Detecting Knowledge Artifacts in Scientific Document Images - Comparing Deep Learning Architectures

|  |  |  |
| --- | --- | --- |
| Mohamed Kerwat  *Department of*  *Cyber-Physical Systems*  *Clark Atlanta University*  Atlanta, GA 30314  mohamed.kerwat@students.cau.edu | Roy George  *Department of*  *Cyber-Physical Systems*  *Clark Atlanta University*  Atlanta, GA 30314  [rgeorge@cau.edu](mailto:rgeorge@cau.edu) | Khalil Shujaee  *Department of*  *Cyber-Physical Systems*  *Clark Atlanta University*  Atlanta, GA 30314  kshujaee@cau.edu |

***Abstract*—there is a vast store of scientific knowledge contained in traditional archival media, such as paper, film, photographs, etc., created prior to the development of portable formats for documents. There are very useful scientific summaries such as tables and graphs embedded within these documents. While converting these to documents to images is straightforward, identifying these artifacts automatically is still challenging. Researchers have shown interest in this area by proposing numerous techniques for the detection of such artifacts. In this paper we review previous work, and present a comparison of three deep learning algorithms that address this problem. A dataset comprising of the ICDAR 2013 benchmark data set supplemented with data from the “wild” is used to train and test the models. The models are compared on ease of training, and accuracy of each model. The results of this comparison and practical suggestions for using deep learning models for document image recognition are presented.**

***Keywords— Object detection, Image classification, Document Images, Deep Learning.***

1. INTRODUCTION

Some of this scientific knowledge applies to the solution of business and scientific problems, and consequently, [20] it is possible to manually extract the content; however, this technique is not feasible for large knowledge repositories stored on traditional media due to cost and the lack of timeliness. Prior to the internet era, the primary mechanism for the archival of scientific data was through documents preserved in media such as paper, film, and photographs [1]. Scientific summaries in the form of tables and graphs are often embedded in these documents. There is considerable value in extracting such scientific artifacts for a variety of reasons, including the fact that many of the reports are not reproducible today (for instance, due to changes in the environment, or ecology or even through the extinction of species). Tables and graphs enable the readers to read, compare, evaluate and easily understand the information as well as facts that have been presented in scientific papers, book, etc. [21] Automatic extraction of such data is needed, since it is infeasible to do it manually. A multi-step approach has to be deployed to extract the information- the documents are converted to an images, followed by automated detection of the artifact, extraction of the artifact and Object Character Recognition (OCR) of the artifact (as applicable).

Table detection in document images is an important intermediate step, and has been a productive field of research with the application of several different types of technologies. Kieninger et al. [2]-[3] describe the T-Recs algorithm for table detection and structure extraction from documents called T-Recs.” It is a framework that covers a block division, table area and a model free structural analysis of tables. In [7] Tupaj et al. propose an OCR based table detection technique. That searches for sequences of table-like lines based on the keywords that might be present in the table headers. The line that contains keyword is regarded as the starting line while subsequent lines are then analyzed to match with predefined set of tokens which are then categorized as table structure. The limitation of this technique is that it depends highly on the keywords that might appear in table headers. In Gatos et al. [9] detection is performed by finding area of intersection between the horizontal and vertical lines. Tables are then reconstructed by drawing corresponding horizontal and vertical lines that are connected to intersection pairs. Kasar. [11] presents a method that locates tables by detecting horizontal and vertical lines from input image. A set of 26 low level features are extracted and passed to Support Vector Machine (SVM) which then detects the table. The major limitation of this approach is that it cannot handle tables without ruling lines. Wang et al. [4] proposes an algorithm to detect table lines based on distance between consecutive words. After that, Horizontal consecutive words are then grouped together with vertical adjacent lines to propose table entity candidates. Hu et al. [5] presents an approach that assumes that input images are single columned, which however, cannot be applied on multi-column layouts. Shafait et al. [6] describes an approach for table detection in heterogeneous documents. Their system is integrated into open source Tesseract OCR engine and works on documents with varying layouts (reports, newspaper articles, magazine, ...) . Harit et al. [8] describes a technique for table detection based on the identification of unique table start and trailer pattern. The major limitation of this method is it breaks if the table start patterns are not unique in document images. Costa e Silva. [10] presents an alternate technique for table detection using Hidden Markov Models (HMMs). The system extracts text from PDF files using the pdftotext Linux utility. Feature vectors are then computed on basis of spaces present between the texts. However, this approach is restricted to non-raster PDF files that are not noisy. Jahan et al. [12] uses local thresholds for word spacing and line height for localization and extraction of table regions. This method is limited in that it detects table regions only along surrounding text regions. Hence it cannot be used for localization of table regions only. Anh et al. [13] presents a hybrid approach by first classifying the document into text and non-text regions. These regions are then examined to get table regions. This approach will fail if table is spanned across multiple columns in the document. Moreover, it will not work for scanned images as it does not use any heuristic filter (type of algorithms are used to examine text or content in images in specific ways) to cater for noisy images.

