

Object detection deep learning networks for Optical Character Recognition

In this article, we show how we applied a simple approach coming from deep learning networks for object detection to the task of optical character recognition in order to build image features tailored for documents. In contrast to scene text reading in natural images using networks pretrained on ImageNet, our document reading is performed with small networks inspired by MNIST digit recognition challenge, at a small computational budget and a small stride. The object detection modern frameworks allow a direct end-to-end training, with no other algorithm than the deep learning and the non-max-suppression algorithm to filter the duplicate predictions. The trained weights can be used for higher level models, such as, for example, document classification, or document segmentation.

Introduction

Document images make the use of deep learning networks a complex task, since most deep learning network architectures have been designed and trained for natural images, making them useless for document images which are mainly white and black characters and figures. This is in particular the case for classification networks (VGG, ResNets, ...), object detection networks (Fast RCNN, Faster RCNN, Yolo, SSD, ...), segmentation networks (FCN, U-Net, SegNet, DeconvNet, Dilated-Net, ParseNet, DeepLab...) which cannot be applied directly, even with finetuning.

Two challenges arise with deep learning and document data. First, we need to train specific features for the type of data. Second, the available datasets can be of smaller size than classical datasets for computer vision (ImageNet, COCO, ...), in particular when it is required to annotate images for a specific purpose.

To reduce the amount of data to train high level descriptions of the document images, such as document zones, segments, lines, or elements, the idea is to train a smaller network on OCR data which exists at massive scale, and use the weights of this small network as pretrained early layers in bigger networks to perform high level tasks with less required data.

In our experiments, we show that best performing approaches currently available for object detection on natural images can be used with success at OCR tasks. Code is released on Github at https://github.com/lvalua/object_detection_ocr, so that the open research community can bring the best model architectures in terms of accuracy and speed/size efficiency.

Related work:

In this section we quickly review the literature on OCR and object detection.

Approaches for OCR

Most deep learning approaches using Object Detection methods for OCR are applied to the task of *scene text recognition* also called *text spotting*, which consists in recognizing image areas of text, such as a sign or a wall plaque. Once the text area is recognized, a reading method is applied inside the zone. Some approaches use *weakly supervised training* either using a CTC loss leaving the alignment between the character positions and the output result to a recurrent network such as bi-directional LSTM ([4], [5], [6], [8], [9], [10]) or using a fixed number of softmax classifiers ([11], [12]) ; some other approaches use *guided learning* [7]. These approaches are mainly driven by the Street View SVHN, Uber-Text [3], FSNS [2], Coco-text [1], ICDAR 2003 [17] and 2015 [18], SVT and IIIT5K [13], Synth90k [16] datasets.

Rather than recognizing at word level or scene text level, few approaches concern direct detection of characters in natural images, using a localization network in ST-CNN [11], or modern object detection approach in yolo-digits [38] to recognize digits in natural images.

This work is the first to apply modern object detection deep learning approaches to document data with small convolutional networks, without converting them to natural images as in [26]. [27] shows that document classification accuracy decreases with deeper networks.

Approaches for object detection

Modern object detections approaches are divided into two classes.

The first class yields to the highest accuracy object detectors, such as Fast-RCNN [35], Faster-RCNN [36], Mask-RCNN (Detectron) [37], and is based on the two-stage approach of R-CNN [34]. In the first stage, an algorithm, such as Selective Search, or a deep learning model, generates a set of candidate proposals for object regions. In the second stage, a deep learning network classifies the presence of the object (the *objectness*), its class, as well as estimates the precise object bounding box.

In the second class of object detectors, the objectness, the class as well as the bounding box regression, are directly predicted by a single dense deep learning network. These approaches include OverFeat [33], Yolo [32, 38] or SSD [31].

Our approach to OCR

In our work, as a first attempt to use object detection networks to OCR, we design a single stage object detector, predicting the confidence of an object presence, the class, and the regression for the bounding box. In order to cope with multiple scales we use the feature pyramid approach of SSD [31].

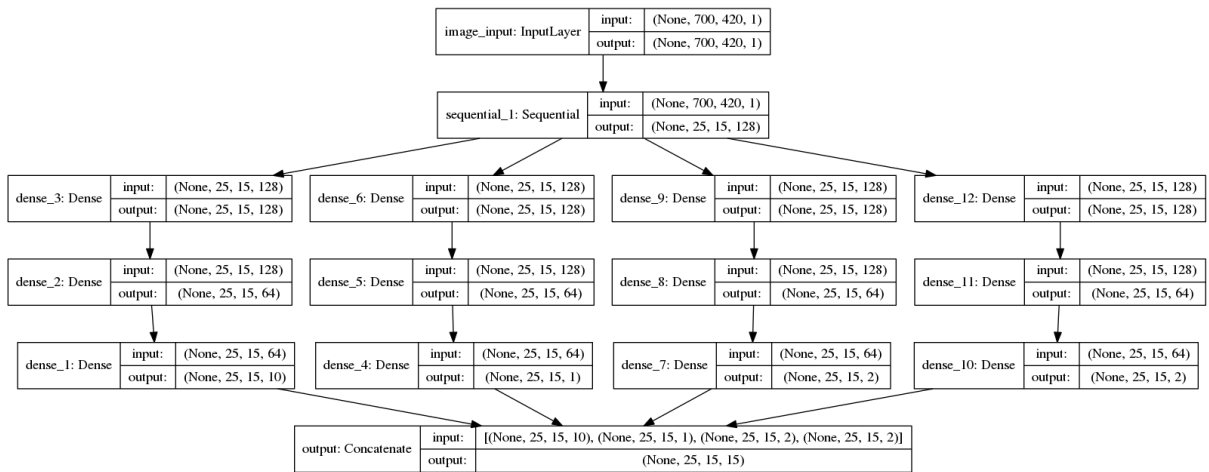
Architectures

Our 1-scale models are inspired by the LeCun model for digit classification except that the dense layer have been converted to convolutions (locally linear) in order to compute a prediction at multiple positions in the image on a grid defined by the stride of the whole network. These models are composed of 2 convolution layers of kernel 3 and 32 and 64 features respectively, followed by a max pooling of stride 2 and 1 convolution layers of kernel 12 and 128 features, so that the receptive field of the network is 28 pixel large and wide. Offset of the model with 'valid' paddings will be 14. We consider the stride of the last convolution as a parameter, *stride_scale*, to adjust the stride of the whole model which will be $2 \times \text{stride_scale}$. On top of these features, 4 stacks of dense layers are used for objectness, classification, position and scale regressions. We named this 1-scale model **CNN_C32_C64_M2_C128_D**.

