# Space Displacement Localization Neural Networks to locate origin points of handwritten text lines in historical documents

Bastien Moysset
A2iA SAS, Paris, France
INSA-Lyon, LIRIS, UMR5205, F-69621
bm@a2ia.com

Pierre Adam
A2iA SAS, Paris, France
pad@a2ia.com

Christian Wolf
Université de Lyon, CNRS, France
INSA-Lyon, LIRIS, UMR5205, F-69621
christian.wolf@liris.cnrs.fr

Jérôme Louradour
A2iA SAS, Paris, France
jl@a2ia.com

## ABSTRACT

We describe a new method for detecting and localizing multiple objects in an image using context aware deep neural networks. Common architectures either proceed locally per pixel-wise sliding-windows, or globally by predicting object localizations for a full image. We improve on this by training a semi-local model to detect and localize objects inside a large image region, which covers an object or a part of it. Context knowledge is integrated, combining multiple predictions for different regions through a spatial context layer modeled as an LSTM network.

The proposed method is applied to a complex problem in historical document image analysis, where we show that is capable of robustly detecting text lines in the images from the ANDAR-TL competition. Experiments indicate that the model can cope with difficult situations and reach the state of the art in Vision such as other deep models.

## 1. INTRODUCTION

In this work we are concerned with the detection and precise localization of multiple objects in images, in particular in the context of document image analysis. The detection and localization of text lines in document images is a challenging task requiring the integration of information on a local scale, such as signal and geometry, as well as information of contextual nature on a wider scale. The former models the appearance of text lines, whereas the latter allows (i) to distinguish them from objects with similar appearance, such as words, and (i) to integrate cases with high inter-class variance in appearance, such as text lines structured into multiple columns.

The problem of detecting and localization has been dealt with several types of methods in computer vision. Local methods like sliding windows need to calculate a complex prediction function for each pixel of the input image, which can be expensive to compute and makes it difficult to take into account context account.

Global methods directly predict object bounding boxes given a full input image. In the case of methods based on machine learning, they require the regressor to be highly invariant to translations, i.e. the system has to be able to find the object at different positions in the image. Methods based on deep learning are the state of the art in object detection and localization, especially Convolutional Neural Networks (CNN) [17]. However, invariances are not the strong points of these models, in particular invariance to translations. Instead, this property is enforced through massive amounts of training data, often with the help of artificial data augmentations, i.e. creating new data samples from existing training data through shifting, cropping and rotating.

In this paper, we propose a new method for text line detection and localization, which proceeds by a direct regression of line positions with a deep neural network. Compared to the state of the art, our model is a compromise between pixel wise classification and a global model, effectively combining the advantages of both approaches.

The contributions are twofold:

- A semi-local detection and localization model, which combines the advantages of sliding windows and global detection methods. As for global methods, object bounding boxes are directly predicted through regression. However, predictions are fused from local regions and fused globally. The main advantage is decreased variation in network input, because the filters are shared between the different positions in the image.
- A context layer in the deep network, which models dependencies between different local regions in the input image. The layer makes it possible to detect objects which are only partially visible in a given single image region.
- An application of the proposed model to document image analysis and text line detection in historical documents.

## 2. RELATED WORK

We will here briefly review existing literature on text line detection in document analysis, but also more generally on object detection and localization in computer vision.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

*Text Line Detection.*
Early work is purely based on image processing. They proceed through projection, where lines are separated by finding minima in the horizontal projection histograms [22] or they employ morphological operations or blurring [15, 4] to fuse letters belonging to the same line. The methods that extract the text components and group them by finding alignment with the Hough transform [12] or sweep-line [16] also belong to this category. More recently, Nicolaou et al. [14] find text interline whitespace by following low intensity areas. Shi et al. [19] use steerable filters to filter the image, where text line objects are remaining components after binarization. Moysset et al. regroup connected components obtained after binarization to form lines [13].

Most of these state of the art techniques employ image-based heuristics. This means that heavy engineering efforts have to be done to design them and, more importantly, to adapt them to new tasks and to new datasets. They are also arguable less powerful on datasets with highly diversified document content.

Few techniques based on machine learning, and especially using deep neural networks, have been applied on document segmentation. However, outside the document analysis community, these models have received tremendous interest in the last few years, especially for the detection and localization of objects in natural images, for instance of the ImageNet dataset [17]. We will review these techniques in the following paragraphs.

*Sliding windows.*
A popular yet classical technique consists in sliding a window over the input image and classifying the contents of each window. The raw output then requires post-processing to find bounding boxes, for instance non-maxima suppression.

Sermanet et al. use a first classifier applied to sliding windows to extract candidate objects which are then fed into a Convolutional Neural Network (CNN) for validation [18]. Similarly, Garcia et al. apply this principle to face detection [8]. Uijlings et al. produce region candidates from an over-segmentation of the input image, which are then classified [20]. R-CNNs, introduced by Girshick et al., proceed in a similar way, classifying candidate bounding boxes from features extracted with CNNs [9].

The specific task of text line segmentation is challenging for these methods, as the appearance of a part of a line is often very similar to the appearance of a full line. Moreover, the shape of the line is highly variable whereas most of these approaches use windows of fixed size.