These techniques have significant drawbacks- they are not able to completely detect table regions from document images, or they need domain information such as clear borders between columns and rows or using predefined keywords or use a particular class of document images. The use of deep learning techniques for automated extraction of scientific artifacts is still relatively new in comparison to traditional Image Processing approaches. However, these techniques have the ability to overcome the problems described above.

Hao et al. [18] present a deep learning based approach for table detection. They compute region proposals using a predefined set of rules. These region proposals are then used by a Convolutional Neural Network (CNN) to detect whether a certain region proposal belongs to table region or not. The limitation is that it works well for tables with rule lines but fails to localize table regions if the table is spanned across multiple columns. A second limitation is that it works only on non-raster PDF documents.

Gilani and Shafait. [14] propose the Faster Recurrent Convolutional Neural Network (Faster RCNN). Images are used directly in this model since Image transformation is applied to separate regions and Fast RCNN for table detection.

The RCNN technique shows promise, however there are no systematic studies on the applicability of deep learning architectures on table detection or extensions to scientific artifact extraction. This paper addresses this problem- by exploring the performance of three well known deep learning architectures- Faster RCNN [16], the Single Shot Multi-box Detector (SSD) [17] and You Only Look Once (YOLO) [15] to the problem of table detection. We posit that other useful scientific artifacts embedded in scientific papers and could also be extracted automatically. To this purpose we include graph extraction as a component of this effort. These architectures are trained and tested with a dataset composed from the benchmark data, the ICDAR 2013 set, and document images from the “wild” retrieved from the web. This paper compares performance of the three deep learning models on document images and deployment issues discussed in detail.

Section II of this paper introduces the deep learning models tested, and Section III describes the performance metrics used in the paper. Section IV describes technical details related to the training process .Section V discusses the results of the comparisons and Section VI concludes the paper and discusses future work.

1. DEEP LEARNING NEURAL NETWORK MODELS

Deep Learning is a branch of machine learning that uses artificial neural networks for the process of machine learning. While number of different architectures has been developed, in this paper we use a supervised deep learning architecture for the purpose of artifact detection.

Three popular deep neural network models, Faster RCNN, SDD, and YOLO are chosen for comparison here. The models are briefly described below.

*Faster RCNN*: It is a state of art computer vision method that recognizes and localizes objects in static or dynamic images. It uses two networks; region proposal network (RPN) for generating proposed regions and Recurrent Convolutional Neural Network (RCNN) for detecting those regions and classifies them.

*Single Shot Detector (SSD)*: SSD is an algorithm designed for a single neural network to detect object in an image. The task of localizing and classifying objects is done using the single network as mentioned, as a result this network act as an object detector.

*You Only Look Once*: [15] YOLO is a new fast method for object detection. A single neural network is applied to the image, which then creates bounding boxes for image regions, and weights the boxes, by predicted probability. The base model of YOLO processes images in real-time at rate of 45 frames per second and other faster models could reach 155 frames per second, but with less accuracy compared to the other sate of art object detection algorithms.