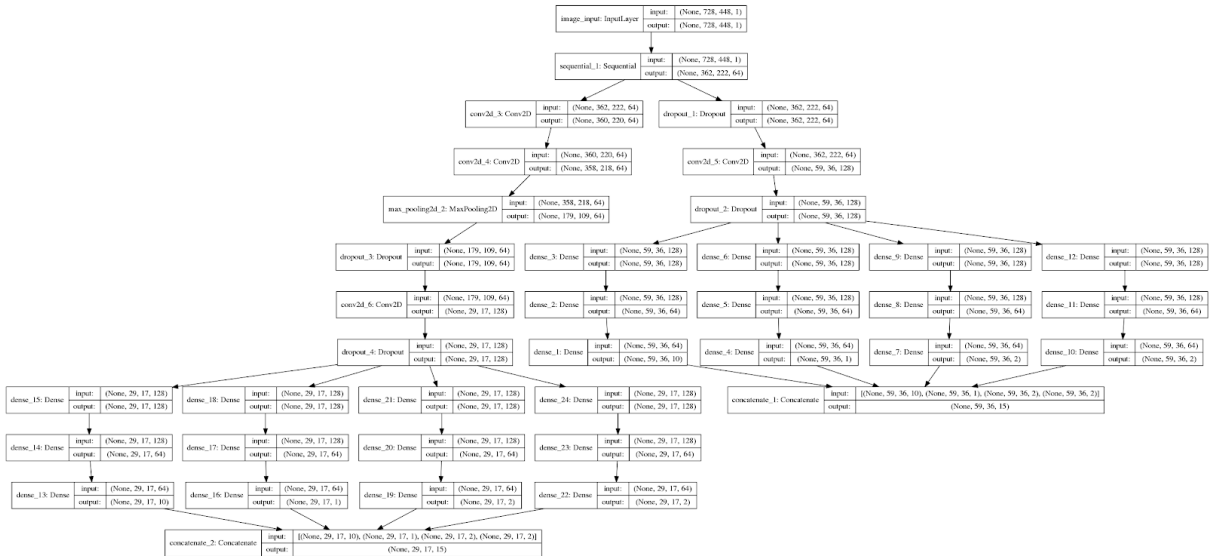
Our 2-scale models are composed of 2 convolution layers of kernel 3 and 32 and 64 features respectively, followed by a max pooling of stride 2 and 2 other convolution layers of kernel 3 and 64 features and another max pooling of stride 2. Each max pooling layer is followed by a convolution layer of kernel 11 and 12 respectively, so that the receptive field for each output is 28 and 56 pixel large and wide. Offset for each output is 14 and 28 respectively. We consider the stride of the output convolutions as a variable parameter, *stride_scale*, to adjust the stride of the whole model which will be $2 \times \text{stride_scale}$ for the first output, and $4 \times \text{stride_scale}$ for the second output. On top of these 2 stages of features, 4 stacks of dense layers are used as well. We name this 2-scale model **CNN_C32_C64_M2_C64_C64_M2_C128_D_2**.

Each layer is followed by a ReLU activation, except for the outputs: objectness is computed with a sigmoid, classification with a softmax, position with hyperbolic tangent and scale with sigmoid.

1-scale model:



2-scale model:



Loss

For objectness, we need to consider the abundance of negative positions compared to positive positions. That is why we use the Tensorflow expression of weighted crossentropy designed to ensure stability and avoid overflow:

$$(1 - z) * x + l * (\log(1 + \exp(-\text{abs}(x)))) + \max(-x, 0))$$

where $l = (1 + (q - 1) * z)$ and $x = \text{logits}$, $z = \text{targets}$, $q = \text{pos_weight}$. We found that a positive weight of 1000 works well on our OCR dataset.

The loss for classification and regression are crossentropy loss and MSE loss. We found that adding a multiplier of 100 to the regression loss help converge faster.

Computation of average precision

It is common to use the mAP score as the final metric for object detection. In our case, we consider all classes as one class in order to use average precision as metric to measure the capacity of the models in terms of objectness and not classification. We use the name *object mAP* to distinguish it from classical mAP score.

The reason for this choice is that we focus on character detection in this work. For full character recognition, early results suggest that two-stage detectors might be of better fit than a one-stage detector, because in our 1-stage setting, classification accuracy drops when the classification network is not well positioned on the character (see Stride experiments on Mnist), and this argument could give an advantage to 2-stage detectors.

Later on, we might add a second stage on top of this network as in Mask RCNN or Faster RCNN and this network might become a *region proposal network*. We leave this as future work, which purpose will be to improve the classification accuracy.

Datasets

1. Toy dataset

We build a toy dataset in order to test our implementations on a simpler task and check that the implementation is correct. For that purpose, we use the MNIST handwritten digits dataset to create pages with handwritten digits, at fixed or variable scales, with or without noise. The number of object classes is 10, the digits ["0", "1", "2", "3", "4", "5", "6", "7", "8", "9"].

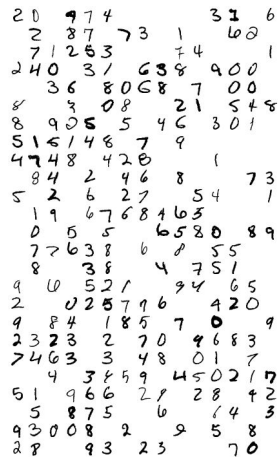
Our MNIST dataset is composed of 1600 images of size 728x448, consisting of 28x28 square digits randomly placed on a 2D grid of stride 28.

MNIST (default digit size 28)	MNIST with white prob .4 (≠ .9)	MNIST with noise	MNIST with digit size in 14-28 range
random digits at scale 28 at different positions on a grid	more digits per positions	cluttered with noise added randomly	random digit scale between 14 and 28. position is random



MNIST digit size 56

random digits at scale 56 at different positions on a grid



MNIST with digit size in 28-56 range

random digits scale between 28 and 56. random position



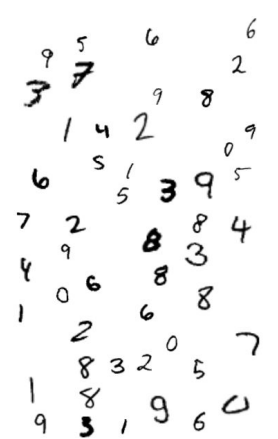
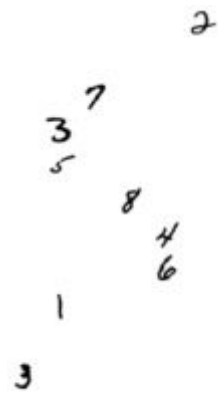
MNIST with 2 digit sizes 28,58

digits at scales 28 and 56. Positions on a grid



MNIST with 2 digit ranges 14-28,28-56

random digit scales between 14 and 56. random positions.



Our MNIST dataset with noise adds random distortions to create a high level of noise on the images and test the robustness of the models.

To check the performance of the position prediction, we set a **different network stride**, for example 12 (setting stride scale to 6), so that the network grid of positions where the model is evaluated in the convolutions, do not fall exactly on the grid of characters. That way, some digits will appear cropped, up to 6 pixels off horizontally and vertically, in the viewpoint of the network, ie its 28x28 receptive field.

To check the performance of the scale prediction, we build a MNIST dataset with digits randomly resized in the [14-28] range.

Before adding a layer to our network architecture as in SSD[31], we also check larger models at a bigger scale, with a MNIST dataset of 56 pixel wide digits.

Last, we build a two-scale dataset for two-layer predictions as in SSD[31], with digits at size 28 and 56, and add a middle output to CNN_C32_C64_M2_C64_C64_M2_C128_D to build CNN_C32_C64_M2_C64_C64_M2_C128_D_2, a two-scale network.

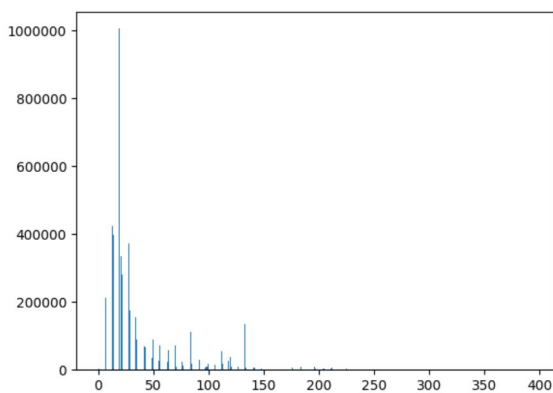
For a continuous scale between 14 pixels and 56 pixels, we build another two-scale dataset with 2 digit size ranges, 14-28 and 28-56.

