})$
AE$_{\text{final}}$ $\leftarrow$ train AnswerExtractor on $D_1 \cup \widehat{D}$
return AE$_{\text{final}}$                       ▷ refined model
```

a passage span), nor are the table cell coordinates (when the answer is in a table cell). Furthermore, high level supervision of whether the correct answer is a table cell or passage span, is also not available. This makes the training of AnswerExtractor a challenging task. We tackle this challenge using a multi-span training paradigm.

Multi-instance (-span) training: Recent systems (Devlin et al., 2018; Segal et al., 2020) simply consider the first span matching the gold answer text as the correct span and use that for training. *This is often an incorrect policy.* In Figure 1, the correct answer, '2018', occurs multiple times in $t[r_\oplus, \star]$.psg, where $r_\oplus$ is the relevant row. There is absolutely no guarantee that the first span in $t[r_\oplus, \star]$.psg matching the gold answer text will be true evidence for answering the question. Therefore, using the first, or all, matches for training AnswerExtractor can introduce large volumes of training noise and degrade its accuracy.

Let $\Sigma[r_\oplus]$ be the set of spans in $t[r_\oplus, \star]$.psg that match the gold answer. Our problem is when $\left|\Sigma[r_\oplus]\right|>1$. Inspired by data programming methods (Ratner et al., 2016), we propose a multi-span training (MST) paradigm for AnswerExtractor, shown in Algorithm 1. Assuming there is a sufficient number of single-match instances, we train an initial model AE1 on these. We then use this initial model AE1 to score spans from the noisy instances in $D_{>1}$. Note that this is different from end-task inference, because we are in a highly constrained output space — we know the answer can only be among the few choices. The best-scoring span σ^* should therefore give us a 'denoised' instance. These, combined with the earlier single-span instances, give us a much better training set on which we can train another answer extractor, leading to the final model AE2. Appendix A.4 has more details.

Algorithm 2 Joint row+span reranker training.

```
Input: Trained RowRetriever and AnswerExtractor; $K$: number of rows to retain; $K'$:
       number of spans to retain; search space of combining weights $\mathcal{W}$; development fold
       $D = \{(q, t, a)\}$
  for $w \in \mathcal{W}$ do                              ▷ grid search for weights $w$
      $\widehat{D} \leftarrow \varnothing$
      for $(q, t, a) \in D$ do
          $R = \{(r, s)\} \leftarrow$ top-$K$ rows from
          RowRetriever$(q, t, K)$ with scores
          for $(r, s) \in R$ do
              $\Sigma = \{(\sigma, s_{\text{st}}, s_{\text{en}})\} \leftarrow$
                    AnswerExtractor$(q, t, r, K')$
              $\vec{s} \leftarrow \begin{bmatrix} s & s_{\text{st}} & s_{\text{en}} \end{bmatrix}$
              score$(r, \sigma) \leftarrow w \cdot \vec{s}$          ▷ combo score
          $r_\oplus \leftarrow \text{argmax}_r\, \text{score}(r, \sigma)$
          $\widehat{D} \leftarrow \widehat{D} \cup \{(q, t, r_\oplus, a)\}$
      perf$(w) \leftarrow$ evaluate AnswerExtractor on $\widehat{D}$
  return $\text{argmax}_w\, \text{perf}(w)$
```

4.4 RowRetriever feedback (RF)

In Algorithm 1, note that a single row $r_\oplus$ is identified in each instance as relevant. As we have noted before, this is not directly available from training data, because the gold answer may match multiple rows, with no certificate that they are evidence rows. A trivial approach involves invoking Algorithm 1 on all rows containing the gold answer. As expected, this method produced a sub-optimal AnswerExtractor. Instead, we use the trained RowRetriever to identify the most probable row as $r_\oplus$.

4.5 Joint row+span reranker (RSR)

The final piece in MITQA combines the confidence scores of RowRetriever and AnswerExtractor. Despite the efforts outlined in the preceding sections, they are both imperfect. E.g., if we retain the top five rows from RowRetriever, gold row recall jumps 8–9% compared to using only the top one row. To recover from such situations, we retain the top five rows, along with their relevance scores. These rows are sent to AnswerExtractor, which outputs its own set of scores for candidate answer spans. The row+answer reranker implements a *joint selection* across RowRetriever and AnswerExtractor, through a linear combination of their scores, to select the best overall answer. The weights in the combination are set using a development fold. These weights can be selected using either grid search or gradient descent, after pinning module outputs. We do a grid search, shown as Algorithm 2. We shall see that such reranking leads to significant accuracy improvements.

