Divide Rows and Conquer Cells: Towards Structure Recognition for Large Tables

Huawen Shen[1,2], **Xiang Gao**[1], **Jin Wei**[4,1], **Liang Qiao**[3], **Yu Zhou**[1,2*], **Qiang Li**[1,2] and **Zhanzhan Cheng**[3]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
[3]Hikvision Research Institute, China
[4]School of Information and Communication Engineering, Communication University of China
{shenhuawen, zhouyu, liqiang}@iie.ac.cn, gaoxiang181@mails.ucas.ac.cn, weijin@cuc.edu.cn,
{qiaoliang6, chengzhanzhan}@hikvision.com,

Abstract

Recent advanced Table Structure Recognition (TSR) models adopt image-to-text solutions to parse table structure. These methods can be formulated as image caption problem, *i.e.,* input a single-table image and output table structure description in a specific text format, *e.g.*, HTML. With the impressive success of Transformer in text generation tasks, these methods use Transformer architecture to predict HTML table text in an autoregressive manner. However, tables always emerge with a large variety of shapes and sizes. Autoregressive models usually suffer from the error accumulation problem as the length of predicted text increases, which results in unsatisfactory performance for large tables. In this paper, we propose a novel image-to-text based TSR method that relieves error accumulation problems and improves performance noticeably. At the core of our method is a cascaded two-step decoder architecture with the former decoder predicting HTML table row tags non-autoregressively and the latter predicting HTML table cell tags of each row in a semi-autoregressive manner. Compared with existing methods that predict HTML text autoregressively, the superiority of our row-to-cell progressive table parsing is twofold: (i) it generates an HTML tag sequence with a vertical-and-horizontal two-step 'scanning', which better fits the inherent 2D structure of image data, (ii) it performs substantially better for large tables (long sequence prediction) since it alleviates error accumulation problem specific to autoregressive models. Extensive experiments demonstrate that our method achieves competitive performance on three public benchmarks.

1 Introduction

Extracting information from text images has become an increasing hot topic, and much work has been done in scene text detection [Qin *et al.*, 2021], text recognition [Qiao *et al.*, 2020], text understanding [Zeng *et al.*, 2023], document analysis [Xu *et al.*, 2020], and so on. Table is one of the most common forms of structured data that facilitates information management and retrieval. Similarly, in real-life scenarios, a huge number of tables are not born-digital and often stored in unstructured image formats, *e.g.*, photos, scanned documents. These unstructured table images are hard to be analyzed and processed by computers directly, bringing difficulties to automatic procedures like archiving and retrieving. Therefore, table structure recognition (TSR), which aims at parsing logical table structure in images, plays a critical role in table understanding.

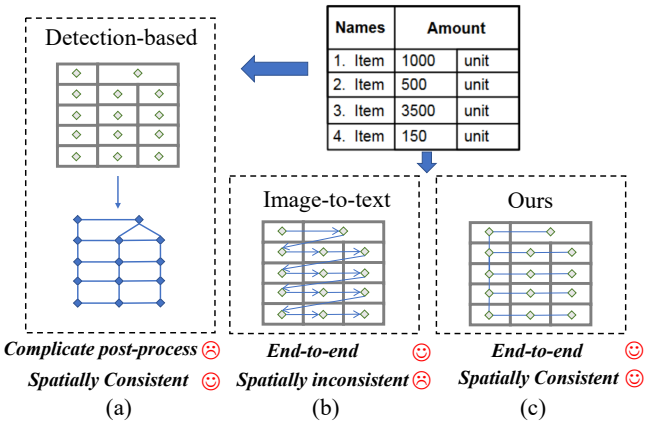


Figure 1: The detection-based methods need complicated post-processing, and the previous image-to-text methods' 1D text output is inconsistent with 2D table image. Our proposed method keeps the end-to-end manner and relieves the spatial inconsistency problem.

Detection-based methods, such as line-based models [Schreiber *et al.*, 2017; Ma *et al.*, 2023; Raja *et al.*, 2022] and cell-based models [Prasad *et al.*, 2020; Raja *et al.*, 2020; Long *et al.*, 2021], are first proposed to solve the TSR problem. These solutions are inspired by the intuition that a table is composed of basic elements such as rows/columns or cells. However, these methods face another difficulty: it is not capable of reconstructing the logical structure using detection network. This means that they have to add an addi-

*Corresponding author

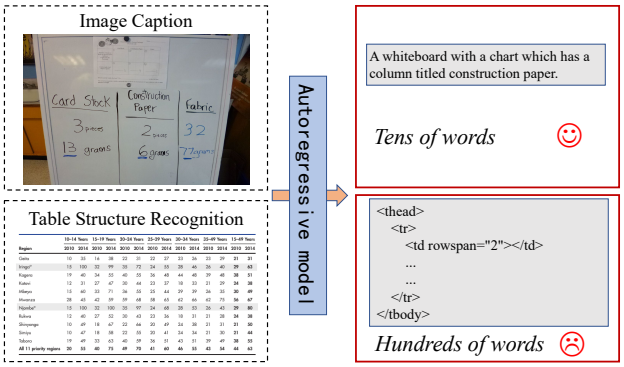


Figure 2: **The length of TSR output is extremely long**, which makes a great difference when compared with other image-to-text tasks such as image caption.

tional architecture, like graph network [Xue *et al.*, 2021] or rule-based post-processing strategies [Qiao *et al.*, 2021a], to recover the relationship between the detected objects, which brings longer procedures and complex processes, as shown in Fig. 1 (a).

