electronics

MDPI

Article

Table Structure Recognition Method Based on Lightweight Network and Channel Attention

Tao Zhang, Yi Sui, Shunyao Wu, Fengjing Shao and Rencheng Sun *

College of Computer Science and Technology, University of Qingdao, Qingdao 266071, China
* Correspondence: src@qdu.edu.cn

Abstract: The table recognition model rows and columns aggregated network (RCANet) uses a semantic segmentation approach to recognize table structure, and achieves better performance in table row and column segmentation. However, this model uses ResNet18 as the backbone network, and the model has 11.35 million parameters and a volume of 45.5 M, which is inconvenient to deploy to lightweight servers or mobile terminals. Therefore, from the perspective of model compression, this paper proposes the lightweight rows and columns attention aggregated network (LRCAANet), which uses the lightweight network ShuffleNetv2 to replace the original RCANet backbone network ResNet18 to simplify the model size. Considering that the lightweight network reduces the number of feature channels, it has a certain impact on the performance of the model. In order to strengthen the learning between feature channels, the rows attention aggregated (RAA) module and the columns attention aggregated (CAA) module are proposed. The RAA module and the CAA module add the squeeze and excitation (SE) module to the original row and column aggregated modules, respectively. Adding the SE module means the model can learn the correlation between channels and improve the prediction effect of the lightweight model. The experimental results show that our method greatly reduces the model parameters and model volume while ensuring low-performance loss. In the end, the average F1 score of our model is only 1.77% lower than the original model, the parameters are only 0.17 million, and the volume is only 0.8 M. Compared with the original model, the parameter amount and volume are reduced by more than 95%.

Keywords: table structure recognition; deep learning; SE module; lightweight network

Citation: Zhang, T.; Sui, Y.; Wu, S.; Shao, F.; Sun, R. Table Structure Recognition Method Based on Lightweight Network and Channel Attention. *Electronics* **2023**, *12*, 673. https://doi.org/10.3390/electronics12030673

Academic Editors: Yu-Chen Hu, Praveen Kumar Donta, Piyush Kumar Pareek and Chinmaya Kumar Dehury

Received: 23 November 2022
Revised: 23 January 2023
Accepted: 25 January 2023
Published: 29 January 2023

1. Introduction

Tables are a common data storage method in daily life. With the development of the information age, various documents use tables more and more widely. In the early days, people mainly used rule-based methods to identify tables. Reference [1] and reference [2] used hand-made rules to analyze tables, which could only be used for table recognition in certain fixed formats, which had certain limitations, and the design of the rules was also more complicated. With the continuous development of deep learning, deep learning methods have achieved remarkable results in various fields such as music, natural language, and images. Reference [3] uses different conventional algorithms to solve multiple types of table structure recognition, but requires many preprocessing operations. In recent years, many researchers have used deep learning methods to parse the structure of tables. Reference [4] uses the method of object detection to identify the table structure, and proposed complicated table structure recognition with local and global pyramid mask alignment (LGPMA) based on Mask R-CNN [5], which detects the local and global boundaries of the table, and aligns and fuses the results. Then, three post-processing steps of cell matching, blank cell search, and blank cell merging are added, which solves the problem that blank cells are difficult to detect. Qasim et al. [6] used a convolutional neural network [7] and graph neural network (GNN) [8] to identify table structures, the former for extracting image features and the latter for improving the correlation between vertices.

Reference [9] proposed a table graph reconstruction network for table structure recognition (TGRNet), which uses ResNet50 [10] to extract the rows and columns of the table image and the features of the original image for fusion, predicting the spatial coordinates, and used the graph convolutional networks (GCN) [11] to predict the logical coordinates. Khan et al. [12] tried to use a variant of recurrent neural network (RNN) [13–15], gated recurrent units (GRU) [16], to identify table structure. The receptive field of a convolutional neural network (CNN) is not enough to capture complete row and column information in one step, so RNN can effectively make up for this deficiency. After comparing two improved RNN models, namely, long short-term memory network (LSTM) [17] and GRU, GRU shows greater advantages. Khan et al., therefore, choose to use a pair of bidirectional GRUs, one for row detection and the other for column detection. Siddiqui et al. [18] reduced table structure recognition to the prediction of table columns and table rows. Shen et al. [19] designed a semantic segmentation network for the problem of high fault tolerance of rows and columns, and added feature slicing and tiling operations to the rows aggregated (RA) module and the columns aggregated (CA) module, segmenting the rows and columns of the table. Reference [20] proposed a transformer-based method for table structure identification (TableFormer), which achieved better results in predicting the table structure and bounding boxes of cells. Reference [21] proposes a spatial CNN and grid-CNN-based method for table structure recognition, which can be robust on curved table datasets.

At present, the methods based on deep learning have achieved good results in the task of table structure recognition, but after investigation, it is found that the volume and parameters of the model are often relatively large. From the perspective of the backbone network, many models use a backbone network with a large number of parameters. TGRNet adopts the ResNet50 network, and the parameter amount reaches 25.56 million, while the method in reference [22] uses the Resnet101 [10] network, and the parameter amount reaches 44.55 million. Judging from the size of the model, the volume of the LGPMA model reaches 177 M, while the model volume of reference [23] reaches 256 M. At the same time, the structure of the model is very complex, the training consumption is large, and it is difficult to deploy to a lightweight server or apply it to mobile devices. How to simplify the model complexity and make the table structure recognition model lightweight is still a problem to be solved.

In summary, this paper optimizes the volume of the table recognition model from the perspective of model compression and proposes an improved lightweight table recognition model lightweight rows and columns attention aggregated network (LRCAANet) based on rows and columns aggregated network (RCANet). We used the ShuffleNetv2 [24] backbone network, rows attention aggregated (RAA) module, and columns attention aggregated (CAA) module to replace the original ResNet18 backbone network, rows aggregated (RA) module, and columns Aggregated (CA) module. In the case of ensuring low loss of model performance, the volume and parameters of the model are greatly reduced.

The main innovations are as follows:

1. In this paper, we use a more lightweight network. The backbone network ResNet18 of the RCANet [19] model is replaced by a lightweight ShuffleNetv2 network, which greatly reduces the volume and parameters of the model;
2. In this paper, we add the squeeze and excitation (SE) [25] module to the rows and columns aggregated module of RCANet, so that the row–column feature information has channel attention and improves the performance of the lightweight model;
3. Finally, we combine the lightweight backbone shufflenetv2 with RAA and CAA modules to propose the end-to-end lightweight table structure identification model LRCAANet.

