Information Processing and Management 62 (2025) 104135

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

Retrieval augmentation for text-to-table generation☆

Jiahua Zhang [a], Meijuan Tan [b], Jing Zhang [c], Xiaolu Zhang [d], Jun Zhou [d], Chenliang Li [a,*]

[a] Key Laboratory of Aerospace Information Security and Trusted Computing, School of Cyber Science and Engineering, Wuhan University, Wuhan, 430072, Hubei, China
[b] Minzu University of China, Beijing, 100081, China
[c] Central China Normal University, Wuhan, 430072, Hubei, China
[d] Ant Group, Hangzhou, 310013, Zhejiang, China

ARTICLE INFO

Keywords:
Retrieval augmentation
Text-to-table generation

ABSTRACT

The task of text-to-table generation is important for many real-world scenarios. The current solutions mainly take the textual information as input and transform it into a table structure. However, the original textual information often does not contain the structural information explicitly for table generation: it is difficult to formalize the table headers. Additionally, it is often inappropriate to directly copy the textual segments from the original input as cell values. That is, we need more background knowledge to help derive the cell values instead. In this paper, we propose a simple yet effective framework to exploit retrieval augmentation for better text-to-table generation (namely RAGTABLE). The idea is to retrieve the relevant exemplar knowledge as the guidance to help generate table for the target input. Specifically, we comprehensively integrate three perspectives: Text Matching(TeM), Iterative Table Generation (ITG) and Noise-aware Learning(NaL), under a unified pipeline in RAGTABLE. Experimental results show that our RAGTABLE significantly surpasses the baseline models and achieves state-of-the-art (SOTA) performance on commonly used datasets, especially, the generation performance gain for non-header cells reaches up to 20%.

1. Introduction

Text-to-table generation is aimed at generating tabular description of important information covered by the given text. As shown in Fig. 1, the given input is a descriptive unstructured text, and the desired output is a structured table. Typically, this task finds wide applications in extracting crucial structured information, such as restaurant reviews (Novikova et al., 2017), player statistics (Wiseman et al., 2017) and Wikipedia infoboxes (Bao et al., 2018), aiding humans in effective information consumption.

Recently, several efforts have been made under the seq2seq architecture. Wu et al. (2022) trained seq2seq model fine-tuned from BART (Lewis et al., 2020) while exploiting the relationships of table headers during decoding. Furthermore, Li, Wang, et al. (2023) proposed a two-stage pipeline by generating the table headers during decoding, followed by parallelly generating the non-header

☆ This work was supported by the National Natural Science Foundation of China (No. 62272349, No. 12071172), Natural Science Foundation of Hubei Province (No. 2023BAB160), the Ministry of Education Humanities and Social Sciences Project (No. 24JDSZ3073), and CAAI-Ant Group Research Fund (No. CAAI-MYJJ2024-01). The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University. Chenliang Li is the corresponding author.

* Corresponding author.
E-mail address: cllee@whu.edu.cn (C. Li).

https://doi.org/10.1016/j.ipm.2025.104135
Received 13 October 2024; Received in revised form 14 January 2025; Accepted 6 March 2025
Available online 25 March 2025

Text	
barbara schett and patty schnyder defeated martina hingis and jana novotná in wta hamburg at hamburg, germany on 3 may 1998.	
Table	
title	barbara schett
subtitle	career finals
date	3 may 1998
tournament	wta hamburg
location	hamburg, germany
partner	patty schnyder

Fig. 1. An example of a text-to-table pair.

Test Text	
michelle schimel was new york state assemblywoman in portuguese heritage society.	
Ground Truth	
title	portuguese heritage society
subtitle	other activities
name	michelle schimel
office	new york state assemblywoman

Original Generation	
title	portuguese heritage society
subtitle	other activities
name	michelle schimel
office	new york state assemblywoman

Text of a Sample in Training Set	
liz kreuger was the new york state senator of portuguese heritage society.	
Ground Truth	
title	portuguese heritage society
subtitle	other activities
name	liz kreuger
office	new york state senator

Generation after Deleting those Samples	
title	portuguese heritage society
subtitle	leadership
position	new york state assemblywoman
name	michelle schimel

Fig. 2. Case study of solely rely on the original textual information, where the blue portion represents information that can be clearly extracted from the text, the green portion indicates information that is difficult to extract or summarize from the text, and the red portion denotes content incorrectly generated by the model after certain training data is removed. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

cells. This not only enhances the generation of more consistent table structure but also prevents error accumulation from one row to subsequent rows.

However, these existing approaches focus on decoding process and typically rely solely on the original textual information for table generation. We observe that table generation involves learning various patterns and specific background knowledge. It is often the case that certain table contents cannot be straightforwardly inferred from the original textual information alone. Consider the example shown in Fig. 2, it is very difficult to generate "other activities" for header "subtitle" by simply examining the given input. Interestingly, upon training a basic seq2seq model, it accurately generates this entry during inference, owing to the fact that the training set contains a few instances including this header-value pair. When we eliminate only four relevant training instances from the training process, it instead fails to generate the correct content for the 'subtitle' row. The model not only needs to learn how to extract important information from the given input to generate the table content, but also to determine table structure and elements that are not explicitly covered in the training data.

It is intuitive to explicitly provide the model with relevant structure patterns and background knowledge for generation enhancement. As illustrated on the right side of Fig. 3, from a more straightforward perspective, with the assistance of the retrieval table and by referencing its content, the model is able to correct header generations that do not meet expectations and produce headers that better reflect the textual semantics. Furthermore, compared to merely considering the original text input, this approach provides a clearer understanding of how to generate more accurate table values. Hence, we propose a simple yet effective framework

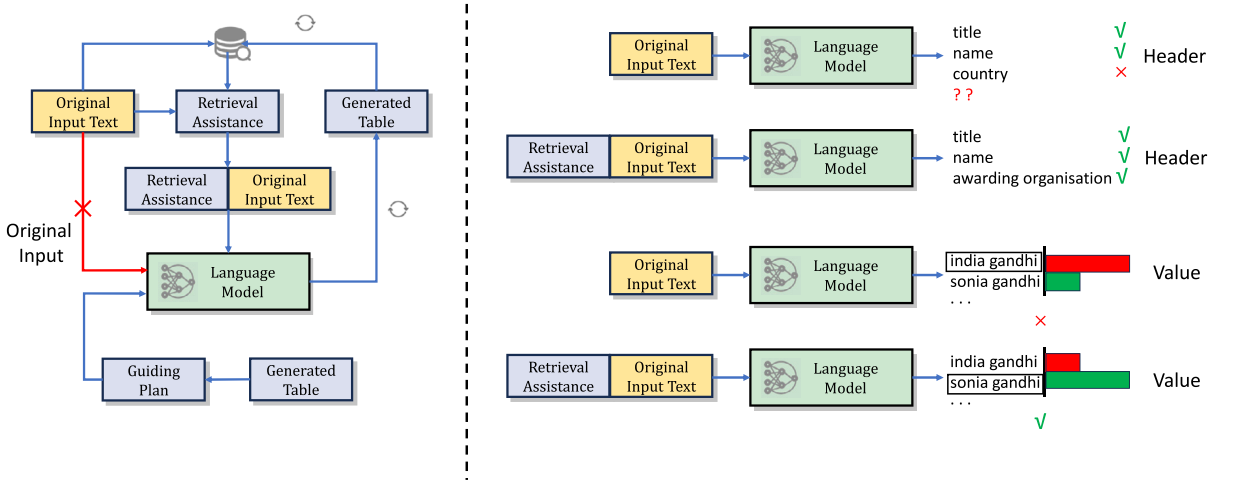

Fig. 3. Framework of RAGTABLE. On the left is the flowchart of our approach, while on the right is why we place a × between the original input and the model.

to exploit retrieval augmentation for better table generation. As shown on the left side of Fig. 3, a unified pipeline is devised in the proposed RAGTABLE, aiming at exploiting *text matching*, *iterative table generation* and *noise-aware learning* in one go. Initially, during training, we use the target textual input as query to match relevant text from the training set. We then concatenate the corresponding text-table pair with the original input, allowing the model to generate its output based on this enriched context. After training, we introduce an iterative process for table generation. Specifically, we firstly generate the table following the text matching scheme, by augmenting the original input with the most relevant text-table pair. Then, the generated table is taken as query to retrieve the most relevant table from the training set. Afterwards, we concatenate the corresponding text-table pair retrieved by this table matching with the original input to generate the table again. Inspired by the two-stage pipleline proposed in Su et al. (2021), we then take the headers of the resultant table as another kind of auxiliary information, and ask the model to generate the final table.

Note that the retrieved text-table pairs often contain noisy information irrelevant to the target table. Similar observations can also be made for the intermediate tables generated in the above iterative process. To endow the model with the noise-aware robustness, we further introduce the noise augmentation to match the aforementioned iterative table generation process. Specifically, during the model training, we take the headers of the ground-truth table as another kind of auxiliary input. By randomly replacing these headers with other values, we explicitly stimulate error-prone table structure generated in the iterative process. This noise augmentation would allow model to learn how to identify the useful auxiliary information, leading to robust generation performance.

Compared to solely providing the model with the original textual input, our method, which incorporates retrieval augmentation, not only enhances the table structure generation, but also enables the model to utilize auxiliary information that are challenging to extract from the text alone, leading to more precise generation of table content. The experimental results over real-world datasets well demonstrate that our approach surpasses the SOTA solutions by a large margin. Specially, in the WikiTableText dataset (Bao et al., 2018), the performance gain for generating non-header cells reaches up to 20%.