4.6 Modules for open-domain applications

TableRetriever: For open-domain scenarios where questions are not accompanied by tables, this module retrieves the tables most relevant to a given question. For this task, we linearize the tables using different special delimiters to distinguish header information, cells and rows. we also prefix the table title in front of the linearized table with a separator. Then we train a dense passage retriever (DPR) (Karpukhin et al., 2020) to give a higher score for a table if it is relevant to the question while computing the dot product of the encoded table and question. Details about table linearization and DPR training are in Appendix A.1.

RowPassageLinker: This module iterates over each row of the tables retrieved by *TableRetriever* and links relevant passages to the row. For every cell in the row, RowPassageLinker first searches for nearest neighbour in the passage corpus using a BM25 retriever (Chen et al., 2017b). Similar to Chen et al. (2021), RowPassageLinker additionally uses a pre-trained GPT-2 model as context generator for each row and uses the generated context to retrieve more relevant passages from passage corpus. Details are in Appendix A.2.

5 Experiments

5.1 Datasets

HybridQA (Chen et al., 2020) is the first large scale multi-hop QA dataset that requires reasoning over hybrid contexts of tables and text. It contains 62,682 instances in the train set, 3466 instances in the dev set and 3463 instances in the test set. HybridQA provides the relevant table and its linked passages with each question, so TableRetriever and RowPassageLinker are not needed.

OTT-QA (Chen et al., 2021) extends HybridQA. It is a large-scale open-domain QA dataset over tables and text which needs table and passage retrieval before question answering. This dataset provides 400K tables and 5M passages as corpus. It has 42K questions in the training set, 2K questions in the dev set, and 2K questions in the test set.

Multiple rows containing the answer text pose a major challenge for question answering on these datasets. In HybridQA, ∼40% instances have more than one row in the table matching the answer text exactly. This makes learning to retrieve the most relevant row nontrivial.

Multiple answer spans pose additional challenges. Further analysis on HybridQA revealed that ∼34.5% instances in the training set have multiple answer spans. Details are in Appendix B.1.

5.2 Baselines and competing methods

We compare MITQA's performance with **HYBRIDER** (Chen et al., 2020), **CARP** (Zhong et al., 2022), **MATE** (Eisenschlos et al., 2021) and the methods proposed by Chen et al. (2021): iterative/fusion retrieveal (**IR/FR**) + single/cross block reader (**SBR/CBR**). Appendix B.2 has details.

5.3 Performance Summary

HybridQA: In Table 1, we compare the performance of the proposed models on the dev and test sets of HybridQA dataset. We evaluate the performance in terms of exact match (EM) and F1 scores between predicted answer and ground truth answer. We observe that MITQA, which incorporates passage filtering, multi instance training and joint row+span reranking achieves the best performance on dev as well as test set in terms of both EM and F1. The final best model achieves ∼21% absolute improvement over HYBRIDER in both EM and F1 on the test splits. At the time of writing, our system also has a ∼4% lead in both EM and F1 over the next best submission on the public leaderboard. Our system outperforms MATE (Eisenschlos et al., 2021) (a contemporary work reporting performance on HybridQA dataset) by ∼1.5–2%.

OTT-QA: In Table 2, we compare the performance of the best performing method, MITQA, on the dev and test sets of OTT-QA dataset. We report the final answer prediction performance in terms of exact match (EM) and F1 scores. Table 2 shows MITQA achieves the best performance on dev as well as test set in terms of both EM and F1. It delivers ∼10% absolute improvement over the best performing baseline by (Chen et al., 2021) in both EM and F1 on the test splits. It also achieves ∼4% higher EM on test set when compared to the very recent CARP (Zhong et al., 2022).

5.4 MITQA Ablation Setup

MITQA is a complex system with many modules working in concert. It starts from a base system (RATQA, see below) and then adds several enhancements. In this section, we compile a list of these enhancements, show their effects on performance, and analyze the results.

