



A Saliency-Based Convolutional Neural Network for Table and Chart Detection in Digitized Documents

I. Kavasidis^{1(✉)}, C. Pino¹, S. Palazzo¹, F. Rundo³, D. Giordano¹, P. Messina²,
and C. Spampinato¹

¹ Department of Electrical, Electronic and Computer Engineering,
University of Catania, Via S. Sofia 6, 95125 Catania, Italy
kavasidis@dieei.unict.it

² Tab2Ex, San Jose, USA

<http://www.tab2ex.com>, <https://www.snapchart.co/>

³ STMicroelectronics - ADG Central R&D, Catania, Italy

Abstract. Within the realm of information extraction from documents, detection of tables and charts is particularly needed as they contain a visual summary of the most valuable information contained in a document. For a complete automation of the visual information extraction process from tables and charts, it is necessary to develop techniques that localize them and identify precisely their boundaries. In this paper we aim at solving the table/chart detection task through an approach that combines deep convolutional neural networks, graphical models and saliency concepts. In particular, we propose a saliency-based fully-convolutional neural network performing multi-scale reasoning on visual cues followed by a fully-connected conditional random field (CRF) for localizing tables and charts in digital/digitized documents. Performance analysis, carried out on an extended version of the ICDAR 2013 (with annotated charts as well as tables) dataset, shows that our approach yields promising results, outperforming existing models.

Keywords: Document analysis · Image classification · Object detection · Saliency detection

1 Introduction

Production and storage of digital documents have increased exponentially in the last two decades. Extracting and retrieving information from this massive amount of data have become inaccessible to human operators and a large amount of information captured in digital documents may go lost or never seen. As a consequence, a large body of research has focused on automated methods for document analysis. Most of these efforts are directed towards the development of Natural Language Processing (NLP) methods, that analyze both grammar and semantics of text with the goal of automatically extracting, understanding

and, eventually, summarizing key information from digital documents. However, while text is, inarguably, a fundamental way to convey information, there are contexts where graphical elements are much more powerful. For example, in scientific papers, many experiments, variables and numbers need to be reported in a concise way that fits better with tables/figures than text. “*A picture is worth a thousand words*” describes exhaustively the power that graphical elements possess in conveying information that would be otherwise cumbersome, both for the writer to express and the reader to understand. Thus, it is of primary importance for an effective automatic document processing approach to gather information from tables and charts. Several commercial software products that convert digitized and digital documents into processable text already exist. However, most of them either largely fail when dealing with graphical elements or require an exact localization of such elements to work properly. For this reason, a crucial pre-processing step in automated data extraction from tables and charts is to find their exact location. The problem of identifying objects in images traditionally falls in the object detection research area, where, nowadays, Deep Convolutional Neural Networks (DCNNs) play the leading role [14, 18]. However, naively employing DCNN-based object detectors in digital documents, suitably transformed into images, leads to failures mainly because of the intrinsic appearance difference between digital documents and natural images (the data for which models are mainly thought for). Trying to train models from scratch may be unfeasible due to the large number of images required and to the lack of suitably annotated document datasets. Moreover, such approaches generally exploit the visual differences between object categories: while the visual characteristics of certain graphical elements (e.g., charts) significantly differ from text, the same cannot be said for tables, whose main differences from the surrounding content lie mostly in the layout. Finally, many of the existing object detectors are often prone to potential errors by upstream region proposal models and are not able to detect simultaneously all the objects of interest in an image [5, 19].

In this paper we propose a general deep neural network model for pixelwise dense prediction, and consequently for detection of *arbitrary graphical elements*, rather than only for tables as existing methods, in digitized documents. The key intuition to make the whole model generalizable to arbitrary graphical elements is to pose the detection problem as a saliency detection one as those elements generally stand out in documents. Additionally, in order to provide a stronger supervision to the internal saliency detectors, we employ the approach introduced in [16], by adding a loss term related to the capability of the saliency maps to identify regions that are distinctive for visual classification in one of the four target categories. Finally, predictions of our network are enhanced with a fully connected Conditional Random Field (CRF) [12]. We demonstrate that our method generalizes well as demonstrated by the performance achieved on an extended version of ICDAR 2013 dataset (with annotated charts as well as tables and we also release).

