

This is a pre print version of the following article:

Watch Your Strokes: Improving Handwritten Text Recognition with Deformable Convolutions / Cojocaru, Iulian; Cascianelli, Silvia; Baraldi, Lorenzo; Corsini, Massimiliano; Cucchiara, Rita. - (2021), pp. 6096-6103. ( 25th International Conference on Pattern Recognition, ICPR 2020 Milan, Italy 10-15 January 2021) [10.1109/ICPR48806.2021.9412392].

Institute of Electrical and Electronics Engineers Inc.

*Terms of use:*

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

24/12/2025 11:20

(Article begins on next page)

# Watch Your Strokes: Improving Handwritten Text Recognition with Deformable Convolutions

Iulian Cojocaru, Silvia Cascianelli, Lorenzo Baraldi, Massimiliano Corsini and Rita Cucchiara  
University of Modena and Reggio Emilia  
Email: {name.surname}@unimore.it

**Abstract**—Handwritten Text Recognition (HTR) in free-layout pages is a valuable yet challenging task which aims to automatically understand handwritten texts. State-of-the-art approaches in this field usually encode input images with Convolutional Neural Networks, whose kernels are typically defined on a fixed grid and focus on all input pixels independently. However, this is in contrast with the sparse nature of handwritten pages, in which only pixels representing the ink of the writing are useful for the recognition task. Furthermore, the standard convolution operator is not explicitly designed to take into account the great variability in shape, scale, and orientation of handwritten characters. To overcome these limitations, we investigate the use of deformable convolutions for handwriting recognition. The kernel of this type of convolution deforms according to the content of the neighborhood, and can therefore be more adaptable to geometric variations and other deformations of the text. Experiments conducted on the IAM and RIMES datasets demonstrate that the use of deformable convolutions is a promising direction for the design of novel architectures for handwritten text recognition.

## I. INTRODUCTION

Handwritten Text Recognition (HTR) aims at automatizing document processing by providing natural language transcriptions of handwritten texts. As such, it plays an important role in automated services, document processing, and Digital Humanities. In this last field, the applications range from the transcription of large document corpora to the analysis of toponyms on ancient maps. Despite Optical Character Recognition (OCR) being a mature and well-established technology, HTR is still a challenging task even when tackled with approaches based on feature learning, especially when it comes to free-layout pages.

As Deep Learning has advanced the state of the art in many image and text understanding tasks, most of the current approaches for handwriting recognition employ architectures based on Deep Neural Networks. The input image is usually encoded by applying a Convolutional Neural Network (CNN), while the underlying text is decoded by employing a Recurrent Neural Network (RNN), in charge of generating the output character sequence [1]. Typically, these approaches rely on standard convolutional layers, in which features from the input image are extracted by sliding kernels with fixed shape and parameters. However, handwritten texts can more effectively be thought as a sparse structure, in which only a small part of the input can actually be used for the recognition task (*i.e.*, the ink pixels). Indeed, the handwritten text is basically a curve, hence a dense 2D kernel may be the not proper solution

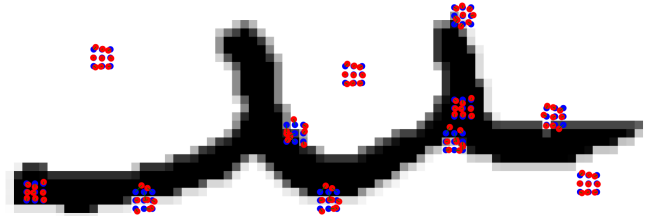


Fig. 1: Sampling grid of a convolution (in blue) and of a deformable convolution (in red) kernel over a handwritten character. Deformable convolutions can adapt better to handwritten strokes (best seen in color).

to process it. Moreover, handwritten characters and words are inherently varying in shape, scale, and orientation. With standard convolutions this variability is not effectively taken into account unless ad hoc data augmentation or preprocessing are performed.