RATQA: Row retrieval Augmented Table-text Question Answering (RATQA) is a minimal ablation of MITQA. RATQA includes a BERTLARGE (Devlin et al., 2018) based row retriever trained on standard cross-entropy loss and a BERTLARGE based answer extractor. The answer extractor is trained with all the rows having a string match with the answer text. During inference, we get the best

	Table				**Passage**				**Total**			
	Dev		**Test**		**Dev**		**Test**		**Dev**		**Test**	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Table-Only	14.7	19.1	14.2	18.8	2.4	4.5	2.6	4.7	8.4	12.1	8.3	11.7
Passage-Only	9.2	13.5	8.9	13.8	26.1	32.4	25.5	32.0	19.5	25.1	19.1	25.0
HYBRIDER ($\tau = 0.8$) (Chen et al., 2020)	54.3	61.4	56.2	63.3	39.1	45.7	37.5	44.4	44.0	50.7	43.8	50.6
POINTR + MATE [†] (Eisenschlos et al., 2021)	**68.6**	**74.2**	66.9	72.3	62.8	71.9	62.8	72.9	63.4	71.0	62.8	70.2
MITQA	68.1	73.3	**68.5**	**74.4**	**66.7**	**75.6**	**64.3**	**73.3**	**65.5**	**72.7**	**64.3**	**71.9**

Table 1: End-task performance on dev and test folds of HybridQA, comparing prior systems against MITQA. [†]—Systems contemporary to MITQA.

	Dev		**Test**	
	EM	F1	EM	F1
HYBRIDER (Top-1) (Chen et al., 2020)	8.9	11.3	8.4	10.6
HYBRIDER (best Top-K)	10.3	13.0	9.7	12.8
IR+SBR (Chen et al., 2021)	7.9	11.1	9.6	13.1
FR+SBR (Chen et al., 2021)	13.8	17.2	13.4	16.2
IR+CBR (Chen et al., 2021)	14.4	18.5	16.9	20.9
FR+CBR (Chen et al., 2021)	28.1	32.5	27.2	31.5
CARP[†] (Zhong et al., 2022)	33.2	38.6	32.5	38.5
MITQA	**40.0**	**45.1**	**36.4**	**41.9**

Table 2: End-task performance on dev and test folds of OTT-QA. IR=iterative retriever, FR=fusion retriever. SBR=single block reader, CBR=cross block reader. Best numbers overall are in bold. [†]—Systems contemporary to MITQA.

row from the retriever and apply AnswerExtractor.

MIL: This is the novel multi instance loss function (Section 4.2) used to deal with multiple rows getting incorrect labels if they contain the answer text. Without MIL i.e. if a naive cross entropy loss is used, they lead to a noisy training regime.

RF: As described in Sec. 4.4, we use a pre-trained row retriever to score rows in the train set. This score is used to select the most relevant row while constructing the training data for AnswerExtractor. For the control case (no RF), we create separate instances for AnswerExtractor from all rows where the gold answer text occurs.

MST: Multi-span answer extractor training (Algorithm 1) is used. For the control case, the leftmost answer span is used.

RSR: Algorithm 2 is used for joint row+span reranking, with K=5. For the control case, K=1.

PF: PassageFilter (Sec. 4.1 and Appendix A.3) is used to select a limited number of tokens to attach to a linearized row, to fit within the input capacity of BERT. In the control setting without PF, we concatenate connected passages in left-to-right cell order while constructing the context, and retain the largest prefix accepted by BERT.

Ablations					**Total**			
MIL	RF	MST	RSR	PF	**Dev**		**Test**	
					EM	F1	EM	F1
					51.6	59.5	54.0	62.1
	✓				53.5	61.2	57.3	64.6
	✓	✓			53.8	61.5	57.1	64.6
	✓		✓		58.8	66.0	59.1	66.2
	✓	✓	✓		58.9	67.0	59.3	67.1
				✓	60.2	68.0	57.1	65.5
	✓			✓	63.0	70.3	61.0	68.0
	✓	✓		✓	64.1	71.3	62.2	69.3
	✓		✓	✓	63.9	71.1	62.6	69.7
	✓	✓	✓	✓	64.8	71.9	63.4	70.6
✓				✓	60.7	68.4	58.1	66.6
✓	✓			✓	64.7	71.7	63.4	70.7
✓	✓	✓		✓	64.8	71.9	63.5	70.8
✓	✓		✓	✓	65.3	72.4	**64.3**	71.7
✓	✓	✓	✓	✓	**65.5**	**72.7**	**64.3**	**71.9**

Table 3: Ablations of MITQA, starting from the RATQA baseline and progressing to the full MITQA system.

5.5 MITQA Ablation results and analysis

Table 3 shows the results of ablation experiments. In the rest of this section, we will discuss the key takeaways.

Benefits of retrieving row, then span: Comparing HYBRIDER in Table 1 and RATQA in Table 3, we see that our strategy to retrieve correct rows first works better than HYBRIDER, producing ∼12% F1 score improvement even without any other enhancements and without retriever feedback. This shows that identifying the correct/best rows is of utmost importance and brings large benefits.