Therefore, recent attention to the TSR problem has been drawn on image-to-text solution [Li *et al.*, 2020; Deng *et al.*, 2019a], *i.e.*, input a table image and directly output a sequence of text to depict the table such as HTML code. In this area, encoder-decoder-based autoregressive methods have made remarkable progress [Jimeno Yepes *et al.*, 2021; Ye *et al.*, 2021; Nassar *et al.*, 2022]. They use Convolution Neural Network (CNN) to extract table image features and use Transformer to predict HTML table text word by word. Meanwhile, they also predict the bounding box of each cell with a bounding box regression branch. However, these methods usually face two critical problems. (i) Autoregressive image-to-text models suffer from severe error accumulation problems (intermediate erroneous prediction has a long influence on future predictions), especially for the large table corresponding to long text sequence, as shown in Fig. 2. (ii) The spatial inconsistency problem for the 2D image to 1D text translation can also degrade model performance. For example, the second row's leftmost cell and the third row's leftmost cell are adjacent, but in the HTML sequence, they are separated by several table cells, as illustrated in Fig. 1 (b). Autoregressive methods could be especially sensitive to such problems.

In this paper, we propose a novel end-to-end TSR model, **D**ivides **R**ows and **C**onquers **C**ells, called **DRCC**. Similar to existing image-to-text based models, DRCC also directly outputs HTML format table text, and meanwhile regresses the bounding box for each predicted table cell as well. Observing that predicting long text sequence word by word is the main cause of error accumulation, and tables are organized in a row-column manner, inspired by the (semi-)autoregressive attempt in machine translation [Akoury *et al.*, 2019] and scene text recognition [Qiao *et al.*, 2021b], we build a two-step coarse-to-fine Transformer decoder network to parse table images from row to cell progressively, as illustrated in Fig. 1 (c). Specifically, DRCC first uses a row decoder to detect table rows. The row decoder predicts both HTML row tag category and row location determined by an upper boundary and a lower boundary. Then for each row detected by the row decoder, a cell decoder is applied to predict all cells of the current row. The prediction of cells includes both HTML cell tag category and cell location determined by a bounding box. The row features extracted by the former row decoder are input to the latter cell decoder as guiding information. For cell prediction, we propose a novel decoding scheme that predicts cells of the current row semi-autoregressively with the guiding information of previously predicted rows and cells.

With our proposed two-step architecture, the error accumulation problem of text prediction is dramatically alleviated. As an example, for a 20×25 size table, previous image-to-text methods have to predict **500** cells one by one autoregressively [Nassar *et al.*, 2022], any early mistakes could cause a large influence on later predictions. By contrast, our method bypasses autoregressive manner and only needs to predict 20 rows at first and then 25 cells for each predicted row. With the predicted row tags and cell tags, a simple predefined tag combination process is applied to yield the final HTML table text.

Overall, the main contributions of our work are as follows:

- We propose a novel two-step Transformer architecture for image-to-text TSR task. It alleviates error accumulation problems specific to existing autoregressive methods. To the best of our knowledge, we are the first to concentrate on the error accumulation problem for TSR task.
- We design a semi-autoregressive row-to-cell progressive decoding scheme, which eliminates the inherent spatial inconsistency problem for the 2D image to 1D text translation.
- Extensive experiments demonstrate that we have outperformed previous work on three public benchmarks: PubTabNet(+1% in TEDS and 1.4% in TEDS-Struct), SciTSR(0.3% in complicated test set), SynthTabNet(2% in TEDS on structure result).

2 Related Work

TSR is a long-standing and challenging computer vision problem aiming to parse table images into structured data format [Itonori, 1993]. With the thriving of deep learning technologies, TSR has been fully developed and can be basically divided into detection-based approaches and image-to-text-based approaches.

2.1 Detection-based TSR

Detection-based TSR methods parse table images from the perspective of object detection or segmentation. These methods can not predict table logical structure without post-processing. According to the basic components focused on in these methods, detection-based TSR can be categorized into line-based methods and cell-based methods.

Line-based methods [Siddiqui *et al.*, 2019; Tensmeyer *et al.*, 2019] resort to detection or segmentation methods to locate lines of a table. Then all the detected lines are intersected

to obtain the cell grids. With the success of Transformer in object detection, TSRFormer [Lin *et al.*, 2022] proposes a coarse-to-fine method to detect table lines based on DETR [Carion *et al.*, 2020], it also extends line-based methods to curved and rotated table images.

Cell-based methods focus on cells, the fundamental element of the table. The cell element can be easily detected by classic off-the-shelf object detection models such as Faster-RCNN [Ren *et al.*, 2015] and YOLO [Redmon *et al.*, 2016]. Besides, some table-specific models [Prasad *et al.*, 2020; Qiao *et al.*, 2021a; Long *et al.*, 2021] are proposed and achieve much better performance than generic object detection methods. After obtaining the cell bounding boxes, cell relationships could be recovered with heuristic rules or separate deep learning methods.

2.2 Image-to-text based TSR

Inspired by the success of vision-language tasks, *e.g.*, image caption, visual question answering, etc., researchers propose to directly perceive the whole table image and output the final table logical structure in text format, *e.g.*, HTML, with an encoder-decoder architecture. Attention-based encoder-decoder models are first proposed in image caption area [Xu *et al.*, 2015], and are soon extended to other structural prediction tasks such as mathematical formulas recognition [Deng *et al.*, 2017] and table structure recognition [Deng *et al.*, 2019b]. However, the performance of directly transferring generic encoder-decoder models to TSR is not satisfactory.

