Cloud Computer Research on Table Detection Model Based on the DC-LSTM Model

Xingming Zhang, Yunchao Bai*, Naike Wei, Huadong Pan and Jun Yin
ZheJiang Dahua Technology CO., LTD., LTD., Hangzhou, China

*Corresponding author email: bai_yunchao@dahuatech.com

Abstract. In view of the fact that tables are not easy to detect, this paper designs a table detection model based on the DC-LSTM module, which references the Convolutional Long Short-Term Memory (ConvLSTM). The model uses the backbone network of target detection to extract convolution features. The feature pyramid network is used to complete the detection task, and the DC-LSTM module is embedded in a special position in the feature pyramid network. In order to evaluate the performance of the DC-LSTM module, we added the DC-LSTM module to the YOLO v3, SSD network. Specifically, we added the DC-LSTM module to the YOLO v3 network, the new model can achieve an accuracy of more than 98% on the TableBank data set and own data set. The model in this paper can realize the automatic extraction of table document images, which is of great significance for the realization of automated data collection.

Keywords: Deep learning; Table detection; The DC-LSTM module.

1. Introduction

In many document analysis applications, the detection and identification of tables is an important task because tables typically present basic information in a structured manner. These table data are very important for the research and analysis of market [1]. How to obtain data and analyze the law of data in a short time is very important. If manual entry is used, it will not meet the market demand, so it is very important to use automatic entry. Due to the variety of forms in the business reports, these tables include borders and no borders, with separate lines and no dividers inside the table. Therefore, it is not effective to use the straight line method to detect the form [2]. The rapid development of object detection technology, has significantly boosted the data-driven image-based approaches for table analysis. But there are many types of tables, and the general target detection model is not effective enough to extract the features of the table based on the convolution, so it is still difficult to expand in practical applications.

Deep Convolutional Neural Networks (CNN) have recently implemented general object detection [3]. The current object detection technology has become increasingly mature, Faster R-CNN, SSD and YOLO can meet people's needs for ordinary object detection [4][5][6]. However, ordinary targets have a well-defined closed boundary, but there may not be such a well-defined boundary in the table because the table is composed of many word groups or one sentence. In contrast, table recognition is a regional identification that requires extracting the relationship between the table and the context and the relationship within the table.

For table detection, it is best to learn the spatial characteristics of the table. If we want to get the spatial characteristics inside the table, it is difficult to achieve the effect using only the traditional convolution method. Therefore, this paper considers the use of Recurrent Neural Networks. The main
reason is that Recurrent Neural Networks have a certain memory function and can process time series data or spatial data [9]. But ordinary RNNs only have short-term memory functions, and they will produce gradient explosion or gradient disappearance problems when processing long sequences. In order to solve this problem, we introduced the ConvLSTM. The ConvLSTM is an improved version of LSTM, and it adds convolution operation based on the original LSTM [10]. This way, we can not only learn time series data, but also process two-dimensional spatial data. In the work of this paper, we have designed a two-dimensional spatial the ConvLSTM module for learning the spatial characteristics inside a table. To achieve this goal, we have added the ConvLSTM in both horizontal and vertical directions. Finally, we stitch the original convolutional layer and the output layer after the ConvLSTM. The concatenated results are processed by convolution, Batch Normalization, and Leaky Relu, then the final results are output. We call it the Double Direction ConvLSTM Module (DC-LSTM Module).

2. Related Work

Accurate positioning and searching of forms are the biggest difficulty of form detection. The research on table detection dates back to the 1990s. Watanabe and colleagues presented an article that used classification trees to identify the structure of table layouts [11]. With the continuous development of technology, convolutional shared computing has received increasing attention due to its efficient and accurate visual recognition. Deep convolutional neural networks have become a better method for detecting tables. For example, Minghao Li uses Faster R-CNN for common table detection [13]. Basilios Gatos et al. proposes inputting word bounding box information and then outputting text block logical units to identify the table environment and cells [14]. Although these methods solve some of the problems in table detection, with the widespread use of deep convolutional neural networks, we find that the shortcomings of table detection models are also increasing. For example, the shapes of the tables are different, and there are many types of tables, which bring great difficulties and interference to the detection of the tables.