Multi-Row Training (MIL) benefits: Table 3 also gives evidence that training with the new Multi Instance Loss helps RowRetriever increase overall F1 score beyond passage ranking alone.

Multi-Span Training (MST) benefits: Multi Span Training (Algorithm 1) usually boosts performance by 0.5-1%. This demonstrates the effectiveness of training on denoised data.

Joint Row+Span Reranking (RSR) benefits: Beyond multi-span training (MST) of AnswerExtractor, the joint row+span reranker (RSR) improves F1

K	Table retrieval accuracy (%)
1	41.28
5	68.15
10	76.51
50	88.07

Table 4: TableRetriever HITS@K, OTT-QA dev set.

Ablations		Row Retrieval
MIL	PF	Accuracy (%)
		81.39
	✓	84.30
✓	✓	86.38

Table 5: RowRetriever accuracy, HybridQA dev fold.

score as compared to model variations not applying these strategies. In fact, these enhancements can be applied together — as seen in Table 3, model variations with MST+RSR produce the best results.

PassageFilter (PF) benefits: While designing PassageFilter, our intent was to minimize the damage from discarded text. Comparing RATQA+PF against RATQA, we find that not only is PassageFilter effective in this role, but it can, in fact, *increase* F1 score by pruning irrelevant passages before invoking RowRetriever and AnswerExtractor.

Retriever Feedback (RF) benefits: In all ablations of MITQA that include multi-row training, RF acts as a positive influence, always yielding better F1 scores than ablations without RF. This translates to better AnswerExtractor performance using less data. With RF, the model is only trained on the best row, while without RF, thrice as much training data is available, but it is more noisy. This also demonstrates the superiority of our row retriever in enhancing answer extractor performance.

5.6 Performance of additional modules

TableRetriever: Given a question, *TableRetriever* retrieves top-k tables from ∼400K tables provided in the corpus of OTT-QA. Table 4 gives the hit rates at top-k predictions for various values of k.

RowRetriever: In Table 5, we present row retrieval accuracy of our models on the dev split of HybridQA dataset. We also present ablations corresponding to all the modules affecting the accuracy i.e. MIL and PF. We observe that passage filtering improves the row retrieval accuracy by ∼3%. Changing standard cross entropy loss to multi instance loss (Section 4.2) further boosts the row retriever accuracy by ∼2%.

PassageFilter: We find that average number of tokens in the context for the dev set is 585, with 49% examples exceeding BERT's maximum token count of 512 (thus needing truncation). We see that, if we follow our passage ranking and filtering strategy before truncation, the answer is retained in the truncated context in around ∼1–2% more dev set examples. Interestingly, the observed performance gain for our answer extractor is slightly larger than this. This can be attributed to the fact that with passage ranking, the correct span more often appears as the first one and gets correctly chosen during back-propagation training for answer extraction.

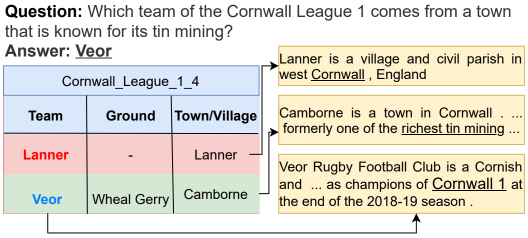

Question: Which team of the Cornwall League 1 comes from a town that is known for its tin mining?
Answer: Veor
Cornwall_League_1_4
Team
Ground
Town/Village
Lanner
-
Lanner
Veor
Wheal Gerry
Camborne
Lanner is a village and civil parish in west Cornwall , England
Camborne is a town in Cornwall . ... formerly one of the richest tin mining ...
Veor Rugby Football Club is a Cornish and ... as champions of Cornwall 1 at the end of the 2018-19 season .

Figure 4: The benefit of MITQA over HYBRIDER.

5.7 An anecdotal example

Figure 4 shows how MITQA can outperform at answering questions where the context might cause confusion to both retriever and reader because of multiple matches of important question keywords. As shown, HYBRIDER got confused by the presence of 'Cornwall' in the first row and produced an incorrect answer 'Lanner'. In contrast, MITQA predicts the correct answer 'Veor'. Appendix C shows more examples.