Dedicated to table images, [Zhong *et al.*, 2020] proposes EDD, a CNN + LSTM model improving the performance of TSR to a new stage. Its key ingredient is a dual-branch decoder architecture with a tag branch predicting the table in text format and a content branch recognizing the content of each table cell. However, it integrates Optical Character Recognition (OCR) into the overall framework for cell content detection, which is computationally costly. TableMaster [Ye *et al.*, 2021] makes further progress by using Transformer based encoder-decoder architecture. It also abandons cell content detection and proposes to locate cells with bounding boxes instead. TableFormer [Nassar *et al.*, 2022] proposes a similar Transformer based architecture for TSR, which replaces bounding box regression branch with a separate Transformer to increase model capacity.

All existing image-to-text TSR models adopt autoregressive manner and suffer from error accumulation and spatial inconsistency problems. Our proposed method is focused on alleviating these problems.

3 Method

Below we will first describe the overall architecture of our model and then elaborate on individual modules.

3.1 Overall Architecture

The overall framework of DRCC is illustrated in Fig. 3. Given an input table image, a CNN backbone is used to extract image features, which are serialized as input token embeddings to a Transformer encoder.

The kernel ingredient of our method is a cascaded two-step decoder architecture. A Transformer row decoder takes in a learnable query embedding sequence as input and interacts with the Transformer encoder output through multi-head cross attention. At the output end, the row decoder non-autoregressively predicts rows of the entire table, including HTML row tag category and row location. Then, by iterating over all the predicted rows in top-to-down order, a Transformer cell decoder is applied to predict HTML cell classification and regression of each row. The cell decoder adopts a novel semi-autoregressive decoding scheme. That is, at the micro level, the cell decoder predicts all cells corresponding to the current row non-autoregressively, and at the macro level, if regarding the cells belonging to one row as a cell set, the cell decoder predicts cell sets one set by another set autoregressively following top-to-down order.

When processing cell prediction, auxiliary visual features are added to better capture visual context. The auxiliary visual features are all other row features and predicted cell features from the backbone, with 1×1 ROIAlign [He *et al.*, 2017] applied on the final CNN output feature maps.

At inference time, when we get the regression of all rows in the row decoder, a simple rearranging strategy is applied to sort rows by comparing the upper/lower bound of the rows. As for cells, since we apply the semi-autoregressive strategy introduced above, all cells can be grouped into one particular row. Cells in one row can be easily ordered by their bounding box. Combining all prediction results, the final logical structure of the table is reconstructed.

The details of each component module as well as our tag reorder scheme are described below.

3.2 CNN Feature Extractor

For all input table images, we resize them to a fixed size $R^{H \times W \times 3}$. The CNN feature extractor maps input images to feature maps of shape $R^{\frac{H}{32} \times \frac{W}{32} \times C}$, namely it downsamples images 32 times to encode rich contextual information. We use the ResNet [He *et al.*, 2016] modified by removing the final classification layer as our CNN backbone.

3.3 Transformer Encoder

The CNN output features are tokenized as the input sequence of the Transformer encoder, which is of shape $N \times C$ where $N = \frac{HW}{32^2}$. The positional encoding used in VIT [Dosovitskiy *et al.*, 2020] is also added to the input sequence to introduce image spatial information. The Transformer encoder is stacked from standard Transformer encoder layers [Vaswani *et al.*, 2017] with a multi-head self-attention (MHSA) module and a feed-forward network (FFN). It outputs a sequence with the same shape of $N \times C$ as the input sequence.

3.4 Transformer Row Decoder

The Transformer row decoder takes a learnable row query sequence as input, which is of shape $L_{row} \times C$. The sequence length L_{row} is the predefined maximum number of rows in a table. It is stacked from DETR [Carion *et al.*, 2020] decoder layers with each one composed of a MHSA module, a multi-head cross-attention (MHCA) module and a FFN. At the output end, 2 separate three-layer FFNs are applied to map the output sequence of shape $L_{row} \times C$ into 2