This paper is organized as follows: Section 2 presents the structure of the original model, the structure of the replaced lightweight backbone, and the compression and optimization strategies. Section 3 describes the process of the experiments and the analysis of the experimental results. Section 4 describes the main conclusions of this paper and the prospect on future research directions.

2. Methods

2.1. RCANet Model

The research in this paper is based on the RCANet [19] model, the main structure of RCANet is shown in Figure 1. RCANet is mainly composed of three main parts, namely, the ResNet18 backbone network, rows aggregated (RA) module, and columns aggregated (CA) module. The ResNet18 backbone network is mainly used to extract the features of table images, and the outputs of the layer1, layer2, layer3, and layer4 layers of ResNet18 [10] are extracted to be used as the input of the row–column aggregation module. The outputs of layer4 and layer3 are used as inputs to RA3 and CA3. The input of RA2 and CA2 consists of two parts, one is the result of conducting element-wise addition to the output of RA3 and CA3, and the other is the output of layer2. Finally, the output of RA1 and the output of CA1 are convolved by a 1×1 convolution to obtain the final mask prediction result.

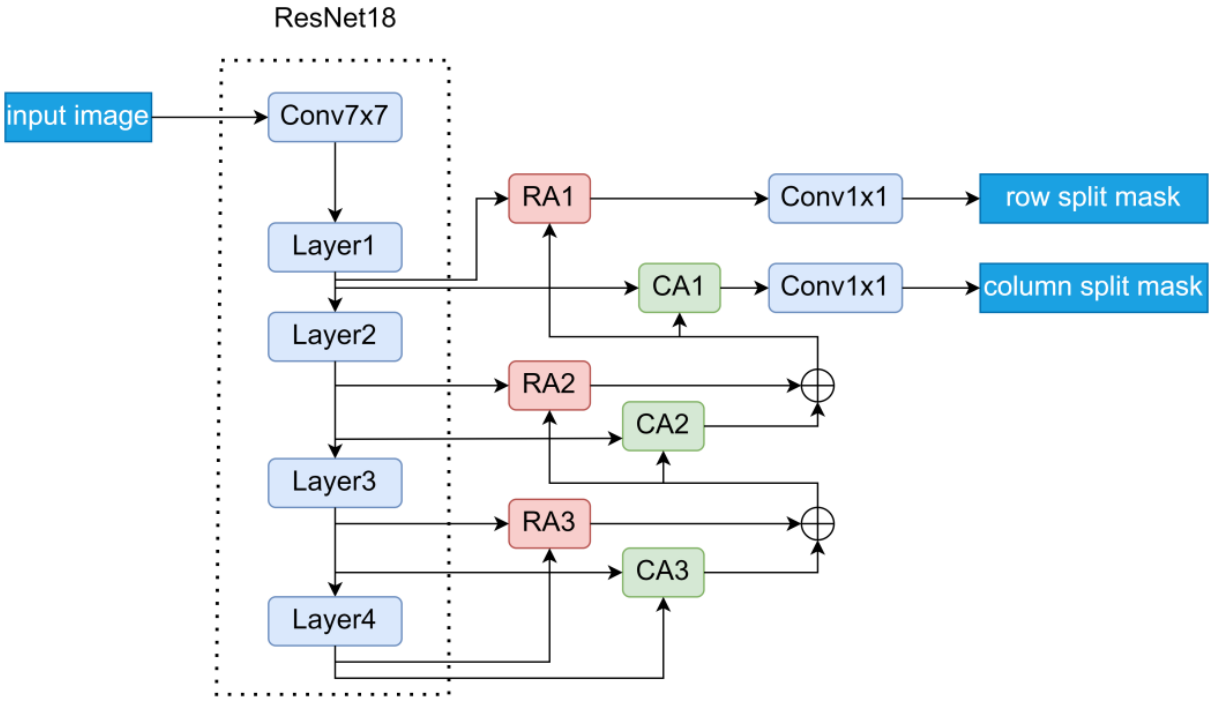


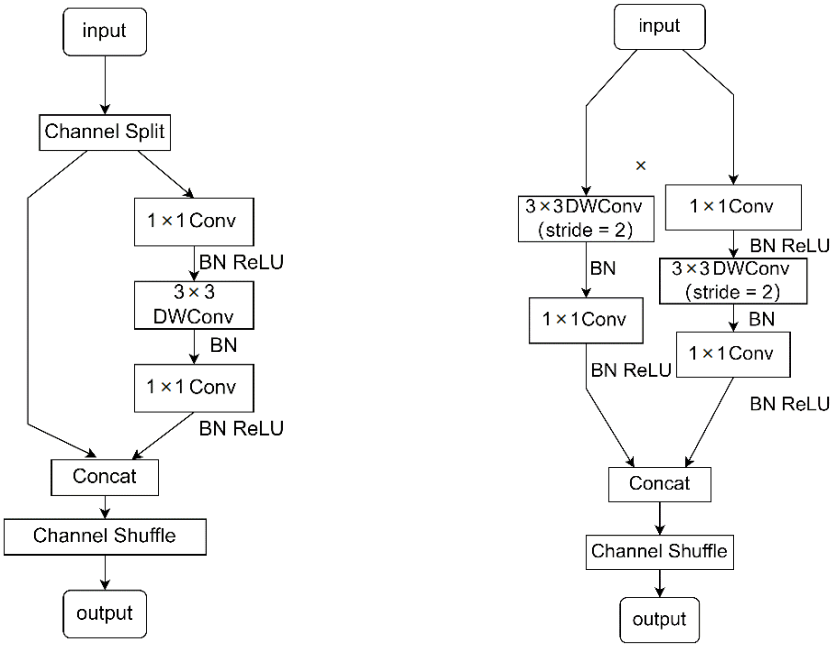
Figure 1. RCANet network structure.

2.2. ShuffleNetv2 Model

As an efficient network, the ShuffleNetv2 [24] is mainly composed of a basic unit and a down-sampling unit; the structure of the basic unit and the down-sampling unit is shown in Figure 2.

In the basic unit, the input features are firstly channel split to obtain left and right branches with the same number of channels. The left branch performs the identity mapping, and the right branch undergoes two 3×3 convolutions and one 1×1 convolution and keeps the number of channels before and after the output unchanged. The left and right branches are merged by channel splice, and the feature information fusion of the left and right branches is enhanced by channel shuffling.