1. PERFORMANCE METRICS

We use several measures for evaluating table and graph detection from the document images. The three models are evaluated and compared similarly. The metrics are meant to measure various aspects of object detection (tables and graphs) including partials, and multiple detections. This evaluation is based on a simplified framework presented in [14].

*a) Correct Detections*: The numbers of tables and graphs that has been detected correctly by the trained model. When the detection box correctly detects 90% or more of the object, it is considered a correct detection.

*b) Partial Detection:* The numbers of tables and graphs that have been detected partially correct by the trained model. This means that the detection box has detected more than 10% and less 90% of the object.

*c) Over-Segmented Detections:*The numbers of tables and graphs that have been detected more than once at the same time by the trained model. i.e., More than one detection box was on the same object at the same time.

*d) Under-Segmented Detection:*The number of detections that detected two or more parts from different objects. It means detecting different parts from different objects, and the model consider it as one as one object.

*e) Missed Detection:* The model did not detect any object on the image at all.

*f) Incorrect Detection:* This indicates the number of tables and graphs that has been detected incorrectly, for example, detecting table as a graph or detecting graph as a table.

*Accuracy:* We measured and evaluated the overall performance of tables and graphs detections. The percentage is calculated for each type of detection based on the number of the detections divided by the number of the total objects we have in our dataset.

1. TRAINING CONFIGURATION

As with all supervised learning models, the deep learning models go through a training phase and a testing phase. Roughly 85 % of the data set is used to train the models, which are then tested using the 15% test data set. The training process is performed on a desktop with a Nvidia GPU “GeForce GTX 1070 Ti” with 8GB VRAM. [19]. The training process takes approximately seven hours to reduce the error to 98% for YOLO, 86% for SSD and 99% for Faster RCNN. (Based on comparisons of the error at the first step to the error at the last step of training). There are several possible configurations for each model, for example the configuration that is used for SSD and Faster RCNN is the “inception\_v2\_coco” [22]. This configuration permits training on a GPU that has only 8GB of VRAM, but there are other configurations that require a graphic cards with higher VRAM.

1. EXPERIMENT AND RESULTS

In order to evaluate the performance of the three models that we chose, we trained and tested all the models on a dataset consisting of the ICDAR 2013 benchmark data set supplemented with data from the “wild”. This dataset contains of a wide variety of document images, different types of tables and graphs from many sources.

FIGURE 1: EXAMPLE FROM ICDAR DATASET

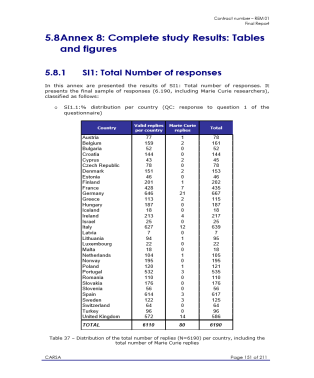
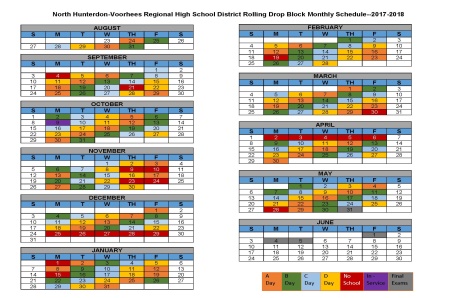
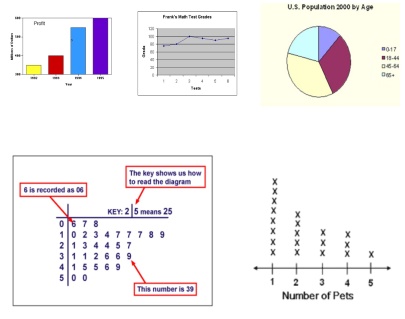


FIGURE 2: OTHER EXAMPLES





Details of the data set are as below:

* Training Data (701 images contain 1480 objects): 750 labeled graphs and 730 labeled tables
* Testing Data (119 images contain 224 objects): 125 Tables and 99 Graph

FIGURE 3.1: FASTER RCNN DETECTIONS

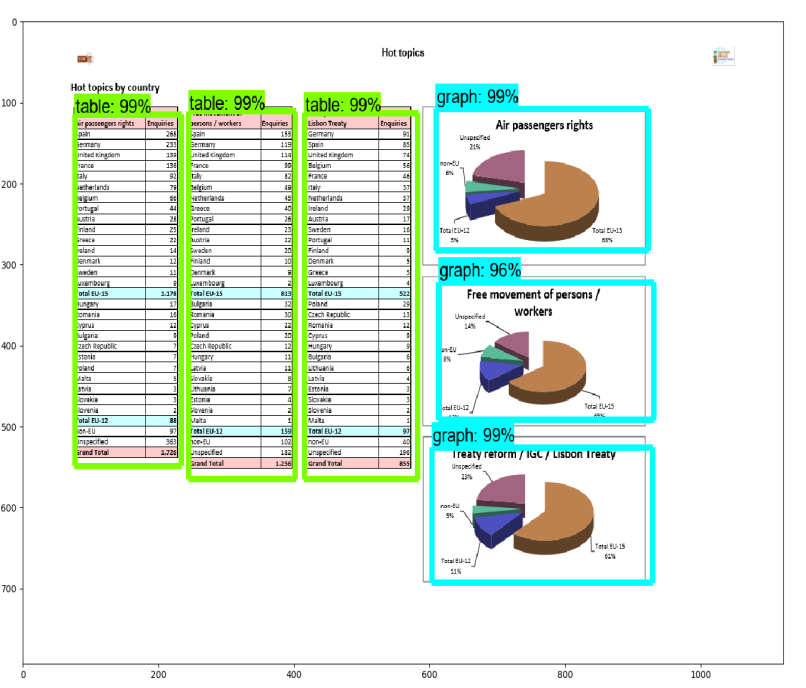


FIGURE 3.2: FASTER RCNN DETECTIONS

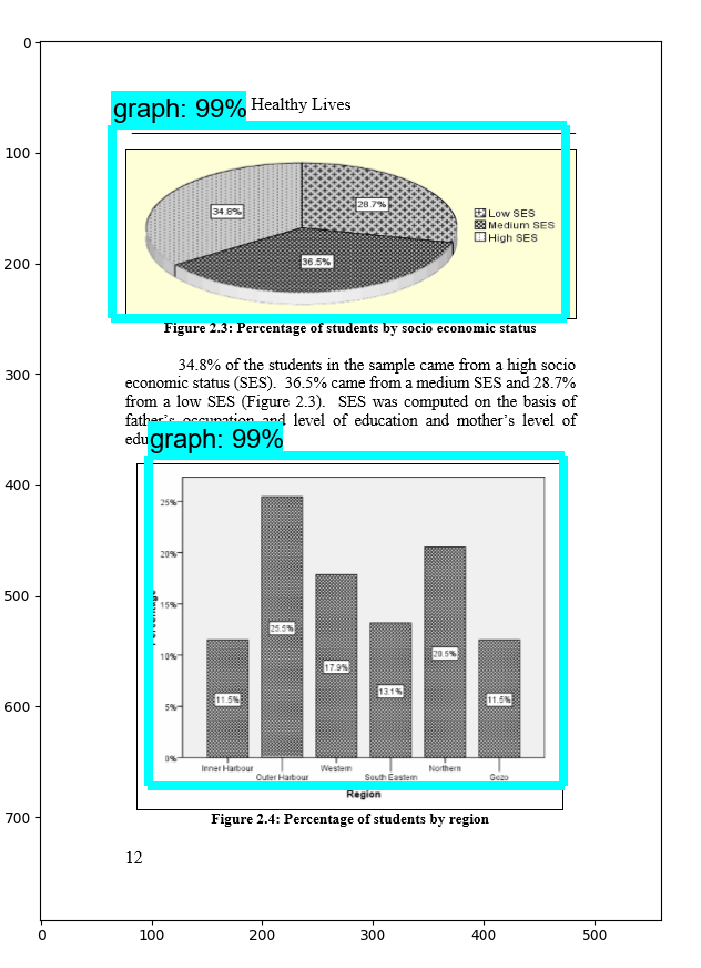


FIGURE 4: SDD DETECTIONS



FIGURE 5: YOLO DETECTIONS

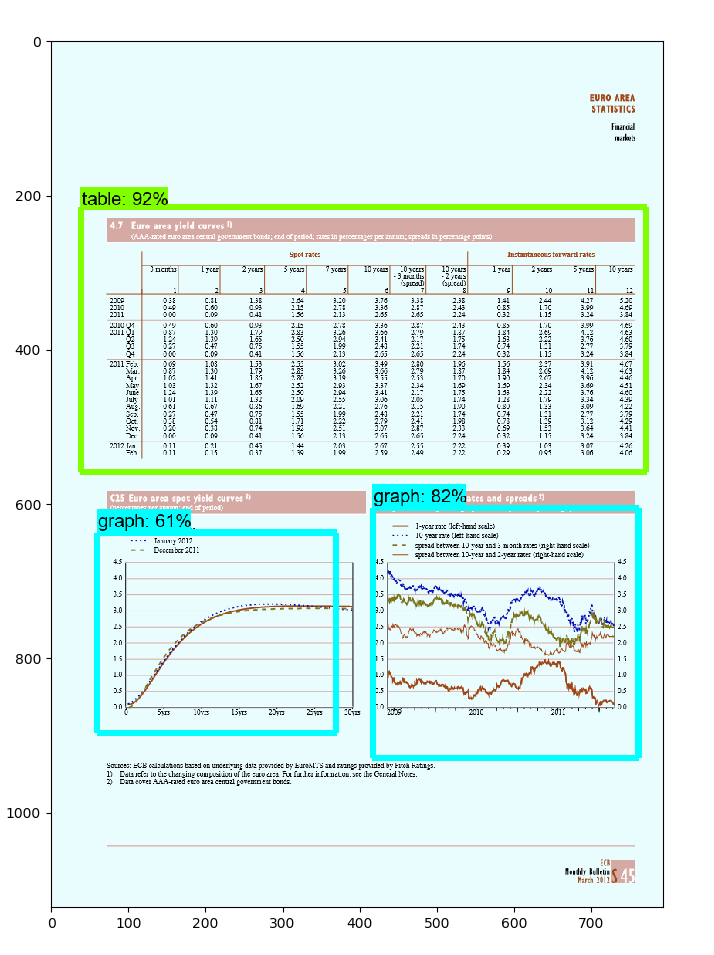


TABLE 1: COMPARISON BASED ON NUMBER OF TOTAL DETECTIONS

|  |  |  |  |
| --- | --- | --- | --- |
|  | Faster-RCNN | SDD | YOLO |
| Correct | 83.7% | 72% | 45.8% |
| Partial | 1.7% | 11% | 36% |
| Under\_Segment | 6.3% | 9.8% | 13.6% |
| Over\_Segment | 5.5% | 1.4% | 4% |
| Incorrect | 2.8% | 5.6% | 0.6% |

The Faster-RCNN has the best overall performance, with the largest percentage of correct detections, 83.7%, and the lowest number of incorrect detections, 2.8%. YOLO has the poorest performance among the three. It was able to detect only 33.7% of the test set. It should be noted that it partially detects and segments artifacts frequently. SSD has intermediate results; and its results are closer to Faster-RCNN than YOLO. We further break out these results with respect to table and graph artifacts.