In a nutshell, we make the following contributions in this paper:

- We are the first to highlight the importance of exploiting relevant auxiliary information for better text-to-table generation. Specifically, we argue that providing the model with relevant structure patterns and background knowledge would help generate more precise table data.
- We propose a simple yet effective framework that take the retrieved text-table pair to guide the table generation. In our proposed RAGTABLE, we comprehensively integrate three perspectives (*i.e.,* text matching, iterative table generation and noise-aware learning) under a unified pipeline.
- We conduct extensive experiments over real-world datasets with different characteristics. The results well demonstrate that RAGTABLE achieves significantly superior generation performance than the existing baselines. Further analysis also validate the rationality of our solution.

2. Related work

Table-to-Text Generation aims to generate natural language descriptions from the input structured tables. Among recent advances table-to-text models, Gong et al. (2020), Li, Fang, et al. (2021), Xing and Wan (2021) introduce novel ways to efficiently encode tabular data. Jiang et al. (2020), Sha et al. (2018) propose copy and point decoding mechanism to deal with out-of-vocabulary(OOV) problems. Various interesting learning strategies have also been introduced, such as multi-task learning (Gong et al., 2019; Li, Ma, et al., 2021) and reinforcement learning (Liu et al., 2019; Nishino et al., 2020; Zhao et al., 2021). Pre-training for table-to-text generation (Clive et al., 2021; Harkous et al., 2020; Qian et al., 2022; Zhao, Zhang, et al., 2023) is beneficial as well.

Furthermore, to solve the problem of poor faithfulness and low coverage, neural modular approaches are proposed: Moryossef et al. (2019), Puduppully et al. (2019), Su et al. (2021) introduce two-stage approaches(content planning and text generation) and neural template-based approaches are adopted by Upadhyay et al. (2021), Wiseman et al. (2018), Ye et al. (2020).

In this paper, we primarily focus on text-to-table generation, the dual task of table-to-text generation, aiming to generate structured tables from natural language descriptions. Information extraction is deeply associated with this task. Previous works continuously improve the effectiveness of information extraction, including Relation Extraction (RE) (Hu et al., 2024; Nayak & Ng, 2020; Zeng et al., 2018, 2024; Zhao, Gao, et al., 2023), Named Entity Recognition (NER) (Chen & Moschitti, 2018; Feng et al., 2024; Geng et al., 2023; Liu et al., 2024; Yan et al., 2021), and Event Extraction (EE) (Li, Ji, et al., 2021; Lu et al., 2021). Adhering to this trend, researchers employ unified models advocated by Paolini et al. (2021) and Lu et al. (2022), modeling multiple IE tasks as the production of sequences in a consistent format. However, compared to traditional information extraction tasks, the text-to-table generation task does not rely on predefined schemas, and we do not predefine which information to extract. Wu et al. (2022) first explored this task as a seq2seq generation task by fine-tuning the BART model (Lewis et al., 2020) for generation. During decoding, the table is generated row by row sequentially, and an interesting design is embedding table relationships during decoding. Building on this, Li, Wang, et al. (2023) proposed a method where row headers are outputted first during decoding, followed by parallel outputs of table body columns, achieving faster and better decoding. Coyne and Dong also attempted to solve this problem using the GPT3.5 large language model.

However, in existing methods, the approach utilizing supervised learning only analyzes from the original input, making it difficult for the model to determine which table headers to generate. Additionally, many table values are challenging to generate solely based on the textual information. Using few-shot learning to directly generate results with GPT3.5 lacks a large amount of data to learn different generation methods, resulting in less than ideal performance.

In recent years, there has been a growing interest in retrieval-augmented text generation across various domains, encompassing neural machine translation (Agrawal et al., 2022; Cheng et al., 2022; Gu et al., 2018), dialogue response generation (Cai et al., 2018; Li et al., 2022; Song et al., 2016), open domain question answering (Chen et al., 2017; Izacard & Grave, 2021), conversational question answering (Li, Yang, et al., 2023), aspect-level sentiment classification (Jian et al., 2024) and language modeling (Cheng et al., 2023; Khandelwal et al., 2019; Shi et al., 2023). We have found that we can adopt a retrieval-augmented approach to retrieve relevant content from the training corpus and provide suggestions to the model to generate higher-quality results.

3. Preliminary

3.1. Problem formulation

Formally, given an input text $X = x_1, \ldots, x_{|X|}$ where x_n represents the nth token, the task of text-to-table generation is to produce the table that covers the major semantics of X. Assuming a $n_r \times n_c$ table T is generated, it can be formulated as a sequence:

$$\begin{aligned}\mathbf{T} = &\langle \mathbf{s} \rangle, \mathbf{t}_{1,1}, \langle \mathbf{s} \rangle, \ldots, \langle \mathbf{s} \rangle, \mathbf{t}_{1,n_c}, \langle \mathbf{s} \rangle, \langle \mathbf{n} \rangle, \\ &\langle \mathbf{s} \rangle, \mathbf{t}_{2,1}, \langle \mathbf{s} \rangle, \ldots, \langle \mathbf{s} \rangle, \mathbf{t}_{2,n_c}, \langle \mathbf{s} \rangle, \langle \mathbf{n} \rangle, \\ &\ldots\ldots\ldots \\ &\langle \mathbf{s} \rangle, \mathbf{t}_{n_r,1}, \langle \mathbf{s} \rangle, \ldots, \langle \mathbf{s} \rangle, \mathbf{t}_{n_r,n_c}, \langle \mathbf{s} \rangle\end{aligned} \tag{1}$$

where $t_{i,j}$ represents the cell value in the ith row and jth column of table T, symbol $\langle \mathbf{s} \rangle$ denotes the separator between table cells, and $\langle \mathbf{n} \rangle$ represents a line break.

3.2. Vanilla Seq2Seq

Given the sequential nature of both X and T, it is a common choice to achieve the text-to-table generation with a Seq2Seq architecture. Specifically, many tasks have been resolved by using the Seq2Seq approach, such as machine translation (Liu et al., 2020; Raffel et al., 2020) and text summarization (Huang et al., 2020). Following the common setting in earlier work Wu et al. (2022), an encoder is utilized to derive the hidden states H as follows:

$$H = \text{Encoder}(X) \tag{2}$$

Then, the derived hidden states H are fed into a decoder which generates the table T in an autoregressive manner as follows:

$$P_G(T_i \mid T_{<i}, X) = \text{Softmax}\left(\text{Decoder}(H, T_{<i})\right) \tag{3}$$

where $T_{<i}$ refers to the sequence generated before ith step, T_i refers to the token to be generated at ith step and function *Softmax* projects the output of the decoder into a probability simplex with an MLP layer, indicating the generation probability of each token. As to model training, the cross-entropy loss is utilized as follows:

$$\mathcal{L}_{\text{LM}} = -\sum_{i=1}^{|T|} \log P_G\left(T_i \mid T_{<i}; X\right) \tag{4}$$

As mentioned in Section 2, the common practice is to utilize the Transformer networks as both the encoder and decoder. Moreover, the existing efforts have mainly dedicated on optimizing the decoder, ignoring the possibility of enhancing the input with a more enriched context through retrieval augmentation.

4. Method

In this section, we provide a detailed description of our proposed RAGTABLE. Specifically, RAGTABLE firstly choose to retrieve relevant table via Text Matching (TeM). The relevant table is then used as the input augmentation to guide the table generation. Then, for the inference phase, we introduce an Iterative Table Generation process (ITG), which enhances the quality of the retrieved content. As to the model optimization, we further enhance noise-aware robustness with Noise-aware Learning (NaL), asking the model to identify the true useful auxiliary information. Fig. 3 illustrates the whole pipeline. The training algorithm is detailed in Algorithm 1, and the testing algorithm is detailed in Algorithm 2.

4.1. Text matching

Let $\mathcal{D} = \{< X_i, T_i > \}$ denote the training set where X_i and T_i refer to the textual description and the corresponding table associated with this text-table pair respectively, we can also form the text training set and table training set: $\mathcal{X} = \{X_i\}$ and $\mathcal{T} = \{T_i\}$, for notation conciseness. Given an input textual description X, we treat it as the query and retrieve the relevant texts from $\mathcal{X}$. Specifically, we choose the widely known BM25 as the retrieval algorithm. The most similar text from $\mathcal{X}$ is retrieved as follows:

$$X_s = \arg\max_{X_j \in \mathcal{X}} \text{BM25}(X, X_j) \tag{5}$$

where X_s is the most similar textual description towards query X, and function BM25 returns the similarity score calculated by utilizing BM25 algorithm. After retrieving the closest training neighbor X_s, we can then obtain the corresponding text-table pair $< X_s, T_s >$.

Subsequently, we enrich the original input by concatenating it with the corresponding text-table pair. This augmented input provides guiding cues for header generation, and additional contextual information that cannot be derived solely from X. In detail, we utilize a standardized template to integrate the retrieved text-table pair with the original input:

$$X^a = \text{Template}(X_s, T_s, X) \tag{6}$$

where *Template* takes the form of:

$$\text{Template}(X_s, T_s, X) = \text{“}\texttt{example text : }\text{”} + X_s + \text{“}\texttt{; example table : }\text{”} + T_s + \text{“}\texttt{; test text : }\text{”} + X \tag{7}$$

Finally, we process the augmented input X^a using the standard Seq2Seq model described in Section 3.2. The loss function can be written as follows:

$$\mathcal{L}_{\text{txt}} = -\sum_{i=1}^{|T|} \log P_G(T_i \mid T_{<i}; X^a) \tag{8}$$

Note that for the training set, we retrieve the most similar text excluding itself. A running example of the text matching process, which is part of the overall workflow, is illustrated across Figs. 4 and 5, with Fig. 4 focusing on the training process and Fig. 5 on the testing process. We can see that the model can learn useful patterns from both original input and relevant contextual information for better table generation.