In the down-sampling unit, the input is directly sent to two branches without channel split, and the left and right branches perform 3×3 depth-wise separable convolution and 1×1 point convolution with stride 2 respectively. After channel splicing, the number of channels becomes twice that of the input, and the spliced features are channel-shuffled, similar to the basic unit, to enhance feature information fusion.



(**a**) (**b**)

Figure 2. ShuffleNetv2 units (**a**) basic unit; (**b**) down-sampling unit.

2.3. Compression and Optimization Strategy

We found that most of the volume of the model comes from the backbone network, and replacing the backbone network with a lightweight network can greatly reduce the size of the model. ShuffleNetv2 enhances the flow of information between channels, ensures the correlation between input and output channels, and ensures a lower amount of parameters. The parameter comparison between ShuffleNetv2 and the original backbone network ResNet18 is shown in Table 1. Compared with ResNet18, ShuffleNetv2 reduces the number of parameters by 88%. Therefore, we optimize the backbone network of RCANet and select the lightweight network ShuffleNetv2.

Table 1. Comparison of ResNet18 network and ShuffleNetv2 network parameters.

Model Name	Model Parameters (Million)
ResNet18	11.69
ShuffleNetv2	1.37

Assuming that the original image size is W × H, the output feature map size and number of channels of each layer of Resnet18 and ShuffleNetv2 are shown in Tables 2 and 3.

Table 2. The output feature map size and number of channels of each layer of Resnet18.

Layer	Output Size	Output Channels
Conv1	W/2 × H/2	64
Layer1	W/4 × H/4	64
Layer2	W/8 × H/8	128
Layer3	W/16 × H/16	256
Layer4	W/32 × H/32	512

Table 3. The output feature map size and number of channels of each layer of ShuffleNetv2.

Layer	Output Size	Output Channels
Conv1	W/2 × H/2	24
MaxPool	W/4 × H/4	24
Stage2	W/8 × H/8	48
Stage3	W/16 × H/16	96
Stage4	W/32 × H/32	192

However, by comparing the number of channels in Tables 2 and 3, we find that the number of feature map channels output by ShuffleNetv2 is reduced to a certain extent compared to Resnet18. The reduction in the number of channels increases the difficulty of the network learning the features between channels, which has a certain impact on the performance of the model. Therefore, in order to strengthen the correlation learning between feature channels and optimize the performance of the model, we propose the rows attention aggregated (RAA) module and the columns attention aggregated (CAA) module. The RAA module and the CAA module are obtained by adding the squeeze and excitation (SE) [25] module to the RA module and CA module of the original model RCANet, respectively. We enhance the correlation learning between channels by increasing the SE module, thereby improving the overall performance of the model.

The implementation idea of the SE module is very simple, and it is flexible to use, so it is easy to join various networks. The specific structure of the SE module is shown in Figure 3. The module first uses global average pooling to pool the W × H size feature map compressed to 1 × 1, and then through a full connection layer to compress the channel, compress the feature of the original channel C into channel C/r, and use the Relu function to activate it. In this paper, r is set to 16. Then the compressed channel is mapped back to the original channel C, and the Sigmoid function is used to activate it. The activation result is multiplied with the original feature to obtain a feature map with channel attention.

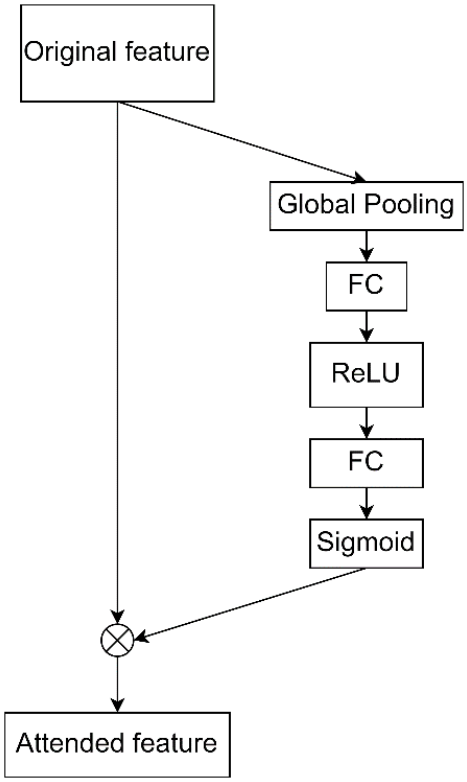


Figure 3. SE module structure diagram.

Finally, we name the proposed model lightweight rows and columns attention aggregated network (LRCAANet), and the structure of the model is shown in Figure 4. The structure of the RAA and CAA module is shown in Figure 5.

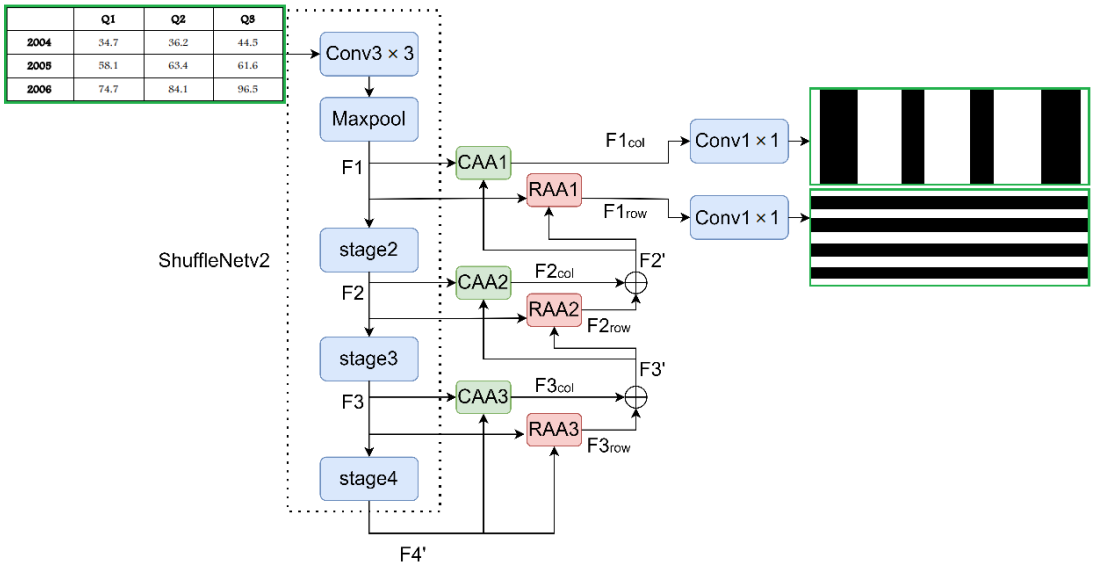


Figure 4. The overall structure of LRCAANet.