6 Conclusion

TextTableQA requires reasoning over table cells and linked passage contents. Weak supervision poses a challenge: the target answer might be mentioned in multiple row cells and/or as multiple spans in linked passages. We design a novel QA pipeline that uses multiple row and multiple answer based novel training strategies to identify correct rows first and then use the row cells for relevant passage lookup. We propose efficient strategies for filtering linked passages to retain the most relevant ones for the question, and a novel re-ranker to rank the answers obtained from different rows and their respective linked passages. Our system, MITQA, performs better than recent systems on HybridQA and OTT-QA benchmarks, with large improvements in F1 scores. We have also tried different combinations of our proposed strategies to substantiate the benefit from each of them separately. In future, we would like to explore the following directions: (1) answering complex numerical questions over hybrid context of table and text, (2) handling more complex table with structural hierarchies, and (3) enhancing MITQA to provide interpretable explanations for answers.

7 Limitations

Although MITQA achieves the best results for TextTableQA benchmarks to date, it still has some limitations, owing to its design, and the type of training data it can access.

Design policy: We have designed MITQA as a collection of trainable modules, which are used in a specific sequence. This design has helped us to focus our innovations in specific modules such as multi-row training for RowRetriever, multi-span training for AnswerExtractor, etc., with an eye to boost overall accuracy. However, the modular design also means that MITQA is not fully end-to-end trainable. Therefore MITQA is, in principle, susceptible to compounding error propagation across modules. We view this as an acceptable trade-off while working on HybridQA and OTT-QA, but other data sets may force us to revisit this decision.

Types of queries: TextTableQA, being a relatively new task, has only two major benchmarks available (HybridQA and OTT-QA), where OTT-QA is an open domain extension of HybridQA. Therefore, the types of queries to which MITQA during training are limited to effectively a single large benchmark (HybridQA). HybridQA — and consequently OTT-QA — corpora are similar to Wikipedia articles, not confined to any specific domain. Further experiments in specific verticals, such as Finance, Retail, and Health are needed to check if MITQA affords practical cross-domain adaptation.

Moreover, only a small fraction of queries in HybridQA and OTT-QA need aggregation. Due to their rareness, we have not considered handling aggregation queries through MITQA, which needs additional work in future.

References

Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. 2003. Support vector machines for multiple-instance learning. *Advances in neural information processing systems*, pages 577–584.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Razieh Baradaran, Razieh Ghiasi, and Hossein Amirkhani. 2020. A survey on machine reading comprehension systems.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017b. Reading Wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Wang, and William W. Cohen. 2021. Open question answering over tables and text.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *Findings of EMNLP 2020*.

Pradeep Dasigi, Matt Gardner, Shikhar Murty, Luke Zettlemoyer, and Eduard Hovy. 2019. Iterative search for weakly supervised semantic parsing. 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 2669–2680, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1-2):31–71.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *CoRR*, abs/1903.00161.

Julian Martin Eisenschlos, Maharshi Gor, Thomas Müller, and William W. Cohen. 2021. Mate: Multi-view attention for table transformer efficiency.

Michael Glass, Mustafa Canim, Alfio Gliozzo, Saneem Chemmengath, Vishwajeet Kumar, Rishav Chakravarti, Avi Sil, Feifei Pan, Samarth Bharadwaj, and Nicolas Rodolfo Fauceglia. 2021a. 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, Online. Association for Computational Linguistics.

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, and Alfio Gliozzo. 2021b. Robust retrieval augmented generation for zero-shot slot filling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1939–1949, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *CoRR*, abs/1705.03551.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

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, Copenhagen, Denmark. Association for Computational Linguistics.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth 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)*, pages 23–33.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.

Alexander J. Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. *NeurIPS*, 29:3567–3575.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2021. Contrastive learning with hard negative samples. *ICLR*.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big Bird: Transformers for longer sequences. *NeurIPS*, abs/2007.14062.

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

Wanjun Zhong, Junjie Huang, Qian Liu, Ming Zhou, Jiahai Wang, Jian Yin, and Nan Duan. 2022. Reasoning over hybrid chain for table-and-text open domain qa.

Multi-Row, Multi-Span Distant Supervision For Table+Text Question Answering (Appendix)

A Further details of MITQA modules

A.1 TableRetriever and its training

In the open domain QA setting (like in OTT-QA) where a designated table t and linked passages $t[\star,\star].\mathsf{psg}$ are not provided, we employ the module $\text{TableRetriever}(q)$ to retrieve the most promising tables $T_q \subseteq T$, where T is the corpus of tables.

The training of the TableRetriever module follows the original DPR work (Karpukhin et al., 2020) and its recent application (Glass et al., 2021b), where we first index the linearized tables with Anserini. TableRetriever is trained using triplet loss over instances of the form $\langle q, t_\oplus, t_\ominus \rangle$, where $t_\oplus$ is a ground-truth table and $t_\ominus$ is a *hard negative* (Robinson et al., 2021) — an irrelevant table that scores highly with respect to the current scoring model.