The feature pyramid network is a basic component in a recognition system for detecting objects of different scales. The feature pyramid constructing module can be easily revised and fit into state-of-the-art Deep Neural Networks based detectors. MS-CNN, SSD, FPN and RefineDet adopted this strategy in different ways [15]. However, we found through experiments that table features learned using only convolutional neural networks are not generalized and robust. The main reason is that the features cannot be extracted to the table through the convolution operation. Therefore, many people have used recurrent convolutional networks to learn more feature spatial position relationships. For example, Sean Bell and others proposed Inside-Outside Net [10]. This network uses RNNs to encode pixels on the top, bottom, left, and right of the target, and distinguishes the target from the background in this way.

Because the gradient disappears and the gradient explosion is easy to occur during the training of the RNN, this causes great trouble to the model training. However, the LSTM can solve the problems encountered by the RNN very well. The LSTM mainly solves the problem of one-dimensional sequences, but cannot be used directly for two-dimensional images. The ConvLSTM just solves this problem. The core essence of the ConvLSTM is the same as LSTM, with the output of the previous layer as the input of the next layer. The difference is that after the ConvLSTM joins the convolution operation, it can not only get the timing relationship, but also extract the spatial features like the convolution layer. In this way, the ConvLSTM can extract both temporal and spatial features (spatial-temporal features), and the state-to-state switching is replaced by a convolution operation. This paper combines the feature pyramid network and the ConvLSTM ideas to construct a network model for extracting the internal features of tables of different sizes.

3. Model Construction

3.1. The DC-LSTM Module

In this article, we use the DC-LSTM Module as a mechanism for learning table features, and combine it with a feature pyramid network for table detection. The structure of DC-LSTM module is shown in
the figure below. As shown, we add two ConvLSTMs in both the horizontal (x) and vertical (y) directions. Their convolution kernels are 3x3. Then the original convolution layer is combined with the output of the ConvLSTM. The splicing result is passed through a 1x1 convolution layer, and finally outputted after a BatchNormalization and Leaky ReLu. We combine the feature map output by the ConvLSTM and the feature map output by convolution, so as to ensure that the feature data extracted by the original convolution layer will not be changed, and spatial features are also added. Finally, the output is incorporated into the backbone network again. We call this structure the DC-LSTM module.

Figure 1. Structure of the DC-LSTM module.

However, the DC-LSTM module is only a module that assists with table detection, and the output result is not the final result of table detection. The DC-LSTM module continues to extract table information under the convolution feature map, and uses the ConvLSTM module to learn the spatial features of the table, and finally fuses the extracted spatial features with the original convolution features. The output of the DC-LSTM module realizes the fusion of the target's type features and spatial features, and then feeds these features into the regression function or classification function to achieve the detection and classification of the target. The iterative update process of the DC-LSTM module is as follows. Assume that the output of the convolutional feature network is:

\[ z_{i,j,k} = \sum_{l,m,n} V_{i,j,m+l-1,n+l-1} K_{i,j,m,n} \]  

(1)

The result of the convolution output will be fed into the DC-LSTM module, and the new formula of the ConvLSTM in the DC-LSTM module is as follows:

\[ i_t = \sigma(W_{ix} Z_t + W_{ih} H_{t-1} + W_{ic} o C_{t-1} + b_i) \]  

(2)

where \( i_t \) is the sigmoid layer of the input gate that determines the value to be updated.

\[ f_t = \sigma(W_{xf} Z_t + W_{hf} H_{t-1} + W_{fc} o C_{t-1} + b_f) \]  

(3)

where \( f_t \) is the output of the sigmoid layer of the forget gate.

\[ C_t = f_t \circ C_{t-1} + i_t \circ \tanh(W_{xc} Z_t + W_{hc} H_{t-1} + b_c) \]  

(4)

where \( C_t \) is to update the old cell state to the new cell state.