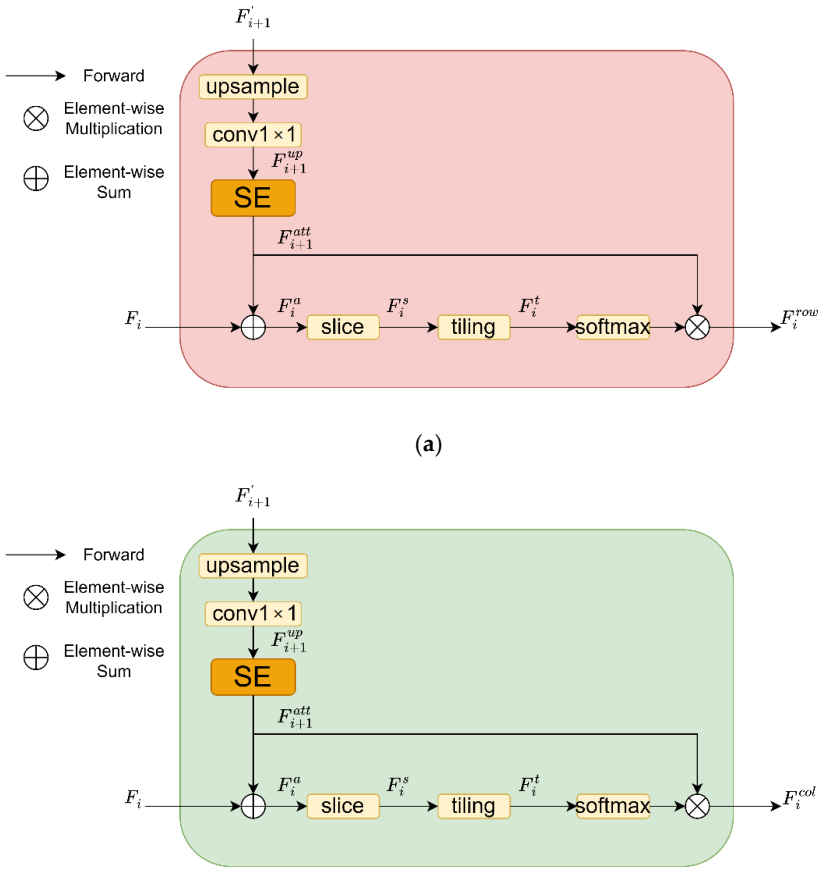


Figure 5. RAA module and CAA module (**a**) RAA module; (**b**) CAA module.

As shown in Figure 4. All input images are normalized and then passed into the model. After a 3 × 3 convolutional layer, four outputs are obtained through the Maxpool

layer, stage2 layer, stage3 layer, and stage4 layer as the input of the RAA module and the CAA module.

We use RAA_i to denote the ith RAA module and CAA_i in the same way. First, the input of the RAA_i and CAA_i modules is divided into two parts, F_i and F'_{i+1}. F'_{i+1} can be denoted as $I^{W/2\times H/2\times 2C}$ and F_i can be denoted as $I^{W\times H\times C}$, where W and H denote the width and height of the feature map and C denotes the number of channels. The number of channels of F'_{i+1} is twice that of F_i, while the height and width of the feature map of F'_{i+1} are both half that of F_i. In order to adjust F'_{i+1} to the same feature map size and number of channels as F_i, as shown in Figure 5a,b we perform up-sampling and 1×1 convolution on F'_{i+1} to obtain F^{up}_{i+1} with the same feature map size and number of channels as F_i, and F^{up}_{i+1} goes through the SE module to obtain F^{att}_{i+1} with channel attention.

F^{att}_{i+1} and F_i perform element-wise addition to obtain F^a_i, slicing and tiling operations are performed on F^a_i. The slicing operation is a process of taking the maximum value. In the RAA module, the maximum value is obtained by row for the feature map, and the maximum value is obtained by column in the CAA module. The calculation formulas for the two modules are shown in Equations (1) and (2), respectively.

$$F^s_i[m,1] = \max_{1\le n\le w} F^a_i[m,n] \quad \forall m = \{1,\ldots,H\} \tag{1}$$

$$F^s_i[1,\mathrm{n}] = \max_{1\le m\le H} F^a_i[m,n] \quad \forall n = \{1,\ldots,W\} \tag{2}$$

The tiling operation is performed to copy the features obtained after slicing. For example, the features obtained by row in the RAA module need to be copied W times to restore the feature map of the same size as before. Similarly, in the CAA module, it will be obtained by column. The features are replicated H times to restore the feature map size. The tiling operations of the RAA module and the CAA module are shown in Equations (3) and (4), respectively.

$$F^t_i = \begin{bmatrix} F^s_i[1,1] & F^s_i[1,1] & \cdots & F^s_i[1,1] \\ F^s_i[2,1] & F^s_i[2,1] & \cdots & F^s_i[2,1] \\ \vdots & \vdots & \ddots & \vdots \\ F^s_i[H,1] & F^s_i[H,1] & \cdots & F^s_i[H,1] \end{bmatrix} \tag{3}$$

$$F^t_i = \begin{bmatrix} F^s_i[1,1] & F^s_i[1,2] & \cdots & F^s_i[1,W] \\ F^s_i[1,1] & F^s_i[1,2] & \cdots & F^s_i[1,W] \\ \vdots & \vdots & \ddots & \vdots \\ F^s_i[1,1] & F^s_i[1,2] & \cdots & F^s_i[1,W] \end{bmatrix} \tag{4}$$

F^t_i is obtained after slicing and tiling of F^a_i. After passing through a softmax layer, F^t_i is multiplied element-wise with the previously obtained F^{att}_{i+1} with channel attention to obtain the final module output, where the RAA_i module outputs Fi_{row} and the CAA_i module outputs Fi_{col}. Fi_{row} and Fi_{col} perform element addition to obtain F'_i,and F'_i and F_{i-1} as the input of the RAA_{i-1} module and CAA_{i-1} module. As shown in Figure 4, the inputs of RAA_3 and CAA_3 are the same, both F'_4 and F_3. The output of RAA_3 and CAA_3 modules have the same feature map size and channels as F_3. The output of RAA_3 and CAA_3 perform element addition to obtain F'_3, and F'_3 and F_2 are used as the input of RAA_2 and CAA_2, and so on, to finally obtain the output $F1_{row}$ of RAA_1 and output $F1_{col}$ of CAA_1. $F1_{row}$ and $F1_{col}$ perform 1×1 convolution to obtain the final mask prediction result.