2 Related Work

Given the large quantity of digital documents that are available today, it is mandatory to develop automatic approaches to extract, index and process information for long-term storage and availability. Consequently, there is a large body of research on document analysis methods attempting to extract different types of objects (e.g., tables, charts, pie charts, etc.) from various document types (text documents, source files, documents converted into images, etc.). Before the advent of deep learning, most works on document analysis for table detection were based on exploiting *a priori* knowledge on object properties by analyzing tokens extracted from source document files [2, 3, 21]. For example, [3] proposes a method for table detection in PDF documents, which uses tags of tabular separators to identify the table region. Of course, the main shortcoming of all methods that rely on detecting horizontal or vertical lines for table detection is that they fail to identify tables without borders. Alternatively, methods operating on image conversion of document files and exploiting only visual-cues for table detection have been proposed [9, 15].

Similar computer vision-based methods have been proposed for detecting other types of graphical elements (e.g., charts, diagrams, etc.) than tables [8]. These methods basically employ simple computer vision techniques (e.g., connected components, fixed set of geometric constraints, edge detection, etc.) to extract chart images, but, as for the table detection case, they show scarce generalization capabilities. Low-level visual cues (e.g., intensity, contrast, homogeneity, etc.) in combination with shallow machine learning techniques have been used for specific object classification tasks [17] with fair performance, but these methods are mainly for classification as they tend to aggregate global features in compact representations which are less suitable for performing object detection. With the recent rediscovery of deep learning, in particular convolutional neural networks, and its superior representation capabilities for high-level vision tasks, the document analysis research community started to employ DCNNs for document processing, with a particular focus on document classification [1, 7] or object (mainly chart) classification – after accurate manual detection [22]. One recent work presenting a DCNN exclusively targeted to table detection is [4], which employs Faster R-CNN for object detection. Nevertheless, this method suffers from the limitations mentioned in the previous section, i.e., its performance is negatively affected by the region proposal mistakes and it does not provide multiple detections for each image.

In this paper, we tackle the detection problem from a different perspective, i.e., we pose it as semantic image segmentation problem, by densely predicting—according to visual saliency principles—for each pixel of the input image the likelihood of being part of a salient object. This allows us to detect arbitrary graphical elements by only fine-tuning the classification stream as the salient objects have been already outlined by the saliency network of the proposed approach.

3 Deep Learning Models for Table and Chart Detection and Classification

The approach for table and chart extraction presented here works by receiving an image as input and generating a set of binary masks as output, from which bounding boxes are drawn. Each binary mask corresponds to the pixels that belong to objects of four specific classes: *tables*, *pie charts*, *line charts* and *bar charts*. The main processing engine driving our approach consists of a fully-convolutional deep learning model that performs table/chart detection and classification, followed by a conditional random field for enhancing and smoothing the binary masks (see Fig. 1).

