Biometrika
written in biometrika style Article written on June 10, 2021

OCR for Handwritten Digits

BY DIVYANSHU BHAIK, AYUSH KESARWANI AND VAISHNAVI MADHEKAR
Under Menon Labs, London, UK for Optidash, Germany
dvbhaik@gmail.com kaayush112@gmail.com madhekarvaishnavi@gmail.com

SUMMARY

Handwritten digit recognizer models are usually trained on a fixed set of classes, so the model would locate and classify only those classes in the image. Also, the location of the object is generally in the form of a bounding rectangle. So, this involves both localization of the object in the image and classifying that object. In Region-Based methods, the first objective is to find all the regions which have the objects and then pass those regions to a classifier, which gives us the locations of the required objects. So, it is a two-step process. Firstly, it finds the bounding box and afterward, the class of it. whereas Single Shot detectors, however, predict both the boundary box and the class at the same time. Being a single-step process, it is much faster. SSD and YOLO are Single Shot detectors. YOLOv3 (You Only Look Once, Version 3) has accuracy on par with the contemporary object detection algorithms while being faster than the other algorithms by a wide margin. So, we have employed the YOLOv3 architecture to carry out our task.

1. ALGORITHM FOR OCR

The whole algorithm for making an OCR that can detect the Handwritten digits includes the use of YOLO algorithm and then then an algorithm for series prediction. The Training and implementation of the YOLO algorithm for this task are explained in this article along with the introduction and key ideas for the series prediction part. So the whole work can be broadly divided into four parts which are:

1) Making of the dataset
2) YOLO Algorithm
3) Training the model
4) Implementation and Further Algorithm

Each of these parts are explained in the subsections below.

1·1. *Making of the dataset*

The making of a dataset for the YOLO algorithm is a very important task because this training dataset will make the algorithm understand that what it gave to detect and in what kind of environment it has to do so. Likewise, in our case we have to detect the handwritten digits from any hand-filled form. So for making the YOLO algorithm understand this thing we need to provide the algorithm with thousands of sample images as input and their labels as the class of the digits along with the place where they are placed in the image and the size of the bounding boxes covering those digits. For preparing this dataset we used the mnist digits to imitate the written digits in form and some images of random forms which will be considered as the background/noise images. Then Using a script named make_data.py we make the final dataset along with the y labels of the digits in the image.

Fig. 1. sample of the training images produced during the making of dataset.

_localization\YOLOv3_for_Handwritten_OCR\mnist\mnist_train\000036.jpg 152,654,180,682,1 174,654,202,682,5 196,654,224,682,3 218,654,246,682,0

Fig. 2. sample of the label for the training image produced during the making of dataset.

In Fig. 1 a sample of the training image produces during the making of the dataset can be seen where we have a random form/document as a background/noise and on it a series of handwritten number is placed. These numbers are taken from the mnist dataset and then aligned in such a manner that it can form a sense for human brain in the form of a written number. This was all about the input image to the YOLO model. Now as explained earlier, in order to detect where the numbers are in the image we also need to give the labels to the model for it's supervised learning. So, for this purpose along with the making of input images for the dataset, we also produced the related labels for the image.

As shown in Fig. 2, which is a part of the label used for the same image as shown in Fig. 1, the sample of the labels produced during the making of the dataset, includes the address of the related image along with a vector of data. Each vector is associated with each numbers which are placed in the same image. This vector include the position of the digit in the image i.e. bx and by, the dimensions of the bounding box i.e. bh and bw and the identification/label of the digit. This is a very important part for this project as it will make the YOLO model understand that what and where it have to detect the concerned object. Also, it should be noted that the more realistic and accurate the data is, the more accurate will be the performance of the YOLO algorithm.

1·2. *YOLO Algorithm*

YOLO is an algorithm that uses neural networks to provide real-time object detection. It has been used in various applications to detect traffic signals, people, parking meters, and animals, etc. Here, Fig. 3 gives the diagram of the neural network which is used during the making of the YOLO algorithm.

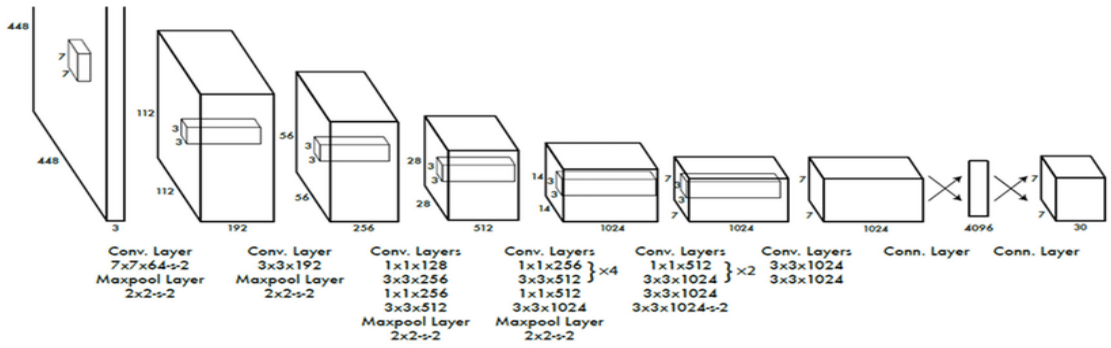

Fig. 3. Diagram of neural network used in making of the YOLO algorithm.