We use \( o_t \) to filter the output. Then we use a sigmoid layer to decide which parts of the cell state are to be output.

\[ o_t = \sigma(W_{ox} Z_t + W_{ho} H_{t-1} + W_{oc} o C_t + b_o) \]  

(5)

Finally, we scale the state cell to a value between -1 and 1 through \( \tanh \), and multiply the output of the sigmoid gate to determine the output part.

\[ H_t = o_t \circ \tanh(C_t) \]  

(6)

where ‘x’ denotes the convolution operator and ‘◦’ denotes the Hadamard product. So the output after passing through the DC-LSTM module is:

\[ z'_{i,j,k} = \sum_{l,m,n} (H_{i,j,m+l-1,n+l-1} + V_{i,j,m+l-1,n+l-1}) K_{i,j,m,n} \]  

(7)

3.2. The Insertion Position of the DC-LSTM Module

We mentioned that the DC-LSTM module is an auxiliary module, so we need to embed it in the target detection model. The tables are not uniform in size, the length-to-width ratio is large across domains,
and the training set of the tables cannot include all scales, we can detect objects from the pyramid features extracted from the inherent layers in the network, and solve the problem of different table sizes through multi-scale feature maps. Therefore, we embed the DC-LSTM module in the feature pyramid network. The convolutional neural networks have different levels of semantic features at different depths. The shallow network has high resolution, learns more detailed features, with low-level detailed semantic information and smaller receptive fields. Deep networks have low resolution and learn more semantic features with advanced semantic information and larger receptive fields. However, for table detection, the distance between the text in the picture and the line spacing will cause some interference to the table detection, especially for the output of the shallow network. For the output of deep networks, because the semantic information is high, it is easy to confuse the semantic information of tables with the semantic information of paragraphs.

The DC-LSTM module can improve the difference between text and tables in pictures, and put the processed results on the output layer after simple fusion processing, which will improve the generalization ability and robustness of the model. But it's not that the more the DC-LSTM module is added, the better, nor can it be added anywhere in the feature pyramid model. If the DC-LSTM module is added after the output of the deep network, the deep convolution has higher semantic information and less information can be learned. If you add the DC-LSTM module after the output of the shallow network, it can learn more information. And it will correlate the horizontal and vertical information of the table to improve the receptive field of convolution. Therefore, for the table detection, the DC-LSTM module is embedded on the shallow network, and the effect may be better.

4. Experiments

4.1. Experiment Preparation
In this section, we will introduce the test results of table detection on our dataset and Tablebank dataset. Among them, our dataset sets produced by ourselves are mainly tables in the financial statements of some companies. The format of the tables is very diverse, with a total of 9,736 table pictures. Tablebank is an image-based table detection and recognition data set. It is based on Word and
Latex documents on the Internet and uses a novel weak supervision mechanism to construct. It contains 417K high-quality label tables. We tested the ability of different models to detect tables, including Faster R-CNN model, SSD model, Yolo V3 model, etc. At the same time, we add the DC-LSTM module to different feature pyramid models, and we test the ability of these models to detect tables again. In order to verify the ability of each model to detect forms, we mainly evaluate the accuracy (AP) of detection forms, because there is only one kind of detection object.

4.2. Experimental Results

We put all the experimental results in Table 1. These experiments include adding DC-LSTM module in different locations for different models and the same model. Then we compare the experimental results of all models, and compare the accuracy of table detection after adding the DC-LSTM module in different locations. As shown in Table 1, the experimental results of the Faster R-CNN model are shown first. As a classic two-stage target detection model, the accuracy of the model has been recognized by most people. Then we look at the experimental results of the Tiny-YOLO network detection table. Tiny-YOLO removes some feature layers based on YOLO v3, and only retains the model of 2 independent prediction branches. The table shows the accuracy of the table detection after adding the DC-LSTM module to the "Bridge" position of the Tiny-YOLO network. First, we observe that adding the DC-LSTM module to b1 or b2 improves the detection rate of the table, but the accuracy of model detection by adding the DC-LSTM module at b1 is the highest. It is worth noting that the network added to the DC-LSTM module at both b1 and b2 has a lower accuracy rate of table detection than Tiny-YOLO.