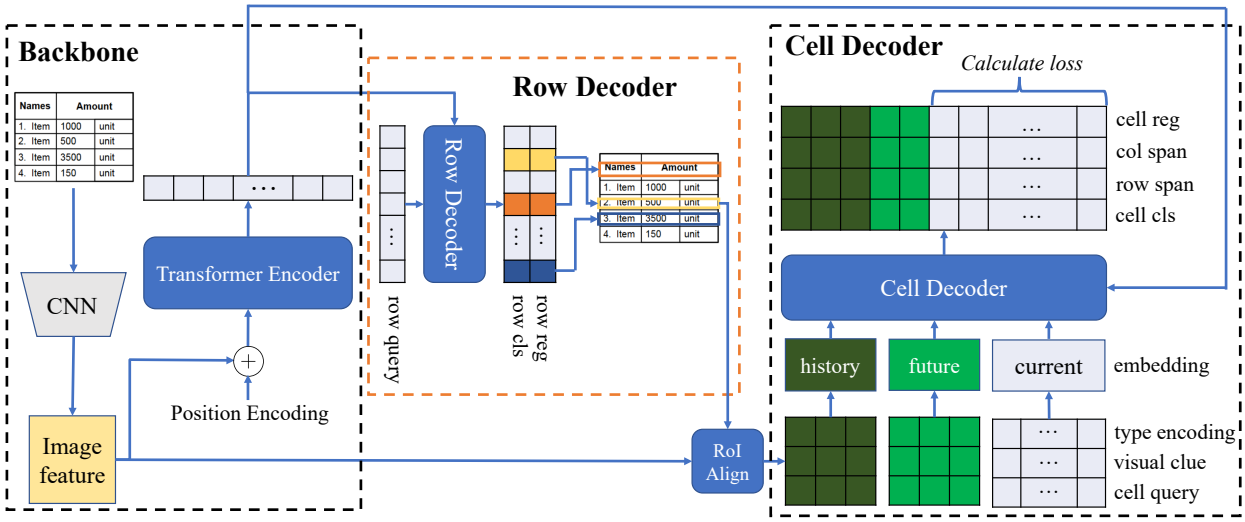


Figure 3: **An overview of the proposed DRCC**. DRCC is composed of three parts: backbone, row decoder, and cell decoder. The backbone is used to extract image features, the subsequent row decoder will regress and categorize all rows, and cell decoder will predict all table cells.

sets of predictions: (i) row tag category predictions P_{rCAT} of shape $L_{row} \times 3$; (ii) row location predictions P_{rLOC} of shape $L_{row} \times 2$. Meanwhile, the row localization is parameterized with a 2-dimensional vector representing the normalized upper and lower boundary of the row. Similar to DETR, Hungarian Algorithm is applied to map each ground truth row of the table to a predicted one based on both P_{rCAT} and P_{rLOC}. It is worth mentioning that the row decoder predicts in a non-autoregressive manner, the order of the predicted rows is not in line with the true row order of the table. Therefore, row reordering according to the predicted row locations is performed, such that the predicted rows can be iterated in the correct top-to-down order.

3.5 Transformer Cell Decoder

The transformer cell decoder has 4 similar Transformer layers to the ones in the row decoder. As illustrated in Fig. 3, the input query sequence is the summation of three different embedding sequences: (i) type encoding; (ii) visual clue; (iii) cell query. All three sequences have the same shape of $L_{cell} \times C$, where sequence length L_{cell} is the predefined maximum number of cells in a table. Before introducing these three input sequences. It is worth mentioning our proposed semi-autoregressive decoding scheme first.

Semi-autoregressive Decoding

As shown in Fig. 3, all the three input embedding sequences of the cell decoder are partitioned into three sections: (i) history section; (ii) future section; (iii) current section. The embeddings in the current section act as a query to non-autoregressively predict cells of the current row being iterated. To further utilize contextual information from other rows to boost model performance, we append the history section and future section to the input sequence, where embeddings in the history section correspond to visual features from rows and cells that have already been iterated, while embeddings in the future section correspond to features from rows that have not been iterated yet. As the row iteration goes on, the history section of the input sequence is gradually expanded. Considering the non-autoregressive manner in predicting cells in one row and autoregressive characteristic of predicting cells row by row, we term this hybrid sequence prediction scheme as semi-autoregressive prediction. Below we describe the three types of input embedding.

Type Encoding

Each token in the type encoding sequence is a marker that indicates which of the three sections the current token belongs to. Meanwhile, it also points out whether the visual clue token at the same location is a row feature or a cell feature. Therefore, each token belongs to the following four situations: (history, row), (history, cell), (future, row), (current, row). Since the cell features for the current and future rows are still unavailable, we do not have (current, cell) and (future, cell) items. Actually, the type encoding is obtained by a learnable embedding layer which maps each 4-dimensional one-hot encoding into a C-dimensional feature vector.

Visual Clue

The token in the visual clue sequence is either a row feature or a cell feature. Based on the CNN backbone output feature maps, the row feature is the 1×1 ROIAlign [He *et al.*, 2017] result of a row region determined by the row locations as predicted by the row decoder. Similarly, the cell feature is the ROIAlign result of a cell region determined by the bounding box as predicted by the cell decoder. In the history section, the visual clue tokens include all row features and cell features corresponding to all the rows having been iterated. In the future section, each visual clue token is the row feature of the corresponding row. In the current section, all visual clue tokens are the same row feature of the current row being iterated. Note that with the overall sequence length L_{cell} fixed, the length of the history section increases while the length of the other two sections decreases during row iteration.

Cell Query

With only type encoding and visual clue sequences, we find that tokens in the current section are the same , which can not query different cells. To circumvent this problem, we add the third input sequence, cell query embedding, which is a learnable embedding acting as an object query for cell detection.

Based on the output sequence of the cell decoder, 4 three-layer FFNs are applied to generate 4 sets of predictions. They are cell tag category prediction P_{cCAT} of shape $L_{cell} \times C_{cell}$, cell row span attribute prediction $P_{rowspan}$ of shape $L_{cell} \times 10$, cell column span attribute prediction $P_{colspan}$ of shape $L_{cell} \times 10$, and cell bounding box prediction P_{cBbox} of shape $L_{cell} \times 4$. The cell tag category number C_{cell} differs for different datasets. The row span and column span attributes prediction are 10-way discrete classification problems, the target label ranges from 1 to 10, *i.e.*, we set the maximum value of these two attributes to 10. The cell bounding box parameterized with a 4-dimensional vector is regressed to the ground truth for cell localization. Similar to the row decoder, Hungarian Algorithm is applied to establish a one-to-one mapping between the predicted and the ground truth cells.

3.6 Loss Function

The training objective of our model is a multi-task loss function.

For both row decoder and cell decoder, we use the negative log-likelihood loss for all tag classifications. For row and cell localization, we use a linear combination of l_1 loss and generalized IoU (GIOU) loss, which is the same as the localization loss in DETR. Since we only regress the upper and lower bound of each row in the row decoder, we manually add a 0 left bound and a 1 right bound when calculating loss functions, such that both the localization of rows and cells could be unified with the GIOU loss.