Motivated by the above considerations, in this paper we propose to apply deformable convolutions (DefConvs) [2] in place of standard convolutions for the HTR task. Since now, DefConvs have been employed for the task of object recognition, showing great adaptability to geometric variations and to part deformations, and the ability to model transformations in the object scale, pose, and viewpoint. To the best of our knowledge, this is the first work exploring the suitability of DefConvs for handwriting recognition. Our expectation is that their kernel adaptability (see Fig. 1) helps to improve the efficiency and the performance in the task. A deep analysis and quantitative and qualitative results on two benchmark HTR datasets confirm this behavior.

The rest of this paper is organized as follows. After providing a review of the most relevant literature for HTR, in Section III we will present our architecture for recognizing handwritten characters. In Section IV we then assess the role of DefConvs by performing comparisons and experiments on two benchmark datasets. Through a series of qualitative and quantitative results, we will show the benefit of using DefConvs in the design of HTR architectures.

## II. RELATED WORK

In early works [3], [4], HTR was performed by building Hidden Markov Models (HMM) upon heuristic visual features

to recognize text, eventually combining them with N-gram Language Models (LMs) to enhance the recognition accuracy. See [5] for a detailed survey. This approach has been outperformed by recent Deep Learning-based strategies [6].

HTR can be performed at *character level* [7], *i.e.*, the text is recognized as one isolated handwritten character at a time. This task was the first tackled using LeNet [8], and is what is currently done for ideographic languages such as Chinese [9] and Japanese [10]. For alphabetic languages, HTR can be also performed at *word level* [11], [12], [13], *i.e.*, decoding single words that are detected in the image. This task is performed both on digitalized documents, and in scene images [14]. Furthermore, many works focus on HTR at *line level*, *i.e.*, the full text of a single line is transcribed, also taking into account spaces which are disregarded in the word level HTR task. Text lines recognition can be performed either on pre-segmented text [15], [1], [16], [17], [18] or integrated into a joint detection-and-recognition system that automatically detects and segments the line in a document image [19], [20]. In this work, we tackle the HTR problem at line level, starting from pre-segmented text lines. Finally, recent works tackle the HTR problem at *paragraph level* [20], [21] or *page level* [10], [19] directly. These works combine layout analysis techniques such as paragraph or line segmentation [22], [23], [24], [25], [26] with line level HTR [27] strategies. In the above-mentioned variants of the HTR task, for highly represented languages, *i.e.*, those for which a sufficiently big textual corpus is available, character-level and word-level language models can be integrated into the recognition process to enhance accuracy.

A major challenge in the HTR task is the non-ideality of handwritten characters, which can vary in shape and size non-uniformly. To face this issue, specific data augmentation [28], [29], [30] and preprocessing [1], [16], [31] strategies have been proposed. At architecture level, Zhong *et al.* [32] proposed to apply a Spatial Transformer Network (STN) [9] for recognizing Chinese handwritten characters. In the context of word-level HTR, Bhunia *et al.* [12] proposed to warp the features extracted by the intermediate layers of a CNN by inserting an Adversarial Feature Deformation Module in-between. In STNs, spatial transformations are applied on the input image to enhance geometric invariance. DefConvs add learnable offsets to the regular grid sampling locations in the standard convolution, so it can be thought as a local, dense and light-weight spatial transformer in STN.

In this work, we propose to investigate the role of deformable convolutional kernels in comparison with standard fixed kernels for the task of HTR. Starting from the state-of-the-art architecture proposed in [14], we replace the convolutional layers with deformable ones, and evaluate their effect. We will see that focusing on the most suitable parts of the image only, *i.e.*, along the curve forming the text instead of treating it and the background uniformly permits to increase the performance and reduce overfitting, even without severe data augmentation.