2.OCR data

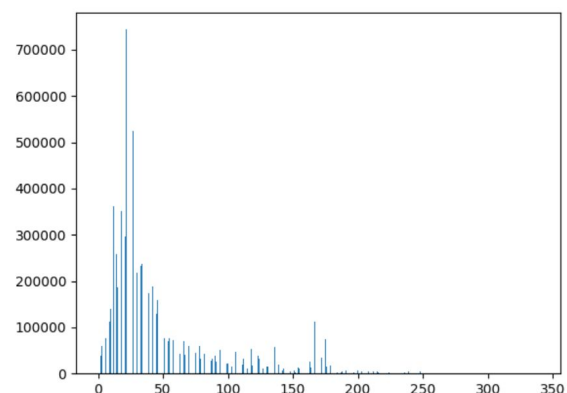
We build our own dataset of 8000 document pages, split into train (90%) and validation (10%) sets, for a fast check of our approach. Document PDF are converted to images with a resolution chosen automatically to have normal sized characters. To have fixed-sized image input for the network batched training, document images are then randomly cropped on a 728x448 area with characters, to have the same sized inputs as our mnist dataset.

We consider uppercase and lowercase letters, digits, the two parenthesis and the % sign. The number of classes is 65: ["0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z", "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z", "(", ")", "%"]

Letters are filtered by their size to fall in the range of [14-56] pixels and we start with two-scale networks ([14-28] and [28-56]) tested on our MNIST dataset.



Character widths



Character heights

Experiments

Implementation details

Code has been developed under Python with Keras deep learning framework, for Tensorflow and CNTK compute engines. It is compatible with Python 2.7 and 3.5 and allows multi-gpu training.

For training, batch size is 3, the optimizer is Adam and the learning rate 0.001. Hyperparameters are searched by simple grid search.

To create the OCR dataset, we use Tesseract OCR on 10 000 documents.

Toy dataset

Digits centered in network field of view

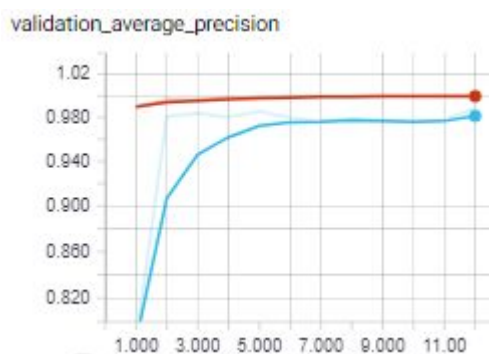
On the MNIST toy dataset, digits are always centered on a grid (of 28x28).

A 28-pixel-strided LeCun convolutional model offers a class accuracy above 99.2% since every positive position falls centered on the digit centered on a grid of stride 28. Object mAP score is >0.99 at 12 epochs with our simple model CNN_C32_C64_M2_C128_D.

With noise, object mAP score with our simple model CNN_C32_C64_M2_C128_D is >0.98. Classification accuracy drops to 98.7.

mnist (#ce7562)

mnist_noise (#98cfe6)



Dataset	Obj acc	Class acc	Reg acc	Obj mAP
---------	---------	-----------	---------	---------

MNIST	100	99.2827	1.60e-10	99.93
MNIST with noise	99.62	98.92	4.65e-6	98.41

The effect of stride and IOU on average precision

To test the network capability to predict position, we need to use a network stride different than the data grid stride. For example, with stride 12 instead of 28, most digits are not anymore in the center of the network reception field (except first row and column of our image).

Also, most digits will appear cropped in the field of view of the network and the IOU threshold defines how much crop ratio will be allowed to still consider the position on the grid as positive.

In the worst case, the network stride can be so large that some digits do not appear on the output grid, equivalent to an Intersection Over Union (IOU) of zero (intersection area is zero). Usually, the stride is not larger than the receptive field, and under this condition, the maximal intersection area between any digit and any network field is 50% times 50% = 0.25, while the union is 1.75, leading to a **minimal IOU of 0.14**. In the case of a smaller stride, for example 12 as below, the IOU threshold can be set higher without losing any digit for reconstruction:

IOU	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
.15	96.37	36.25	0.010	99.97	100
.2	98.42	28.56	0.012	99.75	100
.25	97.05	36.42	0.015	99.52	100
.3	98.35	92.78	0.0013	99.88	100
.35	98.99	83.72	0.0069	99.22	100
.4	98.70	94.96	0.0066	98.37	100

.5	96.71	95.46	0.0062	91.09	95.71
.6	99.92	98.23	4.8e-05	51.80	54.32
.8	99.90	97.90	7.67e-05	8.5	10.63
.95	99.94	97.27	3.7-07	10.80	12.21
.99	99.91	97.66	7.06e-07	9.3	11.71

The large drop in classification accuracy for a small stride suggests that classification would benefit from better localized digits in the receptive field, which would encourage the use of 2-stage detectors.

To reduce the impact of the stride, we set a *stride margin* (see Experiment section on OCR data) on the digit max size to consider at a layer scale so that there is always one position on the network grid for which the character is fully seen by the network.

Reconstruction of ground truth from target data at 100% is only possible until an IOU threshold of 0.4, after which the network stride should be decreased. With a smaller stride of 4, reconstruction at 100% is possible at most IOU range:

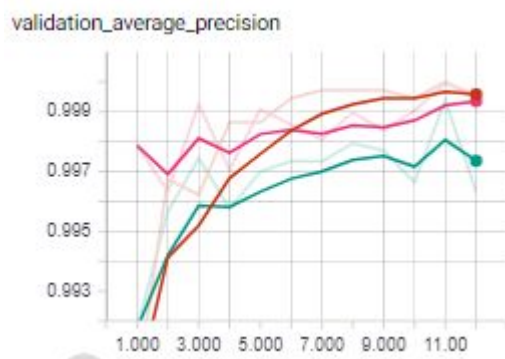
IOU	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
.2	98.51	72.71	0.034	99.99	100
.25	98.63	78.53	0.018	100	100
.3	97.88	94.54	0.0098	99.89	100
.4	96.85	97.41	0.0098	99.93	100
.5	94.14	98.81	0.0099	99.61	100
.6	99.80	98.57	0.00031	99.93	100

.7	99.64	98.21	0.0016	99.77	100
.8	100	98.19	1.7e-8	82.24	100
.8 (30 epochs)	99.98	99.35	1.73e-9	91.05	100