It is worth mentioning that for the cell decoder, loss is only calculated based on the output tokens in the current section. Since the other two sections are only used to provide rich contextual information for the current section.

3.7 Parameter Settings

We set $H = W = 960$, *i.e.*, rescale all training and testing table images to 960×960 resolution, the resolution of the final extracted CNN feature maps are 30×30. We set $C = 512$, *i.e.*, the feature dimension at all network modules are fixed at 512. The sequence length of the Transformer encoder is 900, which is in line with CNN feature map size. The row decoder sequence length L_{row} is set to be 50, and the cell decoder sequence length L_{cell} is set to 500.

4 Experiments

4.1 Datasets

Our models are conducted on three popular public benchmarks, including PubTabNet [Zhong *et al.*, 2020], SciTSR [Chi *et al.*, 2019] and SynthTabNet [Nassar *et al.*, 2022] to verify the effectiveness of our model.

PubTabNet is a large-scale table dataset that contains 500,777 training images, 9,115 validating images, and 9,138 testing images. PubTabNet is generated from scientific articles and contains both the structure of the table and the text within each cell in HTML format. As the annotations of the testing set are not released, we report the result in the validation set following previous work [Lin *et al.*, 2022]. We extract 2 row types and 11 cell types following the setting in [Ye *et al.*, 2021]. TEDS [Zhong *et al.*, 2020] and TEDS-Struct [Qiao *et al.*, 2021a] are used to evaluate performance.

Methods	TEDS(%)	TEDS-Struct(%)
EDD[2020]	88.3	-
TableFormer[2022]	93.6	96.8
TableMaster[2021]	96.8	-
LGPMA[2021a]	94.6	96.7
FLAG-Net[2021]	95.1	-
RobusTabNet[2023]	-	97.0
TSRFormer[2022]	-	97.5
Ours	97.8	98.9

Table 1: **Result on PubTabNet.**

Methods	SciTSR	SciTSR-COMP
LGPMA[2021a]	98.8	98.0
FLAG-Net[2021]	99.5	98.5
RobusTabNet[2023]	99.3	98.7
TSRFormer[2022]	99.4	98.9
Ours	99.5	99.2

Table 2: **Result on SciTSR.**

SciTSR contains 12,000 training images and 3000 testing images cropped from PDF of scientific literature, with structure labels obtained from LaTex source files. 716 complicated table images are selected as SciTSR-COMP test set to further compare the performance in complicated situations like spanning cells. There are only 1 row type and 1 cell type in this dataset, and the cell adjacency relationship [Chi *et al.*, 2019] metric is used as the evaluation metric.

SynthTabNet is a large synthetically generated dataset with complicated structure and diverse appearance. It contains 600k images and their annotations, and all images are divided into train, test and val splits (80%, 10%, 10%). Table is defined as "complex" if there are row spans or column spans. There are 2 row types and 1 cell type in SynthTabNet, and TEDS is applied to evaluate the performance.

4.2 Result on Benchmark

DRCC is conducted on three public benchmarks, and it achieves state-of-the-art performance on all of them.

As shown in Table 1, DRCC has achieved 97.8% TEDS score and 98.9% in TEDS-Struct, outperforming SOTA methods in TEDS by 1% and TEDS-Struct by 1.4%. Table 2 shows our method is comparable with the previous SOTA in SciTSR in F1-score, and in SciTSR-COMP, we are 0.3% better. As reported in Table 3, although table in SynthTabNet has much more complicated structure and various appear-