### III. PROPOSED METHOD

Convolutions have been the key ingredient to the success of CNNs, and they are the main actors in the feature extraction steps carried out inside any CNN. Typically, the convolutional operator consists in a learned and weighted sum conducted over a regular neighborhood of the image, which favours a position-independent and local feature extraction process. Formally, given a kernel  $k$  of learnable weights and a regular grid  $\mathcal{N}$ , convolution on a pixel  $p$  can be defined as follows:

$$y(p) = \sum_{d \in \mathcal{N}} k(d) \cdot x(p + d), \quad (1)$$

where  $d$  is a displacement vector and  $\cdot$  is the inner product between channel-wise feature vectors. The neighborhood  $\mathcal{N}$  depends on the receptive field and dilation of the kernel.

The recently proposed deformable convolutions [2], on the contrary, sample the input image on an irregular grid whose geometry is learned as a function of the context, thus allowing a non-local and position-dependent feature extraction. Conceptually speaking, this can help to handle geometric transformations of patterns as well as their sparse structure. The deformation of the grid is achieved by adding 2D offsets to each of the sampling positions of a regular grid (see Fig.2). The offsets are learned alongside with the kernel weights in an additional convolutional layer, thus ensuring a content-dependent deformation. Formally, Eq.1 in deformable convolution is replaced by

$$y(p) = \sum_{d \in \mathcal{N}} k(d) \cdot x(p + d + \Delta d), \quad (2)$$

so that the set of points of the deformed kernel becomes  $\{d + \Delta d\}_{d \in \mathcal{N}}$ . Since the offsets produced by the additional convolutional layer are generally fractional, and introducing a quantization step would harm the training phase, Eq. 2 is implemented through bilinear interpolation. Formally,

$$y(p) = \sum_{d \in \mathcal{N}} k(d) \cdot \sum_{s \in \mathcal{S}} B(s, p + d + \Delta d) \cdot x(s), \quad (3)$$

where  $B(\cdot, \cdot)$  is the 2D bilinear interpolation kernel, and  $\mathcal{S}$  is the set of points in the input feature map which are close to the sampling locations  $\{p + d + \Delta d\}_{d \in \mathcal{N}}$ . Despite the addition of a convolutional layer for computing the deformation offsets, the number of parameters increases only slightly. In particular, for each kernel in a standard convolutional layer, the number of parameters needed to model the offsets is  $2K$ , where  $K$  is the kernel size.

A sample of the grids obtained with a  $3 \times 3$  deformable kernel, when applied on the image of a handwritten character, is reported in Fig. 1, where we also compare them with those of a standard convolutional kernel. Noticeably, the kernel is only slightly deformed when applied on uniform regions (background or ink). On the contrary, when the kernel operates on stroke edges, it undergoes a more significant deformation. The same trend can be observed also in Fig. 3, where we report the cumulative magnitude of the offsets applied to a  $3 \times 3$  kernel grid in each point of an image of a word.

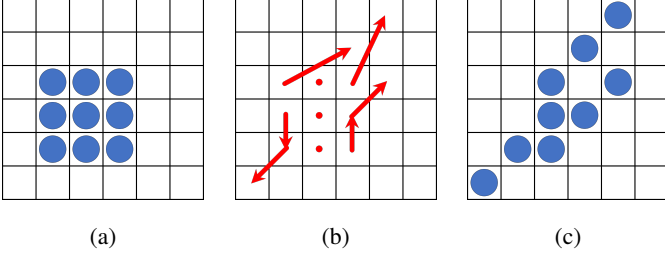


Fig. 2: The regular sampling grid of standard convolution  $d$  (2a) is deformed by applying a set of offsets  $\Delta d$  (2b), obtaining the deformed sampling grid of the deformable convolution (2c).

Due to the capability of DefConvs to adapt to geometric transformations in their inputs, we propose to apply them instead of standard convolutions for the HTR task. To this end, we adapt the sequence recognition network proposed in [14], commonly used as a base for HTR schemes (see for example [29], [27], [18]), and replace all its standard convolution layers with DefConv layers.