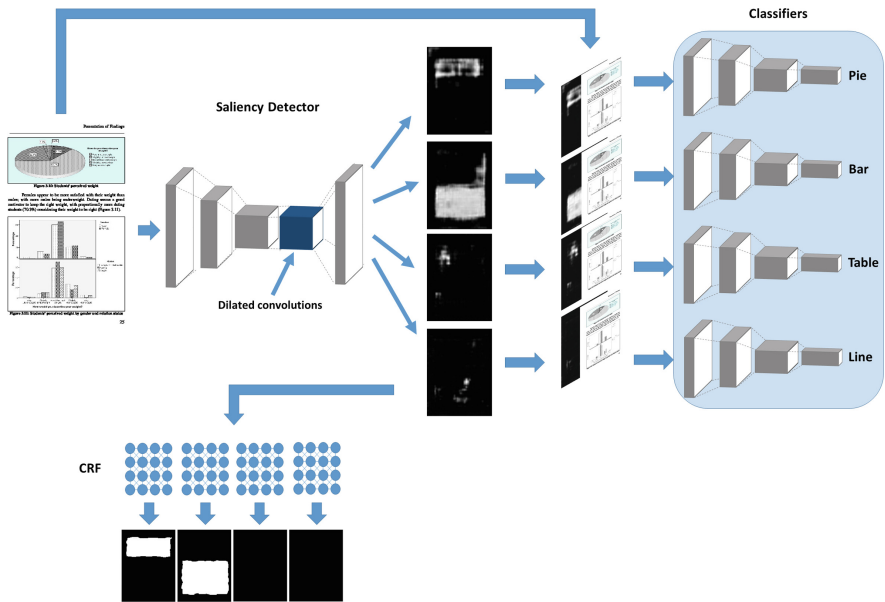


Fig. 1. The proposed system. An input document is fed to a convolutional neural network trained to extract class-specific saliency maps, which are then enhanced and smoothed by CRF models. Moreover, binary classifiers are trained to provide an additional loss signal to the saliency detector, based on how useful the computed saliency maps are for classification purposes.

Given an input document page transformed into an RGB image and resized to 300×300 , the output of the system consists of four binary masks, one for each of the aforementioned classes. Pixels set to 1 in a binary mask identify document regions belonging to instances of the corresponding class, while 0 values are background regions (e.g., regular text).

We leverage visual saliency prediction to solve our object detection problem. As a result, the first processing block in our method is a fully-convolutional

neural network (i.e., composed only by convolutional layer) that extracts four class-specific heatmaps from document images. Our saliency detection network is based on the feature extraction layers of the VGG-16 architecture. However, we applied a modification aimed at exploiting inherent properties of tables and charts: in particular, the first two convolutional layers do not employ traditional square convolution kernels, as in the original VGG-16 implementation, but use rectangular ones instead, of sizes 3×7 and 7×3 . This set up gives our network the ability to extract table-related features (e.g., lines, spacings, columns and rows) even at the early stages. Padding was suitably added in order to keep the size of the output feature maps independent of the size of the kernels.

After the cascade of layers from the VGG-16 architecture, the resulting 75×75 feature maps are processed by a *dilation block*, consisting of a sequence of dilated convolutional layers. While the purpose of the previous layers is that of extracting discriminative local features, the dilation block exploits dilated convolutions to establish multi-scale and long-range relations. Dilation layers increase the receptive field of convolutional kernels while keeping the feature maps at a constant size, which is desirable for pixelwise dense prediction as we do not want to spatially compact features further. The output of the dilation block is a 4-channel feature map, where each channel is the saliency heatmap for one of the target object categories. After the final convolutional layer, we upsample the 75×75 maps back to the original 300×300 using bilinear interpolation.

In theory, the saliency maps could be the only expected output of the model, and we could train it by just providing the correct output as supervision. However, [16] recently showed that posing additional constraints to saliency detection—for example, forcing the saliency maps to identify regions that are also class-discriminative—improves output accuracy. This is highly desirable in our case as output saliency maps may miss non-salient regions (e.g., regular text) inside salient regions (e.g., table borders), while it is preferable to obtain maps that entirely cover the objects of interest.

For this reason, we add a classification branch to the model. This branch contains as many binary classifiers as the number of target object classes. Each binary classifier receives as input a crop of the original image around an object (connected component) in the saliency detector outputs, and aims at discriminating whether that crop contains an instance of target class. The classifiers are based on the Inception model and are architecturally identical, except for the final classification layer which is replaced by a linear layer with one neuron followed by a sigmoid nonlinearity.