3. Experimental Results and Analysis

3.1. Experimental Environment and Parameter Setting

Table 4 shows the software and hardware environment of this paper. The experiments use the Python language and are based on the deep learning open-source framework Pytorch.

Table 4. Experimental environment.

Category	Version
CPU	Intel Xeon W-2133
GPU	NVIDIA GeForce GTX 3090
RAM	32 GB
CUDA	CUDA 11.1
Operating system	Ubuntu 20.04.1
Pytorch	torch 1.10.1
Hard-disk	1 TB SSD

We used Adam [26] as the optimizer for model training. When reproducing RCANet, we resized the input image to $640 \times 640 \times 3$ pixels and set the learning rate to 1×10^{-4}, which is the same as the original author. For the compressed model, we adjusted the learning rate to 4×10^{-4} based on experience, and other parameters are consistent with the original model, and a total of 200 training iterations are performed. During training, data enhancement techniques such as horizontal flipping, random movement, and scaling are used to enhance the training set data to prevent overfitting. The data augmentation operation is shown in Figure 6.

As shown in Figure 6, where Figure 6a is the image in the original dataset, we flipped the image in Figure 6a horizontally to obtain the image shown in Figure 6b as the expansion of the data. We randomly translate\d the image in Figure 6a to obtain the image shown in Figure 6c as the expansion of the data. We randomly scaled the image in Figure 6a to obtain the image shown in Figure 6d as an expansion of the data.

The loss function, such as RCANet, uses dice loss, and the loss function is derived from the dice coefficient. The dice coefficient is a metric function used to measure the similarity of sets, usually used to calculate the similarity between two samples. The original dice coefficient calculation function is shown in Equation (5).

$$\text{Dice} = \frac{2|X \cap Y|}{|X| + |Y|} \tag{5}$$

The corresponding dice loss is defined as shown in the following formula:

$$\text{Dice Loss} = 1 - \frac{2|X \cap Y|}{|X| + |Y|} \tag{6}$$

3.2. Datasets

The experiments use the public tabular dataset ICDAR2013 [27], which contains 67 PDF documents from the EU and US governments. The dataset we obtained comes from the cropped table area images in the original PDF document, with a total of 156 table images. We visualized according to the text area annotation labels, and the result is shown in the following Figure 7:

	THRESHOLD FOR RELEASES		
	to air kg/year	to water kg/year	to land kg/year
Carbon dioxide (CO2)	100 million	-	-
Hydro-fluorocarbons (HFCs)	100	-	-
Methane (CH4)	100 000	-	-
Nitrous oxide (N2O)	10 000	-	-
Perfluorocarbons (PFCs)	100	-	-
Sulphur hexafluoride (SF6)	50	-	-

(**a**)

THRESHOLD FOR RELEASES			
to air kg/year	to water kg/year	to land kg/year	
100 million	-	-	Carbon dioxide (CO2)
100	-	-	Hydro-fluorocarbons (HFCs)
100 000	-	-	Methane (CH4)
10 000	-	-	Nitrous oxide (N2O)
100	-	-	Perfluorocarbons (PFCs)
50	-	-	Sulphur hexafluoride (SF6)

(**b**)

	THRESHOLD FOR RELEASES		
	to air kg/year	to water kg/year	to land kg/year
Carbon dioxide (CO2)	100 million	-	-
Hydro-fluorocarbons (HFCs)	100	-	-
Methane (CH4)	100 000	-	-
Nitrous oxide (N2O)	10 000	-	-
Perfluorocarbons (PFCs)	100	-	-
Sulphur hexafluoride (SF6)	50	-	-

(**c**)

	THRESHOLD FOR RELEASES		
	to air kg/year	to water kg/year	to land kg/year
Carbon dioxide (CO2)	100 million	-	-
Hydro-fluorocarbons (HFCs)	100	-	-
Methane (CH4)	100 000	-	-
Nitrous oxide (N2O)	10 000	-	-
Perfluorocarbons (PFCs)	100	-	-
Sulphur hexafluoride (SF6)	50	-	-

(**d**)

Figure 6. (**a**) Original image; (**b**) image after horizontal flip; (**c**) image after random translation; (**d**) image after random scaling.

Faculty cluster	Population size	Sample size
Sciences	1269 (19.9%)	101(20.4%)
Social Sciences	3212 (50.6%)	247(50.0%)
Humanities	1168 (18.4%)	95(19.3%)
Civil Sciences	705 (11.1%)	51(10.3%)

Figure 7. Text box callout visualization.

We use Opencv [28] to process the original data according to the text box annotation information, and draw the mask labels according to the position information between the text boxes. We divide the segmented area into white and the non-segmented area into black, and the generated mask label is shown in Figure 8.