The model consists of three main parts: a CNN to extract sequences of features from the input image, an RNN to produce labels probabilities based on the sequence, and a decoding block to output the final transcription. Note that in this setting, the input images are rescaled to have the same height. As customary for HTR at line-level, the proposed network is trained to maximize the Connectionist Temporal Classifier (CTC) probability of the transcribed sequence. For this reason, the labels scored by the RNN are textual characters and a special character (called *blank*) meaning “no other characters”.

For the convolutional part, we take the architecture of VGG-11 [33] up to the fourth convolution blocks and add a 7<sup>th</sup> convolution layer with a  $2 \times 2$  kernel. All the standard convolutional layers are replaced with DefConvs. A DefConv layer is obtained through the concatenation of a standard convolutional layer, for the offsets, and another convolutional layer for the kernel weights. Also, we change the receptive field of the 3<sup>rd</sup> and 4<sup>th</sup> max-pooling layers from squared  $2 \times 2$  to rectangular  $2 \times 1$ . This way, we obtain wider feature maps, that better reflect the height-width ratio of text-lines images. The feature map of the last layer is used to obtain the sequential input for the RNN. In particular, given a feature map of size  $H \times W \times C$ , we build  $W$  ( $H \cdot C$ )-elements feature vectors from left to right, each one by concatenating the  $w^{\text{th}}$   $C$ -dimensional vector of each of the  $H$  map’s rows. Each feature vector of the sequence corresponds to a region of the original image, *i.e.*, its receptive field. Since we use DefConvs, the receptive fields have irregular, non-rectangular shape, but better follow the handwriting strokes and cover a wider area. Nevertheless, given the way the feature vector sequence is collected, also these receptive fields are considered left to right. A pictorial representation of such receptive fields is given in Fig. 4 both for our model using DefConvs and the original model in [14], which employs standard convolutions.

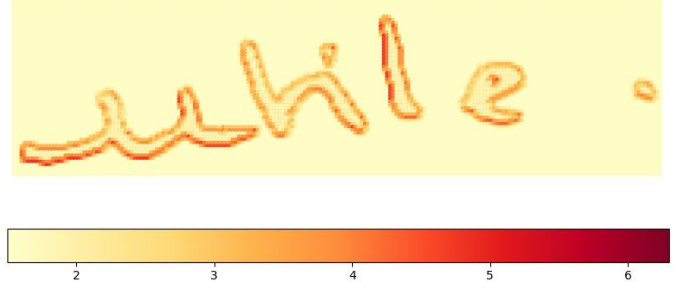


Fig. 3: Cumulative magnitude of the offsets applied to a  $3 \times 3$  kernel’s grids on points of an image of a word. Grids sampling in uniform regions are less deformed than those sampling on edges.

The recurrent part of the scheme consists of a stack of two Bidirectional Long Short-Term Memory networks (BLSTMs). It takes as input one feature vector at a time and produces the label probabilities of the image region corresponding to the feature vector.

Finally, the decoding block produces the transcription by taking the most probable label at each timestep, removing duplicate characters not separated by a *blank*, and then removing the blanks.

#### A. Analysis of deformable kernels

One of the main intuitions in using DefConvs in this task is that the kernel should deform itself focusing on the writing instead of the background. This is confirmed by the following analyses here reported.

To locate the pixels where the kernel is subject to a more severe deformation, in Fig. 3, the cumulative magnitude of the offsets is represented for each pixel. As expected, the deformations are concentrated around the writing parts.

Moreover, Fig. 5 depicts the activations computed with the Saliency algorithm [34], when standard convolutions and deformable convolutions are used. In the first row, it is reported the maximum activations given by considering each class, while in the second and third rows, it is reported the activation when a specific character is recognized (‘u’ and ‘l’ respectively). As it can be observed, just a few background pixels are activated in the case of DefConvs. Arguably, this behavior makes the recognition more robust against a noisy background *e.g.*, due to small scratches and stains caused by paper acidification.