To train the model we employ a multi-loss function that combines the error measured on the computed saliency maps with the classification error of the binary classifiers. The saliency loss function measured between the computed saliency maps \mathbf{Y} (expressed as a $N \times 300 \times 300$ tensor, with N being the number of object classes, 4 in our case) and the corresponding ground-truth mask \mathbf{T} (same size) is given by the mean squared error between the two:

$$\mathcal{L}_S(\mathbf{Y}, \mathbf{T}) = \frac{1}{N \cdot h \cdot w} \sum_{k=1}^N \sum_{i=1}^h \sum_{j=1}^w (Y_{kij} - T_{kij})^2 \quad (1)$$

where h and w are the image height and width (in our case, both are 300), and Y_{kij} and T_{kij} are the values of the respective tensors at location (i, j) of the k -th saliency map. The binary classifiers are first trained separately from the saliency network, so that they can be used to provide a reliable error signal to the saliency detector. Training is performed using original images cropped with ground-truth annotations. For example, for training the table classifier, we use table annotations (available in the ground truth) and crop input images so as to contain only tables: these are the “positive samples”. “Negative samples” are, instead, obtained by cropping the original images with annotations from other classes (pie chart, bar chart and line chart) or with random background regions. This procedure is performed for each classifier to be trained. Since cropping may result in images of different sizes, all images are resized to 299×299 to fit the size required by the Inception network.

Each classifier is trained to minimize the negative log-likelihood loss function:

$$\mathcal{L}_{C_i}(\mathbf{I}, t_i) = -t_i \log C_i(\mathbf{I}) - (1 - t_i) \log (1 - C_i(\mathbf{I})) \quad (2)$$

where C_i ($1 \leq i \leq 4$) is the classifier for the i^{th} object class, and returns the likelihood that an object of the targeted class is present in image \mathbf{I} : t_i is the target label, and is 1 if i is the correct class, 0 otherwise. After training the classifiers, they are used to compute the classification loss for the saliency detector, as follows:

$$\mathcal{L}_C = \sum_{i=1}^N \mathcal{L}_{C_i}(\mathbf{I}, t_i) \quad (3)$$

The saliency detector, in this way, is pushed to provide accurate segmentation maps so that whole object regions are passed to the downstream classifiers. Indeed, if the saliency detector is not accurate enough in identifying tables, it will provide incomplete tables to the corresponding classifier, which may be then misclassified as non-table objects with a consequent increase in loss. Note that while training the saliency detector, the classifiers themselves are not re-trained, and are only used to compute the classification loss. This prevents the binary classifiers to learn to recognize objects from their parts, thus forcing the saliency network to keep improving its detection performance.

The multi-loss used for training the network in an end-to-end manner is, thus, given by the sum of the terms \mathcal{L}_C and \mathcal{L}_S :

$$\mathcal{L} = \mathcal{L}_C + \mathcal{L}_S \quad (4)$$

The outputs of our fully convolution network, usually, show irregularities such as spatially-close objects fused in one object or one object oversegmented in multiple parts. In order to mitigate segmentation errors, we integrate in our system a downstream module based on the fully-connected CRFs employed in [13].

4 Performance Analysis

4.1 Dataset

Training Dataset. To train our method, we developed a web crawler that searched and gathered images from Google Images, using the following queries: “*tables in documents*”, “*pie charts*”, “*bar charts*”, “*line charts*”. Additionally, we manually collected a set of documents available online related to banking reports, using queries constructed by prefixing the name of a banking institute to “research report” and “financial report”. All retrieved documents were then converted to images (one per page), resulting in a total of 50,466 images.

Training Annotations and Splits. Annotation was carried out on the web-crawled training dataset by paid annotators using an adapted version of the annotation tool in [10]. In particular, among the 50,442 retrieved images only 19,564 had at least one object of interest, while the remaining 30,878 images did not. The 19,564 images with positive instances had in total 22,544 annotations. From the set of retrieved images (and related annotations), 10% were used as a validation set for model selection, while the remaining 90% as training set. The distribution of instances of the four target classes between the training and validation sets were approximately equal.