TABLE 2: COMPARISON BASED ON NUMBER OF TABLES DETECTIONS

|  |  |  |  |
| --- | --- | --- | --- |
|  | Faster-RCNN | SDD | YOLO |
| Correct | 81.2% | 70.7% | 33.7% |
| Partial | 0.7% | 11% | 45.5% |
| Under\_Segment | 8.7% | 14.6% | 16.9% |
| Over\_Segment | 8.7% | 2.5% | 2.6% |
| Incorrect | 0.7% | 1.2% | 1.3% |

In Table 2 a separate comparison between the three models for table detection. Faster-RCNN has the largest value of correct detection 81.2% and lowest of incorrect detection 0.7%.

TABLE 3: COMPARISON BASED ON NUMBER OF GRAPHS DETECTIONS

|  |  |  |  |
| --- | --- | --- | --- |
|  | Faster-RCNN | SDD | YOLO |
| Correct | 87% | 73.8% | 56.3% |
| Partial | 3% | 11.5% | 27.6% |
| Under\_Segment | 3% | 3.3% | 10.4% |
| Over\_Segment | 1% | 0.0% | 5.75% |
| Incorrect | 6% | 11.5% | 0.0% |

For the graph detection we still have almost the same results as the previous ones, Faster-RCNN still achieving the highest performance while YOLO is still in the bottom.

TABLE 4: COMPARISON OF BASED ON NUMBER OF MISSED OBJECTS

|  |  |  |  |
| --- | --- | --- | --- |
|  | Faster-RCNN | SDD | YOLO |
| Number of missed objects | 5 | 59 | 68 |
| Percentage to the total number of objects | 2.2% | 26% | 30% |

For the results presented in Table 4, Faster-RCNN barely misses any object; it missed only 5 objects out of 224, which is only 2.2%.

Noticeably, all three models have better performance with detecting graphs than detecting tables. This indicates that even the deep learning models are subject to the wide variability in table appearance. Tables that have elements separated in cells constructed from horizontal and vertical lines have a higher probability of being recognized. In the case of graphs the training dataset has more graphs than tables and the variety of graphs are more than tables- permitting a wider range of detection. That’s why large variety in datasets is very important for higher accuracy.

The results that we presented give us a strong indicator that YOLO is less preferable for the task of detecting artifacts in image documents for the purpose of extraction. It should be noted that these particular deep learning models are rapidly being improved and in fact, within the experimental phase of this project, new models have been defined [22] and it is to be expected that these configurations will provide better results. However, many of these models require systems with much larger hardware capabilities and advanced GPUs. Future plans for the project include performing the same tests more complex deep learning models executing on more powerful machines.

Another point that needs to be emphasized is that the dataset for these experiments were obtained largely from the “wild”, which was used to supplement the ICDAR 2013 benchmark set. The ICDAR 2013 is small with only 23 images. Literature indicates that there is a larger UNLV dataset – however, this does not seem to be accessible anymore. The results are promising and seem to indicate that with a much larger and more targeted data set, i.e., the accuracies could be improved much further.

V. CONCLUSION

This paper presents a comparison of three state of the art deep learning models for tables and graphs detection in document images. Experimental results show that Faster-RCNN provides the best results and performance for object detection in scientific document images. The deep learning models that are compared are all data driven and not dependent on hand engineered features. This provides guidance to the researcher as to the efficacy of the deep learning models for this task. To have a practical model with results that could be used outside the lab, training should be based on a dataset that contain even larger variety of targeted images with a careful labeling. We plan to extend this work in the direction of detect more complex artifacts objects in image documents, including diagrams, and text features, such as titles, authors, references. etc. We also feel that the deep learning models could be used in conjunction with more traditional image processing techniques to further improve the quality of detection. In addition to the correct detection, the models are able to detect a large proportion of the remaining artifacts. This information could be used as the basis of investigating the document using traditional techniques for very high accuracies.

Acknowledgement

This research is funded in part by the Department of Energy and NGC under Contract Number DE-NA 0002686, Contract No. 2017-2007 respectively. Any opinions, findings, and conclusions expressed here are those of the author(s) and do not reflect the views of the sponsor.