### IV. EXPERIMENTAL EVALUATION

In this section, we evaluate the suitability of the proposed DefConvs-based method for the HTR task when compared to a baseline that features standard convolutions. In this section, we refer to the proposed approach as Full-DefConv.

#### A. Experimental Setup

1) *Datasets*: To validate the proposed approach, we use the benchmark line-level IAM [35] and the Reconnaissance et Indexation de données Manuscrites et de fac similés (RIMES) [36] datasets.

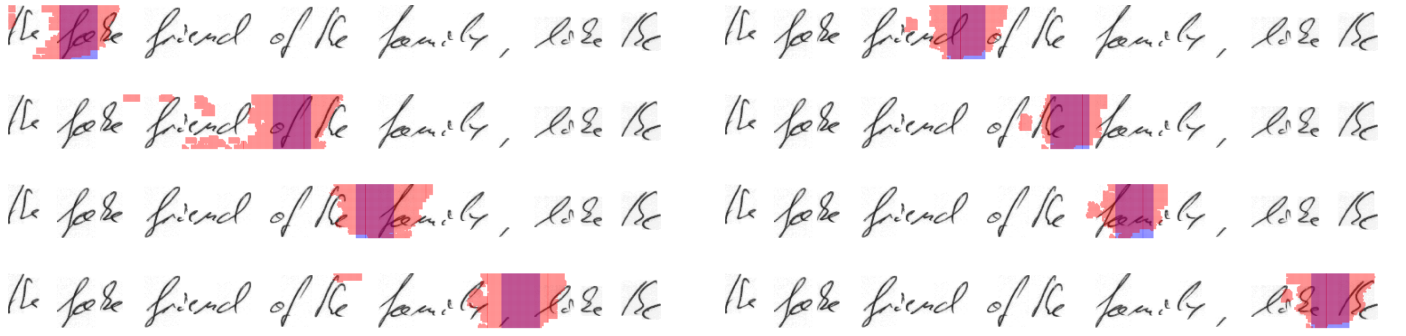


Fig. 4: Some receptive fields of the HTR network using standard convolutions (in transparent blue) and DefConvs (in transparent red) on a text line image. DefConvs lead to non-connected areas of irregular shape that better adapt to handwritten strokes and cover a wider portion of the image thanks to the limited amount of additional offsets parameters (best seen in color).

The IAM Handwriting dataset features unconstrained textual documents in modern English, handwritten by multiple users copying paragraphs from the Lancaster-Oslo/Bergen (LOB) corpus [37]. The dataset comes with an official writer-independent splitting, specified on the dataset website<sup>1</sup> for the Large Writer Independent Text Line Recognition Task (6161 lines for training, 900 for validation, and 1861 for test). However, in our experiments, we use the so called Aachen University splitting<sup>2</sup>, since it is more commonly applied in the HTR literature. This splitting provides 6482 training lines, 976 validation lines, and 2915 test lines. The total number of non-*blank* characters in this dataset is 95, and the line images width and height are  $1698 \pm 292$  pixels and  $124 \pm 34$  pixels respectively. Some exemplar images from this dataset can be observed in Fig. 6a.

The RIMES dataset features handwritten free-layout letters written by multiple authors in modern French. The official splitting for this dataset is 11333 lines for training, and 778 lines for test. Since no official validation splitting is given, we retained the lines contained in the 10% of documents in the training set for validation. Non-*blank* characters in this dataset are 79, and the images contained are  $1637 \pm 555$  pixels wide, and  $130 \pm 36$  pixels high. Some exemplar images from this dataset can be observed in Fig. 6b.

2) *Compared Approaches*: As explained in Section III, we build upon the method proposed in [14] and replace all its standard convolutional layers with deformable convolutions. In the experiments, we use our implementation of [14] as baseline. This way, we can evaluate the effect of using DefConvs instead of standard convolutions on the HTR task.

