# Checkbox Detection

## Abstract

In most industries and organizations, a large part of organizational knowledge resides in documents, but these documents are in unstructured form. Extracting data from such documents has been difficult. One of those industries is the life sciences, and one of the major processes in life sciences is to bring a drug to market with safety and efficacy.

The unique nature of this domain is that on the one hand, we want to ensure safety because it can adversely affect human lives, and on the other, extract maximum value out of the drugs, experiments, and clinical trials.

Traditional systems to support the clinical trials collect data in a document-centric way. Much of this information ends up as PDF documents that are scanned and used in the business process.

The information from these PDF documents is then manually entered into systems for analytics and further processing, which is an expensive exercise. AI, with its advances in robust optical character recognition and natural language processing, can make this conversion much more efficient and accurate. However, extracting data from clinical forms that includes labels, checkboxes, tables, and lines are more challenging because we need to extract the labels and the corresponding values.

At DataFoundry, we have created an intelligent data extraction system using AI technologies to automate the data extraction to any one of many structured formats. The system does minimal manual annotations to capture the semantics of specific sections for any particular document template. Once that has been done then millions of documents can be fed through the system to extract information automatically.

In this paper, we will describe the business drivers behind such a system, the architecture of the system, show how the system performs compared to human levels, and also showcase examples of documents processed. We will discuss the difficulties around exacting information from checkboxes and share details about the neural network architecture we used to achieve high accuracy.

## Background

One of the challenges in the field of Document Analysis is, document structure detection. Here, techniques from computer vision, a sub-discipline of artificial intelligence, are used for (1) information detection and (2) extraction. In their article “Histograms of oriented gradients for human detection” [1] Navneet Dalal and Bill Triggs describe the histogram of oriented method to detect the objects from images. While this approach addresses many practical object detection and extraction problems, more advanced techniques use deep neural networks, yet another sub-discipline of AI, to address more complex problems. Such classical deep convolutional neural networks extract feature maps to extract more complex objects as shown in the work by Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus and Yann LeCun [2]. Specifically for document analysis, Neural Networks such as RCNN [3], fast RCNN [4], Faster RCNN [5], Yolo [6], RetinaNet [7,8] and then CornerNet [9] has shown good results to detect and extract complex document content such as graphs, figures, tables, checkboxes etc.

Checkbox detection is a unique problem in document analysis, because in addition detecting the checked box, we also need to detect the text which corresponds to the checked box. In his Master's thesis "Optical character recognition for checkbox detection" [10] the author uses a three-step procedure for checkbox detection:

1. image preprocessing,

2. box detection through optical character recognition (OCR), and,

3. checkmark detection through optical mark recognition.

IBM Knowledge Center [11] also proposes OCR usage for box detection, but they use a pixel threshold evaluation method, wherein the ratio of black to white pixels of the box area determines whether the box is checked. Another approach was offered by the authors of "Automatic Recognition Method for Checkbox in Data Form Image" [12]. They detect skews in image and then use the handwritten symbol recognition to solve the problem. However, this works only for handwritten boxes checked.

However, none of the approaches solve the problem of the text detection in the checked box. In the clinical trial life sciences space specifically, our analysis of different documents shows, that text can be both on the left and on the right of the checkbox. The RCNN (recurrent convolution neural network) family can be implemented for this task, but since they are based on ruled base Selective Search or Edge Boxes detection algorithms, they are extremely time and resources consuming. After analyzis and experimentation to find a more efficient approach, we implemented a variation of the RetinaNet NN model which achieved high performance mean Average Precision (mAP).

**Problem Statement**

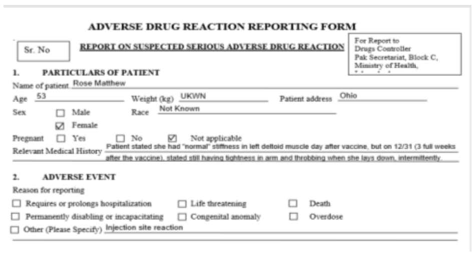
The intent of the document analysis project was to create a web application, which will disaggregate multiple PDF format documents of a same type and reaggregate the informational content to make it identifiable, retrievable and easily readable. Generally, data in the pdf documents is semi-structured. As opposed to well-structured data, which conforms to a schema or data model and can be queried using structured query language to answer questions, the content does not adhere to any rigorous format.