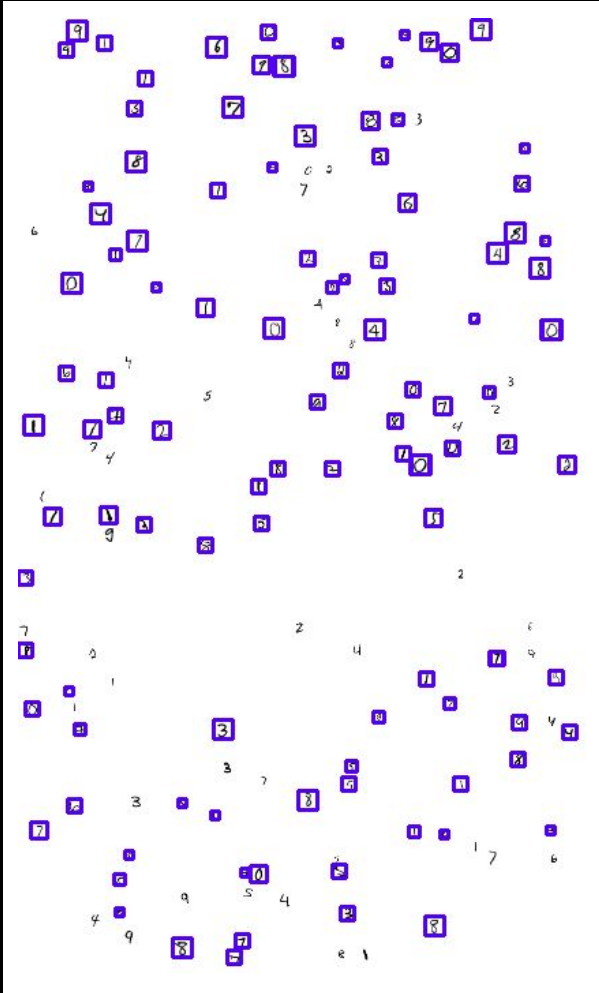
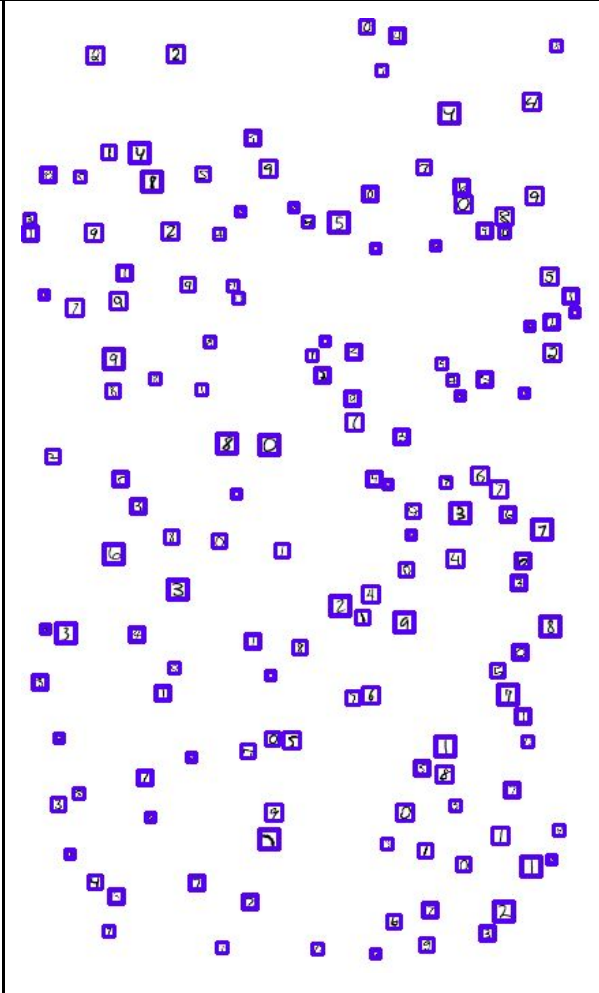
mnist (#d77b66)

mnist_stride_6 (#da88a7)

mnist_stride_12 (#7abfb7)



The images below show the target for an IOU of .2 for digits at scale between 7 and 14 pixels. The right image shows that with a large stride, small digits cut in the receptive field are dropped because of a too small IOU with the anchor, while with smaller stride, the IOU threshold does not remove good candidates. A smaller stride enables to work with higher IOU and better mAP scores.

	
With stride 14, final object mAP is 72.31	With stride 4, final object mAP is 98.61

Digit scale prediction

The second reason (after the *cropped* digits) to use a smaller IOU threshold is to capture *small* digits.

For example, for digits two times smaller, the maximal intersection of a digit with a network receptive field is 0.5 times 0.5 times 0.25 (the maximal intersection area for full size digits of 20), hence 0.0625, while the union is $1+0.5 \times 0.5 \times (1-0.25)=1.1875$ leading to a **minimal IOU of 0.052**. About 3 times smaller than for the full digit size.

With a range scale of 14-28 for the digit sizes, the target object mAP (obtained when we reconstruct the bounding boxes from the target) remains 100% at IOU 0.25 for a stride of 12 pixels. The predicted object mAP is 99.58. The classification accuracy drops down to 89.37%.

Higher capacity networks

Let's double the number of kernels to create CNN_C64_C128_M2_C256_D model. At stride 12 and IOU .3, classification accuracy increases from 92.78 to 96.84, while objectness remains roughly perfect. At stride 6 and IOU .2, it increases from 78.53 to 95.78%.

Parameters	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
Stride 12, IOU .5	99.59	98.02	0.00078	92.32	94.89
Stride 12, IOU .4	99.17	97.23	0.0047	99.79	100
Stride 12, IOU .3	99.74	96.84	0.00043	100	100
Stride 12, IOU .2	97.57	91.14	0.0016	99.98	100
Stride 12, IOU .15	98.02	83.85	0.0083	99.95	100
Stride 4, IOU .5	99.80	98.87	0.00053	100	100
Stride 4, IOU .25	99.48	95.78	0.00054	100	100
14-28 pixel wide, Stride 12, IOU .25	96.56	91.42	0.0045	99.85	100

Multi-stage networks

In order to capture digits in a bigger range than 28 pixels, we try networks with double reception field size, adding more layers (CNN_C32_C64_M2_C64_C64_M2_C128_D model), and possibly, multiple outputs at multiple layer stages (CNN_C32_C64_M2_C64_C64_M2_C128_D_2 model) as in SSD [31].

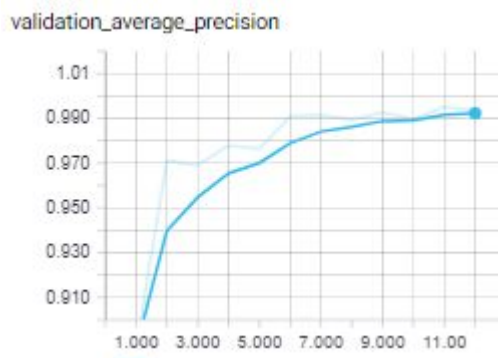
First, we check our model with bigger field, CNN_C32_C64_M2_C64_C64_M2_C128_D model, on the MNIST dataset of 56 pixel wide digits. Object mAP score is 1 while classification accuracy is 99.2% at 12 epochs, meaning this first architecture 56x56 receptive field deals well with digits twice big.

Then we add a second output to our network architecture as in SSD [31] to build CNN_C32_C64_M2_C64_C64_M2_C128_D_2 model, and on a 2-scale dataset with digits at size 28 and 56, object mAP scores remain stable at 99.44 and 99.64 for network strides 12 and 4 respectively.

On a 2-scale dataset with digits at size ranges 14-28 and 28-56, object mAP score with our CNN_C32_C64_M2_C64_C64_M2_C128_D_2 model is 98.82% and for the double size CNN_C64_C128_M2_C128_C128_M2_C256_D_2 is 99.11%.