Other methods proceed by labelling pixels or super-pixels of an image as belonging or not to the special class of targeted object. Delakis et al., for instance, use sliding windows and a CNN to classify parts of natural images as text or non-text [3]. Whereas the group described above classifies a pixel as being the center pixel of a targeted object, these methods classify a pixel as being part of an object, be it center pixel or not. As a consequence, the raw output is a segmented image which then requires non-trivial post-processing to find bounding boxes. In particular, overlapping and touching objects are challenging configurations. This is the case in document text segmentation, especially for handwritten historical texts, where successive lines are close, if not overlapping one each other, making a correct and robust grouping challenging to develop.

*Deformable parts models.*
Most methods based on sliding windows use machine learning to classify each window. Deformable parts models, which technically belong to the group of sliding windows techniques, integrate deformable terms into the prediction model making it particularly invariant to complex transformations and deformations [7, 6]. An object is modeled as a set of parts, each part being defined as a filter, an anchor position w.r.t.t. object's center, and a deformation cost. At test time, the most probable configuration of part positions is searched by solving a combinatorial problem with dynamic programming.

*Straightforward regression of positions.*
Sliding windows across a possibly large input window and performing a complex prediction at each position is computationally complex. A different approach proceeds by feeding the full input image into a single predictor, which learns to predict the set of output bounding boxes for each image. Ehran et al. recently proposed this alternative solution [5] using a CNN as a regressor. The deep network comprises multiple vectorial outputs, each one belonging to a different bounding box. During training, the set of outputs needs to be matched to the set of ground truth bounding boxes, which is done solving a combinatorial problem with the Hungarian algorithm.

Our method is inspired by [5]. However, instead of proceeding in an entirely global way, which is sub-optimal in situations involving a high variation in input image sizes, the model is semi-local.

# 3. SPACE DISPLACEMENT LOCALIZATION (SDL) NEURAL NETWORKS FOR LOCATING POINTS OF INTEREST

In the lines of [5], our network directly predicts the position $(x, y)$ of one or several objects in an image. We also restrict ourselves to the setting where a particular point of each object has to be located. This is the case in the ANDAR-TL competition, where the points of interest of the handwritten lines are the lower left corners of the bounding boxes.

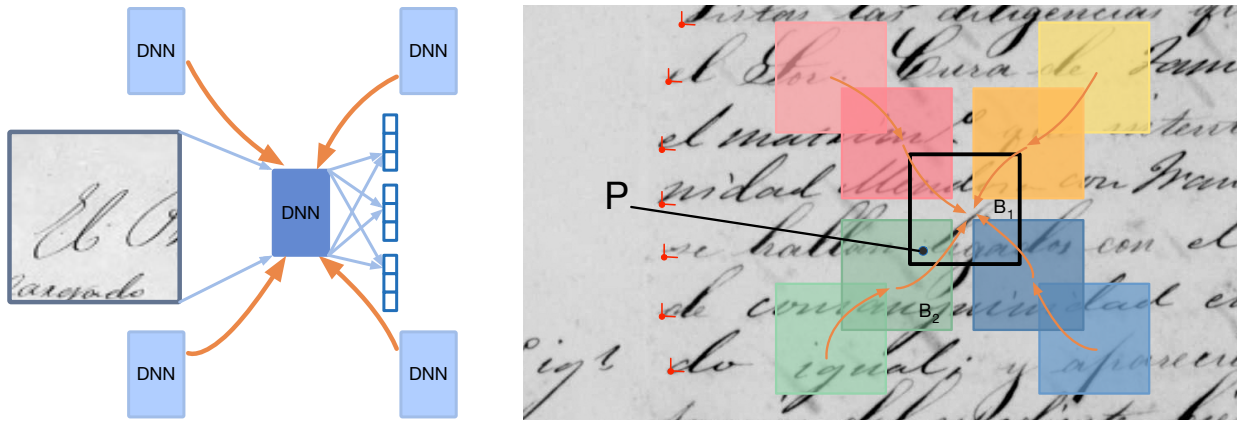
In [5], object locations are predicted from the full image, which requires a high number of parameters on the fully connected layers to train with a limited number of samples. To deal with this issue, we created the *Space Displacement Localization* (SDL) layer. This layer predicts object positions from a part ("block") of the image, effectively sharing parameters between different parts of the image.

The full input image is fed into a deep neural network consisting of several convolutional layers, recurrent layers and fully connected layers (see section 5 for the precise architecture). These different layers are shared over a grid of $W \times H$ blocks, each block corresponding to a window in the input image. The topmost hidden representation consists in $W \times H$ vectors $\mathbf{h}(i, j), i = 1 \dots W$ and $j = 1 \dots H$. On top of this comes a special layer, that we call Space Displacement Localization (SDL).

In this section, we first present the new SDL output layer that we propose. It was designed especially to perform localization in relation to the position of each window in the input image. Then we stress the importance of taking into account the context: to achieve robust localization, the hidden representation $\mathbf{h}(i, j)$ for each block $(i, j)$ should encompass information coming from the other blocks around $(i-1, j), (i, j-1), \dots$ We finally show that [5] on the one hand, and sliding-window approaches on the other hand, are two special cases of our proposed model, that is a compromise between these two kinds of approaches.