**Solution**

To extract structured data as described in problem statement, we used the following steps:

| **Step 1:** | **Step 2:** | **Step 3:** | **Step 4:** |
|---|---|---|---|
| Annotate a single document of each type | Extract data from multiple documents in accordance with initial annotation | Identify only information from checked boxes for extraction | Create structured data in a csv format |

**Step 1** of the algorithm is to annotate a single document of a given type. See Figure 1 for examples of uploaded and annotated documents.



ADVERSE DRUG REACTION REPORTING FORM

Sr. No

REPORT ON SUSPECTED SERIOUS ADVERSE DRUG REACTION

For Report to
Drugs Controller
Pak Secretariat, Block C,
Ministry of Health,

1. PARTICULARS OF PATIENT

Name of patient Rose Matthew

Age 53 Weight (kg) UKWN Patient address Ohio

Sex ☐ Male [✓ Female] Race Not Known

Pregnant ☐ Yes ☐ No [✓ Not applicable]

Relevant Medical History Patient stated she had "normal" stiffness in left deltoid muscle day after vaccine, but on 12/31 (3 full weeks after the vaccine), stated still having tightness in arm and throbbing when she lays down, intermittently.

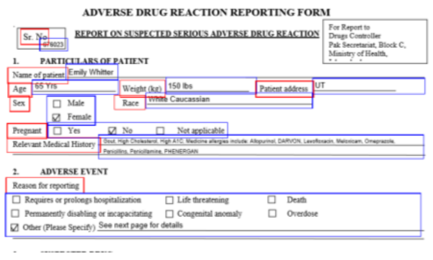
2. ADVERSE EVENT

Reason for reporting

☐ Requires or prolongs hospitalization ☐ Life threatening ☐ Death

☐ Permanently disabling or incapacitating ☐ Congenital anomaly ☐ Overdose

☐ Other (Please Specify) Injection site reaction



ADVERSE DRUG REACTION REPORTING FORM

Sr. No 678023

REPORT ON SUSPECTED SERIOUS ADVERSE DRUG REACTION

For Report to
Drugs Controller
Pak Secretariat, Block C,
Ministry of Health,

1. PARTICULARS OF PATIENT

Name of patient Emily Whitter

Age 65 Yrs Weight (kg) 150 lbs Patient address UT

Sex ☐ Male [✓ Female] Race White Caucassian

Pregnant ☐ Yes [✓ No] ☐ Not applicable

Relevant Medical History Gout, High Cholesterol, High A1C, Medicine allergies include: Allopurinol, DARVON, Levofloxacin, Meloxicam, Omeprazole, Penicillins, Penicillamine, PHENERGAN

2. ADVERSE EVENT

Reason for reporting

☐ Requires or prolongs hospitalization ☐ Life threatening ☐ Death

☐ Permanently disabling or incapacitating ☐ Congenital anomaly ☐ Overdose

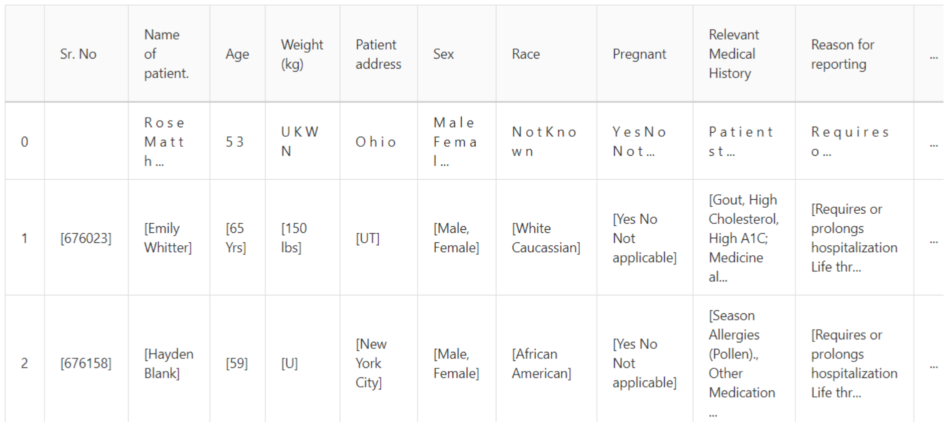
[✓ Other (Please Specify)] See next page for details

*Figure 1: An example of an unannotated document (left) and annotated document (right).*

The document are annotated so that the key and value pairs would be identifiable. For example, (name, Smith), and (age, 58 years) where the key is the question, and value is the corresponding answer. In Figure 1 we show the annotated keys with red boxes, and the annotated values with blue boxes.