Moreover, we report the results of other approaches in literature. All these exploit standard convolution. To better appreciate the role of the deformable convolution w.r.t the standard one, we consider only methods that do not apply any Lexicon or Language Model (LM). Furthermore, for each compared approach, we specify the training/validation/test splitting applied by the authors on the considered datasets.

<sup>1</sup> [www.fki.inf.unibe.ch/databases/iam-handwriting-database/](http://www.fki.inf.unibe.ch/databases/iam-handwriting-database/)  
iam-handwriting-database

<sup>2</sup> [www.tbluche.com/resources.html](http://www.tbluche.com/resources.html)

Some details about the considered approaches are given in the following. Bluche [21] is a method for paragraph-level HTR that applies the commonly used multi-dimensional long short-term memory recurrent neural network (MDLSTM-RNN) [11]. MDLSTM-RNNs build a 2D representation of the textual image and collapse it in a sequence of vectors used for decoding. In [21], the collapsing mechanism is performed by a MDLSTM-based network, which implicitly performs line segmentation. Wigington *et al.* [27] is a page-level HTR system, whose major strength is a mechanism to segment and dewarp text lines, even if curved. For the text recognition component of their system, the authors build upon [14], as we do in our approach, and employ a specifically designed data augmentation strategy [29] to modify the words' shape. In their experiments on the line-level IAM and RIMES datasets, both [29] and [21] used their line segmentation strategy instead of the provided line segmentation. Voigtlander *et al.* [16] built upon the MDLSTM-RNN network proposed in [11] and devised a deeper and wider architecture, by stacking alternating convolutional layers and MDLSTMs before the collapsing layer. Also Pham *et al.* [15] built upon the MDLSTM-RNN network and explored the effect of dropout as a regularization strategy for HTR models. Puigcerver [1] proposed a simpler alternative to MDLSTM-RNNs for line-level HTR, consisting of a CNN to extract a sequence of feature vectors from the text image, and 1D-LSTMs to output characters' probabilities for the CTC decoding. Additionally, random distortions (affine transformation, gray-scale erosion, and dilation) are applied to the input images during training.

3) *Implementation Details*: Both for our approach and the baseline, we rescale the text line images in height so that all the images become 60 pixels high, keeping the original aspect ratio. Moreover, the images are normalized between -1 and 1. In TABLE I a scheme of the convolutional part of the proposed model is reported, specifying the number of channels, kernel size, stride and padding of each layer. The offsets of each DefConv layer are learnt in a paired standard convolutional layer. The feature map at the last layer is a  $2 \times W \times 512$  tensor, which is collapsed in a sequence of  $W$  vectors of 1024 elements. The two BLSTMs that constitute