4.2. Iterative table generation

In TeM, we treat the original input text as query. However, what we actually need is the guidance of possible table information relevant to that text. Given the case that we could use the table as the query to retrieve the corresponding table from the training set, the retrieved knowledge would be more directly relevant for the target table.

More specifically, although we cannot directly obtain the ground truth table, we can utilize the generated table T_{tem} after TeM scheme as query to retrieve the relevant table from the training set:

$$T_t = \arg\max_{T_j \in \mathcal{T}} \text{BM25}(T_{tem}, T_j) \tag{9}$$

where T_t refers to the most similar table towards query T_{tem}. Following Eq. (6), we concatenate the text-table pair associated with T_t with the original input X for augmentation as follows:

$$X_a^t = \text{Template}(X_t, T_t, X) \tag{10}$$

Then, similar to Eq. (8), we generate the new table T_{tab} with X_a^t again as follows:

$$T_{tab,i} \sim P_G(T_{tab,i} \mid T_{tab,<i}; X_a^t). \tag{11}$$

where $T_{tab,i}$ refers to ith token of the generated table, and $T_{tab,<i}$ refers to the table sequence generated before ith token. The generated table T_{tab} is expected to contain more precise table structure since we choose the table matching for input augmentation. Hence, we can further perform table generation again by explicitly exploiting the table structure of T_{tab}. Specifically, we choose to extend the input augmentation X_a^t as follows:

$$X_a^{ts} = \text{Template}(X_t, T_t, X) + \text{“}\texttt{; guiding plan:}\text{”} + Struct(T_{tab}) \tag{12}$$

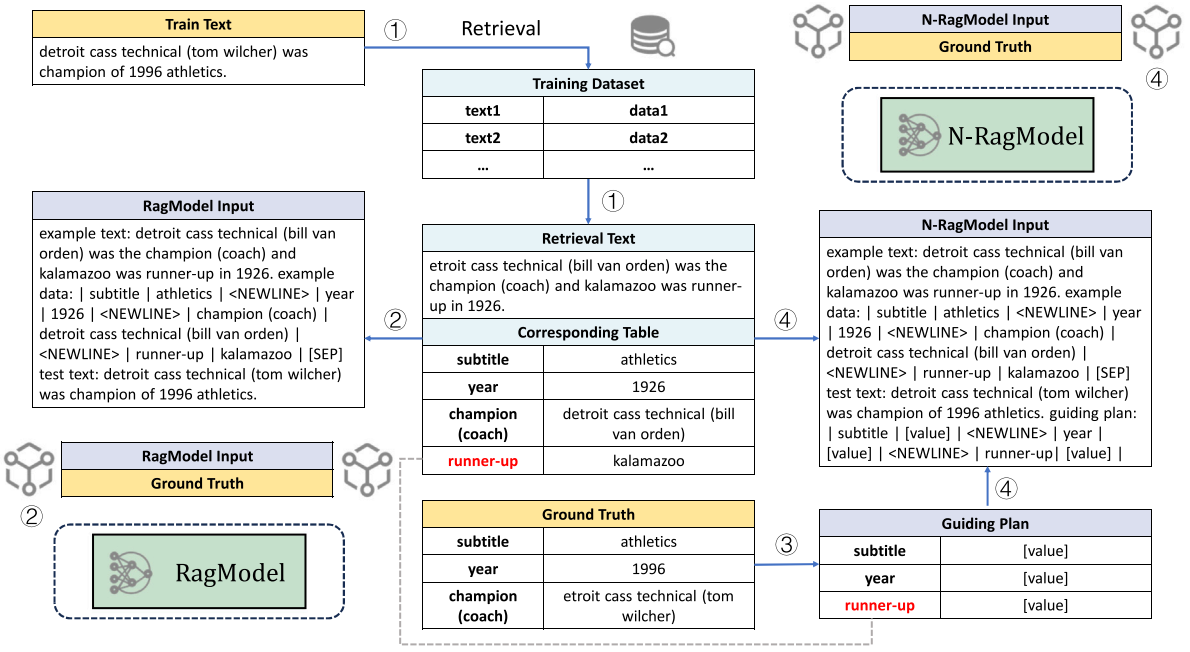

Fig. 4. Flowchart of Instances of the Training Method. We train the **RagModel** and the **N-RagModel**. Specifically, in step ①, for each **Train Text**, the most similar **Retrieval Text** is retrieved from the **Training Dataset**, along with its **Corresponding Table**. The combination of the original input and the retrieved text-table pair is then, in step ②, trained alongside the **Ground Truth** to develop the **RagModel**. For the **N-RagModel**, in step ③, by introducing noise from the **Corresponding Table** to the **Ground Truth** table structure, we generate a **Guiding Plan**. In step ④, it, together with the original input and the retrieved text-table pair, serves as the input and is trained against the **Ground Truth** to obtain the **N-RagModel**.

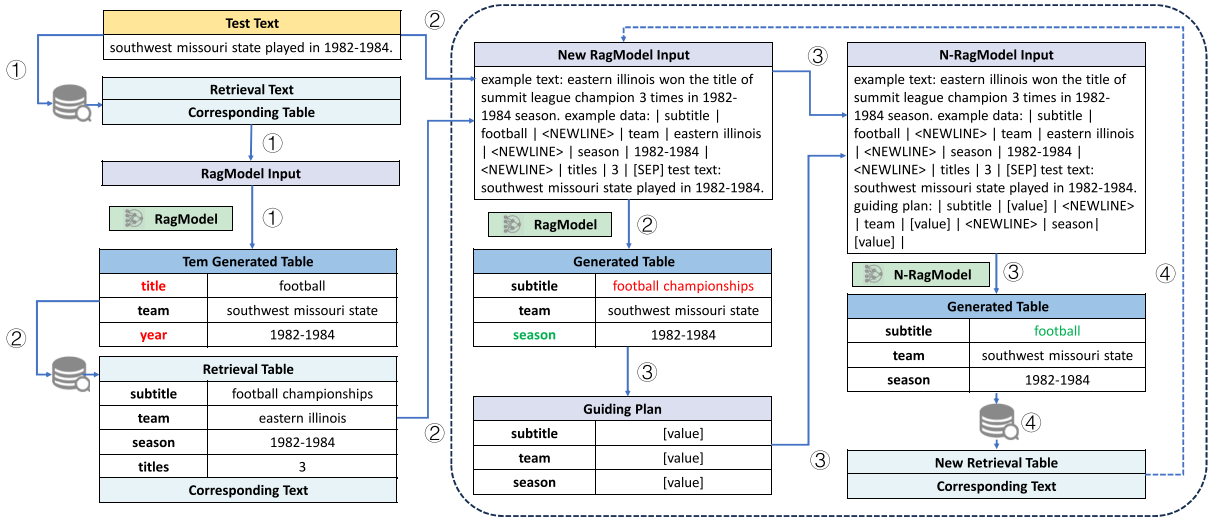

Fig. 5. Flowchart of Instances of the Testing Method. For each **Test Text**, firstly in step ①, we retrieve its text-table pair from the training set to generate the **RagModel Input**. This input is then processed by the pre-trained **RagModel** to produce a preliminary **Tem Generated Table**. Subsequently, in step ②, this generated table serves as a query to retrieve the table from the training set, yielding the **New RagModel Input**. From this input, another table is generated, and based on its structure, in step ③, a **Guiding Plan** is formulated to create the **N-RagModel Input**. This input is then processed by the **N-RagModel** for generation. Using this newly generated table, in step ④, we once again retrieve the table from the training set, obtaining the next round of retrieved text-table pair. This iterative process continues until the generated table no longer undergoes changes.

Algorithm 1 Training Method of Ragtable Algorithm

procedure TrainingRagtable($X, T, \mathcal{X}, \mathcal{T}$)
  $X_s \leftarrow \arg\max_{X_j \in \mathcal{X}} \text{BM25}(X, X_j)$ ▷ Perform text matching with BM25 algorithm
  $T_s \leftarrow \text{GetCorrespondingTable}(X_s, \mathcal{T})$ ▷ Get the corresponding table
  $X^a \leftarrow$ "`example text:`"$+ X_s +$ "`;example table:`"$+ T_s +$ "`;test text:`"$+ X$ ▷ Concatenate retrieval assistance using the template
  $RagModel \leftarrow \text{TrainSeq2SeqModel}(X^a, T)$ ▷ Train seq2seq model
  $X^{plan} \leftarrow \text{GenerateGuidingPlan}(T_s, T)$ ▷ Get guiding plan from ground truth
  $\hat{X}_a^{ts} \leftarrow X^a +$ ";guiding plan:"$+ X^{plan}$ ▷ Concatenate guiding plan using the template
  $N - RagModel \leftarrow \text{TrainSeq2SeqModel}(\hat{X}_a^{ts}, T)$ ▷ Train seq2seq model
  return $RagModel, N - RagModel$ ▷ Return the trained models
end procedure
function GenerateGuidingPlan(T_s, T)
  $headers_{retrieved} \leftarrow \text{ExtractHeaders}(T_s)$ ▷ Extract retrieval headers
  $headers_{gt} \leftarrow \text{ExtractHeaders}(T)$ ▷ Extract ground truth headers
  $headers_{noise\_candidates} \leftarrow \text{FilterHeaders}(headers_{retrieved}, headers_{gt})$ ▷ Filter out headers present in ground truth to get noise candidates
  for all headers in $headers_{gt}$ **do**
    if Random(0, 1) < 0.15 **and** $headers_{noise\_candidates} \neq \emptyset$ **then**
      $noise\_header \leftarrow \text{SelectRandomHeader}(headers_{noise\_candidates})$ ▷ Select a random noise header
      $headers_{gt}[index] \leftarrow noise\_header$ ▷ Replace the current header with the noise header
    end if
  end for
  $X^{plan} \leftarrow \text{IntegrateHeaders}(headers_{gt})$ ▷ Display (possibly noisy) headers in table structure
  return X^{plan} ▷ Return the guiding plan
end function

where function $Struct(T_{tab})$ returns the table header sub-sequence for T_{tab}. Then, we perform another round of table generation by using X_a^{ts} as the input augmentation again:

$$T_{f,i} \sim P_G(T_{f,i} \mid T_{f,<i}; X_a^{ts}). \tag{13}$$

where $T_{f,i}$ refers to ith token of the generated table T_f. Note that we can continue this iterative process by treating the generated table as the query for further data augmentation. When the generated table T_f is identical to the result from the previous iteration, we stop the iteration and obtain the final result T_{final}. Fig. 5 shows a running example of the whole process described above. The content retrieved by TeM may greatly deviate from the ground truth. Consequently, retrieving the relevant table information through the table matching can provide more relevant background knowledge from different perspective.