**Step 2.** Using the Python programming library fitz, we can detect the red and blue boxes and extract their corresponding coordinates. We also detect the text within the boxes and extract them into a Python data structure. Thus, keeping the coordinates of the boxes, we can extract the information from the boxes from any document of the same type. As you can see in *Figure 2*, the data from three analogous documents for three different patients was extracted from the annotated boxes into a single document.

**Commented [RR1]:** Should this be two instead of three? If you have three, can you update the figure 2?



| | Sr. No | Name of patient. | Age | Weight (kg) | Patient address | Sex | Race | Pregnant | Relevant Medical History | Reason for reporting | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | | R o s e M a t t h ... | 5 3 | U K W N | O h i o | M a l e F e m a l ... | N o t K n o w n | Y e s N o N o t ... | P a t i e n t s t ... | R e q u i r e s o ... | ... |
| 1 | [676023] | [Emily Whitter] | [65 Yrs] | [150 lbs] | [UT] | [Male, Female] | [White Caucassian] | [Yes No Not applicable] | [Gout, High Cholesterol, High A1C; Medicine al... | [Requires or prolongs hospitalization Life thr... | ... |
| 2 | [676158] | [Hayden Blank] | [59] | [U] | [New York City] | [Male, Female] | [African American] | [Yes No Not applicable] | [Season Allergies (Pollen)., Other Medication ... | [Requires or prolongs hospitalization Life thr... | ... |

*Figure 2: Structured data received from three documents of the same type.*

However, to retrieve only information from a checked box is yet another challenge. In *Figure 3* we can see that for section “Sex” both options – male and female are retrieved. Information from the annotated boxes is retrieved fully irrespective of the checked box in front of only one of the options. That is a problem.

**Step 3.** To overcome that problem and detect the box that is checked, we used a RetinaNet deep learning network. We propose a completely new solution to the problem of checkbox detection: instead of detecting the box itself, we trained the network in a manner, that it could identify the bounding box of the text near the checked box (see *Figure* 3). This facilitates the work, because it reduces the number of steps in the code. It also gives a possibly higher accuracy, since we don't have to rely on engineering solutions to find out whether the check box is on the left of the text or on the right of it.

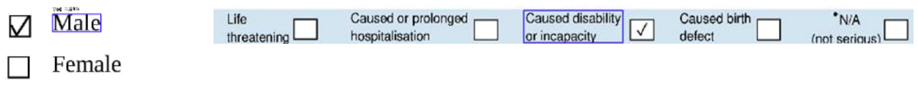

Male
Female
Life threatening
Caused or prolonged hospitalisation
Caused disability or incapacity
Caused birth defect
N/A (not serious)

*Figure 3: Detection of the text near the checked box using RetinaNet deep learning network.*

We have also considered cases where there is an additional information not only near the checkboxes, or the words selected are circled instead. For instance, in *Figure 4*, "56 years" is the output of our program.

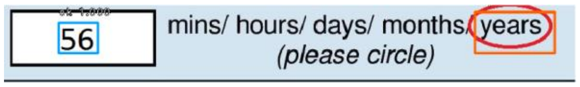


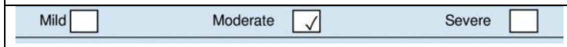
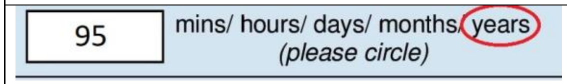
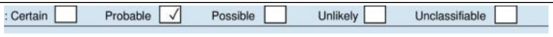
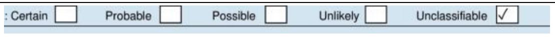
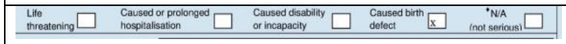
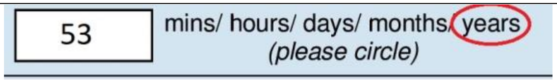
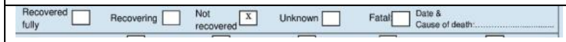
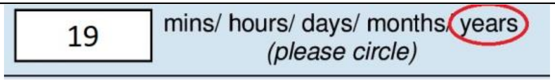
*Figure 4: Detection of circled words and additional words.*