Model	Digit size	Stride	IOU	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
S	28-56	12	.25	98.99	93.92	0.0018	99.89	100
S	14-28, 28-56	12	.25	98.92/ 98.04	64.06/ 91.08	0.0037/ 0.0056	98.82	99.90
D	14-28, 28-56	12	.2	98.57/ 97.73	58.30/ 79.84	0.0058/ 0.0036	98.31	99.90
D	14-28, 28-56	12	.25	99.10/ 98.16	93.64/ 95.28	0.0016/ 0.0014	98.42	99.93
D, 50 epochs	14-28, 28-56	12	.25	99.26/ 98.78	93.91/ 94.02	0.0010/ 0.0014	98.81	99.93
D, 50 epochs	14-28, 28-56	12	.2	99.05/ 98.05	89.88/ 91.97	0.0021/ 0.0022	99.11	99.97
S	14-56	12	.02	97.58	30.17	0.10	75.07	100

S	14-56	12	.05	97.92	53.20	0.027	75.49	100
S	14-56	12	.1	97.82	58.44	0.0057	87.45	92.67
S	14-56	12	.2	98.82	79.23	0.0010	72.36	75.78



Low resolution

In order to train full document image rather than a 700 pixel high crop of the images, resolution has to be smaller to fit in the GPU. For that reason, we look at models to recognize digits at a maximum size of 14 pixels instead of 28. We build a model CNN_C32_C64_C128_D by removing the max pooling layer. The network input fields becomes 14 pixel wide, and the stride is divided by 2.

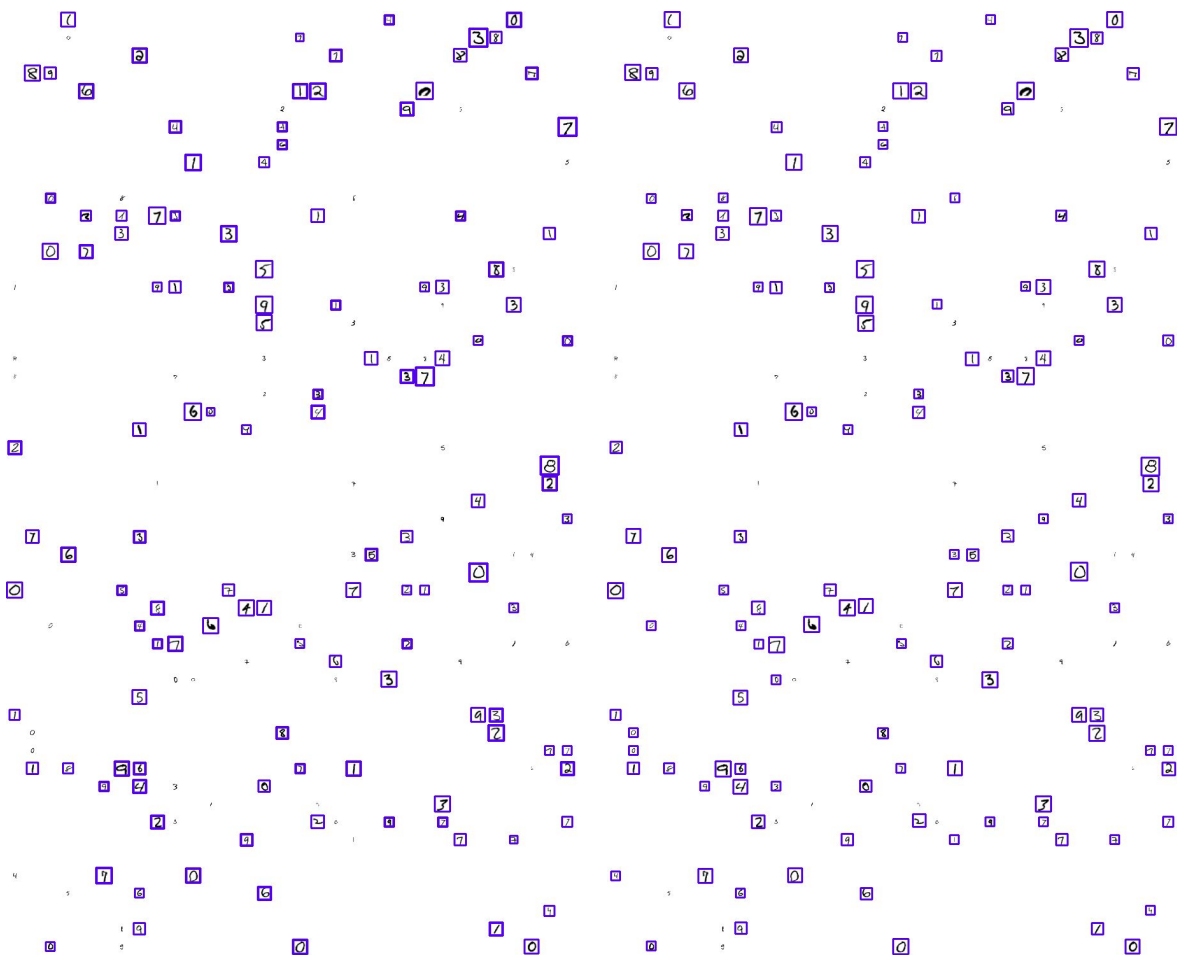
With stride 8 after 30 epochs:

Digit size	IOU	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
14	.3	97.12	94.50	0.012	99.91	100
7-14	.2	98.58	73.07	0.0087	98.61	100
7-14	.25	99.07	75.34	0.012	98.98	100

To capture digits on a larger range **7-28** pixel wide, we remove the 2 max pooling layers from our 56 pixel wide model, to build CNN_C32_C64_C64_Cd64_C128_D. At stride 3,

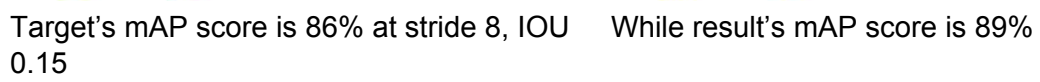
IOU	Epochs	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
.1	30	97.47	73.70	0.010	87.19	95.45
.2	30	99.08	92.84	0.0074	81.01	76.47
.15	50	98.71	88.02	0.0046	87.79	84.76
.1	50	97.97	79.19	0.0096	89.17	95.24

On 7-28 pixel digit range, the network sometimes learns a better reconstruction than the target, due to the hard IOU threshold decision in the target computation :



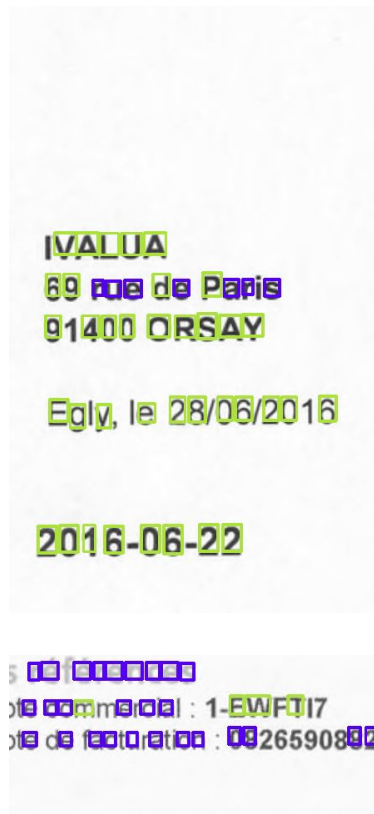
Target's mAP score is 76% at stride 8, IOU 0.2