REFERENCES

[1] Soraya de Chadarevian, The future historian: Reflections on the archives of contemporary sciences, Studies in History and Philosophy of Biological and Biomedical Sciences 55 (2016) 5, Accessed: 06/15/2018, http: <http://socgen.ucla.edu/wp-content/uploads/2016/01/Future-historian-Studies-C-2016.pdf>.

[2] T. Kieninger and A. Dengel, “A paper-to-html table converting system,” in Proceedings of document analysis systems (DAS), vol. 98, 1998.

[3] “Applying the t-recs table recognition system to the business letter domain,” in Document Analysis and Recognition, 2001.

[4] Y. Wang, I. Phillip, and R. Haralick, “Automatic table ground truth generation and a background-analysis-based table structure extraction method,” in Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on. IEEE, 2001, pp. 528–532.

[5] J. Hu, R. S. Kashi, D. P. Lopresti, and G. Wilfong, “Mediumindependent table detection,” in Electronic Imaging. International Society for Optics and Photonics, 1999, pp. 291–302.

[6] F. Shafait and R. Smith, “Table detection in heterogeneous documents,” in Proceedings of the 9th IAPR International Workshop on Document Analysis Systems. ACM, 2010, pp. 65–72.

[7] S. Tupaj, Z. Shi, C. H. Chang, and H. Alam, “Extracting tabular information from text files,” EECS Department, Tufts University, Medford, USA, 1996.

[8] G. Harit and A. Bansal, “Table detection in document images using header and trailer patterns,” in Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing. ACM, 2012, p. 62.

[9] B. Gatos, D. Danatsas, I. Pratikakis, and S. Perantonis, “Automatic table detection in document images,” Pattern recognition and data mining, pp. 609–618, 2005.

[10] A. C. e Silva, “Learning rich Hidden Markov Models in document analysis: Table location,” in Document Analysis and Recognition, 2009. ICDAR’09. 10th International Conference on. IEEE, 2009, pp. 843– 847.

[11] T. Kasar, P. Barlas, S. Adam, C. Chatelain, and T. Paquet, “Learning to detect tables in scanned document images using line information,” in Document Analysis and Recognition (ICDAR), 2013 12th International Conference on. IEEE, 2013, pp. 1185–1189.

[12] M. A. Jahan and R. G. Ragel, “Locating tables in scanned documents for reconstructing and republishing,” in Information and Automation for Sustainability (ICIAfS), 2014 7th International Conference on. IEEE, 2014, pp. 1–6.

[13] T. T. Anh, N. In-Seop, and K. Soo-Hyung, “A hybrid method for table detection from document image,” in Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on. IEEE, 2015, pp. 131–135.

[14] A. Gilani, S. R. Qasim, I. Malik and F. Shafait, "Table Detection Using Deep Learning," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, 2017, pp. 771-776.

[15] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779-788.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”

[17] Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu and Alexander C. Berg. “SSD: Single Shot MultiBox Detector.” ECCV (2016).

[18] L. Hao, L. Gao, X. Yi, and Z. Tang, “A table detection method for pdf documents based on convolutional neural networks,” in Document Analysis Systems (DAS), 2016 12th IAPR Workshop on. IEEE, 2016, pp. 287–292.

[19] Nvidia Corporation, “CUDA Toolkit Documentation”, v9.2.88, May 15, 2018 <https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#compute-capabilities>. Accessed: 06/15/2018

[20] A. Kölsch, M. Z. Afzal, M. Ebbecke and M. Liwicki, "Real-Time Document Image Classification Using Deep CNN and Extreme Learning Machines," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, 2017, pp. 1318-1323.

[21] L. Kang, P. Ye, Y. Li and D. Doermann, "A deep learning approach to document image quality assessment," 2014 IEEE International Conference on Image Processing (ICIP), Paris, 2014, pp. 2570-2574.

[22]<https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md>. Accessed: 06/15/2018