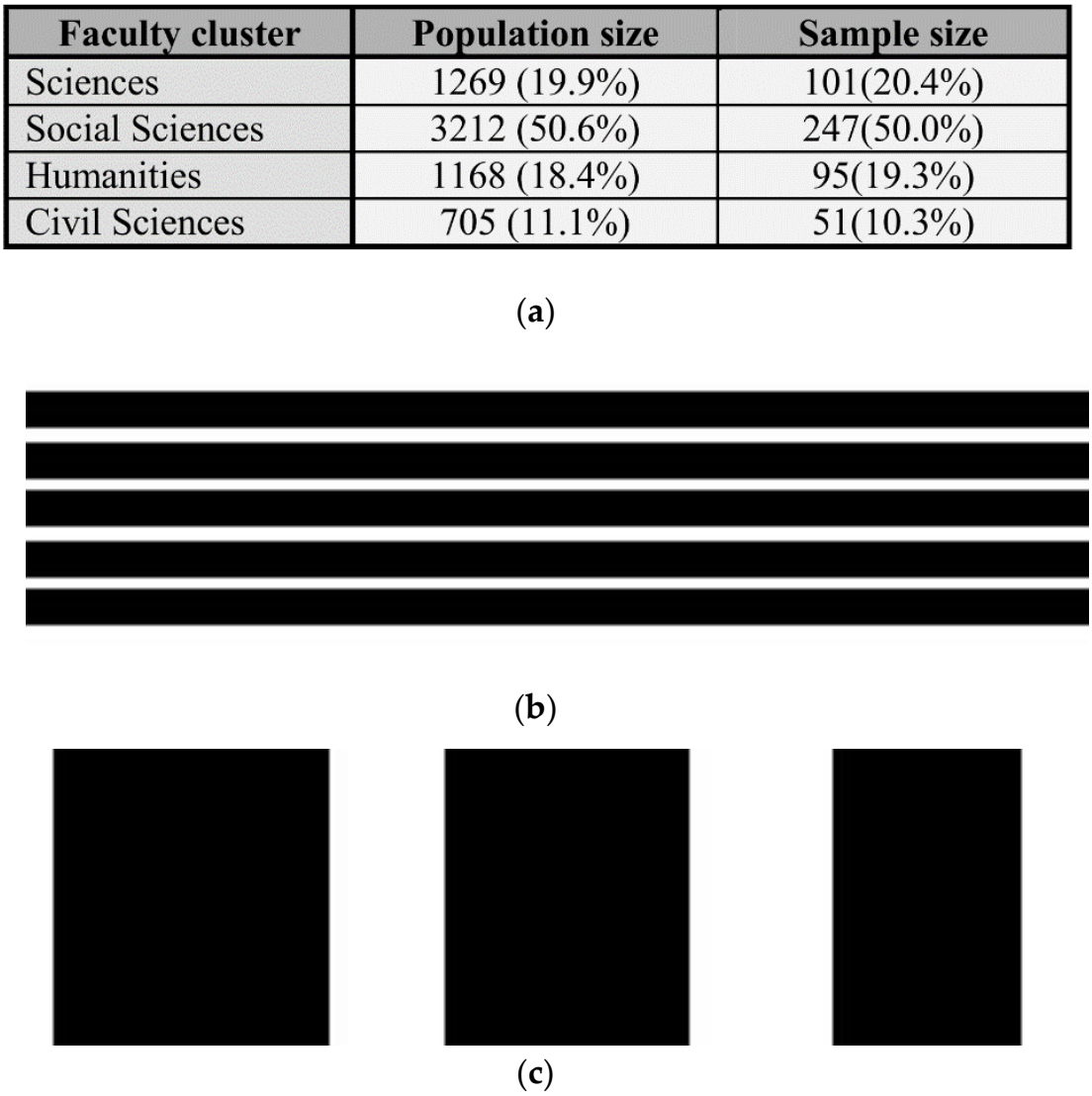


Figure 8. ICDAR2013 tabular dataset (**a**) table image; (**b**) line segmentation mask map; (**c**) column segmentation mask map.

3.3. Evaluation Indicators

For the segmentation performance indicators of the model, we use the same evaluation indicators as RCANet and calculate the precision P, recall rate R, and $F1$ of the model according to the calculated number of true positives (TP), number of false positives (FP), and number of false negatives (TN). The score is calculated as follows:

$$P = \frac{TP}{TP + FP} \tag{7}$$

$$R = \frac{TP}{TP + FN} \tag{8}$$

$$F1 = \frac{2 \times P \times R}{P + R} \tag{9}$$

At the same time, in order to evaluate the compression effect and complexity of the model, we introduced three evaluation indicators, namely, the volume of the model, the parameter amount of the model, and the number of floating-point calculations of the model, to verify the effectiveness of our model compression work.

3.4. Experimental Results and Analysis

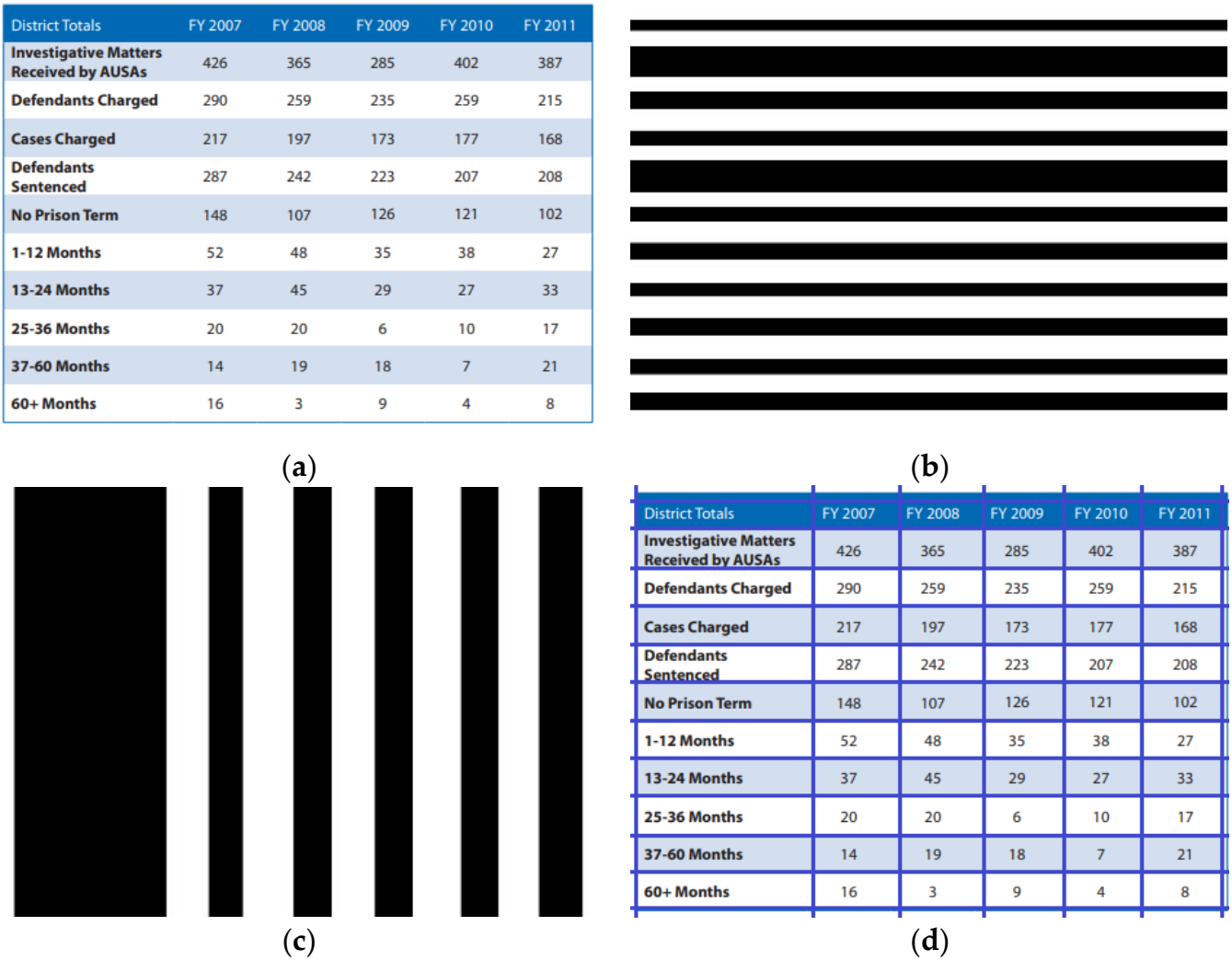
In order to verify the effectiveness of the method proposed in this paper, we conducted a series of ablation experiments to compare the performance of the RCANet (recurrence) model, the model LRCANet after replacing the backbone network, and the model LRCAANet after replacing the backbone network and adding the SE module. The final experiment results are shown in the following table:

From the experimental results in Table 5, it can be seen that when the backbone network is directly replaced by ShuffleNetv2 without adding the SE module, the performance of the model is affected to a certain extent, and the average F1 score drops by 3.78%. The replacement lightweight backbone network ShuffleNetv2 has fewer channels compared to the original Resnet18. The SE module has a channel attention mechanism, and the addition of the SE module can enhance the learning between channels, thus, improving the performance of the model. When the SE module is added, the indicators of the model increase, and the average F1 score is 2.01% higher than that without the SE module, which verifies the effectiveness of adding the SE module.

Table 5. Model performance comparison.

Model	Row			Column			Average		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
RCANet (Recurrence)	0.9586	0.9534	0.9560	0.9654	0.9922	0.9786	0.9620	0.9728	0.9673
LRCANet	0.9066	0.9152	0.9109	0.9166	0.9818	0.9481	0.9116	0.9485	0.9295
LRCAANet	0.9300	0.9434	0.9366	0.9348	0.9922	0.9626	0.9324	0.9678	0.9496

We process the predicted row and column segmentation masks and use Opencv to draw the segmentation lines to obtain the visualization results of the segmentation. Figures 9 and 10 show the comparison of some prediction results and the prediction before and after adding the SE module:



District Totals	FY 2007	FY 2008	FY 2009	FY 2010	FY 2011
Investigative Matters Received by AUSAs	426	365	285	402	387
Defendants Charged	290	259	235	259	215
Cases Charged	217	197	173	177	168
Defendants Sentenced	287	242	223	207	208
No Prison Term	148	107	126	121	102
1-12 Months	52	48	35	38	27
13-24 Months	37	45	29	27	33
25-36 Months	20	20	6	10	17
37-60 Months	14	19	18	7	21
60+ Months	16	3	9	4	8

Figure 9. Prediction results visualization: (**a**) test form image; (**b**) line segmentation mask prediction map; (**c**) column segmentation mask prediction map; (**d**) visualization of the results.

In the previous experiments, we only added a layer of the SE module to the row–column aggregation module. Considering that the SE module only operates on channels and does not change the size of the original feature map, multiple layers can be added. In order to verify the effect of multi-layer SE modules on the model, we designed ablation experiments according to the addition of different layers. The experimental results are shown in Table 6.

Table 6. SE module layers ablation experiment.

Model	SE Layers		Row			Column			Average		
	Row	Column	P	R	F1	P	R	F1	P	R	F1
LRCANet	0	0	0.9066	0.9152	0.9109	0.9166	0.9818	0.9481	0.9116	0.9485	0.9295
LRCANet	1	1	0.9300	0.9434	0.9366	0.9348	0.9922	0.9626	0.9324	0.9678	0.9496
LRCANet	2	1	0.9253	0.9284	0.9268	0.9329	0.9766	0.9539	0.9291	0.9525	0.9404
LRCANet	2	2	0.8806	0.9075	0.8938	0.9382	0.9870	0.9620	0.9094	0.9473	0.9279

Retailer	Own Brands Market Shares
Monoprix	28%
Casino	25%
Intermarché	23%
Carrefour	22%
Auchan	19%
Leclerc	10%

(**a**)

Retailer	Own Brands Market Shares
Monoprix	28%
Casino	25%
Intermarché	23%
Carrefour	22%
Auchan	19%
Leclerc	10%

(**b**)

Figure 10. Comparison before and after adding SE module (**a**) without SE module; (**b**) with SE module.

From the experimental results in Table 6, it can be seen that the multi-layer SE module does not bring better prediction results. When we add two and one layers of SE modules to the rows and columns aggregated modules, respectively, the overall prediction effect decreases. When we add two layers of SE modules to both rows and columns aggregated modules, the overall prediction effect also decreases. On balance, the model has the highest performance when only one layer of the SE module is added.

At the same time, we compared and analyzed the model complexity before and after the improvement, checked the size of the generated model weight file, and used Python language to obtain the model parameter quantity and model calculation volume. We took MByte as the unit of model volume. We used millions as the unit of model parameter quantity, and used giga floating point operations per second (GFLOPs) as the unit of model calculation volume. One GFLOPs is equivalent to one billion floating point calculations per second. The comparison results are shown in the following table.

From the experimental results in Table 7, it can be seen that compared with the original model, our improved model has a very good model reduction in the three indicators of model volume, model parameter quantity, and floating point number of operations. Compared with the original model, the volume of this model is reduced by 98%, the number

of model parameters is reduced by 99%, and the number of floating point calculations is reduced by 96%. A comprehensive analysis of the results in Tables 5 and 7 shows that our proposed method LRCAANet greatly reduces the number of parameters, volume, and calculation of the model while ensuring a low-performance loss of the model, making the model more lightweight.

Table 7. Model complexity comparison.

	Model Volume (MByte)	Model Parameters (Million)	Model Calculation Volume (GFLOPs)
RCANet (Recurrence)	45.5	11.35	16.62
LRCAANet	0.81	0.17	0.64

4. Conclusions

Aiming at the problem of large size and large parameters of the table recognition model, this paper improves the table structure recognition method RCANet based on the lightweight network. We replaced the original backbone network ResNet18 with a lightweight network ShuffleNetv2, and introduced the SE module into the rows and columns aggregated modules, which strengthens the learning between feature channels, generates feature information with channel attention, and improves the performance of lightweight models. Finally, we experimentally verify the effectiveness of the lightweight table recognition method LRCAANet proposed in this paper. Under the premise of ensuring a low-performance loss of the model, the model volume, model parameters, and floating point operations are reduced by more than 95% compared to the original model. The final model size is only 0.81 M, and the number of model parameters is only 1.7 million.