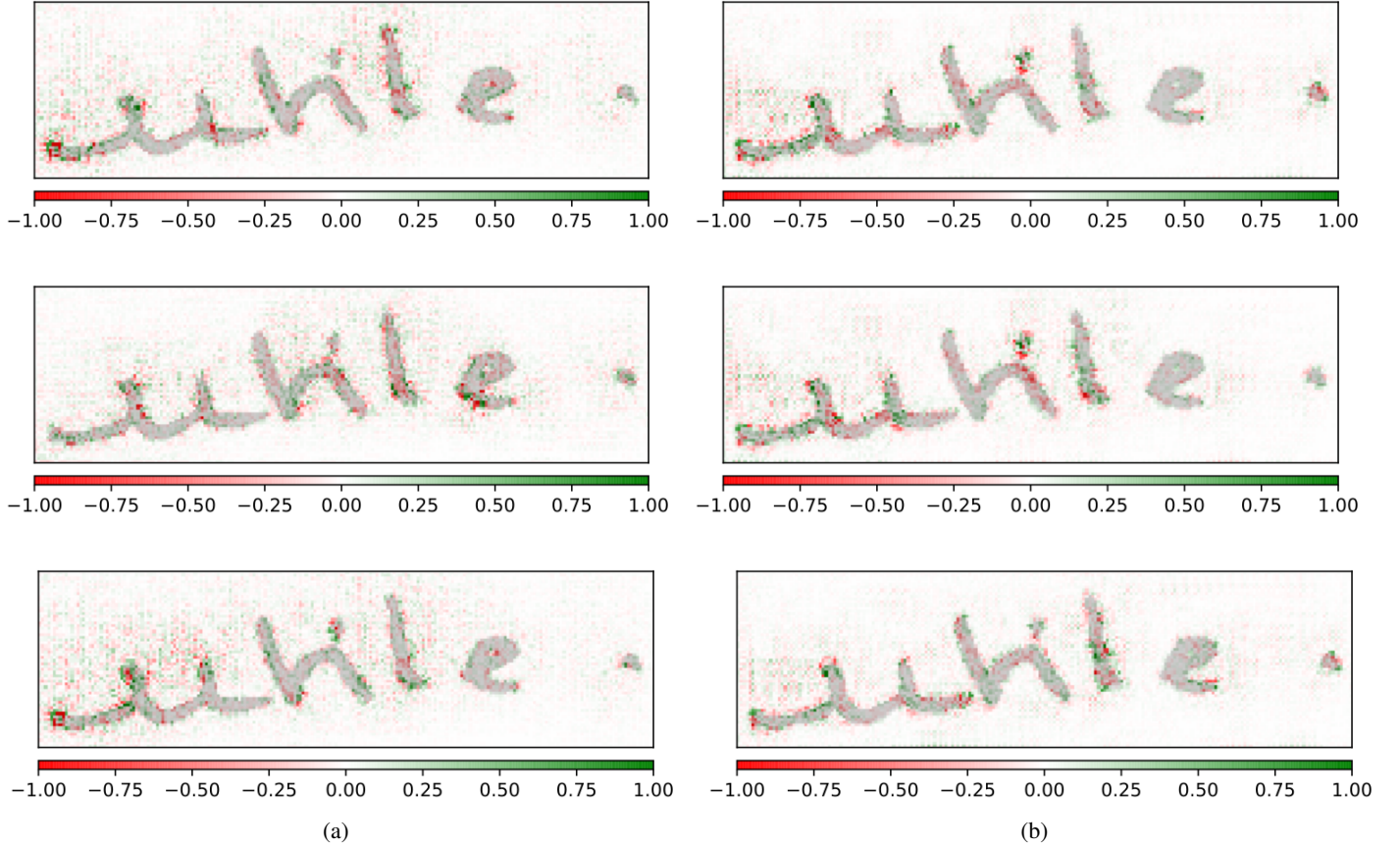


Fig. 5: Activations of the standard convolutions (5a) and deformable convolutions (5b) computed with the Saliency algorithm. DefConvs concentrate the activations on the writing instead of on the background (best seen in color).

the recurrent part of the model have 512 hidden units each and are separated by a dropout layer with dropout probability equal to 0.5. The baseline model has a similar architecture, but without the offsets' layers and with standard convolutions instead of DefConvs.

The proposed model and the baseline have been trained for 500 epochs each (the best model in terms of CER is used for testing), with batch size equal to 8 using Adam [38] as optimizer with  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ , and learning rate equal to 0.0001. The final models have comparable size: 70MB for the baseline and 71MB for the proposed network.

### B. Results and Discussion

The obtained results are summarized in TABLE II and TABLE III. For the IAM and the RIMES datasets the commonly used metrics, Character Error Rate (CER), and the Word Error Rate (WER) are reported.