To collect hard negative tables $t_\ominus$, we retrieve a pool of tables from a BM25 text retrieval system, and remove the gold table if it is retrieved. The surviving tables are considered 'hard'. To further enhance the robustness of TableRetriever, we select the hard negative table at random from some number of top-scoring hard negative tables.

The tables and the questions are encoded independently using the same BERTBASE (Devlin et al., 2018) model. We later calculate the inner product of question embedding and embedding of all tables to locate the top-scoring relevant tables.

A.2 RowPassageLinker

For each table $t \in T_q$ returned by TableRetriever and every row r in table t, we use a $\text{RowPassageLinker}(t, r)$ to retrieve the most appropriate passages (from a large corpus of text) and link them to appropriate cells $t[r, c]$. RowPassageLinker first searches for nearest neighbour of the cell text in the passage corpus using a BM25 retriever (Chen et al., 2017b) and retrieves 10 passages. Similar to Chen et al. (2021), RowPassageLinker additionally uses a pre-trained GPT-2 model to generate text from row $t[r, *]$. and uses the generated text as context to retrieve 10 more relevant passages from the passage corpus. Specifically, the model takes in the text of $t[r, c]$ as input and outputs additional augmented queries, which are then fed again to the BM25 retriever as queries, to retrieve additional relevant passages. The GPT-2 model is fine-tuned on the supervised pairs of table row (i.e., $t[r, \star]$), header (i.e., $t.\mathsf{hdr}$) and their hyperlinks ($t[r, \star] + t.\mathsf{hdr}$, hyperlink) from in-domain (HybridQA) tables.

A.3 PassageFilter (PF) and its training

Given a question, table, and a set of passages connected to cells in the table, PassageFilter ranks the passages based on their relevance to the question. We use Sentence-BERT (Reimers and Gurevych, 2019) to get question and passage embeddings and we perform asymmetric semantic search to rank the passages. Asymmetric semantic search is a feature in Sentence-BERT that allows to find a longer passage/document based on a short question.

Passage ranking plays a vital role in row retrieval as well as answer extraction. BERT encoders (used in RowRetriever and AnswerExtractor) have a limitation that they cannot process sequences of length more than 512 tokens. Passage ranking ensures that even if we truncate the context to fit BERT, we are unlikely to lose passages most relevant to the question. Because we do not have supervision about which passages should be ranked higher, we train PassageFilter on a similar task of passage ranking given a query on the MS MARCO Passage Retrieval dataset (Bajaj et al., 2016).

Moreover, in case the context contains multiple spans, passage filtering helps to bring the correct answer span at the top, thus reducing the possibility of noisy labels. This is particularly important, because the basic model of answer extractor without multi span training (MST), back-propagates through the first span in the passage matching with the gold answer.

A.4 AnswerExtractor text linearization

A training instance for answer extraction consists of a token sequence generated by concatenating linearized row contents and passages (linked to cells in the row), together with start and end span indexes of the ground truth answer. We linearize a row as "`<column-header> is <cell-content>`". This simple linearization bypasses the need to introduce new additional special tokens as column-header and row delimiters, and avoids computationally intensive training of their embeddings. The concatenated sequential context often exceeds BERT's 512-token limit. We reduce the probability of the passage containing the ground truth answer getting truncated, by using PassageFilter (Appendix A.3).

B More Details on Experiments

In this section, we give further details on our experimental approaches.

B.1 Datasets

HybridQA is the first large scale multi-hop QA dataset that requires reasoning over hybrid contexts of tables and text. It contains 62,682 instances in the train set, 3466 instances in the dev set and 3463 instances in the test set. For the test set, ground truth answers are not available. The authors employ Amazon Mechanical Turk crowd-workers to generate questions based on Wikipedia tables with cells linked to Wikipedia pages. We split the tables into rows with column headers attached. This enables us to pose the QA problem as row retrieval and answer extraction from the retrieved row.

OTT-QA extends over HybridQA to make it a large-scale open-domain QA dataset over tables and text which needs table and page retrieval before question answering. This dataset provides 400k tables and 5 million passages as corpus. It has 41,469 questions in the training set, 2,214 questions in the dev set, and 2,158 questions in the test set. According to (Chen et al., 2021) , a remarkable difference from original HybridQA is that a proportion of questions actually have multiple plausible inference chains in the open-domain setting.