## 3.1 SDL output layer

Each topmost hidden representation $\mathbf{h}(i, j)$ is used to predict at most $N$ points of interest. This is achieved by producing $N$ triplets $(\rho, x, y)$ for each block $(i, j)$: a confidence score $\rho$ along




(a) (b)

Figure 1: Decomposition of the calculation into blocks: (a) a single block receives a patch of the input image and outputs $N$ object locations (3 shown). Recurrent connections (shown in orange) connect it to 4 neighboring blocks, which allows to model context. All DNN models share parameters. (b) example of patch layout and recurrence chain. Recurrent connections are shown only for the center patch. The input patch for block $B_1$ contains a point $P$ which resembles a textline origin point. Recurrent connections from block $B_2$ provide context to disambiguate the situation.

with the position $x$ on the horizontal axes and the position $y$ on the vertical axes. For a given image, the model thus predicts at most $(N \times W \times H)$ object positions. If less objects are present in an image, applying a threshold to the confidence value $\rho$ allows to determine unused network outputs. $N$ is a hyper-parameter to be tuned depending on the maximum number of points to retrieve per block.

The triplet of outputs are computed as follows:

$$\rho_{i,j,k} = \sigma\left(\mathbf{w}_k{}^\top \cdot \mathbf{h}(i,j) + b_k\right) \quad (1)$$

$$x_{i,j,k} = \sigma\left(\mathbf{u}_k{}^\top \cdot \mathbf{h}(i,j) + c_k\right) \times \Delta_\mathrm{x} + (i-1) \times \delta_\mathrm{x} \quad (2)$$

$$y_{i,j,k} = \sigma\left(\mathbf{v}_k{}^\top \cdot \mathbf{h}(i,j) + d_k\right) \times \Delta_\mathrm{y} + (j-1) \times \delta_\mathrm{y} \quad (3)$$

with $k = 1 \ldots N$, $i = 1 \ldots W$ and $j = 1 \ldots H$. $\sigma(\cdot)$ is the logit sigmoid function. The weights $(\mathbf{w}_k, \mathbf{u}_k, \mathbf{v}_k)$ and the biases $(b_k, c_k, d_k)$ are free parameters to be learned.

The values of $(\Delta_\mathrm{x}, \Delta_\mathrm{y})$, *resp.* $(\delta_\mathrm{x}, \delta_\mathrm{y})$, are fixed so as to be the size of each input block, *resp.* the step between two consecutive blocks in the input image. They depend on the neural network architecture, namely the sizes of the convolutional filters used to produce the topmost hidden representation $\mathbf{h}(i,j)$. They obey $0 < \delta_\mathrm{x} \leq \Delta_\mathrm{x}$, and blocks are usually overlapping *i.e.* $\delta_\mathrm{x} < \Delta_\mathrm{x}$.

The scaling factors $(\Delta_\mathrm{x}, \Delta_\mathrm{y})$ and $(\delta_\mathrm{x}, \delta_\mathrm{y})$ are used to map the relative coordinates w.r.t. each block ($\sigma$ in $[0,1]$) into absolute coordinates in the image : the vertical, *resp.* horizontal, absolute coordinate naturally lie in $[0, \Delta_\mathrm{x} + (W-1)\delta_\mathrm{x}]$, *resp.* $[0, \Delta_\mathrm{y} + (H-1)\delta_\mathrm{y}]$. The motivation to introduce parameter sharing across positions $(i,j)$ along with these factors is to make the model (roughly) invariant to spatial translations of the input image, with modulo $(\delta_\mathrm{x}, \delta_\mathrm{y})$.

Section 4 will describe how to train a network with a SDL output layer. Inference is straightforward: predictions are calculated with forward passes on the neural network, and outputs with a confidence score $\rho$ above a threshold fixed to 0.5 are considered as detected.

## 3.2 Including context in the network

Producing outputs per block $(i,j)$ allows the SDL model to share parameters, which avoid relearning specific prediction model for each location in the image. However, taking decisions at a local level may lead to problems when an image patch locally indicates the presence of an object, which is not confirmed by a larger context in the image. For instance, the beginning of a word can be easily mistaken for the beginning of a line, if the left side is not embraced by the corresponding block. Hence the importance of taking into account the surrounding context to build each block representation $\mathbf{h}(i,j)$ in equations (1), (2), (3). It is especially critical when the full objects (that define the points of interest to retrieve) are often larger than blocks of interest.

Taking into account the context can be done using convolutions or recurrences. In our work, we used four-directional layers of Long-Short Term Memory (LSTM) units [10]. Recurrent Neural Networks (RNN) with LSTM units are now state-of-the-art in many tasks involving sequential signals[1].

Figure 1 illustrates the concept of incorporating surrounding context into the representation of each block. The prediction model for each block takes a decision based on the visual input of the corresponding image patch and from recurrent connections received from the neighboring blocks. All DNN representations at the different positions share the same modeling parameters. In particular, point $P$ in the image resembles a textline origin point if only the input patch for block $B_1$ is taken into account, since the preceding word in the textline is no visible on this local scope. Recurrent connections from block $B_2$ provide context to disambiguate the situation.