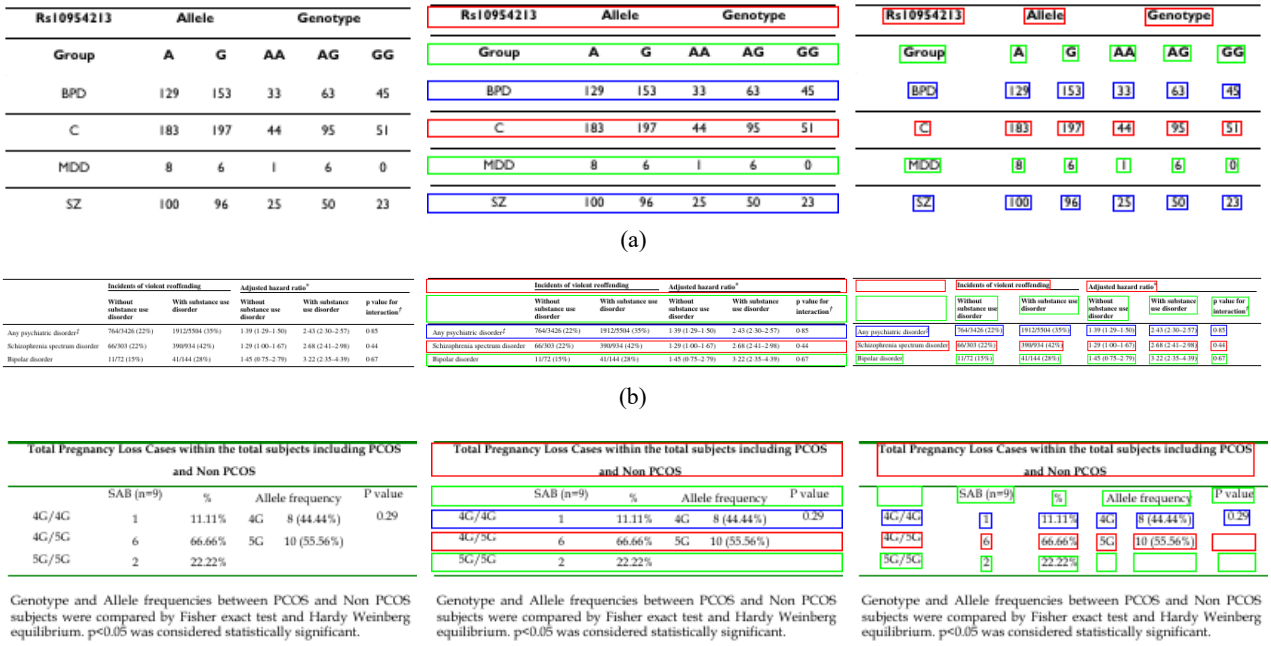


Figure 4: **Visualization of DRCC predictions (row bound and cell bounding box).** Left: table image; middle: row prediction; right: cell prediction. For clarity, we add virtual left and right bound for every row so that it can be represented as a rectangle. The color of the bounding box visualization is looped in red, yellow, and blue in row granularity so that the row-cell relationship can be observed more clearly.

Methods	TEDS(%)		
	Simple	Complex	All
TableFormer[2022]	96.9	95.7	96.7
Ours	98.8	98.0	98.7

Table 3: **Structure Result on SynthTabNet.**

Methods	PTN-small	PTN-medium	PTN-large
EDD	91.79	90.50	87.42
TableMaster	97.35	96.58	94.61
Ours	98.36	97.86	97.25

Table 4: **Result on PubTabNet table of different size.** We trained EDD and TableMaster using the official GitHub code on PubTabNet training set, and illustrate their performance, since this result is not reported by the author in their works [Zhong *et al.*, 2020; Ye *et al.*, 2021]. PTN is the abbreviation of PubTabNet.

ances, we achieve 98.7% in TEDS on structure result, which is 2% better than that of TableFormer, and the improvement increases to 2.3% in "complex" tables. These results show that DRCC achieves great performance in public benchmarks, especially for complicated situations.

4.3 Error Accumulation Analysis

As we mentioned above, error accumulation severely drops the performance of image-to-text methods, especially when facing huge tables and long output. Here, we are trying to show the effort we make to relieve the error accumulation.

Prediction Length

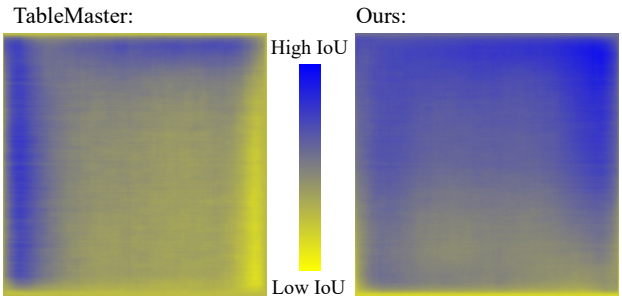
Table 4 shows the result of three image-to-text methods: EDD, TableMaster, and DRCC, on PubTabNet tables of different sizes. We split the PutTabNet into three splits: PTN-small, PTN-medium, and PTN-large according to the output text length. Namely, we categorize the shortest 1/3 tables into PTN-small, the longest 1/3 tables into PTN-large, and others into PTN-medium.

As the length of output text increases, the performance of all three methods drops down. However, when comparing the gap among different splits, it can be observed that DRCC has a much more robust performance of the three. EDD drops down by 1.29 comparing PTN-medium with PTN-small, and 3.08 comparing PTN-large with PTN-medium. When it comes to TableMaster, the gap is 0.77 and 1.97 separately. It can be observed that the gap between PTN-large and PTN-medium is much larger than that between PTN-medium and PTN-small on both TableMaster and EDD. As for DRCC, the gap is only 0.5 and 0.61, and the performance decreasing among different splits is much closer.

Overall, when facing huge tables, DRCC has a much more stable performance.

Spatial Distribution

A "matching map" is shown in Fig 5. The algorithm to generate "matching map" can be referred to in the supplementary material. With the help of "matching map", we can have much more fine-grained view of why and where the error ac-



(a) Matching map of predicted bounding box and ground truth.

TableMaster:

153	140	146	145	128
163	126	119	112	96
162	119	108	101	82
157	112	101	94	72
135	100	92	87	66

Ours:

164	166	168	170	172
163	162	163	167	170
159	152	152	156	163
137	129	129	131	144
133	130	127	121	121

(b) Digtization of matching map.

Figure 5: **(a) "Matching map" of TableMaster and DRCC.** We only show the result from TableMaster and DRCC, as EDD does not predict the bounding box of cells. Blue means better accuracy and yellow represents worse accuracy. **(b) Split the "matching map" into 5 $\times$ 5 grids, and the number in each grid shows the average value in the grid area.** Bigger digit means better accuracy. The algorithm to generate "matching map" can be referred in the supplementary material.

cumulation occurs.

As we can see in Fig 5, TableMaster's performance drop dramatically in the right and bottom side of the image. This can be explained that HTML, the output format for previous image-to-text method, describe the table in a top-to-down left-to-right spatial order. In horizontal direction, DRCC has almost stable performance while TableMaster owns a trend that accuracy drops from left to right. Unfortunately, in vertical direction, both of the methods drop down from top to bottom. This might come from that the cell decoder of DRCC "scans" the table image using a up-to-down manner.