Multiple rows containing the answer text pose a major challenge for question answering on these datasets. As depicted in Figure 3, for HybridQA, $\sim$40% instances have more than one row in the table matching the answer text exactly. This makes retrieving the most relevant row highly nontrivial.

Multiple answer spans pose additional challenges. Further analysis on HybridQA revealed that $\sim$34.5% instances in the training set have multiple answer spans.

B.2 Baselines and Competing Methods

HYBRIDER We compare our model's performance with the standard HYBRIDER (Chen et al., 2020) baseline. HYBRIDER uses a two phase process of linking and reasoning to answer questions over heterogeneous context of table and text. This approach attempts to use cell as a unit for linking, hopping and answer prediction.

Iterative and Block Retrieval These models are proposed by Chen et al. (2021) and are combinations of Iterative/Fusion retrievers and Single/Cross readers. Fusion retrieval uses “early fusion" strategy to group tables and passages as fused blocks before retrieval. Single Block Reader feeds top-k blocks independently to the reader and selecting the best answer. Cross Block Reader concatenates top-k blocks together to the reader, and generates a single joint answer string.

MATE MATE (Eisenschlos et al., 2021) models the structure of large Web tables. It uses sparse attention in a way that allows heads to efficiently attend to either rows or columns in a table. To apply it on HybridQA, the authors propose $PointR$, which expands a cell using description of its enitities, selects an appropriate expanded cell and then reads the answer from it.

CARP CARP (Zhong et al., 2022) is a chain-centric reasoning and pre-training framework for table-and-text question answering. It first extracts explicit hybrid chain to reveal the intermediate reasoning process leading to the answer across table and text. The hybrid chain then provides a guidance for QA, and explanation of the intermediate reasoning process.

Other baselines These can be found on the respective challenge leaderboards.[2] There are no linked papers to the submissions as yet. We compare MITQA's test performance against all of them.

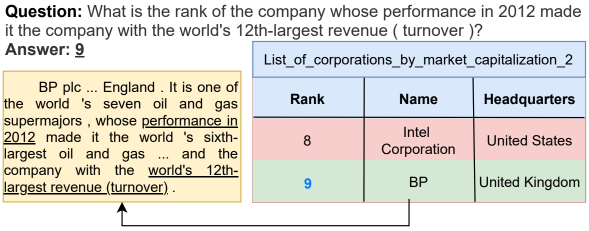
B.3 Implementation Details

MITQA is implemented using Pytorch version 1.8 and Huggingface's transformers[3] (Wolf et al., 2020) library. We train our models using two NVIDIA A100 GPUs. We train the row retriever and answer extractor for 5 epochs and select the best model based on dev fold performance. We optimize the model parameters using AdamW algorithm with a learning rate of 5×10^{-5} and a batch size of 24. We set per-GPU train batch size to 16 while training the answer extractor. We evaluate final answers using EM (exact match) and F1 metrics.

Average Runtime: Overall training of MITQA takes approximately 24 hours on A100 gpu.

[2] HybridQA: https://competitions.codalab.org/competitions/24420
OTT-QA: https://competitions.codalab.org/competitions/27324

[3] https://huggingface.co/



Question: What is the rank of the company whose performance in 2012 made it the company with the world's 12th-largest revenue (turnover)?
Answer: 9

BP plc ... England . It is one of the world 's seven oil and gas supermajors , whose performance in 2012 made it the world 's sixth-largest oil and gas ... and the company with the world's 12th-largest revenue (turnover) .

List_of_corporations_by_market_capitalization_2

Rank	Name	Headquarters
8	Intel Corporation	United States
9	BP	United Kingdom

Figure 5: MITQA is able to extract answer even if the answer is only present in the table as a cell value. The correct answer is highlighted in blue. Despite having other numbers in the table and phrases mentioning ranks like 'seven', 'sixth-largest', etc. in the passage MITQA was able to predict the correct answer from the table.