4.3. Noise-aware learning

Since the input for text matching and table matching consist of different kinds of auxiliary information, we choose to train individual generators respectively. Specifically, the generator for text matching is trained following Eq. (8). As to the training for iterative table generation, it is hard to guarantee that the data augmentation after iterative table matching process is noise free, which depends largely on the semantic alignment between the original input and the training set. In other words, it is inevitable that either generated table T_{tab} or the retrieved X_s and T_i could contain irrelevant semantic information. An important property for our Ragtable is the noise resistance.

Therefore, as to the $Struct(T_{tab})$ in Eq. (12) for table matching, we utilize the ground truth to produce it (as shown across Fig. 4). This allows the model to reassess whether the retrieved content in TeM is noisy and preventing the model from directly copying, which prompts it to reconsider and regenerate based on the exemplar and previous generation. Furthermore, for each header in it, with a 15% probability, we randomly select a header which is not included in the ground truth headers from the retrieval table and replace the original header with it. The purpose of this approach is to introduce real-world scenarios where the model needs to handle the augmented input, specifically generated table, that containing incorrect table headers. After obtaining this deteriorated guiding plan X^{plan}, we formalize the resultant $\hat{X}_a^{ts}$ and use it for model training as follows:

$$\hat{X}_a^{ts} = \text{Template}(X_s, T_s, X) + \text{“;guiding plan:”}+ X^{\text{plan}} \tag{14}$$

$$\mathcal{L}_{\text{LM}} = -\sum_{i=1}^{|T|} \log P_G\left(T_i \mid T_{<i}; \hat{X}_a^{ts}\right) \tag{15}$$

Algorithm 2 Testing Method of Ragtable Algorithm

procedure TestingRagtable($X, \mathcal{X}, \mathcal{T}$)
  $X_s \leftarrow \arg\max_{X_j \in \mathcal{X}} \text{BM25}(X, X_j)$ ▷ Perform text matching with BM25 algorithm on the text training set
  $T_s \leftarrow \text{GetCorrespondingTable}(X_s, \mathcal{T})$ ▷ Get the corresponding table
  $X^a \leftarrow$ "`example text:`"$+ X_s +$ "`; example table:`"$+ T_s +$ "`; test text:`"$+ X$ ▷ Concatenate retrieval assistance using the template
  $T_{tem} \leftarrow \text{RagModel}(X^a)$ ▷ Generate initial table using RagModel model
  $T_{query} \leftarrow T_{tem}$ ▷ Initialize query table
  while True **do** ▷ Iterate until no change in retrieved table
    $T_t \leftarrow \arg\max_{T_j \in \mathcal{T}} \text{BM25}(T_{query}, T_j)$ ▷ Perform table matching with BM25 algorithm on the table training set
    $X_t \leftarrow \text{GetCorrespondingText}(T_t, \mathcal{X})$ ▷ Get the corresponding text
    $X_a^t \leftarrow$ "`example text:`"$+ X_t +$ "`; example table:`"$+ T_t +$ "`; test text:`"$+ X$ ▷ Concatenate new retrieval assistance using the template
    $T_{tab} \leftarrow \text{RagModel}(X_a^t)$ ▷ Generate table using RagModel model
    $X_a^{ts} \leftarrow X_a^t +$ "; guiding plan:"$+ Struct(T_{tab})$ ▷ Get table header sub-sequence from generated table and concatenate it to X_a^t
    $T_f \leftarrow \text{N-RagModel}(X_a^{ts})$ ▷ Generate table using N-RagModel model
    if $T_f \neq T_{query}$ **then**
      $T_{query} \leftarrow T_f$ ▷ Update the query table for next iteration
    else
      $T_{final} \leftarrow T_f$ ▷ No change, set query table as final table
      break ▷ No change, launch iteration
    end if
  end while
  return T_{final} ▷ Return the final table
end procedure

Table 1
Statistics of E2E, WikiTableText and Rotowire datasets, including the number of instances in training, validation and test sets, the average number of BPE tokens per instance, and the average number of rows and columns per instance.

Dataset	Train	Valid	Test	# of tokens	# of rows	# of columns
WikiTableText	10.0k	1.3k	2.0k	19.59	4.26	2.00
E2E	42.1k	4.7k	4.7k	24.90	4.58	2.00
Rotowire	3.4k	727	728	351.05	7.26	8.75

When the model mistakenly uses an incorrect table header during the generation process, it has the opportunity to rethink and correct the error. At the same time, it also has the opportunity to reconsider what kind of table values should be generated for those headers, leading to more precise table generation.

5. Experiments

5.1. Setup

Datasets. Following the previous work Wu et al. (2022) and Li, Wang, et al. (2023), we conduct experiment on commonly-used datasets for text-to-table generation: E2E (Novikova et al., 2017), Rotowire (Wiseman et al., 2017) and WikiTableText (Bao et al., 2018). Table 1 presents the statistics of the datasets we used, including the number of instances in the training set, test set, and validation set, the average number of tokens per instance, and the average number of rows and columns per instance table.
Evaluation. We utilize the evaluation script provided by Wu et al. (2022) to assess the performance. The evaluation metric employed is the F1 measure where the precision refers to the percentage of correct cells out of all the generated cells, while recall represents the percentage of correct cells out of all the target cells. The correct cells can be identified in different ways. Here, we investigate the three options as follows:

- `Exact Match:` This metric requires that the generated table cells match the target table cells exactly, with no discrepancies.
- `Chrf Score:` This metric measures the similarity between two text strings based on character-level n-gram overlap, allowing for differences beyond exact character sequences.
- `BERT Score:` This metric leverages a pre-trained BERT model to gauge the semantic similarity between the generated and target texts by comparing their embeddings. It captures higher-level semantic information beyond simple character-level matching.

Implementation Details. We use the model provided by Wu et al. (2022) as the backbone and maintain consistency in hyper-parameters. The details on hyper-parameters is shown in Table 2.

Table 2
The training hyper-parameters on all datasets. We list the warmup updates (warmup), total updates (total upd.), learning rate (lr), and batch size (bsz, in terms of how many tokens per batch).

	warmup	total upd.	lr	bsz
WikiTableText	2000	8000	1e−04	4096
E2E	400	8000	1e−05	4096
Rotowire	400	8000	1e−04	4096

Table 3
The generation performance of our method and other baselines on the three datasets. **The best results** are highlighted in bold font, and the second best results are underlined.

Dataset	Model	Row header F1			Non-header cell F1		
		Exact	Chrf	BERT	Exact	Chrf	BERT
WikiTableText	NER	59.72	70.98	94.36	52.23	59.62	73.40
	Vanilla Seq2Seq	78.15	84.00	95.60	59.26	69.12	80.69
	HAD	78.16	83.96	95.68	59.14	68.95	80.74
	Seq2Seq-set	78.67	84.21	95.88	59.94	69.59	81.67
	Our Method	**83.59**	**87.73**	**96.79**	**72.14**	**77.75**	**86.41**
	LLM few-shot	45.79	58.67	90.28	29.68	41.17	65.11
E2E	NER	91.23	92.40	95.34	90.80	90.97	92.20
	Vanilla Seq2Seq	99.62	99.69	99.88	97.87	97.99	98.56
	HAD	99.63	99.69	99.88	97.88	98.00	98.57
	Seq2Seq-set	99.62	99.69	99.83	98.65	98.70	99.08
	Our Method	**99.81**	**99.84**	**99.93**	**98.89**	**98.97**	**99.29**
	LLM few-shot	91.70	93.42	98.89	70.23	80.12	86.27
Rotowire	Vanilla Seq2Seq	92.16	93.89	93.60	81.96	84.19	88.66
	HAD	92.31	94.00	93.71	82.53	84.74	88.97
	Seq2Seq-set	92.83	94.48	**96.43**	83.51	85.75	**90.93**
	Our Method	**94.01**	**95.25**	96.17	**83.60**	**85.83**	90.44

Baselines. We evaluate our model against the following representative methods:

- `NER`: This model is proposed by Wu et al. (2022). They first define the schemas based on the training data, then use an existing method of named entity extraction (NER) to extract information, and finally create tables based on the schemas and extracted information.
- `Vanilla Seq2Seq`: This model is proposed by Wu et al. (2022). It is a Transformer based Seq2Seq model that models the generation of a table as a sequence. We have already provided detailed description in Section 3.1.
- `HAD`: This model is proposed by Wu et al. (2022). It is a Seq2Seq variant, where the cell number of each table body row is limited to the same as that of table header and the cell correlation are further utilized for table generation.
- `Seq2Seq-set`: This model is proposed by Li, Wang, et al. (2023). During decoding, the table header is initially generated, followed by the parallel generation of table body columns, resulting in faster and more precise decoding.