4.4 Ablation Studies

Row decoder		Cell decoder		TEDS
CB	RB	NAR	SAR	
		✓		83.21
✓		✓		95.89
✓			✓	97.36
	✓		✓	97.82

Table 5: **Ablation study. CB** means row decoder predicts the first **C**ell's **B**ounding box for each row, **RB** means row decoder predicts the **R**ow **B**ound for each row. **NAR** is the abbreviation of **N**on-**A**uto**R**egressive, **SAR** is the abbreviation of **S**emi-**A**uto**R**egressvie.

We will analyze the impact of different designs of the decoder on PubTabNet in this section.

There are two settings for the row decoder. The first is predicting the first "**C**ell **B**ounding" box for each row. Another row decoder is to predict the upper/lower **R**ow **B**ound for each row, namely the row decoder used in DRCC. As for the cell decoder, we apply a non-autoregressive Transformer decoder using easy-first [Goldberg and Elhadad, 2010; Qiao *et al.*, 2021b] decoding strategy, represented as **N**on-**A**uto**R**egressive. Accordingly, the cell decoder adopted by DRCC is called **S**emi-**A**uto**R**egressvie.

As illustrated in Table 5, a completely non-autoregressive decoder severely decreases the performance to an unacceptable level, which makes it meaningless even if the error accumulation problem can be solved by pure non-autoregressive methods, since the long output in TSR enlarges its drawback in capture context information.

A coarse-grained row decoder can improve the performance of the cell decoder significantly, since it splits the whole table into several components and offers useful auxiliary information. The predicting target for the row decoder also exerts a small amount of influence on the final result. Compared with the first cell in row, the strip where the row is located contains more abundant context information, and all cells in the particular row can be involved.

4.5 Visualization

We illustrate some visualization of DRCC in PubTabNet. For clarity, the prediction of the bounding box is looped in red, yellow, and blue in row granularity, namely red for the first row and its cells, yellow for the second, blue for the third and so on. As is shown in Fig. 4, DRCC can deal with complex table *e.g.* row/column-span, multi-line text, ignoring the caption area which does not belong to the table. These illustrations demonstrate the robustness of our proposed method in different situations.

5 Conclusion

In this paper, we propose a novel framework for table structure recognition named DRCC. We spot the problem of error accumulation and spatial inconsistency between the 2D image and 1D text output in previous image-to-text methods. We propose a novel two-step Transformer architecture to relieve the error accumulation problem, and a semi-autoregressive row-to-cell progressive decoding scheme eliminates the inherent spatial continuity. Our results suggest that DRCC is efficient to handle large tables and achieves state-of-the-art in three public benchmarks. In the future, we will utilize text spotting [Wang *et al.*, 2022; Wei *et al.*, 2022] technique to integrate more textual and visual information for multi-modal TSR.

Acknowledgements

This work is supported by the Key Research Program of Frontier Sciences, CAS, Grant NO ZDBS-LY-7024.

References

[Akoury *et al.*, 2019] Nader Akoury, Kalpesh Krishna, and Mohit Iyyer. Syntactically supervised transformers for faster neural machine translation. *arXiv preprint arXiv:1906.02780*, 2019.

[Carion *et al.*, 2020] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[Chi *et al.*, 2019] Zewen Chi, Heyan Huang, Heng-Da Xu, Houjin Yu, Wanxuan Yin, and Xian-Ling Mao. Complicated table structure recognition. *arXiv preprint arXiv:1908.04729*, 2019.

[Deng *et al.*, 2017] Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M Rush. Image-to-markup generation with coarse-to-fine attention. In *International Conference on Machine Learning*, pages 980–989. PMLR, 2017.

[Deng *et al.*, 2019a] Yuntian Deng, David Rosenberg, and Gideon Mann. Challenges in end-to-end neural scientific table recognition. In *International Conference on Document Analysis and Recognition*, pages 894–901. IEEE, 2019.

[Deng *et al.*, 2019b] Yuntian Deng, David Rosenberg, and Gideon Mann. Challenges in end-to-end neural scientific table recognition. In *International Conference on Document Analysis and Recognition*, pages 894–901. IEEE, 2019.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[Goldberg and Elhadad, 2010] Yoav Goldberg and Michael Elhadad. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 742–750, 2010.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[He *et al.*, 2017] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[Itonori, 1993] Katsuhiko Itonori. Table structure recognition based on textblock arrangement and ruled line position. In *International Conference on Document Analysis and Recognition*, pages 765–768. IEEE, 1993.

[Jimeno Yepes *et al.*, 2021] Antonio Jimeno Yepes, Peter Zhong, and Douglas Burdick. ICDAR 2021 competition on scientific literature parsing. In *International Conference on Document Analysis and Recognition*, pages 605–617. Springer, 2021.

[Li *et al.*, 2020] Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. TableBank: Table benchmark for image-based table detection and recognition. In *Language Resources and Evaluation Conference*, pages 1918–1925, 2020.

[Lin *et al.*, 2022] Weihong Lin, Zheng Sun, Chixiang Ma, Mingze Li, Jiawei Wang, Lei Sun, and Qiang Huo. TSR-Former: Table structure recognition with transformers. In *ACM International Conference on Multimedia*, pages 6473–6482, 2022.

[Liu *et al.*, 2021] Hao Liu, Xin Li, Bing Liu, Deqiang Jiang, Yinsong Liu, Bo Ren, and Rongrong Ji. Show, read and reason: Table structure recognition with flexible context aggregator. In *ACM International Conference on Multimedia*, pages 1084–1092, 2021.