At the same time, we also added the DC-LSTM module in the position of "Bridge" of YOLO v3 network and SSD network. The experimental results are shown in the following table. Among them, the YOLO v3 model uses multi-scale features for object detection. It is a typical feature pyramid structure network. It has three feature network output layers. According to the table data, it is known that the accuracy of the two data sets is highest at b1. The SSD also extracts feature maps of different scales for detection. We only add the DC-LSTM module after the SSD's backbone network. As seen in Table 1, the result of adding the DC-LSTM module to the YOLO v3 backbone network is very low. In fact, we have also added the DC-LSTM modules on other backbone networks, such as Darknet53 and VGG16, and found that the experimental results were not very satisfactory. The accuracy rate is lower than without the DC-LSTM module, so those data are not shown in the table.

| Method                   | Self-Made AP | TableBank AP |
|--------------------------|--------------|--------------|
| Faster R-CNN             | 85.45%       | 90.14%       |
| Tiny-YOLO v3             | 44.34%       | 53.12%       |
| DC-LSTM at b1            | 71.42%       | 72.22%       |
| DC-LSTM at b2            | 61.74%       | 63.90%       |
| DC-LSTM at b1 and b2     | 56.87%       | 60.23%       |
| YOLO v3                  | 83.56%       | 86.00%       |
| DC-LSTM at b1            | 98.14%       | 98.82%       |
| DC-LSTM at b2            | 91.33%       | 93.25%       |
| DC-LSTM at b3            | 86.45%       | 87.19%       |
| DC-LSTM at b1 and b2     | 85.12%       | 86.88%       |
| DC-LSTM at Backbone      | 83.44%       | 85.69%       |
| SSD                      | 72.23%       | 75.89%       |
| DC-LSTM at b1            | 81.98%       | 82.06%       |
| DC-LSTM at b2            | 78.37%       | 79.13%       |
| DC-LSTM at b1 and b2     | 72.89%       | 76.24%       |

Through the comparison of the above experimental data, adding the DC-LSTM module to the 'Bridge' structure of the feature pyramid network can improve the accuracy of table detection. In particular, the lifting effect is most obvious on low-level features, and the lifting effect is weaker on smaller feature-scale layers. These are not limited to the model structure. The accuracy of table detection has been greatly improved by adding the DC-LSTM module to the y3 output of YOLO v3. Such experimental
results are almost consistent with our previous analysis. However, it is not that the greater the number of the DC-LSTM modules added, the higher the accuracy of table detection. If a large number of the DC-LSTM modules are added to a simple feature pyramid model, the accuracy of this model detection table will be reduced. These experiences need to be determined based on the network structure and tasks. In this article, we performed evaluation experiments on both the TableBank dataset and our own dataset. Based on the experimental data, we can see that the accuracy of various detection models on the homemade financial dataset is not as good as the TableBank dataset. In order to find the reason, we used the same amount of data for testing and found that the effect was unchanged. Therefore, we don't think it is caused by the amount of data, but because the table in the TableBank dataset is relatively single.

In this paper, we compare the experimental results of the Faster R-CNN with the experimental results of various models. We found an interesting phenomenon. In the original model, the Faster R-CNN network detection table has the highest accuracy, and the Tiny-YOLO network detection table has the worst accuracy. After the YOLO v3 model added the DC-LSTM module at y3, the accuracy of table detection was higher than Faster R-CNN. At the same time, we compared the detection time on the same equipment. Although the detection time of each model after adding the DC-LSTM module has increased, it is still faster than the Faster R-CNN.

4.3. Experiment Preparation
In order to prove that the DC-LSTM module can learn the internal structure of the table, we choose the optimal results of each model and compare them with Faster R-CNN. As shown in the Figure 4, the data in the figure show the results of three different models on the TableBank dataset and the homemade dataset. The accuracy of the original model for the financial statement data set detection is much lower than the accuracy rate on the TableBank data set. It can be seen that the generalization of ordinary convolutional networks is not strong enough to effectively notice the internal structure of the table. The accuracy of the model after adding the DC-LSTM module on the two data sets is almost close. This shows that the DC-LSTM module can enhance the generalization ability of the model for table detection.