After coordinate detection of the bounding box of the text, we proceed to the **Step 4**. The image is cropped and is passed to pytesseract Python library, which uses tesseract OCR. Tesseract is an  optical character recognition engine released under the Apache License and its development has been sponsored by Google since 2006. The OCR transferred text is then inserted to the corresponding space in the data structure shown in Figure 3.

**Data Preprocessing for RetinaNet Neural Network**

The data for the network was collected from various documents containing different type of checkboxes (*Figure* 3 and *Figure 4*). Checkmarks inside the boxes also vary, being X-shaped or V-shaped. Some noise, like dots of different colors, were added to the images to avoid overfitting. Negative examples included both – images with all unchecked boxes and images without any checkboxes on them.

For positive data, we wrote code using pytesseract python library to give as an output an hOCR file. hOCR is an open standard of data representation for formatted text obtained from optical character recognition, which among many other things contains the bounding box coordinates of the words [14]. For each part of the image, the bounding box coordinates of the words were found and were given as labels of the data. Three different type of classes were created: one for words near the checkboxes for cases represented in *Figure* 3, and two additional classes for cases presented in *Figure 4*: one for the text input without checkboxes, and the other for circled words. We named the classes “yes,” “ok,” and “ellipse.” You can see an example of preprocessed data in *Figure 5*.

| **Checkbox image** | **X$_{min}$** | **Y$_{min}$** | **X$_{max}$** | **Y$_{max}$** | **Class name** |
|---|---|---|---|---|---|
|  Mild ☐ Moderate ✓ Severe ☐ | 505 | 18 | 670 | 55 | tick |
| | | | | | |
|  95 mins/ hours/ days/ months/ [✓ years] (please circle) | 80 | 25 | 135 | 65 | notation |
| | 660 | 10 | 77 | 65 | circle |
| | | | | | |
|  : Certain ☐ Probable ✓ Possible ☐ Unlikely ☐ Unclassifiable ☐ | 330 | 15 | 480 | 50 | tick |
| | | | | | |
|  : Certain ☐ Probable ☐ Possible ☐ Unlikely ☐ Unclassifiable ✓ | 1340 | 15 | 1580 | 50 | tick |
| | | | | | |
|  Life threatening ☐ Caused or prolonged hospitalisation ☐ Caused disability or incapacity ☐ Caused birth defect x *N/A (not serious) ☐ | 1385 | 5 | 1620 | 95 | tick |
| | | | | | |
|  53 mins/ hours/ days/ months/ [✓ years] (please circle) | 80 | 25 | 135 | 65 | notation |
| | 660 | 10 | 770 | 65 | circle |
| | | | | | |
|  Recovered fully ☐ Recovering ☐ Not recovered x Unknown ☐ Fatal ☐ Date & Cause of death:........................ | 700 | 10 | 890 | 95 | tick |
| | | | | | |
|  19 mins/ hours/ days/ months/ [✓ years] (please circle) | 85 | 25 | 140 | 65 | notation |
| | 660 | 10 | 770 | 65 | circle |
| | | | | | |
| | 70 | 25 | 125 | 65 | notation |

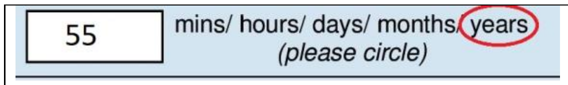
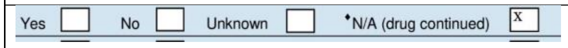
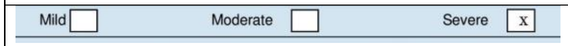
| | | | | | |
|---|---|---|---|---|---|
|  55 mins/ hours/ days/ months/ [✓ years] (please circle) | 660 | 10 | 770 | 65 | circle |
| | | | | | |
|  Yes ☐ No ☐ Unknown ☐ *N/A (drug continued) ☒ | 840 | 25 | 1190 | 70 | tick |
| | | | | | |
|  Mild ☐ Moderate ☐ Severe ☒ | 1100 | 20 | 1225 | 55 | tick |