While result's mAP score is 80%

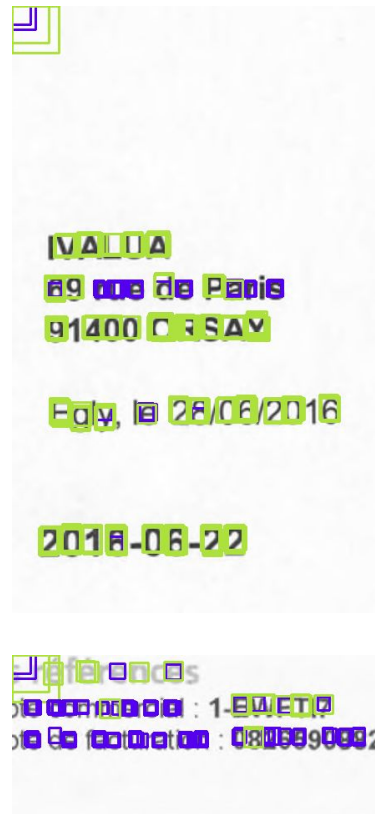


OCR dataset

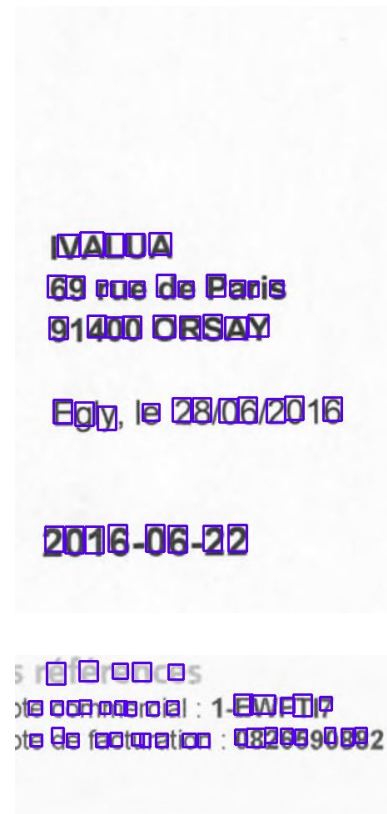
Target (training data)



Detection results (with on the top left corner each layer's receptive field minus stride margin)



Results filtered by NMS



état de paiement à la date prévue
tant TTC dû sera majoré de

état de paiement à la date prévue
tant TTC dû sera majoré de

état de paiement à la date prévue
tant TTC dû sera majoré de

Target

We experiment different settings to define positives on the grid and compute the *target average precision* obtained if we reconstruct the bounding boxes from the target instead of the prediction results. We also compute the final average precision obtained by the trained model on this setting.

We consider *positive* a position on the grid that has a sufficient IOU with the receptive field of the network.

Parameters	Obj acc	Class acc	Reg acc	Obj mAP	Target mAP
Stride 4+8, IOU 0.15	97.00/97.76	69.11/71.78	0.027/0.016	58.82	91.22
Stride 4+8, IOU 0.2	97.89/98.44	75.39/72.75	0.020/0.011	68.09	84.47
Stride 4+8, IOU 0.25	98.19	81.43	0.014	70.98	70.94
Stride 6+12, IOU 0.15	97.52/97.58	72.18/77.03	0.028/0.015	67.05	86.07
Stride 6+12, IOU 0.2	98.24/98.25	79.01/79.47	0.019/0.10	66.25	78.15
Stride 6+12, IOU0.25	98.60/98.90	80.17/78.93	0.015/0.0075	62.71	66.42
Stride 8+16, IOU 0.15	97.90/97.50	72.05/74.58	0.029/0.017	62.87	89.77
Stride 8+16, IOU 0.2	98.42/97.99	78.35/79.15	0.021/0.012	66.30	83.94
Stride 8+16, IOU 0.25	98.88/98.61	77.64/81.11	0.017/0.0077	60.26	69.35

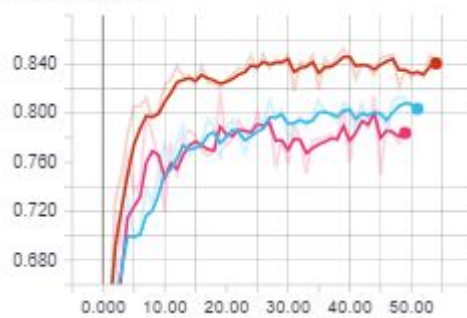
Stride 10+20, IOU 0.15	98.47/97.36	70.94/77.87	0.031/0.018	59.33	85.87
Stride 10+20, IOU 0.2	98.92/97.76	67.94/80.13	0.021/0.014	51.87	77.52
Stride 10+20, IOU 0.25	99.09/98.45	70.41/83.67	0.018/0.0097	44.59	61.57

IOU 0.2 (#ce7562)

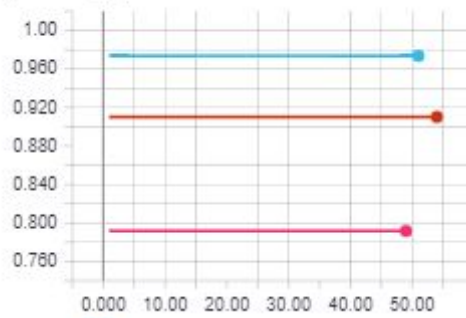
IOU 0.15 (#98cfe6)

IOU 0.25 (#de7ca2)

average_precision



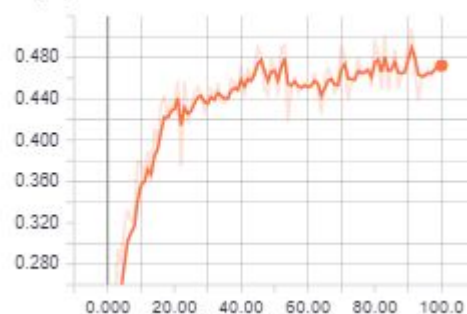
target_average_precision



Target average precision is better when the IOU is low, since the grid misses no ground truth boxes, nevertheless, the model possibly learns better at an higher IOU, which also leads to better classification results.

We also tried considering as *positive* any character that fall in the receptive field of the network. Target average precision is very close to 1 but final average precision remains below 0.48.

average_precision



Stride margin

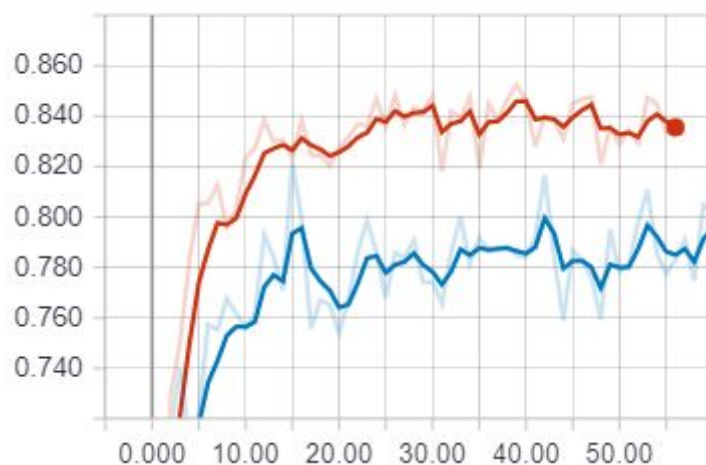
Since the network performs a strided analysis of the input image, we consider that characters should fall entirely into the receptive field of the network on one positive position. For that reason, we consider a stride margin, ie filter characters with a size lower than the receptive field dimension minus the stride.

