Benchmarking with divisions

1. Introduction

In the last sprints a method for extracting information of a pdf considering semantic visual information was defined. It basically consists on the use of a yolo model trained to identify components of a document and a OCR model, until this moment tesseract, that is used for text extraction.

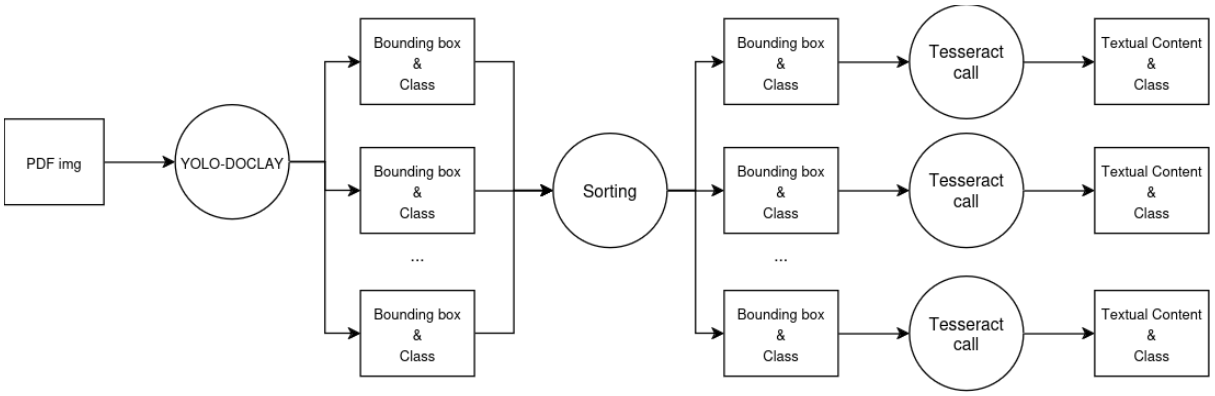
While testing this method, possible performance issues showed up, and with them a necessity of benchmarking, focusing on time consumption of each step. Until this moment, the benchmarking process was made using the document data from Manisha717's pdf documents dataset.

After well defining the benchmarking process, a subdivision of the dataset was proposed. This division could highlight properties or possible inferences that would be possible to make over a document considering a more specialized context.

In this report, along with a recap of the benchmarks steps and metrics, is specified the method used to separate the dataset in multiple independent and disjoint parts and the results yielded from it.

2. Benchmarking Recap

Initially, it is important to have an overview of the whole extraction process.



At the beginning of the data flow, an image of a PDF page is consumed as input for the Yolo model that outputs a series of bounding boxes alongside their classes and metadata.

These boxes are all sorted based on their position. After sorting, for each of the boxes, a tesseract call is made. With tesseract it is possible to get the text contained inside the bounding box area. Joining textual content and the class information for the ordered list of bounding boxes, it is possible to reassemble the document page with different layouts.

It is also important to say that, tesseract is not called for those boxes with classes "picture" and "table" since both of those are handled as having image content.

a. Metrics

Since the situation that fostered the benchmark points to a lack of performance, the metrics are more related to time consumption, and not to accuracy (also because accuracy evaluations over the used models can already be found here).

One of the most important notions to have after the benchmarking is: which step is responsible for the time consumption. To have a clear vision of this, segregate yolo's inference time and tesseract calling phase time. Sorting time is going to be considered irrelevant since it doesn't rely on models, and a pdf page tends to have a really small number of boxes for sorting algorithms metrics.

Since tesseract calling phase consists of many different operations, it is important to have other metrics associating the phase time with the number of non image bounding boxes.

At the end, the selected metrics are:

Metric	Description
Total time	The entire process time
Yolo Time	Time spent on YOLO's detection and classification
Tes Time	Time spent on Tesseract calling phase
N boxes	Number of non image boxes
box/s	"N boxes" divided by "Tes Time"
average box span	The average time spent in a tesseract call for a single non image box
Longest box span	Smallest time spent in a tesseract call for a single non image box
Smallest box span	Smallest time spent in a tesseract call for a single non image box

Division Method

To generate the divisions of the dataset, a script was created. Given the list of files of the dataset and a number of parts K, the script, randomly selects files, filing k-1 parts of size equals to the integer part of $dataset\ size\ /\ k$. After being selected, a file, evidently, is removed from the original list, giving only to the non selected files the opportunity to be chosen. Finally the files that remain at the original list are considered as the final part.