Additionally, we also provide results obtained by using large language models~ Coyne and Dong. The process generates tables in JSON format using a prompt-chaining approach. To enhance performance, example inputs and outputs are randomly selected from the validation set and appended to the current inputs. Every generation is completed through an LLM call. It is important to mention that both our approach and Wu et al. (2022) and Li, Wang, et al. (2023) utilize the BART-base model, which is significantly smaller than the gpt-3.5-turbo-1106 model used in LLM few-shot.

5.2. Overall performance

Table 3 presents the experimental results on the three datasets. It is important to note that each result presented is the average of three runs to ensure robustness and reliability. Additionally, t-tests were conducted to assess the statistical significance of the improvements.

The results indicate that for the WikiTableText dataset, our RAGTABLE achieves significant improvements over the SOTA method in terms of generating table headers and cell values, as measured by exact match, chrf score, and BERTscore. Retrieval augmentation greatly improves the quality of table header generation. For example, in terms of the exact match, RAGTABLE reaches **6.25%** relative improvement against the best baseline (*i.e.,* Seq2Seq-set). Additionally, with the guidance of higher-quality table headers and background knowledge that is difficult to obtain directly from original textual information, for non-header cells, the relative improvement is over **20.35%** in terms of the same metric.

For the E2E dataset, even though previous experimental results have already achieved an impressive 98% accuracy, there is still a noticeable improvement when retrieval augmentation is applied. This demonstrates the powerful impact of retrieval strategies even

Table 4
Ablation study on WikiTableText dataset. "random" means training with our proposed approach, while using randomly sampled content as retrieval assistance for the test set; "w/" denotes the addition of components.

Method	Row header F1			Non-header cell F1		
	Exact	Chrf	BERT	Exact	Chrf	BERT
Vanilla Seq2Seq	78.15	84.00	95.60	59.26	69.12	80.69
HAD	78.16	83.96	95.68	59.14	68.95	80.74
Seq2Seq-set	78.67	84.21	95.88	59.94	69.59	81.67
TeM	82.39	87.02	96.61	69.56	75.80	85.56
TeM random	73.05	80.00	94.82	49.96	59.77	76.51
TeM w. NaL	82.66	87.12	96.69	70.24	76.24	85.65
TeM w. ITG	83.35	87.71	96.66	71.90	77.67	86.37
RAGTABLE	**83.59**	**87.73**	**96.79**	**72.14**	**77.75**	**86.41**

Table 5
Proportion of samples that have achieved consistent results after varying numbers of iterations, across different difficulty levels.

Iteration	Difficulty level			
	Very Hard	Hard	Medium	Easy
Iteration 1	77.8%	82%	80.6%	85.4%
Iteration 2	94.2%	97.6%	96.4%	97.6%
Iteration 3	96.6%	98.6%	97.8%	100%
Iteration 4	98.4%	100%	100%	100%
Iteration 5	100%	100%	100%	100%

in cases where the model performance is nearing saturation. The enhancement underscores the potential of retrieval augmentation to refine and push the boundaries of model accuracy further, even in highly optimized systems. Even with the Rotowire player dataset, where the retrieval texts are particularly lengthy, posing potential challenges for retrieval augmentation, it has still demonstrated remarkably significant improvements in the text-to-table generation task.

Furthermore, a notable observation is that results obtained by Coyne and Dong using gpt-3.5-turbo-1106 large language model is not as good as using a supervised learning approach.

In terms of time cost, during the training process, the time cost for training a single model in our solution is the same as that of the baseline model, making the overall training time cost only double that of the baseline model. For inference, the E2E and Rotowire datasets provided consistent results after just one iteration, while the WikiTableText dataset required three iterations to achieve consistent and accurate results. Due to the inherently short inference time, it does not impose a substantial overall time overhead. Considering both the time cost and the experimental results outlined in Table 3, our method achieves a substantial enhancement in the accuracy of generated output results, all while incurring minimal additional costs.

5.3. Ablation study

We evaluated the impact of each component of our RAGTABLE on the WikiTableText dataset. Specifically, we measured the effect of TeM, ITG and NaL by removing them sequentially. At the same time, we also present the results obtained by performing random retrieval augmentation, *i.e.,* randomly picking a text-table pair for table generation enhancement. The experimental results are presented in Table 4.

Here, it can be observed that TeM based retrieval has already yielded significant improvements. Even with this simple operation, notable performance gains can be achieved. On the other hand, performing trivial random retrieval augmentation is even worse than the vanilla Seq2Seq model. Furthermore, exploiting the iterative table matching enables additional performance gain compared to TeM. This suggests that precise retrieval for the relevant table information is crucial for effective generation enhancement. Beside, we can see that NaL also introduces positive impact towards better generation performance. These indicates that the model can correct errors in the initial generation and produce higher-quality results through further consideration.

Then, we also examine the retrieval quality of both Tem and ITG separately. Fig. 6 shows the Jaccard coefficient score between the headers of the retrieved table and the ground truth table headers for TeM and ITG separately. Note that we excludes the two common headers of "title" and "subtitle" from the score calculation. It can be observed that after adopting ITG, the Jaccard coefficient scores increases significantly, while the proportion of extremely low Jaccard scores (0–0.25) decreases notably, indicating a significant improvement in the quality of the retrieved table information.

Overall, the proposed RAGTABLE achieves the best performance in all three metrics, demonstrating the effectiveness of each design choice.

5.4. Iteration counts

Our method necessitates iteratively generating tables to retrieve new text-table pairs to aid in the generation process. To address concerns about the potential of excessive iterations using this method, we conducted an experiment to analyze the required iterations

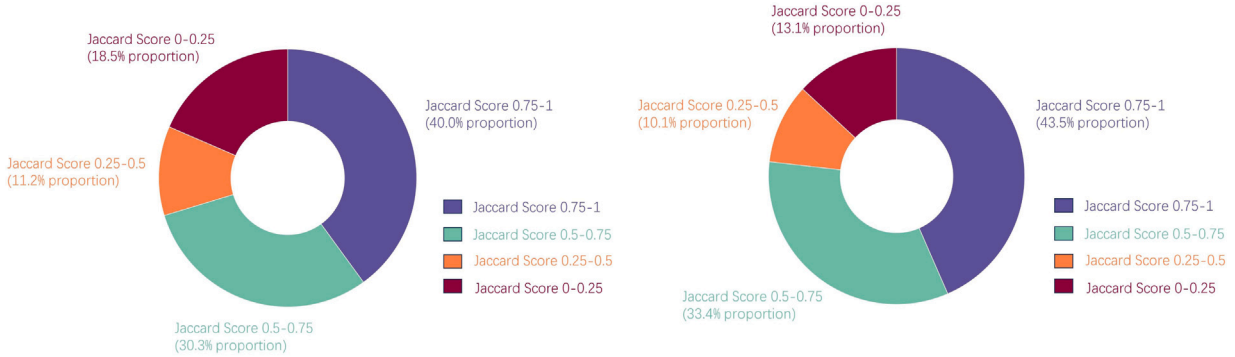

Fig. 6. The distribution of different Jaccard scores between the headers of the retrieved table and the ground truth table headers for TeM (left) and ITG (right) respectively.

Test Text
sonia gandhi was awarded as order of king leopold by the government of belgium in 2006.

Ground Truth	
title	sonia gandhi
subtitle	honours and recognition
year	2006
name	order of king leopold
awarding organisation	government of belgium

Retrieval Assistance	
title	sonia gandhi
subtitle	honours and recognition
name	honorary doctorate (literature)
awarding organisation	university of madras

Original Generation	
title	india gandhi
subtitle	awards and honours
country	belgium
award	order of king leopold
year	2006

Our Method	
title	sonia gandhi
subtitle	honours and recognition
year	2006
name	order of king leopold
awarding organisation	government of belgium

Fig. 7. Case study of how does our method improves the effectiveness of text-to-table generation. Where **red** represents errors generated by the original model, **blue** represents assistance provided by the retrieval table, and **green** represents the corresponding correct content generated by RAGTABLE. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

for different levels of difficulty. We categorize the difficulty levels based on the performance of the vanilla seq2seq model on the dataset, specifically by classifying the test set into four equal difficulty levels based on the results derived from the BERT Score evaluation matrix for the generated tables. Table 5 showcases the proportion of samples from the WikiTableText dataset that have attained consistent results after undergoing varying numbers of iterations, categorized across different difficulty levels. We observed that after just one iteration, 20% of the sample tables had not yet achieved the optimal and required further iterations for refinement. However, after the second iteration, the majority of samples had produced consistent results, and almost all had stabilized after three iterations. For simpler samples, only 2 iterations were sufficient, while for the most complex ones, further iterations were required, indicating no significant overhead.

5.5. Case study

As depicted in Fig. 7, we illustrate how our proposed RAGTABLE outperforms HAD model through a random example. Specifically, the "title" header should correspond to "sonia gandhi," the accurate answer. However, HAD model incorrectly generate "| title | india gandhi |" instead. Notably, "india gandhi" does not occur in the training set, and the original text clearly mentions "sonia gandhi". It is conceivable that the BART-base model, influenced by its knowledge of Sonia Gandhi's Indian origin, incorrectly generates "india

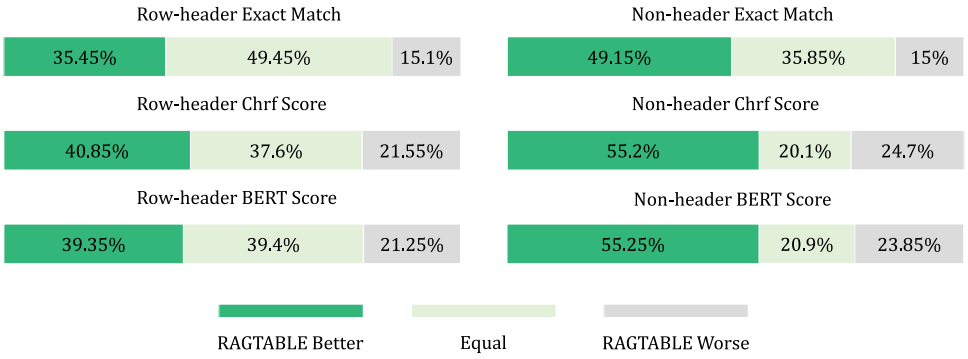

Fig. 8. Proportion of test samples where RAGTABLE outperforms the base model, performs equally, and underperforms on row headers and non-header cells across three evaluation metrics.