![Figure 3. The Location of the "Bridge".](image)

Different models have different accuracy rates for table detection, and the same model has different accuracy rates for table detection after adding the DC-LSTM module at different locations. According to the experimental data on the homemade financial statement data set, the accuracy rate of the YOLO v3 model after adding the DC-LSTM module at the y3 position is 98.14%, which is far more than 88.45% of Faster R-CNN. At the same time, the accuracy of adding the DC-LSTM module at b1 of the SSD model is also close to Faster R-CNN. Therefore, adding the DC-LSTM module to the 'Bridge' structure in all feature pyramid models improves the table detection to some extent. The experimental results verify that the DC-LSTM module has certain advantages in form recognition, especially in the structure of the "Bridge" of the feature pyramid network.

5. Conclusion
This article introduces the DC-LSTM module for table detection and embeds it in a pyramid model. We found that after adding LSTM to the convolution, the feature pyramid model can learn the unique features of the table. In order to verify the superiority of our method, we have done a lot of comparative experiments. This includes putting the DC-LSTM module on different pyramid models.
and different locations on the same pyramid model, etc. The experiments also prove that the method in this paper has strong adaptability and robustness. Applying this method to the detection and extraction of tables will greatly optimize the analysis process of data, reduce waste of manpower and material resources, and will play a significant role in the future Intelligence System field.

References

[1] Vasilopoulos N, Kavallieratou E. Complex layout analysis based classification and morphological operations[C]// Hellenic Conference on Artificial Intelligence. ACM, 2016:32
[2] Duda R O, Hart P E . Use of the Hough transformation to detect lines and curves in pictures [J]. Communications of the ACM. 1972, 15(1):11-15
[3] Yu-Hsin Chen, Tushar Krishna, Joel S. Emer. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks[C]. IEEE International Solid State Circuits Conference. IEEE, 2016
[4] Ren, Shaoqing, He, Kaiming, Girshick, Ross. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015, 39(6):1137-1149
[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan. SSD: Single Shot MultiBox Detector[C]// European Conference on Computer Vision. Springer International Publishing, 2016
[6] Shafiee, Mohammad Javad, Chywl, Brendan, Li, Francis. Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video[J], 2017
[7] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[J], 2018
[8] Tariq Alrifai. The Next Financial Crisis and the New Financial System[M]. John Wiley & Sons Singapore Pte. Ltd, 2015
[9] Bell, Sean, Zitnick, C. Lawrence, Bala, Kavita. Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks[J], 2015
[10] Kim, Seongchan, Hong, Seungkyun, Joh, Minsu. DeepRain: ConvLSTM Network for Precipitation Prediction using Multichannel Radar Data[J], 2017
[11] Watanabe T, Luo Q, Sugie N. Layout recognition of multi-kinds of table-form documents[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995, 17(4):0-445
[12] Stubenrauch, C. J, Del Genio, A. D, Rossow, W. B. Implementation of Subgrid Cloud Vertical Structure inside a GCM and Its Effect on the Radiation Budget[J]. Journal of Climate, 10(2):273-287
[13] Li, Minghao, Cui, Lei, Huang, Shaohan. TableBank: Table Benchmark for Image-based Table Detection and Recognition[J], 2019
[14] Basílios Gatos, Dimitrios Danatsas, Ioannis Pratikakis. Automatic Table Detection in Document Images[C]. Pattern Recognition and Data Mining, Third International Conference on Advances in Pattern Recognition, ICAPR 2005, Bath, UK, August 22-25, 2005, Proceedings, Part I. Springer-Verlag, 2005
[15] Zhang, Shifeng, Wen, Longyin, Bian, Xiao. Single-Shot Refinement Neural Network for Object Detection[J], 2017