The performance of our model may fall short compared to some of the more advanced work. However, from what we know about the model size and the number of model parameters of some advanced works, we can see that the number of parameters and the volume of our model achieve advanced results. For example, the model volume of LGPMA is as high as 177 M, while our model is only 0.81 M. Due to the relatively small number of publicly available row mask datasets, it is difficult to make a comprehensive comparison of our model with most advanced models. Therefore, the main experiments in this paper are compared with the original model RCANet, and from the previously mentioned results, we can see that our model LRCAANet achieves a huge improvement in terms of volume and number of parameters compared to RCANet.

However, our model has only been experimented on smaller datasets so far, and we may produce more row mask segmentation datasets in the future for research purposes to explore the performance of the model on large datasets and further improve the model and enhance its performance based on the experimental results. At the same time, we will consider how to compress the advanced table recognition models to be smaller, drawing on the algorithmic ideas of the currently available advanced work. In recent years, many table recognition models have introduced transformer models, such as TableFormer, and in the future we may consider adding transformer models to the model and adopting some strategies to compress the size of the model. Further improving the performance of the lightweight model will be our main work in the future, and we are looking forward to further exploring a smaller, mobile-friendly table structure recognition model that can guarantee the performance of the model while being lightweight.

Author Contributions: Conceptualization, T.Z. and R.S.; methodology, T.Z., F.S., and S.W.; software, T.Z. and Y.S.; validation, T.Z. and R.S.; formal analysis, T.Z. and R.S.; investigation, T.Z.; resources, T.Z. and R.S.; data curation, T.Z. and R.S.; writing—original draft preparation, T.Z.; writing—review and editing, T.Z. and R.S.; visualization, T.Z.; supervision, T.Z. and R.S.; project administration, T.Z.; funding acquisition, Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by Young Scientists Fund of the National Natural Science Foundation of China (41706198) and Qingdao Independent innovation major special project (21-1-2-1hy).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All Numerical and graph is available in the manuscript and dataset will be provided on request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kim, Y.-S.; Lee, K.-H. Extracting Logical Structures from Html Tables. *Comput. Stand. Interfaces* **2008**, *30*, 296–308. [CrossRef]
2. Masuda, H.; Tsukamoto, S.; Yasutomi, S.; Nakagawa, H. Recognition of Html Table Structure. In Proceedings of the IJCNLP, Hainan Island, China, 22–24 March 2004.
3. Fischer, P.; Smajic, A.; Abrami, G.; Mehler, A. Multi-Type-Td-Tsr–Extracting Tables from Document Images Using a Multi-Stage Pipeline for Table Detection and Table Structure Recognition: From Ocr to Structured Table Representations. In Proceedings of the German Conference on Artificial Intelligence (Künstliche Intelligenz), virtual, 27 September–1 October 2021.
4. Qiao, L.; Li, Z.; Cheng, Z.; Zhang, P.; Pu, S.; Niu, Y.; Ren, W.; Tan, W.; Wu, F. Lgpma: Complicated Table Structure Recognition with Local and Global Pyramid Mask Alignment. In Proceedings of the International Conference on Document Analysis and Recognition, Lausanne, Switzerland, 5–10 September 2021.
5. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-Cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
6. Qasim, S.R.; Mahmood, H.; Shafait, F. Rethinking Table Recognition Using Graph Neural Networks. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 20–25 September 2019.
7. Kim, P. Convolutional Neural Network. In *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence*; Spring: Berlin/Heidelberg, Germany, 2017; pp. 121–147.
8. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The Graph Neural Network Model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef] [PubMed]
9. Xue, W.; Yu, B.; Wang, W.; Tao, D.; Li, Q. Tgrnet: A Table Graph Reconstruction Network for Table Structure Recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021.
10. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
11. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907.
12. Khan, S.A.; Khalid, S.M.D.; Shahzad, M.A.; Shafait, F. Table Structure Extraction with Bi-Directional Gated Recurrent Unit Networks. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 20–25 September 2019.
13. Medsker, L.R.; Jain, L.C. Recurrent Neural Networks. *Des. Appl.* **2001**, *5*, 64–67.
14. Mikolov, T.; Karafiát, M.; Burget, L.; Cernocký, J.; Khudanpur, S. Recurrent Neural Network Based Language Model. In Proceedings of the Interspeech 2010, Makuhari, Japan, 26–30 September 2010.
15. Zaremba, W.; Sutskever, I.; Vinyals, O. Recurrent Neural Network Regularization. *arXiv* **2014**, arXiv:1409.2329.
16. Dey, R.; Salem, F.M. Gate-Variants of Gated Recurrent Unit (Gru) Neural Networks. In Proceedings of the 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), Boston, MA, USA, 6–9 August 2017.
17. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A Review of Recurrent Neural Networks: Lstm Cells and Network Architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef] [PubMed]
18. Siddiqui, S.A.; Khan, P.I.; Dengel, A.; Ahmed, S. Rethinking Semantic Segmentation for Table Structure Recognition in Documents. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, Australia, 20–25 September 2019.
19. Shen, X.K.; Bao, Y.L.; Zhou, Y.; Liu, W. Rcanet: A Rows and Columns Aggregated Network for Table Structure Recognition. In Proceedings of the 2022 3rd Information Communication Technologies Conference (ICTC), Nanjing, China, 6–8 May 2022; pp. 112–116.
20. Nassar, A.; Livathinos, N.; Lysak, M.; Staar, P. Tableformer: Table Structure Understanding with Transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LO, USA, 18–24 June 2022.
21. Ma, C.; Lin, W.; Sun, L.; Huo, Q. Robust Table Detection and Structure Recognition from Heterogeneous Document Images. *Pattern Recognit.* **2023**, *133*, 109006. [CrossRef]
22. Raja, S.; Mondal, A.; Jawahar, C. Table Structure Recognition Using Top-Down and Bottom-up Cues. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020.
23. Ye, J.; Qi, X.; He, Y.; Chen, Y.; Gu, D.; Gao, P.; Xiao, R. Pingan-Vcgroup's Solution for Icdar 2021 Competition on Scientific Literature Parsing Task B: Table Recognition to Html. *arXiv* **2021**, arXiv:2105.01848.
24. Ma, N.; Zhang, X.; Zheng, H.-T.; Sun, J. Shufflenet V2: Practical Guidelines for Efficient Cnn Architecture Design. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
25. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.

26. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
27. Göbel, M.; Hassan, T.; Oro, E.; Orsi, G. Icdar 2013 Table Competition. In Proceedings of the 2013 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013.
28. Bradski, G. The Opencv Library. *Dr. Dobb's J. Softw. Tools Prof. Program.* **2000**, *25*, 120–123.