## 3.3 Special Cases

The SDL neural networks we propose are a compromise between global regressors and purely local models based on sliding windows. Indeed, with two particular (extreme) parameterization, they amount to previously proposed models:

- If $\Delta_\mathrm{x}$ and $\Delta_\mathrm{y}$ are set to the image width and height, then $W = H = 1$ and the model amounts to the method of Ehran et al. [5]. It is fully global: all positions are determined from the whole image, namely from a concatenation of the list of hidden values $\mathbf{h}(i,j)$. In this case, the difference with [5] lies

[1] http://people.idsia.ch/~juergen/rnn.html

in the presence of recurrent connections to model the context at intermediate levels. Another minor difference is the sigmoid activation function on position outputs (1), whose interest is to confine the network responses to valid ranges.

- If we configure a network so that $\delta_x = \delta_y = \Delta_x = \Delta_y = 1$ then a prediction is created at each pixel of the input image. The confidence scores $\rho_k$ can be used to classify each pixel, similarly to a high-resolution sliding-window approach. If $\Delta_x = \Delta_y = 1$, the predicted positions $x_k, y_k$ turn out to be obsolete, given that it is impossible to reach sub-pixel precision on the location. Including context in the hidden representation that feeds the SDL layer is critical for such local approaches, as discussed previously in sub-section 3.2.

A notable advantage of our approach over [5] is that it handles input images with variable sizes, modulo $(\delta_x, \delta_y)$. In [5], images must be scaled to a fixed size imposed by the network architecture, which is awkward when the format of pages vary a lot (*e.g* mix of A4 and A5 page formats), because lines of text will be observed with variable resolutions. With the SDL output layer, as with sliding-window approaches in general, it is possible to naturally adapt the maximum number $N \times W \times H$ of detectable points depending on the size of the input image ($W$ and $H$ can vary without modification to the inference process).

# 4. TRAINING

Training the network involves learning parameters $\mathbf{w}, \mathbf{u}, \mathbf{v}, b, c, d$ of the output regression layer, as well as the weights of all hidden layers including the parameters of the recurrent connections. These parameters are shared over all blocks, i.e. over all patches of the input image, as are the different parameters. Training is done in a supervised way from labeled ground truth images with gradient back propagation, which requires associating ground truth object locations to the different network outputs. There are different network output triplets for different blocks. As opposed to the network parameters, the outputs are not shared over blocks. However, as in [5], we assign ground truth object positions to the different $N \times W \times H$ network output triplets *globally* for the whole image. Once the assignment is done, parameter updates are performed classically using stochastic gradient descent.

## 4.1 Matching network outputs to ground truth

In the lines of [5], matching network output triplets to ground truth points is done minimizing a global energy function designed with two goals in mind: i) assign similar locations, and ii) assign network outputs with high confidence. The matching cost is defined over pairs of points (ground truth — network output), the total cost is calculated as the sum over matched pairs:

$$\begin{aligned} E(X,\theta) &= \alpha \sum_{ij} X_{ij} ||o_i(\theta) - g_j||^p + \sum_i X_{ij} \log\left(\frac{c_i(\theta)}{1-c_i(\theta)}\right) \\ &\text{s.t.} \quad \forall j \sum_i X_{ij} \leq 1 \quad \wedge \\ &\qquad \forall i \sum_j X_{ij} \leq 1 \end{aligned} \tag{4}$$

where $o_i(\theta)$ is the position vector of network output triplet $i$ depending on network parameters $\theta$, $c_i(\theta)$ is the confidence value of the same triplet, and $g_j$ is a ground truth point. $||.||^p$ denotes the Minkowski norm with parameter $p$. The binary matrix $X=\{X_{ij}\}$ denotes a solution of the assignment problem. In particular, $X_{ij}=1$ indicates that output triplet $i$ is matched to ground truth box $j$. Minimizing (4) with respect to the assignment matrix $X$ and constant network parameters $\theta$ is a constraint satisfaction problem which can be done efficiently with the Hungarian algorithm.

The confidence cost value depends only on the confidence $\rho_k$. Its value is $\log(1-\rho_k) - \log(\rho_k)$. The position cost is the squared distance between the ground truth point and the hypothesis point. Two distances will be used in this work, namely the Euclidian $L_2$ distance and the $L_\infty$ distance.

A coefficient $\alpha$ is used to weight these two costs, its aim is to force all the output triplets to be matched with close ground truth points while taking into account the network confidence of a point being there.

## 4.2 Loss function

The loss function used for parameter updates is the same as equation 4. However, as usual, stochastic gradient descent minimizes the loss with respect to the network parameters $\theta$. The assignment matrix $X$ is kept constant during this step. the weight $\alpha$ is not necessarily the same as during matching, as its role is slightly different. During network weight updates, $\alpha$ needs to ensure that the training is balanced between learning geometric positions $o_i(\theta)$ and confidences $c_i(\theta)$

The derivative of this cost function will be used to train the network by gradient backpropagation. Note that for the unmatched output triplets, only the gradient related to the confidence score $s$ will be non-null.