Object detection is a phenomenon in computer vision that involves the detection of various objects in digital images or videos. The main purpose of this algorithm is to what the object is? and where it is located? Although, Object detection consists of various approaches such as fast R-CNN, Retina-Net, and Single-Shot MultiBox Detector (SSD), but the approach of YOLO algorithm is much more faster than these mentioned approaches in detection the object in real-time and hence it is also used in this project also. YOLO is a fast algorithm along with a high accuracy result. Not only this, but this algorithm has excellent learning capabilities that enable it to learn the representations of objects and apply them in object detection.

YOLO is an abbreviation for the term 'You Only Look Once'. This is an algorithm that detects and recognizes various objects in a picture (in real-time). Object detection in YOLO is done as a regression problem and provides the class probabilities of the detected images. YOLO algorithm employs convolutional neural networks (CNN) (as mentioned in Fig. 3) to detect objects in real-time. As the name suggests, the algorithm requires only a single forward propagation through a neural network to detect objects. This means that prediction in the entire image is done in a single algorithm run. The CNN is used to predict various class probabilities and bounding boxes simultaneously.

Now, coming to the working of the YOLO algorithm. This algorithm mainly uses following three techniques.

1) Residual blocks
2) Bounding box regression
3) Intersection Over Union (IOU)

In **Residual blocks** first, the image is divided into various grids. Each grid has a dimension of S x S. Here, Fig. 4 shows how an input image is divided into grids.

A **bounding box** is an outline that highlights an object in an image. Every bounding box in the image consists of Width (bw), Height (bh), Class (c) and Bounding box center (bx,by). Here, Fig. 5 shows an example of a bounding box.

Intersection over union (IOU) is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the objects perfectly. Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The IOU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box. The image in Fig. 6 provides a simple example of how IOU works.

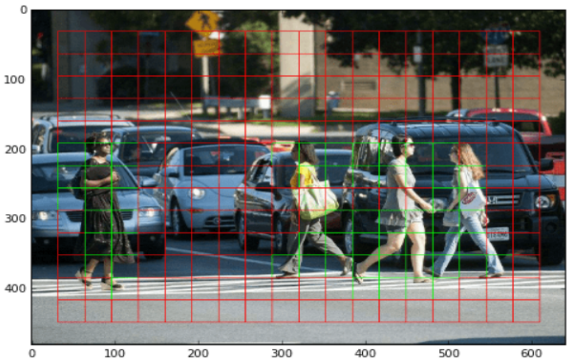

Fig. 4. Image showing the concept of Residual Blocks.

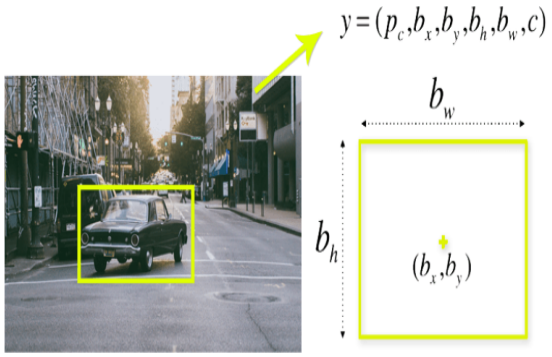

Fig. 5. Image showing the concept of Bounding Box. The bounding box has been represented by a yellow outline.

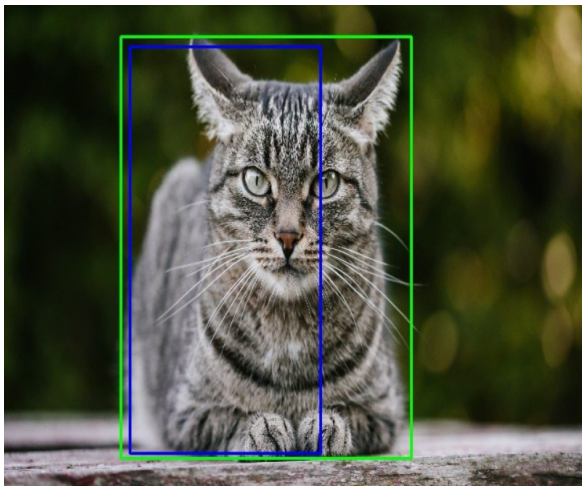

Fig. 6. Image showing the concept of Intersection over union (IOU).