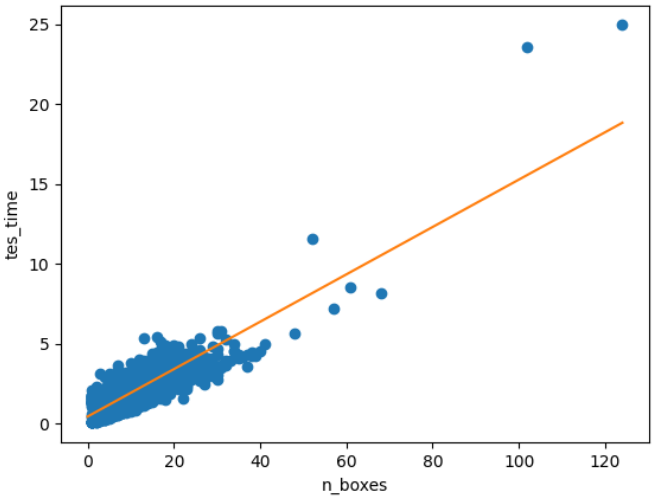
With this script, folds with the following sizes were created:

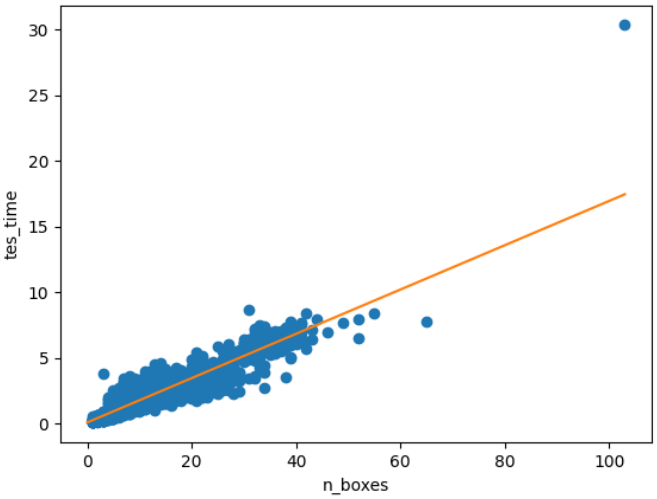
K	size of all k-1 parts	size of the last part
2	531	532
5	212	215
10	106	109

After the division, each part was submitted to the benchmarking process sequentially. With the execution of a part concluded, the logs related to the part were separated, making possible to individually analyse each of the subdivisions

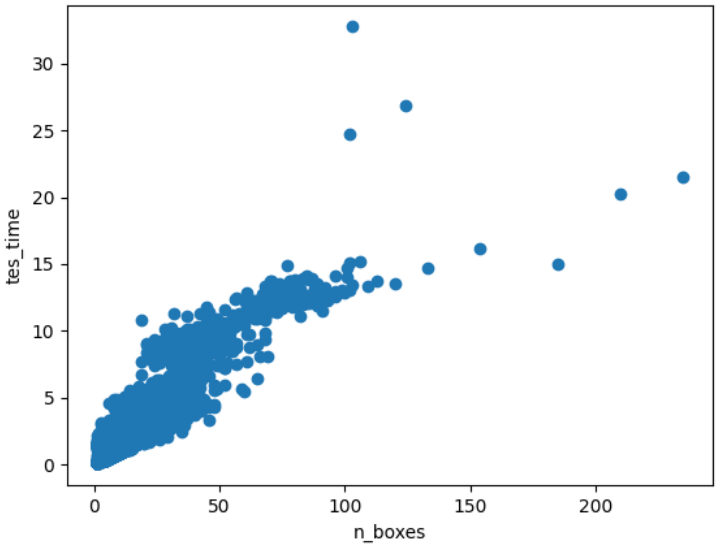
Results

After 12694 document pages and 150704 textual boxes for each division, 38082 logs were generated. The difference between parts intuitively grows with the number of divisions of the dataset, generating subsets that could even indicate a nearly linear relationship between total time/tesseract time with n boxes.