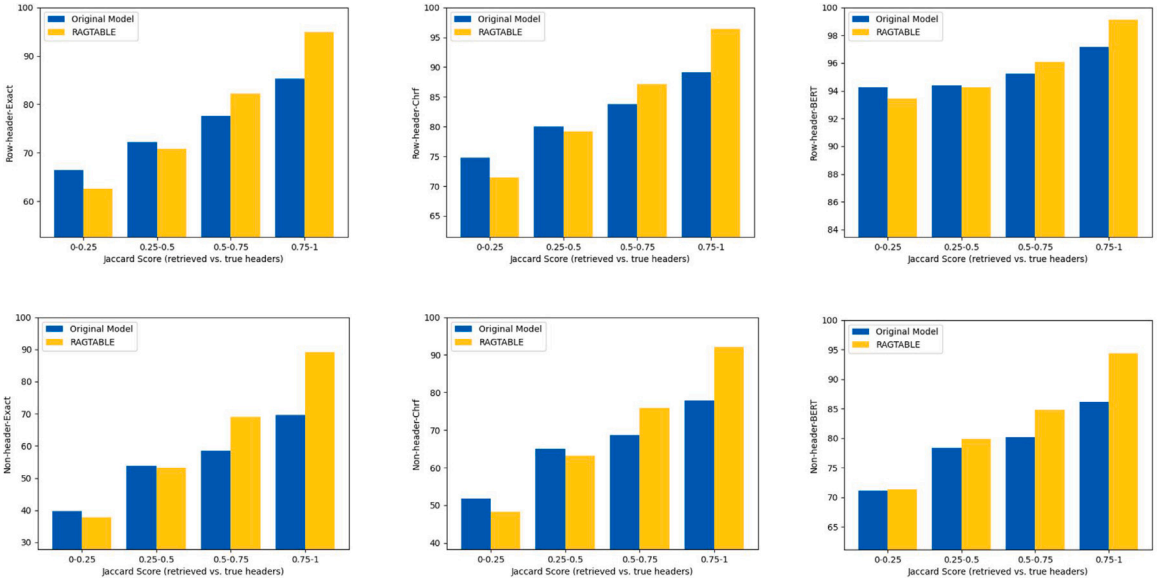

Fig. 9. Results with varied quality levels of retrieval augmentation. The horizontal axis represents the Jaccard score between the headers of retrieved table and the ground truth table headers, while the vertical axis represents the average F1 score in terms of three evaluation matrices for the generated results within each range.

gandhi". In contrast, our approach leverages retrieval augmentation, which reveals "sonia gandhi". Moreover, our approach can disregard irrelevant details, such as "honorary doctorate (literature)" from the retrieved relevant background information, ensuring a more precise generation result.

Additionally, the retrieved table includes many trustworthy header suggestions to guide the model on what to say. As we can see, HAD model does not describe "awarding organization," but this is suggested by the retrieved exemplar and is indeed worth describing. It is obvious that our RAGTABLE accepts the suggestion and generates a table mentioning "awarding organization," making the result more aligned with people's expectation.

5.6. Performance proportions

Fig. 8 illustrates the proportions of test samples that our approach outperformes, matches, and underperformes the base model across three evaluation metrics on the WikiTableText, specifically focusing on row headers and non-header cells. It is evident that, regardless of whether the header or the cell values, RAGTABLE significantly outperforms the base model as twice as the counterpart. This trend is particularly prominent in the generation of non-header content, clearly demonstrating the effectiveness of our solution. Nevertheless, it is also observed that there is indeed a certain proportion of generated samples that fall below the baseline model's performance. The possible reason would be the poor retrieval quality, which we leave as future work.

Table 6
Results of different ways to provide retrieval assistance. "(table)" represents providing only the retrieval table, "(header/format)" represents providing only the retrieval table with its values removed, and "(header)" represents providing only the headers of the retrieval table.

Model	Row header F1			Non-header cell F1		
	Exact	Chrf	BERT	Exact	Chrf	BERT
HAD	78.16	83.96	95.68	59.14	68.95	80.74
Seq2Seq-set	78.67	84.21	95.88	59.94	69.59	81.67
TeM w. table	82.26	86.97	96.48	69.46	75.75	85.14
TeM w. header/format	82.36	87.01	96.41	65.82	73.30	83.23
TeM w. header	81.43	86.37	96.25	64.08	71.93	82.65
TeM	**82.39**	**87.02**	**96.61**	**69.56**	**75.80**	**85.56**

Table 7
RAGTABLE with Oracle.

Method	Row header F1			Non-header cell F1		
	Exact	Chrf	BERT	Exact	Chrf	BERT
Seq2Seq-set	78.67	84.21	95.88	59.94	69.59	81.67
RAGTABLE	83.59	87.73	96.79	72.14	77.75	86.41
RAGTABLE Oracle1	**88.97**	**92.18**	**97.83**	**82.58**	**86.94**	**91.35**
RAGTABLE Oracle2	86.80	90.25	97.27	75.00	80.50	87.53
RAGTABLE Oracle3	86.25	89.71	97.24	74.58	80.00	87.43

5.7. The performance comparison with varied retrieval quality

As shown in Fig. 9, we illustrate the results obtained under each retrieval quality in terms of Jaccard similarity over the table headers. Notably, when the retrieved content is entirely unrelated to the ground truth, it does have a detrimental impact on the model's outcomes.[1] However, once the Jaccard similarity over table headers exceeds 0.25 (*i.e.,* a rather low level of relevance), the negative effect is almost nonexistent. As the relevance surpasses 0.5, despite the considerable difference between the retrieved table and the ground truth, RAGTABLE introduce remarkable performance gain over the base model. Further, when the relevance exceeds 0.75, the relative performance gain is over 30%. The experimental results show that, when the quality of retrieval augmentation is relatively low, RAGTABLE does not significantly underperform compared to the base model. Also, when the quality of retrieval augmentation is fruitful, RAGTABLE can deliver significant performance improvement.

5.8. Content provision experiment

To obtain further insight towards why our RAGTABLE can harness the complementary semantic signals from the retrieval augmentation, specifically, we examine what kind of information should we provide to the model on the basis of TeM. Firstly, we remove the retrieved textual description and only provide the corresponding table instance (namely TeM w. table). Experimental results, as shown in Table 6, indicate that removing the textual description does not lead to a significant drop in performance. It can be seen that unlike in-context learning, we do not have to show the model the complete demonstrations.

Subsequently, we strip the cell values from the retrieved table (namely TeM w. header/format). Experimental results demonstrate that this absence causes a drop in cell value generation. This performance deterioration indicates that the model needs to refer to the content of the retrieved table. However, it still outperforms the base model. This could be attributed to the improved accuracy of the headers, resulting in better guidance for generating table values.

Furthermore, we consider whether maintaining the table structure is necessary (namely TeM w. header). We directly extract the headers from the retrieved table and concatenate them with the original input in the format of 'possible header: $header_1, header_2, ..., header_n$'. Experimental results, as shown in Table 6, indicate that it is not as effective as presenting the complete table, suggesting that maintaining the table structure is beneficial since this structured information contains more semantic guidance. Nevertheless, our approach still outperforms the base model by a significant margin. This demonstrates that the effectiveness of our approach mainly lies in the model's exposure to specific in a structured manner.

5.9. RAGTABLE *with oracle*

We further take ground truth as an oracle to demonstrate the power of retrieval augmentation for text-to-table generation, while also discussing potential future directions. In detail, we investigate the performance with the following three types of oracles:

[1] The retrieval augmentation can be controlled by the retrieval quality estimation. We will leave it as future work.

Table 8
RAGTABLE with advanced retriever.

Method	Row header F1			Non-header cell F1		
	Exact	Chrf	BERT	Exact	Chrf	BERT
Seq2Seq-set	78.67	84.21	95.88	59.94	69.59	81.67
RAGTABLE	83.59	87.73	96.79	72.14	77.75	86.41
RAGTABLE w. Llama-2-7b-hf	79.10	84.75	95.97	62.55	70.30	82.18
RAGTABLE w. e5-Mistral-7b-instruct	84.02	88.15	96.85	72.54	78.33	86.90
RAGTABLE w. gte-Qwen2-7B-instruct	80.65	85.73	96.19	65.96	72.92	83.73
RAGTABLE w. bge-multilingual-Gemma2	**84.41**	**88.42**	**96.90**	**73.55**	**78.98**	**87.19**

- Oracle1: We use the ground truth table as the query to retrieve relevant samples.
- Oracle2: For each retrieved text-table pair, we consider the header-value pair of which the header does not appear in the ground truth as noise and remove it.
- Oracle3: We utilize the oracle to externally filter noise by using ground truth to measure the relevance generated by RAGTABLE and the base model and select the better one as the final result. Here, we choose row-header exact match for relevance measure.