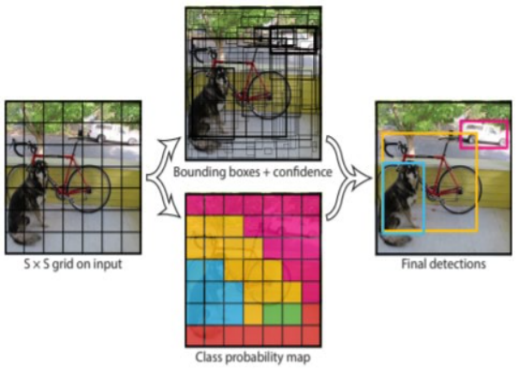

Fig. 7. Image showing that how the mentioned three techniques are applied to produce final detection results.

In the end all the mentioned three techniques are combined in order to implement the YOLO algorithm over the trained neural network mentioned earlier. Image shown in Fig. 7 gives the idea about how the combination of all three techniques is implemented.

1·3. *Training the model*

The training of the model for YOLO is simple. Once the YOLO algorithm is coded, we need to compile the model and feed the training data to it. one main thing to notice here is that we used the transfer learning for the training part and used mAP and total loss as the criteria for evaluation instead of accuracy. These is more important criteria in problems of object detection like this instead of accuracy.Also along with this we Changed the learning rate (gradually increasing and decreasing) during the training procedure to make the training more efficient.

$$mAP = 1/n \sum_{k=1}^{n} AP_k \tag{1}$$

$$AP = \sum_{k=0}^{n-1} [Recalls(k) - Recalls(k+1)] * Precisions(k) \tag{2}$$

The main thing to understand here is mAP which elaborates to Mean Average Precision. The general definition of mAP says that it is calculated by taking mean AP over all classes and overall IoU (Intersection over Union) thresholds.Here AP is Average Precision which is measure that combines recall and precision for ranked relative results. The equation (1) and (2) are the equations of mAP(mean Average Precision) and AP(Average Precision) respectively. In equation (1) AP_k = AP of class k and n = the number of classes and in equation (2) Recalls(n) = 0, Precision(n) = 1 and n = number of thresholds. At the end of our training we had 98.951% mAP score.

1·4. *Implementation and Further Algorithm*

Once the model is trained we need to develop an algorithm for the implementation of the trained model in such a way that it works as an OCR for the provided document image as an input. Currently this part of the OCR development is implementing a simple algorithm which has a possibility of Improvement. In current situation the algorithm first splits the input image

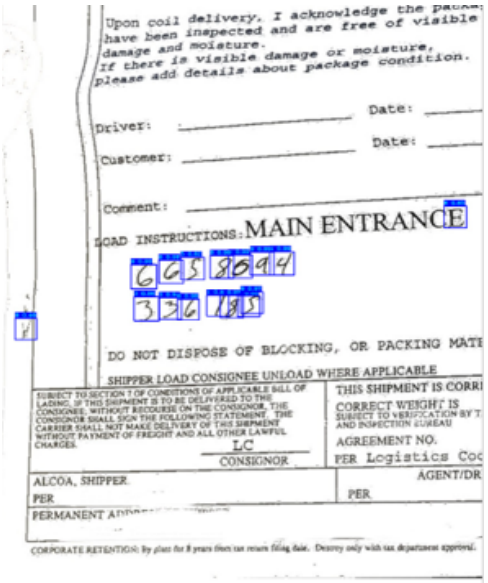

Fig. 8. Output image from the sliced input image once passed from the model.

into 4 sub-parts which are then provided as an input to the trained model. One thing to keep in mind is the resolution of the input image and the image which is fed into the model because the model is trained on a certain resolution of 864x864. So splitting a high resolution image into sub parts while keeping in mind the spaces between the written digits will also expand while splitting the original image.

Once the image is passed through the model it will localize every detected handwritten digit in the frame at once while giving the location, confidence scores and predicted the identity of the digits as an output. After this a threshold is also used on the confidence scores to filter out the miss-detected patterns as digits. As shown in Fig. 8 the output image has some miss-detections, These miss detections are:

1) Miss classification of a number.
2) Not able to detect and classify the number.
3) Miss classifying other printed text as handwritten digits.
4) Miss classifying any random patterns as a digit.

These problems are related to model training and Input image processing part of the project. the problems of can be solved by training the model on much more accurately generated dataset and also using more data along with mnist dataset to generated the training dataset. Also, carefully pre-processing the input image before and after slicing while keeping in mind the inter-digit gaps of written digits will also help to enhance the accuracy of the OCR system. After this we need to tackle one more problem which is giving out the accurate series on numbers as written in the input image by using the detected digit's coordinates, confidence scores and identity. Also keeping in mind the continuity of the series to the next slices of the input image. This will require making of a smart algorithm that can utilize the information coming out of the model as mentioned above and then use it wisely to predict the actual written series.

2. CONCLUSION

So, in conclusion the objective of identifying and localizing the written digits in the form is achieved along with some chances of improvement provided the quality of training dataset. Also, there is some more development left in last part of the project where we need to make an algorithm that can quickly give out the actual written series of handwritten digits from the input image of the form provided the data from the trained model's output.