## 4.3 Model selection

We perform early-stopping and optimize model architectures over the validation set. In particular, the F-Measure (harmonic mean of precision and recall) as optimized. A given text line point was considered as correct if it is within a 6-pixel range of a ground truth point and if no other point has already been assigned to this ground truth point.

# 5. NETWORK ARCHITECTURE

The grayscale document images are downscaled to a fix size corresponding to the convolutional neural network input size. The ratio between the width and height is kept in order to avoid to change the shape of the letters. An extra black margin is added if needed to pad the image to the correct size. No extra pre-processing is done. The pixels of this rescaled image are used as input features for the first layer of our neural network.

The networks are constituted of three successive convolutional layers on top of which is applied the Space Displacement Localization layer. Max-pooling layers are placed after the two first convolutions and three four-direction Long Short Term Memory (LSTM) layers [10] can be added after the convolutional layers. Non-linearities are added after each convolutional, max pooling or LSTM layer. Dropout can be after the last convolutional layer. The detail of the architecture, including the number of hidden units and the filter sizes, can be found in Table 1.

# 6. EXPERIMENTS

### *6.0.1 Dataset: the ANDAR-TL 2015 competition*

The systems have been tuned for the ANDAR-TL Text Line Detection competition task[2]. The ANDAR-TL image set is an historical document database of 726 images. 635 pages have been randomly chosen for training and the remaining 91 images have been used for the validation. Some example of images are shown in figure 2.

[2]http://collections.ancestry.com/ANDAR-TL-2015

| Layer: | Filter size | Number of hidden units | Number of free parameters |
|---|---|---|---|
| (1) Convolution | 12x12 | 12 | 1740 |
| (2) MaxPooling | 3x3 | | |
| (3) LSTM | | 12 | 8880 |
| (4) Convolution | 5x5 | 25 | 7525 |
| (5) MaxPooling | 3x3 | | |
| (6) LSTM | | 25 | 38000 |
| (7) Convolution | 5x5 | 50 | 31300 |
| (8) LSTM | | 100 | 502000 |
| (9) SDL | | 60 | 6060 |

Table 1: The architecture of our networks with the filter sizes, the number of hidden units and the number of free parameters.

Figure 3: Place of the points to detect, shown in red.

The aim of the competition is to detect the origin point of the text lines which is the point at the bottom left of the first character of the line. These points are shown in red on figures 2 and 3.

As in most historical documents, there is the presence of heavy noise in the images. Bleed through, heavy slant or annotation in the margins can be found in these documents. Documents are from different writers, from different books with different layouts and the size of the page can vary.

## 6.1 Evaluation Metrics

The metric of the ANDAR-TL competition has been used in this paper. Tolerance regions are constructed around each ground truth point, an hypothesis point is considered in this tolerance region if the following condition is met:

$$H_y - R_y < \gamma(R_{y+1} - R_y) \wedge |H_x - R_x| < \gamma(R_{y+1} - R_y), \textbf{if } H_y > R_y$$
$$R_y - H_y < \gamma(R_y - R_{y-1}) \wedge |H_x - R_x| < \gamma(R_y - R_{y-1}), \textbf{if } H_y < R_y$$

Where $H_x$,$H_y$ and $R_x$,$R_y$ are respectively the coordinates of the ground truth and hypothesis point and where $R_{y+1}$ and $R_{y+1}$ correspond respectively to the ordinate value of the ground truth points just below and above the current ground truth point. $\gamma$ is a coefficient set to 0.25.

A dynamic alignment is performed to maximise the number of matched points. The error is then defined as:

$$Cost = 20 \times \#Miss + 10 \times \#FalseAcceptance$$

We also used a standard F-Measure to show the influence of the tolerance region size. The mean distance between following lines is computed on the whole dataset. Then squared tolerance regions are set around the ground truth points with a size corresponding to a percentage of this mean distance. Precision and recall are computed with only one hypothesis point kept per ground truth point.

## 6.2 Baselines

In this work we compared our system to two other approach.

### *6.2.1 Image-Processing based Text Line Detection*

The first one is the use of common image processing based line detector algorithms. Because of the complexity of handwritten text in archive images and because the page background may be damaged, pre-processing is performed on the images to enhance the performance of the line detection.

The image pre-processing algorithm is composed of four main steps:

1. Global and a local contrast enhancement [23].
2. Skew correction [1].
3. Dedicated process to delete left and right margins with vertical line detection.
4. Adaptive binary threshold (sliding window algorithm) coupled with a filtering to delete noisy structures.

Three different handwritten text line detectors, designed for fluctuating, touching or crossing text lines have been tested:

1. An adaptation of Nicolaou et al. [14]
2. An adaptation of Shi et al. [19],
3. The algorithm described in Moysset et al. [13].

After getting the bounding box of each line, the baseline of snippet [21] is finally computed to get the positions of the beginning of the line.

### *6.2.2 "Ehran et al. [5]"*

The second one is the algorithm of Ehran et al. [5] that has been adapted to learn to detect points instead of boxes for this special task. The system is similar to the one described in this paper except for the last layer where the neurons corresponding to the different parts of the image are all connected to a last layer that is then fully connected to the outputs that are the point coordinates. The architecture of the network (filter sizes, number of neurons on each hidden layer) has been tuned specifically for this technique.