These experimental results as shown in Table 7, demonstrate the significant value of retrieval augmentation in the context of text-to-table generation. The use of oracles significantly improves the model's performance, highlighting the broad potential for future improvements through retrieval-based approaches.

Each oracle experiment provides insights into how retrieval can be further optimized. Oracle1 achieves outstanding results by using the ground truth table as the retrieval query, demonstrating that precise retrieval significantly enhances the quality of generated tables. A promising future direction could involve developing more sophisticated strategies to approximate this ideal query formulation. Oracle2 shows substantial performance improvement by removing irrelevant header-value pairs, illustrating the power of effective noise reduction. Future work could explore automated noise filtering techniques or develop models that better differentiate between relevant and irrelevant retrieval content, making the system more robust. Oracle3, which leverages an oracle to externally filter noise and select the most relevant output, points to future possibilities in using filtering methods or multi-model comparison techniques to enhance result quality.

Overall, the results suggest that optimizing the retrieval process is a direction worth further exploring in the near future.

5.10. RAGTABLE *with advanced retriever*

In the interest of maintaining fairness in our comparisons, we refrained from introducing new models or additional training data in our earlier chapters, simply relying on the sparse retrieval method BM25. Despite this constraint, our method still demonstrated remarkable superiority over the baseline approach, showcasing its inherent strengths and potential. Recognizing the potential for further enhancement, we decided to incorporate more advanced retrieval models (mainly dense retrieval models). We anticipate that this shift will elevate the retrieval quality and then amplify the improvements observed in our initial results. To this end, we conducted a thorough evaluation of various dense retrieval tools and selected the following models based on their proven capacity.

- `Llama-2-7b-hf`[2]: Llama-2-7b-hf is an auto-regressive language model built on the basis of transformer architecture. This version uses supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) to align to human preferences for helpfulness and safety.
- `E5-Mistral-7b-instruct`[3]: E5-Mistral-7b-instruct is a powerful language model that excels in text embedding tasks, particularly in English. It is well-suited for tasks like passage ranking and retrieval. This model is initialized from Mistral-7B-v0.1[4] and fine-tuned on a wide range of datasets.
- `Gte-Qwen2-7B-instruct`[5]: Gte-Qwen2-7B-instruct is the latest model in the gte (General Text Embedding) model family that ranks No. 1 in both English and Chinese evaluations on the Massive Text Embedding Benchmark MTEB benchmark (as of June 16, 2024).
- `Bge-multilingual-Gemma2`[6]: Bge-multilingual-Gemma2 is a LLM-based multilingual embedding model. It is trained on a diverse range of languages and tasks based on google/gemma-2-9b.[7] It achieves excellent performance on major evaluations, including MTEB, C-MTEB and AIR-Bench.

[2] https://huggingface.co/meta-llama/Llama-2-7b-hf
[3] https://huggingface.co/intfloat/e5-mistral-7b-instruct
[4] https://huggingface.co/mistralai/Mistral-7B-v0.1
[5] https://huggingface.co/Alibaba-NLP/gte-Qwen2-7B-instruct
[6] https://huggingface.co/BAAI/bge-multilingual-gemma2
[7] https://huggingface.co/google/gemma-2-9b

Text	
in 2005, san diego won the season champion and game champion in pioneer football league season held at san diego.	
Ground Truth	
title	pioneer football league
subtitle	league titles by school
season	2005
season champion	san diego
game champion	san diego
location	san diego

Vanilla Generated Table	
title	pioneer football league
subtitle	league titles by school
season	2005
season champion	san diego
game champion	san diego

Retrieval Text	
uc san diego was the champion of the western collegiate roller hockey league in 2009	
Corresponding Table	
title	western collegiate roller hockey league
subtitle	champions
year	2009
champion	uc san diego

Corresponding Text	
in 1980 denver broncos season, broncos defeated san diego chargers which was held at san diego stadium	
Retrieval Table(by Tem Generated Table)	
title	1980 denver broncos season
subtitle	season champions
winner	broncos
opponent	at san diego chargers
game site	san diego stadium

Tem Generated Table	
title	pioneer football league
subtitle	season champions
year	2005
season champion	san diego
game champion	san diego

Final Generated Table	
title	2005 pioneer football league season
subtitle	season champions
winner	san diego
game champion	san diego

Fig. 10. An error instance of RAGTABLE. On the left are the ground truth and the original model's generated result. In the middle are the retrieval assistance of TeM and ITG. On the right are the generated result of TeM and the final generated result. We will address such error in future work.

These models provide more precise feature representations for sentences, enabling more accurate retrieval. By leveraging these tools, we aim to refine our retrieval-assisted content even further, ensuring that it aligns better with the ground truth and meets users' expectations.

Table 8 presents the experimental results on the WikiTableText dataset. We have observed that utilizing bge-multilingual-Gemma2 as the embedder for dense retrieval has yielded substantial improvements compared to our previous approach. Therefore, when resources permit, the continuing performance gain can be obtained by adopting more powerful retrieval models.

5.11. Error analysis

Fig. 10 shows an erroneous example of our RAGTABLE. The upper left part shows the input of the test set and the ground truth, and the lower part shows the output result of the base model. It can be found that almost all the table headers and values are correct. The upper middle part shows the retrieved content of TeM. It can be found that although the text is similar, the table is quite different from the ground truth table. Under the assistance of such noisy retrieved content, the preliminary table generated by TeM, as shown in the upper right part of the figure, is not ideal. With it as the query, as shown in the lower middle part of the figure, the retrieved table is also quite different from the ground truth table. Furthermore, NaL hardly realizes that "winner" is noise because it also has a certain degree of rationality. Due to the aforementioned issues, the final generation result, as shown in the lower right part of the figure, has a big deviation. We will focus on solving such problems in our future work.

6. Limitations

A notable limitation of our work lies in its reliance on high-quality retrieval for substantial improvements. However, as depicted in Fig. 9, the final output is only adversely affected when the retrieved table information bears almost no relevance to the actual results. When the retrieval quality is relatively high, our approach yields considerable enhancements. Additionally, another constraint is that as the number of samples becomes extremely large, the computation burden becomes non-negligible, contributing to the overall cost.

7. Conclusion

In this paper, we recognize the information scarcity is a very important challenge for text-to-table generation, particularly in determining what and how to generate. To address this, we introduce a retrieval augmentation method. the proposed RAGTABLE retrieves the relevant example to guide the model in two ways: what to include in the table headers and how to generate table values that cannot be directly copied from the text. Our experiments reveal that our method significantly outperforms existing solutions.

CRediT authorship contribution statement

Jiahua Zhang: Writing – original draft, Methodology, Investigation, Conceptualization. **Meijuan Tan:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Conceptualization. **Jing Zhang:** Writing – review & editing, Writing – original draft, Methodology, Conceptualization. **Xiaolu Zhang:** Conceptualization. **Jun Zhou:** Methodology, Conceptualization. **Chenliang Li:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Conceptualization.

Ethics statement

We prioritize ethical considerations and adhere to ethical and responsible practices throughout our research methods. Notably, our study does not involve any data collection or dissemination, thus eliminating any privacy concerns. The datasets we employ in this paper are publicly accessible and have been widely embraced by researchers for assessing the proficiency of text-to-table generation. Additionally, we are committed to reporting the findings and conclusions of this study with utmost precision and objectivity.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Agrawal, S., Zhou, C., Lewis, M., Zettlemoyer, L., & Ghazvininejad, M. (2022). In-context examples selection for machine translation. arXiv preprint arXiv:2212.02437.

Bao, J., Tang, D., Duan, N., Yan, Z., Lv, Y., Zhou, M., & Zhao, T. (2018). Table-to-text: Describing table region with natural language. *vol. 32*, In *Proceedings of the AAAI conference on artificial intelligence*. (1).

Cai, D., Wang, Y., Bi, V., Tu, Z., Liu, X., Lam, W., & Shi, S. (2018). Skeleton-to-response: Dialogue generation guided by retrieval memory. arXiv preprint arXiv:1809.05296.

Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. In R. Barzilay, & M.-Y. Kan (Eds.), *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: long papers)* (pp. 1870–1879). Vancouver, Canada: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P17-1171, URL https://aclanthology.org/P17-1171.

Chen, L., & Moschitti, A. (2018). Learning to progressively recognize new named entities with sequence to sequence models. In E. M. Bender, L. Derczynski, & P. Isabelle (Eds.), *Proceedings of the 27th international conference on computational linguistics* (pp. 2181–2191). Santa Fe, New Mexico, USA: Association for Computational Linguistics, URL https://aclanthology.org/C18-1185.

Cheng, X., Gao, S., Liu, L., Zhao, D., & Yan, R. (2022). Neural machine translation with contrastive translation memories. arXiv preprint arXiv:2212.03140.

Cheng, X., Lin, Y., Chen, X., Zhao, D., & Yan, R. (2023). Decouple knowledge from paramters for plug-and-play language modeling. arXiv preprint arXiv:2305.11564.

Clive, J., Cao, K., & Rei, M. (2021). Control prefixes for parameter-efficient text generation. arXiv preprint arXiv:2110.08329.

Coyne, S., & Dong, Y. Large Language Models as Generalizable Text-to-Table Systems.

Feng, J., Xu, G., Wang, Q., Yang, Y., & Huang, L. (2024). Note the hierarchy: Taxonomy-guided prototype for few-shot named entity recognition. *Information Processing & Management*, *61*(1), Article 103557.

Geng, R., Chen, Y., Huang, R., Qin, Y., & Zheng, Q. (2023). Planarized sentence representation for nested named entity recognition. *Information Processing & Management*, *60*(4), Article 103352.