Test Datasets. To test how well our approach generalizes we computed the performance on an extension of the ICDAR 2013 [6] benchmark. In particular we extended the ICDAR 2013 dataset (that contained table-only annotations) into a new version containing annotations of pie charts, line charts and bar charts. We refer to this new dataset as the “extended ICDAR 2013”; chart annotation was carried out as previously described for the training dataset. In terms of number of annotations, only 161 out of 238 images from ICDAR 2013 contained objects of interest. In these 161 images, there were 156 tables (with annotations already available) and 58 charts (of either “pie”, “bar” and “line” types). We did not test on ICDAR 2017 as the test split is unavailable.

Saliency detector and binary classifiers were trained in an end-to-end fashion using an image as input and (a) the annotation masks as training targets for the saliency detector, and (b) presence/absence labels of target objects on image crops for the classifiers. The input image resolution was set to 300×300 pixels. The training phase ran for 45 epochs, which in our experiments was the point where the performance of the saliency network on the validation set stopped improving. All networks were trained using the Adam optimizer [11] (learning rate was initialized to 0.001, momentum to 0.9 and batch size to 32).

4.2 Performance Metrics

Our evaluation phase aimed at assessing the performance of our DCNN in localizing precisely tables and charts in digital documents, as well as in detecting and segmenting table/chart areas.

- **Table/chart localization performance.** To test localization performance we computed precision Pr , recall Re and F_1 score by calculating true positives, false positives and false negatives.
- **Segmentation accuracy** was measured by intersection over union (IOU). While the detection scores above provide information on the ability of the models to detect the tables and charts that overlap with the ground truth over a certain threshold, IOU measures per-pixel performance by comparing the exact number of the pixels that are detected as belonging to a table or chart. In other words, the IOU score reflects the accuracy in finding the correct boundaries of table and chart regions.

The proposed model consists of several functional blocks (saliency detection, binary classification) which are stacked together for final prediction. In order to assess how each block influenced performance, we computed the performance of the model when using only the saliency network (SAL); saliency detector followed by the binary classifiers (SAL-CL) and the whole system including all parts (saliency detection, fully connected CRF and binary classifiers) (ALL).

4.3 Results

The results obtained by the two configurations previously described are reported in Table 1. Our system performed very well in all object types, and this performance increased progressively from the baseline configuration (SAL) to the more complex architecture (i.e, ALL). In particular, the baseline configuration (SAL) achieved an average F_1 score of 69.0%, with the top performance achieved on the “Tables” category (76.3%) and the worst on the “Line charts” category (63.4%). By comparing with the results obtained by SAL-CL model, we can infer that the lower performance was due to the difficulty by the saliency detector alone in extracting discriminative features between these two types of charts. Indeed, this shortcoming was countered by introducing the classification loss in the model (SAL-CL configuration). The classifiers managed to aid the saliency detector network in recognizing the distinguishing features between line charts (increase of F_1 score of about 24%) and bar charts (increase of F_1 score of about 23%) w.r.t. the baseline, bringing the average F_1 score to 87%.

Finally, adding the fully-connected CRF to SAL-CL led to a further 6.4% increase to the system’s performance, reaching a maximum average F_1 score of 93.4%. it appears evident that CRFs influence the number of false negatives and subsequently, the recall, more than they influence the number of false positives. In fact, w.r.t. the SAL-CL configuration, the CRF module increased more the recall (about 7.5%) than the precision (about 4.5%). This can be explained by the fact that the major contribution of CRF models consisted in filling gaps and holes resulting from the deep learning methods, especially for very large tables with extensive white areas. Figure 2 show examples of, respectively, good and bad detections obtained by our method.

Comparison against state of the art methods was done only in terms of table detection. In particular, we compared the performance of the four different

Table 1. Performance—in terms of precision, recall, F_1 and IOU—achieved by the two different configurations of our method on the extended ICDAR 2013 dataset. Values are in percentage.