Deactivating this setting, some characters are not being seen completely by the network anymore and the prediction in position and scale should be harder to perform. Object mAP score becomes 78.5%.

with stride margin (#d0644c)

without stride margin (#6aa6c8)

average_precision



Positive weight and **loss weights**

pos weight 1000 (#4ca89a)

pos weight 100 (#b9bfbf)

pos weight (#df9c83)

validation_average_precision



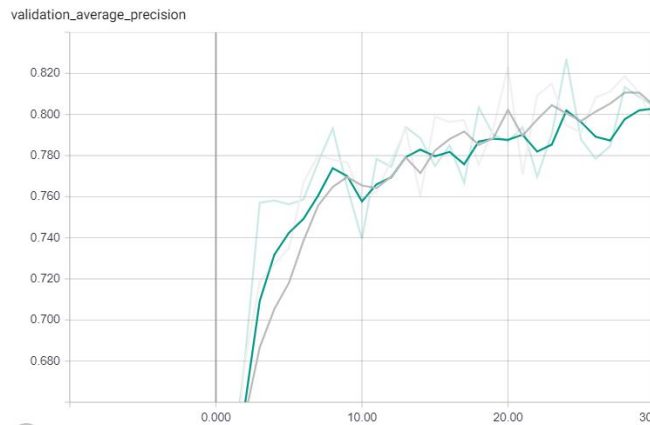
Best results are obtained with pos_weights=100.

Kernel width

To see the influence of the width of the layers, we try to double the number of filters for each layer, leading to the CNN_C64_C128_M2_C128_C128_M2_C256_D_2 model. A wider architecture does not seem to help much.

CNN_C32_C64_M2_C64_C64_M2_C128_D_2 (#f0f2f2)

CNN_C64_C128_M2_C128_C128_M2_C256_D_2 (#b0d7d3)



Parameters	Obj acc	Class acc	Reg acc	Obj mAP	Targ et mAP
Stride 6+12, IOU 0.2	98.45/98.66	83.27/85.42	0.018/0.0097	70.11	78.15

Full document, at low resolution

Since document images are wide, we used a generator for the preprocessing, and do not have ground truth for mAP computation. Results are evaluated qualitatively. Best results with convolution of 28 pixel wide reception fields are obtained with images of max size 1000 pixels, since most characters fall at the right to be recognized.

At 1000 pixels:

6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
10	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4
10	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4
10	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4

[illegible]



Feature

Topic: **Common House Sparrows**

Recture
Any
Value
Rue
Srs.
Grance
Nm

00:03

Def:

Verordening:

Notefollowing:

REDACTED

17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 1047 1048

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1

[illegible]

11b
 11c
 11d
 11e
 11f
 11g
 11h
 11i
 11j
 11k
 11l
 11m
 11n
 11o
 11p
 11q
 11r
 11s
 11t
 11u
 11v
 11w
 11x
 11y
 11z
 11aa
 11ab
 11ac
 11ad
 11ae
 11af
 11ag
 11ah
 11ai
 11aj
 11ak
 11al
 11am
 11an
 11ao
 11ap
 11aq
 11ar
 11as
 11at
 11au
 11av
 11aw
 11ax
 11ay
 11az
 11ba
 11bb
 11bc
 11bd
 11be
 11bf
 11bg
 11bh
 11bi
 11bj
 11bk
 11bl
 11bm
 11bn
 11bo
 11bp
 11bq
 11br
 11bs
 11bt
 11bu
 11bv
 11bw
 11bx
 11by
 11bz
 11ca
 11cb
 11cc
 11cd
 11ce
 11cf
 11cg
 11ch
 11ci
 11cj
 11ck
 11cl
 11cm
 11cn
 11co
 11cp
 11cq
 11cr
 11cs
 11ct
 11cu
 11cv
 11cw
 11cx
 11cy
 11cz
 11da
 11db
 11dc
 11dd
 11de
 11df
 11dg
 11dh
 11di
 11dj
 11dk
 11dl
 11dm
 11dn
 11do
 11dp
 11dq
 11dr
 11ds
 11dt
 11du
 11dv
 11dw
 11dx
 11dy
 11dz
 11ea
 11eb
 11ec
 11ed
 11ee
 11ef
 11eg
 11eh
 11ei
 11ej
 11ek
 11el
 11em
 11en
 11eo
 11ep
 11eq
 11er
 11es
 11et
 11eu
 11ev
 11ew
 11ex
 11ey
 11ez
 11fa
 11fb
 11fc
 11fd
 11fe
 11ff
 11fg
 11fh
 11fi
 11fj
 11fk
 11fl
 11fm
 11fn
 11fo
 11fp
 11fq
 11fr
 11fs
 11ft
 11fu
 11fv
 11fw
 11fx
 11fy
 11fz
 11ga
 11gb
 11gc
 11gd
 11ge
 11gf
 11gg
 11gh
 11gi
 11gj
 11gk
 11gl
 11gm
 11gn
 11go
 11gp
 11gq
 11gr
 11gs
 11gt
 11gu
 11gv
 11gw
 11gx
 11gy
 11gz
 11ha
 11hb
 11hc
 11hd
 11he
 11hf
 11hg
 11hh
 11hi
 11hj
 11hk
 11hl
 11hm
 11hn
 11ho
 11hp
 11hq
 11hr
 11hs
 11ht
 11hu
 11hv
 11hw
 11hx
 11hy
 11hz
 11ia
 11ib
 11ic
 11id
 11ie
 11if
 11ig
 11ih
 11ii
 11ij
 11ik
 11il
 11im
 11in
 11io
 11ip
 11iq
 11ir
 11is
 11it
 11iu
 11iv
 11iw
 11ix
 11iy
 11iz
 11ja
 11jb
 11jc
 11jd
 11je
 11jf
 11jg
 11jh
 11ji
 11jj
 11jk
 11jl
 11jm
 11jn
 11jo
 11jp
 11jq
 11jr
 11js
 11jt
 11ju
 11jv
 11jw
 11jx
 11jy
 11jz
 11ka
 11kb
 11kc
 11kd
 11ke
 11kf
 11kg
 11kh
 11ki
 11kj
 11kk
 11kl
 11km
 11kn
 11ko
 11kp
 11kq
 11kr
 11ks
 11kt
 11ku
 11kv
 11kw
 11kx
 11ky
 11kz
 11la
 11lb
 11lc
 11ld
 11le
 11lf
 11lg
 11lh
 11li
 11lj
 11lk
 11ll
 11lm
 11ln
 11lo
 11lp
 11lq
 11lr
 11ls
 11lt
 11lu
 11lv
 11lw
 11lx
 11ly
 11lz
 11ma
 11mb
 11mc
 11md
 11me
 11mf
 11mg
 11mh
 11mi
 11mj
 11mk
 11ml
 11mm
 11mn
 11mo
 11mp
 11mq
 11mr
 11ms
 11mt
 11mu
 11mv
 11mw
 11mx
 11my
 11mz
 11na
 11nb
 11nc
 11nd
 11ne
 11nf
 11ng
 11nh
 11ni
 11nj
 11nk
 11nl
 11nm
 11nn
 11no
 11np
 11nq
 11nr
 11ns
 11nt
 11nu
 11nv
 11nw
 11nx
 11ny
 11nz
 11oa
 11ob
 11oc
 11od
 11oe
 11of
 11og
 11oh
 11oi
 11oj
 11ok
 11ol
 11om
 11on
 11oo
 11op
 11oq
 11or
 11os
 11ot
 11ou
 11ov
 11ow
 11ox
 11oy
 11oz
 11pa
 11pb
 11pc
 11pd
 11pe
 11pf
 11pg
 11ph
 11pi
 11pj
 11pk
 11pl
 11pm
 11pn
 11po
 11pp
 11pq
 11pr
 11ps
 11pt
 11pu
 11pv
 11pw
 11px
 11py
 11pz
 11qa
 11qb
 11qc
 11qd
 11qe
 11qf
 11qg
 11qh
 11qi
 11qj
 11qk
 11ql
 11qm
 11qn
 11qo
 11qp
 11qq
 11qr
 11qs
 11qt
 11qu
 11qv
 11qw
 11qx
 11qy
 11qz
 11ra
 11rb
 11rc
 11rd
 11re
 11rf
 11rg
 11rh
 11ri
 11rj
 11rk
 11rl
 11rm
 11rn
 11ro
 11rp
 11rq
 11rr
 11rs
 11rt
 11ru
 11rv
 11rw
 11rx
 11ry
 11rz
 11sa
 11sb
 11sc
 11sd
 11se
 11sf
 11sg
 11sh
 11si
 11sj
 11sk
 11sl
 11sm
 11sn
 11so
 1