Hyperparameter Details: We tune hyper-parameters based on loss on validation set. We use the following range of values for selecting the best hyper-parameter

- **Batch Size:** 8, 16, 32
- **Learning Rate:** 1e-3, 1e-4, 1e-5, 1e-6, 3e-3, 3e-4, 3e-5, 3e-6, 5e-3, 5e-4, 5e-5, 5e-6

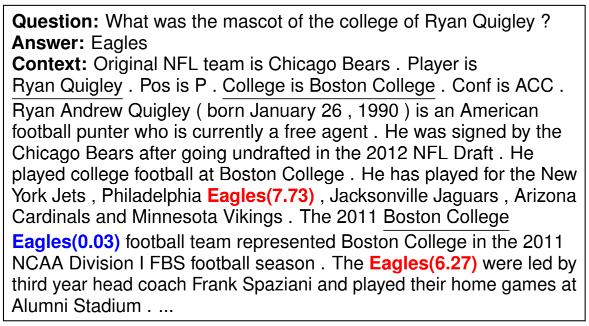
C Anecdotes of Gains

C.1 Answer in Table Cell

We present in Figure 5 an example where MITQA is able to predict the answer correctly even when the correct answer is in a table cell and not a span in the passages.

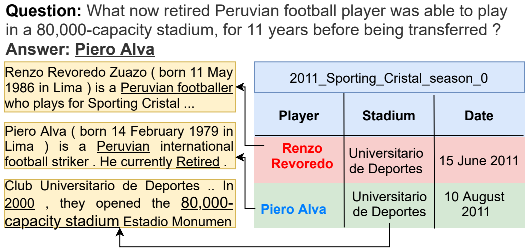
C.2 Benefits of Multi Span Training (MST)

Figure 6 shows an example where MST leads the model to train on the correct answer span, thereby leading to a less noisy training regime.



Question: What was the mascot of the college of Ryan Quigley ?
Answer: Eagles
Context: Original NFL team is Chicago Bears . Player is Ryan Quigley . Pos is P . College is Boston College . Conf is ACC . Ryan Andrew Quigley (born January 26 , 1990) is an American football punter who is currently a free agent . He was signed by the Chicago Bears after going undrafted in the 2012 NFL Draft . He played college football at Boston College . He has played for the New York Jets , Philadelphia **Eagles(7.73)** , Jacksonville Jaguars , Arizona Cardinals and Minnesota Vikings . The 2011 Boston College **Eagles(0.03)** football team represented Boston College in the 2011 NCAA Division I FBS football season . The **Eagles(6.27)** were led by third year head coach Frank Spaziani and played their home games at Alumni Stadium . ...

Figure 6: Benefits from MAT. The model loss is shown in brackets along with the spans. It is clear that the correct mention (in blue) rightly gets the lowest loss while the ones which are irrelevant (in red) have higher losses. Contexts that can potentially help answer the question are underlined. The first 'Eagles' in entirely irrelevant as it refers to a different team. The second one is the best answer by far. The third occurrence refers to the correct team, but lacks as good a context as the second (for model learning).



Question: What now retired Peruvian football player was able to play in a 80,000-capacity stadium, for 11 years before being transferred ?
Answer: Piero Alva

Renzo Revoredo Zuazo (born 11 May 1986 in Lima) is a Peruvian footballer who plays for Sporting Cristal ...

Piero Alva (born 14 February 1979 in Lima) is a Peruvian international football striker . He currently Retired .

Club Universitario de Deportes .. In 2000 , they opened the 80,000-capacity stadium Estadio Monumen

2011_Sporting_Cristal_season_0

Player	Stadium	Date
Renzo Revoredo	Universitario de Deportes	15 June 2011
Piero Alva	Universitario de Deportes	10 August 2011

Figure 7: The benefit of Row Span Re-ranker. The correct answer is highlighted in blue and the incorrect answer is highlighted in red. Both 'Piero Alva' and 'Renzo Revoredo' played in the same stadium ('Universitario de Deportes'). But only 'Piero Alva' (answer after reranking) has retired while 'Renzo Revoredo' (answer before reranking) has not. Thus, the RSR helps rank the correct answers higher than the incorrect ones in similar contexts and confusing scenarios.

C.3 Benefits of Row Span Re-ranker (RSR)

Figure 7 depicts an instance where RSR is able to rectify the error made by MITQA. The incorrect answer also appeared in a context very similar to the context of correct answer but Multi-Row Re-ranker is able to rank the correct answer higher than the incorrect answer.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7

□ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

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

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

B ☑ Did you use or create scientific artifacts?

Section 5

☑ B1. Did you cite the creators of artifacts you used?
Section 2

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

□ 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)?
Not applicable. Left blank.

□ 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?
Not applicable. Left blank.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ 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.
Appendix B

C ☑ Did you run computational experiments?

Section 5

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

☑ 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?

Left blank.

□ 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.?
Not applicable. Left blank.

□ 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)?
Not applicable. Left blank.

□ 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?
Not applicable. Left blank.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.