*Figure 5: Preprocessed data for RetinaNet deep learning neural network.*

Each of the data lines in *Figure 5* contains the x,y coordinates of upper left and lower right corners of the bounding boxes of the label data (colums 2 through 5), and their respective classes (column 6). The data was checked for avoiding possible mistakes and making it work for RetinaNet[1] through keras-retinanet/keras_retinanet/bin/debug.py script designed for debugging images by RetinaNet model developers. To make text detectable by the neural network, anchor parameters of anchor boxes were set to ratios = np.array([0.01,0.03,0.09,0.27, 1], keras.backend.floatx()), scales = np.array([$2^0$, $2^{(1.0 / 3.0)}$, $2^{(2.0 / 3.0)}$, 3.5] in keras-retinanet/keras_retinanet/utils/anchors.py python script. All the coordinates were verified through debug.py

**RetinaNet Training**

One of peculiarities of RetinaNet model is the usage of Focal Loss ($FL(p_t) = -(1 - p_t)^{\gamma} * \log(p_t)$) function during the classification [15]. This conceptual interpretation of this function is that it increases the overall contribution of positive examples in the training process. Data imbalances of having either too many positively tagged data or too many negatively tagged data can result in machine learning models trying to generalize too much, which is why they have to be 'balanced' first. The Focal Loss is designed to address the one-stage object detection scenario in which there is an extreme imbalance between foreground and background classes during training (e.g., 1:1000).

Training data contained 1205 samples. The parameters of training are as follows: batch_size=1, steps=1249, epochs=50. We used pretrained COCO weights for RetinaNet.

1 https://github.com/fizyr/keras-retinanet

We used random-transform and no-resize parameters while training. Random transform usage avoids the need to specify the min_rotation, max_rotation, min_translation, max_translation, min_shear, max_shear, min_scaling, max_scaling, flip_x_chance, flip_y_chance transformation parameters on initial pictures. No-resize is designed to feed the network with pictures of original sizes, and not set one-size-for-all to them

**Evaluation metrics**

The mean average precision (mAP) is a main evaluation metric for RetinaNet [15]. The tables of **Precision and Recall** are calculated for mAP estimation. We assumed that score_threshold=0.4 and iou_threshold=0.5.

| Rank | Precision | Recall | True_positives |
|---|---|---|---|
| 1. | 1.00 | 0.00 | 1.0 |
| 2. | 1.00 | 0.01 | 2.0 |
| 3. | 1.00 | 0.01 | 3.0 |
| … | … | … | … |
| 150. | 1.00 | 0.56 | 150.0 |
| 151. | 1.00 | 0.57 | 151.0 |
| 152. | 1.00 | 0.57 | 152.0 |
| … | … | … | … |
| 282. | 0.90 | 0.95 | 254.0 |
| 283. | 0.90 | 0.95 | 254.0 |
| 284. | 0.89 | 0.95 | 254.0 |
| For class “tick" **average_precision 0.9483, num_annotations 267.0** | | | |

*Figure 6: Evaluation metrics for class “tick”*

| Rank | Precision | Recall | true_positives |
|---|---|---|---|
| 1. | 1.00 | 0.01 | 1.0 |
| 2. | 1.00 | 0.02 | 2.0 |
| 3. | 1.00 | 0.03 | 3.0 |

| | | | |
|---|---|---|---|
| … | … | … | … |
| 50. | 1.00 | 0.51 | 50.0 |
| 51. | 1.00 | 0.52 | 51.0 |
| 52. | 1.00 | 0.53 | 52.0 |
| … | … | … | … |
| 96. | 0.98 | 0.96 | 94.0 |
| 97. | 0.98 | 0.97 | 95.0 |
| 98. | 0.98 | 0.98 | 96.0 |
| For class “notation" average_precision 0.9753, num_annotations 98.0 | | | |