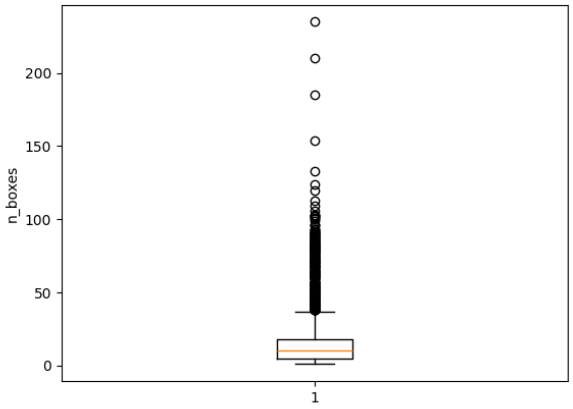


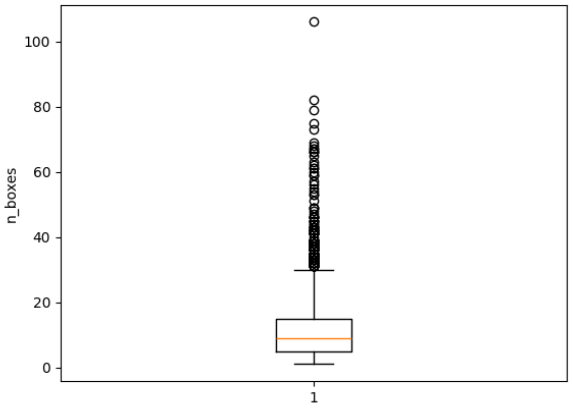


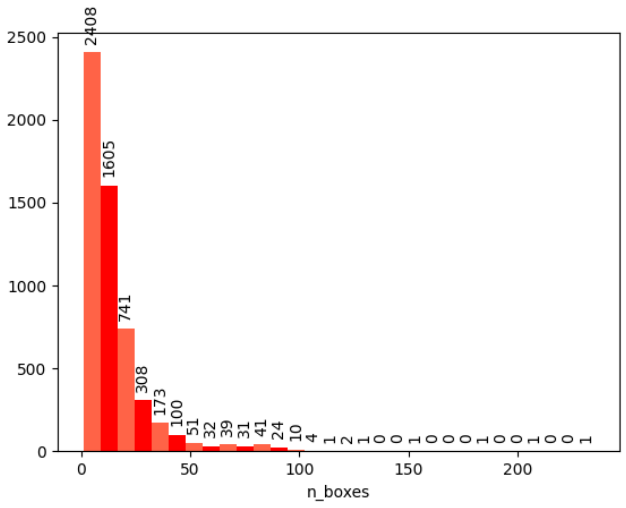
With bigger subsets, the result tended to be similar to what was found with the complete dataset.

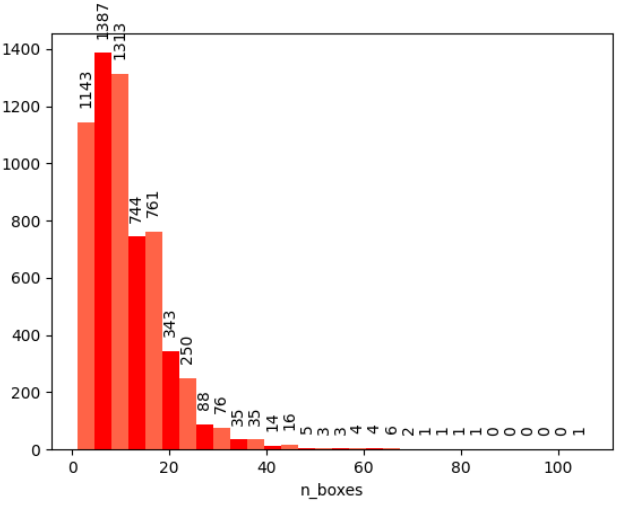


Else than this, it is also possible to see that the number of boxes is condensed below 50 boxes.