Gong, L., Crego, J. M., & Senellart, J. (2019). Enhanced transformer model for data-to-text generation. In *Proceedings of the 3rd workshop on neural generation and translation* (pp. 148–156).

Gong, H., Sun, Y., Feng, X., Qin, B., Bi, W., Liu, X., & Liu, T. (2020). Tablegpt: Few-shot table-to-text generation with table structure reconstruction and content matching. In *Proceedings of the 28th international conference on computational linguistics* (pp. 1978–1988).

Gu, J., Wang, Y., Cho, K., & Li, V. O. (2018). Search engine guided neural machine translation. *vol. 32*, In *Proceedings of the AAAI conference on artificial intelligence*. (1).

Harkous, H., Groves, I., & Saffari, A. (2020). Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. arXiv preprint arXiv:2004.06577.

Hu, Y., Chen, Y., Huang, R., Qin, Y., & Zheng, Q. (2024). A hierarchical convolutional model for biomedical relation extraction. *Information Processing & Management*, *61*(1), Article 103560.

Huang, D., Cui, L., Yang, S., Bao, G., Wang, K., Xie, J., & Zhang, Y. (2020). What have we achieved on text summarization?. arXiv preprint arXiv:2010.04529.

Izacard, G., & Grave, E. (2021). Leveraging passage retrieval with generative models for open domain question answering. In P. Merlo, J. Tiedemann, & R. Tsarfaty (Eds.), *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: Main volume* (pp. 874–880). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.eacl-main.74, URL https://aclanthology.org/2021.eacl-main.74.

Jian, Z., Li, J., Wu, Q., & Yao, J. (2024). Retrieval contrastive learning for aspect-level sentiment classification. *Information Processing & Management*, *61*(1), Article 103539.

Jiang, N., Chen, J., Zhou, R. G., Wu, C., Chen, H., Zheng, J., & Wan, T. (2020). PAN: Pipeline assisted neural networks model for data-to-text generation in social internet of things. *Information Sciences*, *530*, 167–179.

Khandelwal, U., Levy, O., Jurafsky, D., Zettlemoyer, L., & Lewis, M. (2019). Generalization through memorization: Nearest neighbor language models. arXiv preprint arXiv:1911.00172.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In D. Jurafsky, J. Chai, N. Schluter, & J. Tetreault (Eds.), *Proceedings of the 58th annual meeting of the association for computational linguistics*.

Li, Z., Chen, W., Li, S., Wang, H., Qian, J., & Yan, X. (2022). Controllable dialogue simulation with in-context learning. arXiv preprint arXiv:2210.04185.

Li, T., Fang, L., Lou, J. G., & Li, Z. (2021). TWT: Table with written text for controlled data-to-text generation. In *Findings of the association for computational linguistics: EMNLP 2021* (pp. 1244–1254).

Li, S., Ji, H., & Han, J. (2021). Document-level event argument extraction by conditional generation. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, & Y. Zhou (Eds.), *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: Human language technologies* (pp. 894–908). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.naacl-main.69, URL https://aclanthology.org/2021.naacl-main.69.

Li, L., Ma, C., Yue, Y., & Hu, D. (2021). Improving encoder by auxiliary supervision tasks for table-to-text generation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)* (pp. 5979–5989).

Li, T., Wang, Z., Shao, L., Zheng, X., Wang, X., & Su, J. (2023). A sequence-to-sequence&set model for text-to-table generation. In *Findings of the association for computational linguistics: ACL 2023* (pp. 5358–5370).

Li, Y., Yang, N., Wang, L., Wei, F., & Li, W. (2023). Generative retrieval for conversational question answering. *Information Processing & Management*, *60*(5), Article 103475.

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., & Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, *8*, 726–742.

Liu, T., Luo, F., Yang, P., Wu, W., Chang, B., & Sui, Z. (2019). Towards comprehensive description generation from factual attribute-value tables. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 5985–5996).

Liu, P., Wang, G., Li, H., Liu, J., Ren, Y., Zhu, H., & Sun, L. (2024). Multi-granularity cross-modal representation learning for named entity recognition on social media. *Information Processing & Management*, *61*(1), Article 103546.

Lu, Y., Lin, H., Xu, J., Han, X., Tang, J., Li, A., Sun, L., Liao, M., & Chen, S. (2021). Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)* (pp. 2795–2806). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.acl-long.217, URL https://aclanthology.org/2021.acl-long.217.

Lu, Y., Liu, Q., Dai, D., Xiao, X., Lin, H., Han, X., Sun, L., & Wu, H. (2022). Unified structure generation for universal information extraction. arXiv preprint arXiv:2203.12277.

Moryossef, A., Goldberg, Y., & Dagan, I. (2019). Step-by-step: Separating planning from realization in neural data-to-text generation. arXiv preprint arXiv:1904.03396.

Nayak, T., & Ng, H. T. (2020). Effective modeling of encoder-decoder architecture for joint entity and relation extraction. *vol. 34*, In *Proceedings of the AAAI conference on artificial intelligence* (05), (pp. 8528–8535).

Nishino, T., Ozaki, R., Momoki, Y., Taniguchi, T., Kano, R., Nakano, N., Tagawa, Y., Taniguchi, M., Ohkuma, T., & Nakamura, K. (2020). Reinforcement learning with imbalanced dataset for data-to-text medical report generation. In *Findings of the association for computational linguistics: EMNLP 2020* (pp. 2223–2236).

Novikova, J., Dušek, O., & Rieser, V. (2017). The E2E dataset: New challenges for end-to-end generation. arXiv preprint arXiv:1706.09254.

Paolini, G., Athiwaratkun, B., Krone, J., Ma, J., Achille, A., Anubhai, R., Santos, C. N. d., Xiang, B., & Soatto, S. (2021). Structured prediction as translation between augmented natural languages. arXiv preprint arXiv:2101.05779.

Puduppully, R., Dong, L., & Lapata, M. (2019). Data-to-text generation with content selection and planning. *vol. 33*, In *Proceedings of the AAAI conference on artificial intelligence* (01), (pp. 6908–6915).

Qian, J., Dong, L., Shen, Y., Wei, F., & Chen, W. (2022). Controllable natural language generation with contrastive prefixes. arXiv preprint arXiv:2202.13257.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, *21*(140), 1–67.

Sha, L., Mou, L., Liu, T., Poupart, P., Li, S., Chang, B., & Sui, Z. (2018). Order-planning neural text generation from structured data. *vol. 32*, In *Proceedings of the AAAI conference on artificial intelligence*. (1).

Shi, W., Min, S., Yasunaga, M., Seo, M., James, R., Lewis, M., Zettlemoyer, L., & Yih, W.-t. (2023). Replug: Retrieval-augmented black-box language models. arXiv preprint arXiv:2301.12652.

Song, Y., Yan, R., Li, X., Zhao, D., & Zhang, M. (2016). Two are better than one: An ensemble of retrieval-and generation-based dialog systems. arXiv preprint arXiv:1610.07149.

Su, Y., Vandyke, D., Wang, S., Fang, Y., & Collier, N. (2021). Plan-then-generate: Controlled data-to-text generation via planning. arXiv preprint arXiv:2108.13740.

Upadhyay, A., Massie, S., Singh, R. K., Gupta, G., & Ojha, M. (2021). A case-based approach to data-to-text generation. In *Case-based reasoning research and development: 29th international conference, ICCBR 2021, salamanca, Spain, September 13–16, 2021, proceedings 29* (pp. 232–247). Springer.

Wiseman, S., Shieber, S. M., & Rush, A. M. (2017). Challenges in data-to-document generation. arXiv preprint arXiv:1707.08052.

Wiseman, S., Shieber, S. M., & Rush, A. M. (2018). Learning neural templates for text generation. arXiv preprint arXiv:1808.10122.

Wu, X., Zhang, J., & Li, H. (2022). Text-to-table: A new way of information extraction. In S. Muresan, P. Nakov, & A. Villavicencio (Eds.), *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*.

Xing, X., & Wan, X. (2021). Structure-aware pre-training for table-to-text generation. In *Findings of the association for computational linguistics: ACL-IJCNLP 2021* (pp. 2273–2278).

Yan, H., Gui, T., Dai, J., Guo, Q., Zhang, Z., & Qiu, X. (2021). A unified generative framework for various NER subtasks. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)* (pp. 5808–5822). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.acl-long.451, URL https://aclanthology.org/2021.acl-long.451.

Ye, R., Shi, W., Zhou, H., Wei, Z., & Li, L. (2020). Variational template machine for data-to-text generation. arXiv preprint arXiv:2002.01127.

Zeng, X., Zeng, D., He, S., Liu, K., & Zhao, J. (2018). Extracting relational facts by an end-to-end neural model with copy mechanism. In I. Gurevych, & Y. Miyao (Eds.), *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: long papers)* (pp. 506–514). Melbourne, Australia: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P18-1047, URL https://aclanthology.org/P18-1047.

Zeng, D., Zhu, J., Chen, H., Dai, J., & Jiang, L. (2024). Document-level denoising relation extraction with false-negative mining and reinforced positive-class knowledge distillation. *Information Processing & Management*, *61*(1), Article 103533.

Zhao, Q., Gao, T., & Guo, N. (2023). Tsvfn: Two-stage visual fusion network for multimodal relation extraction. *Information Processing & Management*, *60*(3), Article 103264.

Zhao, J., Zhan, Z., Li, T., Li, R., Hu, C., Wang, S., & Zhang, Y. (2021). Generative adversarial network for table-to-text generation. *Neurocomputing*, *452*, 28–36.

Zhao, Y., Zhang, H., Si, S., Nan, L., Tang, X., & Cohan, A. (2023). Investigating table-to-text generation capabilities of LLMs in real-world information seeking scenarios. arXiv preprint arXiv:2305.14987.