*Figure 7: evaluation metrics for class “notation”*

| Rank | Precision | Recall | true_positives |
|---|---|---|---|
| 1. | 1.00 | 0.02 | 1.0 |
| 2. | 1.00 | 0.04 | 2.0 |
| 3. | 1.00 | 0.06 | 3.0 |
| … | … | … | … |
| 22. | 1.00 | 0.47 | 22.0 |
| 23. | 1.00 | 0.49 | 23.0 |
| 24. | 1.00 | 0.51 | 24.0 |
| … | … | … | … |
| 45. | 1.00 | 0.96 | 45.0 |
| 46. | 1.00 | 0.98 | 46.0 |
| 47. | 1.00 | 1.00 | 47.0 |
| For class “circle" **average precision 1, num_annotations 98.0, num_annotations 47.0** | | | |

*Figure 8: Evaluation metrics for class "circle"*

Based on the tables we found mAP=(0.9483+ 0.9753+ 1.0000)/3=0.9745 for test set and analogically for train set we have mAP=0.9816 which are very good.

**Conclusion**

Automating document processing in life sciences for processing clinical trials data is quite challenging. The semi structured nature of PDF documents make it difficult to have a rule-based approach to extract name value pairs, that can then in turn be fed into a structured database for further processing. Machine learning and specifically deep learning approaches hold promise to provide better results.

Checkbook box detection is among the most challenging because the text corresponding to the check box may be on the right, or on the left, or any other location close to the check box itself. In addition, the variation of how the box is checked is high. There is no way to programmatically identify the box's label and its value.

We used a modified RetinaNet model for detecting/predicting checkboxes and their corresponding values. RetinaNet outputs the probabilities of checkboxes (i.e. the probability of existence of a checkbox), and we consider boxes as checked if the probabilities are higher than 0.5. It has been shown that our model detects checkboxes with high accuracy, approximately 90% on a given test data.

While this experiment was done on domain specific PDF documents, the same concept can be applied to documents in any other domain as long as they are trained using the appropriate deep neural network. Often in business, there is a lot of unstructured and semi structured documents (such as in Microsoft Word), and if we can systematically and automatically extract and tag that information, it will be valuable for the business leading to better and faster decision making, reduced cost, and reduced errors.

**References**

1. Navneet Dalal and Bill Triggs Histograms of oriented gradients for human detection //2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05).

2. Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks // arXiv:1312.6229 [cs.CV] 21 Dec2013

3. Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik Rich feature hierarchies for accurate object detection and semantic segmentation // arXiv:1311.2524v5 [cs.CV] 22 Oct 2014

4. Ross Girshick Fast R-CNN // arXiv:1504.08083v2 [cs.CV] 27 Sep 2015

5. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks // arXiv:1506.01497v3 [cs.CV] 6 Jan 2016

6. Joseph Redmon, Santosh Divvala, Ross Girshick , Ali Farhadi You Only Look Once: Unified, Real-Time Object Detection // arXiv:1506.02640v5 [cs.CV] 9 May 2016

7. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollar, Focal Loss for Dense Object Detection // arXiv:1708.02002v2 [cs.CV] 7 Feb 2018

8. Yixing Li, Fengbo Ren Light-Weight RetinaNet for Object Detection // arXiv:1905.10011v1 [cs.CV] 24 May 2019

9. Hei Law · Jia Deng CornerNet: Detecting Objects as Paired Keypoints // arXiv:1808.01244v2 [cs.CV] 18 Mar 2019

10. Istle, J. M. (2004). *Optical character recognition for checkbox detection* [Master's thesis]. https://digitalscholarship.unlv.edu/cgi/viewcontent.cgi?article=2723&context=rtds

11. IBM Knowledge Center https://www.ibm.com/support/knowledgecenter/en/SSZRWV_9.0.0/com.ibm.dc.develop.doc/dcadg363.htm

12. Zhang, S., Yuan, S., & Niu, L. (2014). *Automatic Recognition Method for Checkbox in Data Form Image*. 2014 Sixth International Conference on Measuring Technology and Mechatronics Automation. https://ieeexplore.ieee.org/abstract/document/6802658/authors#authors

13. Tesseract (OCR), Wikipedia, https://en.wikipedia.org/wiki/Tesseract_(software)

14. hOCR, Wikipedia, https://en.wikipedia.org/wiki/HOCR

15. Lin, T., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2017.324

16. Mean average precision. (n.d.). *SpringerReference*. https://doi.org/10.1007/springerreference_65277