*Use of k-Means for initialisation.*

In order to force the different outputs of the system to get some specialisation in predicting points, the initialisation of the network can be performed for this technique.

k-Means clustering is done on the training set points with N clusters. For each cluster, the centroid is associated with an output of the network and the matching step is done comparing the position of the hypothesis points to the position of the centroids instead of the position of the reference point. Note that it does not change the computation of the gradients.

This will help all the neural network outputs to be matched to some points of the training set and thus to learn.

The use of the k-Means technique detailed in section 6.2.2 was improving the results as shown in table 2.

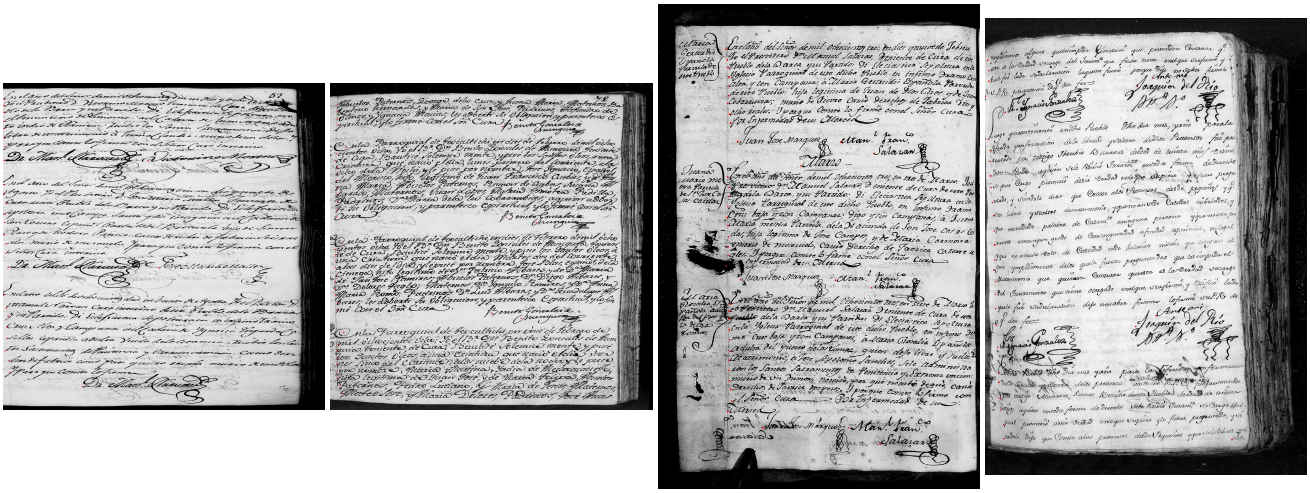


Figure 2: Illustration of ANDAR-TL database images with target points highlighted in red.

| Without K-Means | With K-Means |
|---|---|
| 61.4% | 72.2% |

Table 2: Comparison of the results with and without a k-means initialisation. F-Measure metric is used with a 80% tolerance area.

## 7. RESULTS AND DISCUSSION

All hyper parameters for our technique and for the Ehran et al. method have been optimized on the validation set.

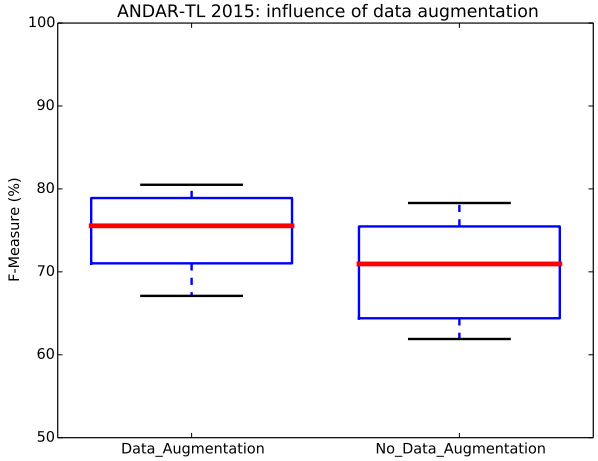
The alpha parameter described in the training section need to be tuned. As mentioned in section 4, there is not necessarily a reason to choose the same alpha parameter for the matching of the reference and hypothesis boxes on the one hand and for the backpropagation of the gradients on the other hand. The goals are different. The objective of the former step is to match outputs and groundtruth such to balance network output triplets and to keep low variances in training positions per triplet. On the other hand, the objective of the latter is to balance training of network parameters associated with geometry and confidence. We therefore optimized both weights independently, as shown in table 3. Surprisingly, both goals lead to similar $\alpha$ values.

| Alpha for weight updates | Alpha for matching | | |
|---|---|---|---|
| | 100 | 300 | 1000 |
| 100 | 35.76% | 41.96% | 39.85% |
| 300 | 36.84% | 46.23% | 42.4% |
| 1000 | 34.4%4 | 41% | 35.08% |

Table 3: Results with the F-Measure metric, and a 25% tolerance area for various alpha choices.