Configuration	Class	Precision	Recall	F_1	IOU
SAL	Tables	78.4	74.4	76.3	65.3
	Pie charts	75.0	66.7	70.6	62.7
	Bar charts	62.5	69.0	65.6	63.5
	Line charts	61.9	65.0	63.4	62.1
	Average	69.5	68.8	69.0	63.4
SAL-CL	Tables	93.8	87.2	90.4	75.5
	Pie charts	87.5	77.8	82.4	72.2
	Bar charts	86.7	89.7	88.1	76.4
	Line charts	89.5	85.0	87.2	74.8
	Average	89.4	84.9	87.0	74.7
ALL	Tables	98.1	98.1	98.1	81.3
	Pie charts	100.0	88.9	94.1	78.1
	Bar charts	90.0	93.1	91.5	79.6
	Line charts	90.0	90.0	90.0	78.5
	Average	94.52	92.52	93.43	79.4

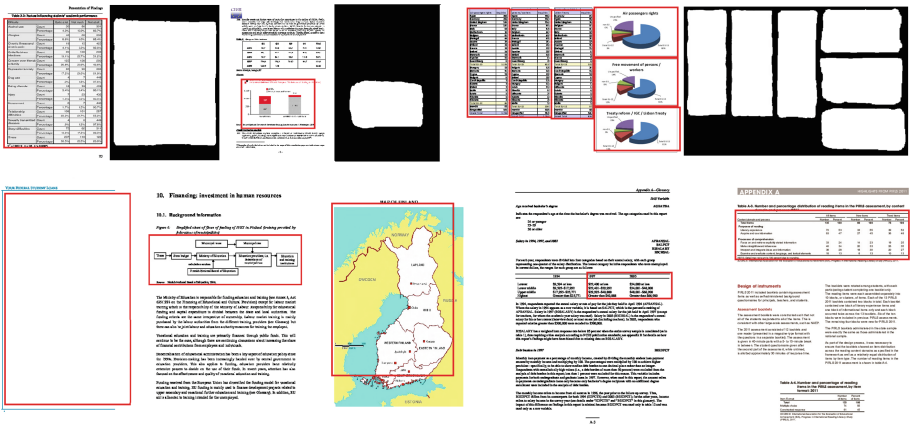


Fig. 2. Examples of good (top row) and bad (bottom row) detections.

configurations of our method to those achieved by both traditional and more recent approaches in detecting only tables on the ICDAR 2013. The comparison is reported in Table 2 and shows that our system achieved an F_1 score of 98.1%, outperforming all of the state-of-the-art methods.

Table 2. Comparison of state of the art methods in table detection accuracy on the standard ICDAR 2013 dataset (IoU = 0.5)

	Table detection		
	Precision	Recall	F_1
DeepDeSRT [20]	97.4	96.1	96.7
Tran [23]	95.2	96.4	95.8
Hao [7]	97.2	92.2	94.6
Silva [21]	92.9	98.3	95.5
Nurminen [6]	92.1	90.8	91.4
Our method	98.1	98.1	98.1

5 Conclusion

The identification of graphical elements such as tables and charts in documents is an essential processing block for any system that aims at extracting information automatically, and finds applicability to the analysis of financial documents, where numeric information is typically represented in tabular format. In this paper, we presented a method for automatic table and chart detection in document files converted to images, hence without exploiting format information (e.g., PDF tokens or HTML tags) that limit the general applicability of these approaches. The core of our model is a DCNN trained to detect salient regions from document images, with saliency based on the categories of objects that we aim to identify (tables, pie charts, bar charts, line charts). An additional loss signal based on the generated saliency maps' discriminative power in a classification task was provided during training, and a fully-connected CRF model was finally employed to smooth and enhance the final outputs. Performance evaluation, carried out on the standard UNLV dataset and ICDAR 2013 benchmark for table detection, and on an extended version of ICDAR 2013 with additional annotations for chart detection, showed that the proposed model achieves better performance than state-of-the-art methods in the localization of tables and charts. At Tab2ex, a technology based on the presented approach is employed at industrial level to extract tabular information from scanned images. Future directions of research in this complex task envision the detection of individual headers, rows and columns of tables, and the extraction of numeric data from charts based on axis values: such technologies would provide businesses with the means to process large amounts of documents (e.g. orders, invoices, financial trends) in an automated way.