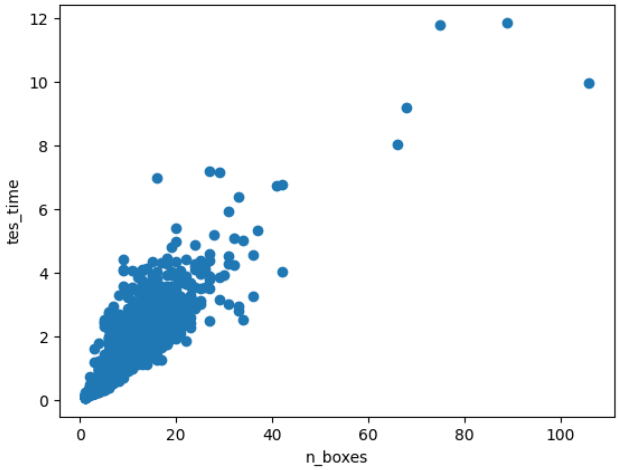


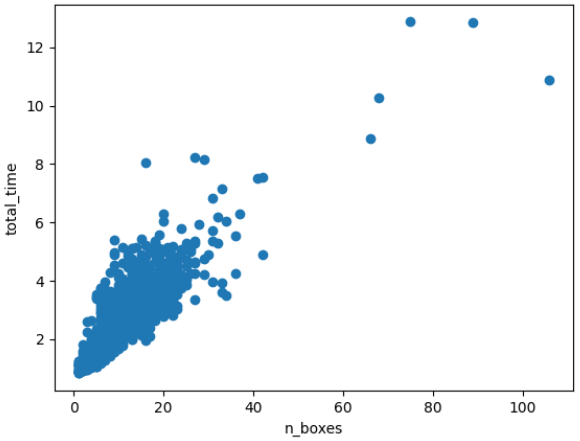


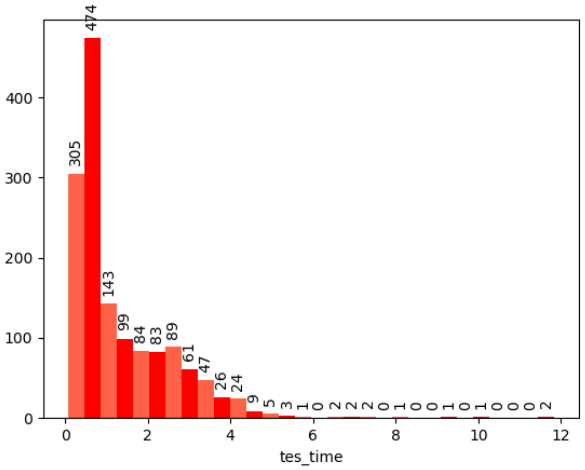


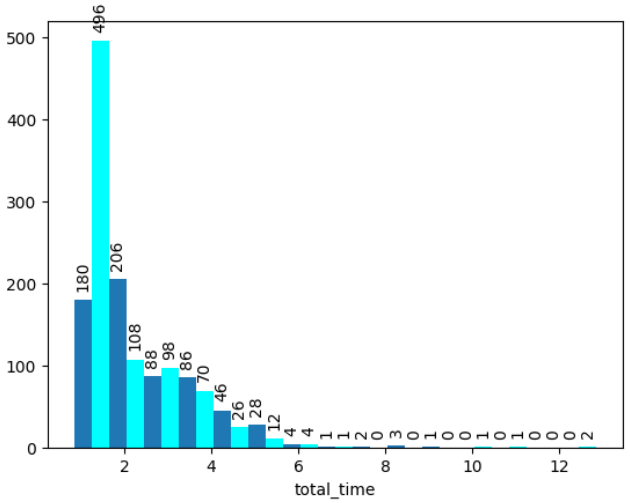


The similarity between the tesseract phase time growth and the total process time growth remains, even in the context of smaller samples.









This occurs because clearly tesseract phase outgrows every other phase time by a large margin in more extreme cases. A proof of this is that the yolo phase doesn't last more than 1.5 seconds in a page in its worst performances, while the tesseract phase surpasses 30 seconds in a single page.