The impact of the context layer based on spatial LSTMs has been evaluated as given in table 4. From one to three recurrent LSTM layers have been added to the network leading to a significant gain in performance. The contextual layer has indeed a very import role in the model. During training, each ground truth point can be seen by multiple blocks. However, global matching will assign it to a single block. The contextual layer ensures coherence of this choice taking into account the global image. A deeper contextual network with additional LSTM layers further improves performance.

Figure 4: Impact of the use of data augmentation on line detection. Results are presented as F-Measure with a 80% tolerance zone and several systems.



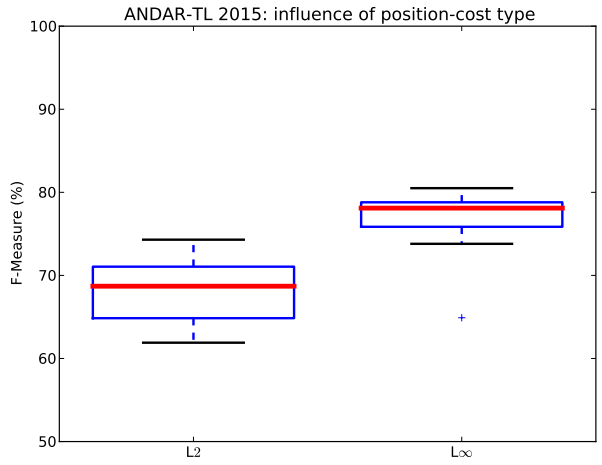

| No LSTM layer | 1 LSTM layer | 2 LSTM layers | 3 LSTM layers |
|---|---|---|---|
| 30.3% | 76.4% | 79.5% | 82.2% |

Table 4: Impact of the number of LSTM layers on line detection. Results are presented as F-Measure with a 80% tolerance zone.

The Minkowski norm $||.||^p$ used to calculate distances between network outputs and groundtruth depends on a parameter $p$. We have observed an improvement of the results when using the infinity norm $L_\infty$ in the matching cost and in the loss function instead of using the Euclidian distance $L_2$. An illustration of these results can be found in figure 5, which reports the distribution of performances over a large set of experiences involving different choices of hyper-parameters (with or without dropout, data augmentation, contextual layer etc.). We think that the better performance of the $L_\infty$ norm could be explained by the fact that our metrics are using squared acceptance zones around the target points.

Dropout [11] has been applied during the training of the networks and improved the results, as shown in table 5.

Figure 5: Impact of the choice of the norm on line detection. Results are presented as F-Measure with a 80% tolerance zone and several systems.




| No dropout | With dropout |
|---|---|
| 61.5% | 82.2% |

Table 5: Impact of the use of dropout. Results are presented as F-Measure with a 80% tolerance zone.

Lack of invariance properties of deep networks often requires data augmentation to ensure high generalization [2]. In our work, data augmentation has been performed to increase the number of training pages and thus improve the generalisation. Pages have been simply moved laterally and expanded or shrunk by a small factor. After this transformation, only the pages with all the target points inside the image have been kept. Figure 4 shows that there is an improvement of the performances when using data augmentation, in particular due to the small size of the database.

We compare the performance of the proposed method to different other techniques of the state of the art. Results using the two metrics given in section 6.1 are provided in table 6 below. For the F-Measure metric, two sizes of acceptance zones have been computed with different values for the $\gamma$ parameter presented in section 6.1. The detail of the F-Measure results with respect to this parameter and therefore to the size of the tolerance zones is shown in figure 6.

As can be seen, our method outperforms the state of the art with a medium sized tolerance region and is only beaten by Shi et al. [19], an image processing method without learning, if the tolerance region is set to a low value. For this reason, we also combined the two methods, as given in the last line in table 6. This combination achieves the highest results over all tested methods.

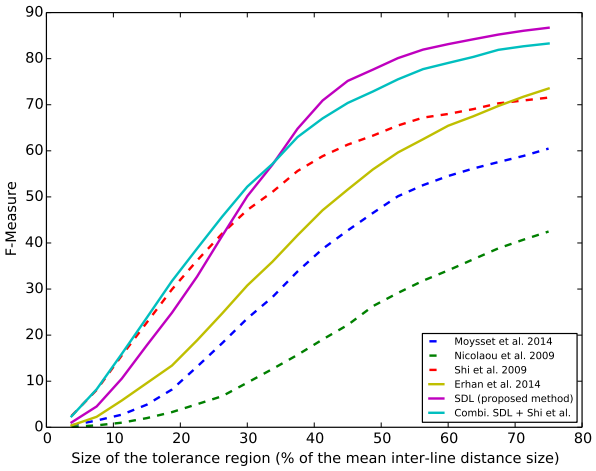
The origin points predicted by the ConvNN/LSTM system are matched to the origin points predicted by the image processing approach system[19], using the Munkes matching algorithm and minimizing the sum of the Euclidian distances. When the matched points are less than 150 pixels distant, the location predicted by the image processing approach system is kept. Otherwise, We keep the point predicted by the ConvNN/LSTM system. Points predicted by the ConvNN/LSTM system and unmatched are also kept. We do so because we observed that the ConvNN/LSTM system is better to predict the right number of ground truth points in the page, while the image processing approach system is more accurate in the location it gives.