References

1. Afzal, M.Z., et al.: DeepDocClassifier: document classification with deep convolutional neural network. In: IEEE ICDAR 2015 (2015)

2. Deivalakshmi, S., Chaitanya, K., Palanisamy, P.: Detection of table structure and content extraction from scanned documents. In: IEEE ICCSP 2014 (2014)
3. Fang, J., Gao, L., Bai, K., Qiu, R., Tao, X., Tang, Z.: A table detection method for multipage PDF documents via visual separators and tabular structures. In: ICDAR 2011, pp. 779–783. IEEE (2011)
4. Gilani, A., Qasim, S.R., Malik, I., Abd Shafait, F.: Table detection using deep learning. In: ICDAR 2017. IEEE (2017)
5. Girshick, R.: Fast R-CNN. In: ICCV 2015 (2015)
6. Göbel, M., Hassan, T., Oro, E., Orsi, G.: ICDAR 2013 table competition. In: IEEE ICDAR 2013 (2013)
7. Hao, L., Gao, L., Yi, X., Tang, Z.: A table detection method for PDF documents based on convolutional neural networks. In: IEEE DAS 2016 (2016)
8. Huang, W., Liu, R., Tan, C.L.: Extraction of vectorized graphical information from scientific chart images. In: IEEE ICDAR 2007 (2007)
9. Jahan, M.A., Ragel, R.G.: Locating tables in scanned documents for reconstructing and republishing. In: IEEE ICIAFS 2014 (2014)
10. Kavasidis, I., Palazzo, S., Di Salvo, R., Giordano, D., Spampinato, C.: An innovative web-based collaborative platform for video annotation. *Multimed. Tools Appl.* **70**(1), 413–432 (2014)
11. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
12. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In: NIPS 2011 (2011)
13. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In: *Advances in Neural Information Processing Systems*, pp. 109–117 (2011)
14. Li, J., et al.: Attentive contexts for object detection. *IEEE Trans. Multimed.* **19**(5), 944–954 (2017)
15. Liu, Y., Mitra, P., Giles, C.L.: Identifying table boundaries in digital documents via sparse line detection. In: 2008 17th ACM Conference on Information and Knowledge Management (2008)
16. Murabito, F., Spampinato, C., Palazzo, S., Giordano, D., Pogorelov, K., Riegler, M.: Top-down saliency detection driven by visual classification. *Comput. Vis. Image Underst.* **172**, 67–76 (2018)
17. Perez-Arriaga, M.O., Estrada, T., Abad-Mota, S.: TAO: system for table detection and extraction from PDF documents. In: FLAIRS Conference, pp. 591–596 (2016)
18. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: CVPR 2017 (2017)
19. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems*, pp. 91–99 (2015)
20. Schreiber, S., Agne, S., Wolf, I., Dengel, A., Ahmed, S.: DeepDeSRT: deep learning for detection and structure recognition of tables in document images. In: IEEE ICDAR 2017 (2017)
21. Silva, A.C.: Learning rich hidden Markov models in document analysis: table location. In: IEEE ICDAR 2009 (2009)
22. Tang, B., et al.: DeepChart: combining deep convolutional networks and deep belief networks in chart classification. *Sig. Process.* **124**, 156–161 (2016)
23. Tran, D.N., Tran, T.A., Oh, A., Kim, S.H., Na, I.S.: Table detection from document image using vertical arrangement of text blocks. *Int. J. Contents* **11**(4), 77–85 (2015)