Fournisseur: JCF

N° de pièce :

Write: 600

Signature: ENAR

Règles :

Tv 6010-100 6010-100 }
6010-100 }

L'Etat - Autorité de régulation au sein du commerce en ligne :

La signature a été vérifiée et l'acte a été enregistré sans cause enregistreuse.

Definición: consiste en la versión de un texto que es original.

Les données soumises au mécanisme de règlement des différends de l'OMC ne sont pas divulguées à l'extérieur de l'OMC, et les données sont traitées de manière confidentielle. Les données sont traitées de manière confidentielle et ne sont pas divulguées à l'extérieur de l'OMC.

While at size 1500 pixels, it misses the address :

Total	4400
-------	------

Conclusion

Object detection architectures sound promising for high quality character reading and the development of document features through end-2-end training. Classical rule-based algorithms can reuse these features to solve higher level tasks, such as line extraction.

This opens the way for best model architectures search by the community. Future work includes reusing improvements from object detection models, such as multiple anchors, two-stage detection, focal loss, optimization tuning for larger images, batches or other input resolution.

Bibliography

- [1] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. arXiv preprint arXiv:1601.07140, 2016.
- [2] Smith, R., Gu, C., Lee, D.S., Hu, H., Unnikrishnan, R., Ibarz, J., Arnoud, S., Lin, S.: End-to-end interpretation of the french street name signs dataset. In: European Conference on Computer Vision. pp. 411–426. Springer (2016)
- [3] Zhang, Y., Gueguen, L., Zharkov, I., Zhang, P., Seifert, K., Kadlec, B.: Uber-text: A large-scale dataset for optical character recognition from street-level imagery. In: SUNw: Scene Understanding Workshop - CVPR 2017. Hawaii, U.S.A. (2017)
- [4] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang. Reading scene text in deep convolutional sequences. arXiv preprint arXiv:1506.04395, 2015.
- [5] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. International Journal of Computer Vision, 116(1):1–20, 2016.
- [6] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. In Pattern Recognition (ICPR), 2012 21st International Conference on, pages 3304–3308. IEEE, 2012.

[7] Guided Attention for Large Scale Scene Text Verification Dafang He^{1*} , Yeqing Li^{2*} , Alexander Gorban² , Derrall Heath² , Julian Ibarz² , Qian Yu² , Daniel Kifer¹ , C. Lee Giles¹

[8] Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks

Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, Vinay Shet

[9] Creating a Modern OCR Pipeline Using Computer Vision and Deep Learning

<https://blogs.dropbox.com/tech/2017/04/creating-a-modern-ocr-pipeline-using-computer-vision-and-deep-learning/>

[10] TextBoxes: A Fast Text Detector with a Single Deep Neural Network Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, Wenyu Liu

[11] Spatial Transformer Networks

Max Jaderberg, Karen Simonyan, Andrew Zisserman, Koray Kavukcuoglu

[12] STN-OCR: A single Neural Network for Text Detection and Text Recognition

Christian Bartz, Haojin Yang, Christoph Meinel

[13] A. Mishra, K. Alahari, and C. Jawahar. Scene text recognition using higher order language priors. In BMVC '12. BMVA, 2012.

[14] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven. Photoocr: Reading text in uncontrolled conditions. In Proceedings of the IEEE International Conference on Computer Vision, pages 785–792, 2013.

[15] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, Computer Vision - ECCV 2014, number 8692 in Lecture Notes in Computer Science, pages 512–528. Springer International Publishing, 2014.

[16] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. arXiv preprint arXiv:1406.2227, 2014.

[17] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions. In ICDAR '03, pages 682–687. IEEE, 2003.

[18] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, et al. Icdar 2015 competition on robust reading. In Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, pages 1156– 1160. IEEE, 2015.

[19] A. Shahab, F. Shafait, A. Dengel, "ICDAR 2011 Robust Reading Competition - Challenge 2: Reading Text in Scene Images", In Proc. 11th International Conference of Document Analysis and Recognition, 2011, IEEE CPS, pp. 1491-1496.

[20] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. Gomez, S. Robles, J. Mas, D. Fernandez, J. Almazan, L.P. de las Heras, "ICDAR 2013 Robust Reading Competition", In Proc. 12th International Conference of Document Analysis and Recognition, 2013, IEEE CPS, pp. 1115-1124.

[21] D. Karatzas, S. Robles Mestre, J. Mas, F. Nourbakhsh, P. Pratim Roy, "ICDAR 2011 Robust Reading Competition - Challenge 1: Reading Text in Born-Digital Images (Web and Email)", In Proc. 11th International Conference of Document Analysis and Recognition, 2011, IEEE CPS, pp. 1485-1490.

[22] Detecting Oriented Text in Natural Images by Linking Segments
Baoguang Shi, Xiang Bai, Serge Belongie

[23] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016

[24] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai. Robust scene text recognition with automatic rectification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4168–4176, 2016.

[25] SVHN yolo-v2 digit detector
<https://github.com/penny4860/Yolo-digit-detector>

[26] Azka Gilani, Shah Rukh Qasim, Imran Malik† and Faisal Shafait, National University of Sciences and Technology (NUST), Islamabad, Pakistan. Table Detection using Deep Learning

[27] Analysis of Convolutional Neural Networks for Document Image Classification
Chris Tensmeyer, Tony Martinez

[28] The Fourth Annual Test of OCR Accuracy Stephen V. Rice, Frank R. Jenkins, and Thomas A. Nartker

[29] Optical Character Recognition using Simple Convolutional Neural Network
Tan D. Lam, Huy V. N. Huynh, Trieu H. Nguyen, Tiep V. Nguyen, Triet M. Tran

[30] RCNN
R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[31] SSD

C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. DSSD: Deconvolutional single shot detector. arXiv:1701.06659, 2016.

[32] YOLO

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi.

You only look once: Unified, real-time object detection. In CVPR, 2016.

[33] OVERFEAT

H. Rowley, S. Baluja, and T. Kanade. Human face detection in visual scenes. Technical Report CMU-CS-95-158R, Carnegie Mellon University, 1995.

[34] Rich feature hierarchies for accurate object detection and semantic segmentation

Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

[35] Fast RCNN

Ross Girshick. Fast R-CNN

[36] Faster RCNN

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

[37] Mask R-CNN

Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. Mask R-CNN

[38] YOLOv3: An Incremental Improvement

Joseph Redmon, Ali Farhadi

[39] <https://github.com/penny4860/Yolo-digit-detector>