[Long *et al.*, 2021] Rujiao Long, Wen Wang, Nan Xue, Feiyu Gao, Zhibo Yang, Yongpan Wang, and Gui-Song Xia. Parsing table structures in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 944–952, 2021.

[Ma *et al.*, 2023] Chixiang Ma, Weihong Lin, Lei Sun, and Qiang Huo. Robust table detection and structure recognition from heterogeneous document images. *Pattern Recognition*, 133:109006, 2023.

[Nassar *et al.*, 2022] Ahmed Nassar, Nikolaos Livathinos, Maksym Lysak, and Peter Staar. TableFormer: Table structure understanding with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4614–4623, 2022.

[Prasad *et al.*, 2020] Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultanpure. CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 572–573, 2020.

[Qiao *et al.*, 2020] Zhi Qiao, Yu Zhou, Dongbao Yang, Yucan Zhou, and Weiping Wang. SEED: Semantics enhanced encoder-decoder framework for scene text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13528–13537, 2020.

[Qiao *et al.*, 2021a] Liang Qiao, Zaisheng Li, Zhanzhan Cheng, Peng Zhang, Shiliang Pu, Yi Niu, Wenqi Ren, Wenming Tan, and Fei Wu. LGPMA: Complicated table structure recognition with local and global pyramid mask alignment. In *International Conference on Document Analysis and Recognition*, pages 99–114. Springer, 2021.

[Qiao *et al.*, 2021b] Zhi Qiao, Yu Zhou, Jin Wei, Wei Wang, Yuan Zhang, Ning Jiang, Hongbin Wang, and Weiping

Wang. PIMNet: A parallel, iterative and mimicking network for scene text recognition. In *ACM International Conference on Multimedia*, pages 2046–2055, 2021.

[Qin *et al.*, 2021] Xugong Qin, Yu Zhou, Youhui Guo, Dayan Wu, Zhihong Tian, Ning Jiang, Hongbin Wang, and Weiping Wang. Mask is all you need: Rethinking mask R-CNN for dense and arbitrary-shaped scene text detection. In *ACM International Conference on Multimedia*, pages 414–423, 2021.

[Raja *et al.*, 2020] Sachin Raja, Ajoy Mondal, and CV Jawahar. Table structure recognition using top-down and bottom-up cues. In *European Conference on Computer Vision*, pages 70–86. Springer, 2020.

[Raja *et al.*, 2022] Sachin Raja, Ajoy Mondal, and CV Jawahar. Visual understanding of complex table structures from document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2299–2308, 2022.

[Redmon *et al.*, 2016] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28, 2015.

[Schreiber *et al.*, 2017] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. DeepDeSRT: Deep learning for detection and structure recognition of tables in document images. In *International Conference on Document Analysis and Recognition*, volume 1, pages 1162–1167. IEEE, 2017.

[Siddiqui *et al.*, 2019] Shoaib Ahmed Siddiqui, Pervaiz Iqbal Khan, Andreas Dengel, and Sheraz Ahmed. Rethinking semantic segmentation for table structure recognition in documents. In *International Conference on Document Analysis and Recognition*, pages 1397–1402. IEEE, 2019.

[Tensmeyer *et al.*, 2019] Chris Tensmeyer, Vlad I Morariu, Brian Price, Scott Cohen, and Tony Martinez. Deep splitting and merging for table structure decomposition. In *International Conference on Document Analysis and Recognition*, pages 114–121. IEEE, 2019.

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

[Wang *et al.*, 2022] Wei Wang, Yu Zhou, Jiahao Lv, Dayan Wu, Guoqing Zhao, Ning Jiang, and Weipinng Wang. TPSNet: Reverse thinking of thin plate splines for arbitrary shape scene text representation. In *ACM International Conference on Multimedia*, pages 5014–5025, 2022.

[Wei *et al.*, 2022] Jin Wei, Yuan Zhang, Yu Zhou, Gangyan Zeng, Zhi Qiao, Youhui Guo, Haiying Wu, Hongbin Wang, and Weipinng Wang. TextBlock: Towards scene text spotting without fine-grained detection. In *ACM International Conference on Multimedia*, pages 5892–5902, 2022.

[Xu *et al.*, 2015] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057. PMLR, 2015.

[Xu *et al.*, 2020] Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. LayoutLMv2: Multimodal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*, 2020.

[Xue *et al.*, 2021] Wenyuan Xue, Baosheng Yu, Wen Wang, Dacheng Tao, and Qingyong Li. TGRNet: A table graph reconstruction network for table structure recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1295–1304, 2021.

[Ye *et al.*, 2021] Jiaquan Ye, Xianbiao Qi, Yelin He, Yihao Chen, Dengyi Gu, Peng Gao, and Rong Xiao. PingAn-VCGroup's solution for ICDAR 2021 competition on scientific literature parsing task B: Table recognition to HTML. *arXiv preprint arXiv:2105.01848*, 2021.

[Zeng *et al.*, 2023] Gangyan Zeng, Yuan Zhang, Yu Zhou, Xiaomeng Yang, Ning Jiang, Guoqing Zhao, Weiping Wang, and Xu-Cheng Yin. Beyond OCR+ VQA: Towards end-to-end reading and reasoning for robust and accurate TextVQA. *Pattern Recognition*, 138:109337, 2023.

[Zhong *et al.*, 2020] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: data, model, and evaluation. In *European Conference on Computer Vision*, pages 564–580. Springer, 2020.