| Method | F-Measure[1] 80% (tolerance) | F-Measure[1] 25% (tolerance) | ANDAR[2] official metric |
|---|---|---|---|
| Moysset et al. [13] with preprocessing | 62.4 | 16.5 | 599.0 |
| Nicolaou et al. [14] with preprocessing | 44.6 | 6.1 | 716.0 |
| Shi et al. [19] with preprocessing | 69.0 | 35.5 | 468.8 |
| Erhan et al. [5] | 72.2 | 27.1 | 512.2 |
| SDL (proposed) | 85.5 | 31.4 | 493.9 |
| SDL (proposed) + Shi et al. [19] | 81.3 | 38.9 | 449.6 |

[1] Higher is better ; [2] Lower is better

Table 6: Comparison of the different systems with several metrics.

Figure 6: F-Measure of the different systems with respect to the size of the tolerance region.




The dependency of performance on the precision used in the evaluation metric (the size of the tolerance region) is plotted graphically in figure 6.

Let us note, in particular, that the proposed local displacement layer significantly outperforms the global method proposed by Erhan et al. [5], confirming the validity of our strategy to share network weights over overlapping blocks.

We also participated in the ICDAR ANDAR-TL competition. However, the official results were still pending at the submission of the paper. Our submission was a combination of the proposed deep learning approach and a second method based on image processing without learning [19].

## 8. CONCLUSION AND PERSPECTIVES

We presented a new semi-local model for the detection of multiple objects in an image. In particular, we present an application to text line detection in document images. The deep model is a compromise between a purely local method based on sliding windows and a pure global method based on regression. This strategy allows to keep the advantages of both extremes. Positions are delivered directly without post-processing, and the semi-local approach allows to share weights over spatially overlapping blocks. Experiments have shown excellent results on the ICDAR 2015 ANDAR-TL dataset. The official ranking in the ANDAR-TL competition is pending.

Future work will tackle the problem of detecting full bounding boxes instead of origin points. This is a more difficult problem,

since the different points of a bounding box may lie on different block of the model. An additional matching stage is required.

# 9. REFERENCES


[1] D. S. Bloomberg, G. E. Kopec, and L. Dasari. Measuring document image skew and orientation. In *Proc. Document Recognition II*, 1995.

[2] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *CoRR*, abs/1405.3531, 2014.

[3] M. Delakis and C. Garcia. text detection with convolutional neural networks. In *VISAPP (2)*, pages 290–294, 2008.

[4] X. Du, W. Pan, and T. D. Bui. Text line segmentation in handwritten documents using mumford–shah model. *Pattern Recognition*, 42(12):3136–3145, 2009.

[5] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable Object Detection using Deep Neural Networks. In *Proc. CVPR*, 2014.

[6] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9):1627–1645, 2010.

[7] P. Felzenszwalb and D. Huttenlocher. Pictorial Structures for Object Recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[8] C. Garcia and M. Delakis. Convolutional face finder: a neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:1408 – 1423, 2004.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to appear.

[10] A. Graves, S. Fernández, and J. Schmidhuber. Multi-dimensional recurrent neural networks. 2007.

[11] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv: ...*, pages 1–18, 2012.

[12] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis. Text line and word segmentation of handwritten documents. *Pattern Recognition*, 42(12):3169–3183, Dec. 2009.

[13] B. Moysset, T. Bluche, M. Knibbe, M. F. Benzeghiba, R. Messina, J. Louradour, and C. Kermorvant. The a2ia multi-lingual text recognition system at the maurdor evaluation. In *Proc. International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2014.

[14] A. Nicolaou and B. Gatos. Handwritten Text Line Segmentation by Shredding Text into its Lines. *International Conference on Document Analysis and Recognition*, 2009.

[15] V. Papavassiliou, V. Katsouros, and G. Carayannis. A Morphological Approach for Text-Line Segmentation in Handwritten Documents. In *International Conference on Frontiers in Handwriting Recognition*, 2010.

[16] I. Rabaev, O. Biller, J. El-Sana, K. Kedem, and I. Dinstein. Text line detection in corrupted and damaged historical manuscripts. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 812–816. IEEE, 2013.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[18] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[19] Z. Shi, S. Setlur, and V. Govindaraju. A Steerable Directional Local Profile Technique for Extraction of Handwritten Arabic Text Lines. In *Proc. International Conference of Document Analysis and Recognition (ICDAR)*, 2009.

[20] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[21] A. Vinciarelli and J. Luettin. A new normalisation technique for cursive handwritten words. *Pattern Recognition Letters*, 22:1043–1050, 2001.

[22] A. Zahour, L. Likforman-Sulem, W. Boussellaa, and B. Taconet. Text Line Segmentation of Historical Arabic Documents. In *International Conference on Document Analysis and Recognition*, 2007.

[23] K. Zuiderveld. Graphics gems iv. chapter Contrast Limited Adaptive Histogram Equalization, pages 474–485. Academic Press Professional, Inc., San Diego, CA, USA, 1994.