With respect to the other State-of-the-Art approaches, Full-DefConv performs competitively, especially compared to the approaches that, as in our case, do not perform any preprocessing or data augmentation (Pham *et al.* [15], Voigtlaender *et al.* [16]). The second-best performing method on the IAM dataset, *i.e.*, Puigcerver [1], and the best-performing method on the RIMES dataset *i.e.*, Wigington *et al.* [29], include

specifically designed data augmentation. Moreover, on the RIMES dataset, which contains many non-straight text line samples, the approaches that combine line segmentation and text recognition (*i.e.*, Wigington [27] and Bluche [21]) are more suitable than line-level approaches. This suggests that the deformations of the kernels learned in our model are effective in handling distortions in words and characters, but not the higher-level line curvature.

Compared to the baseline (Shi *et al.* [14]), Full-DefConv allows decreasing both the CER and the WER. The improvement on the WER is more significant, meaning that errors on characters not only are inferior in number but also are more concentrated, *i.e.*, they are made within the same word more often than in the case of the baseline.

Further, we compare the proposed approach and the baseline by testing on the test images of both datasets when White Gaussian noise or Poisson shot noise with different variance is added. This way, we can evaluate the robustness of our approach with respect to noise that models *e.g.*, low-quality or degraded paper. Exemplar noisy images are reported in Fig. 7 and in Fig. 8. The results of this study are shown in TABLE IV for the IAM dataset and in TABLE V for the RIMES dataset in case of addition of Gaussian noise and in TABLE IV and in TABLE V in case of addition of Poisson noise. It is observed

Layer Type	Channels	Kernel	Stride	Padding
Convolution	18	$3 \times 3$	(1, 1)	(1, 1)
DefConv	64	$3 \times 3$	(1, 1)	(1, 1)
ReLU	—	—	—	—
Max Pooling		$2 \times 2$	(2, 2)	(0, 0)
Convolution	18	$3 \times 3$	(1, 1)	(1, 1)
DefConv	128	$3 \times 3$	(1, 1)	(1, 1)
ReLU	—	—	—	—
Max Pooling		$2 \times 2$	(2, 2)	(0, 0)
Convolution	18	$3 \times 3$	(1, 1)	(1, 1)
DefConv	256	$3 \times 3$	(1, 1)	(1, 1)
Batch Normalization	—	—	—	—
ReLU	—	—	—	—
Convolution	18	$3 \times 3$	(1, 1)	(1, 1)
DefConv	256	$3 \times 3$	(1, 1)	(1, 1)
ReLU	—	—	—	—
Max Pooling		$2 \times 2$	(2, 1)	(0, 1)
Convolution	18	$3 \times 3$	(1, 1)	(1, 1)
DefConv	512	$3 \times 3$	(1, 1)	(1, 1)
Batch Normalization	—	—	—	—
ReLU	—	—	—	—
Convolution	18	$3 \times 3$	(1, 1)	(1, 1)
DefConv	512	$3 \times 3$	(1, 1)	(1, 1)
ReLU	—	—	—	—
Max Pooling		$2 \times 2$	(2, 1)	(0, 1)
Convolution	8	$2 \times 2$	(1, 1)	(0, 0)
DefConv	512	$2 \times 2$	(1, 1)	(0, 0)
Batch Normalization	—	—	—	—
ReLU	—	—	—	—

TABLE I: Implementation details of the convolutional part of the proposed method. The offsets of the DefConvs layers are handled in a standard convolutional layer before the DefConv, that is in charge to learn two parameters for each kernel’s cell of the DefConv.

that the performance of Full-DefConv is more stable than those of the baseline, thus our approach is more robust to noise. This result is in line with the analysis reported in Section III-A, the activations are more concentrated on the writing, hence more robust to the noise present in the input image.

This behavior can be observed also from the qualitative results reported in Fig.6. These allow underlying additional

	CER	WER
<b>Full-DefConv</b> ‡	4.6	19.3
<b>Shi et al. [14]</b> ‡	5.7	23.2
<b>Wigington et al. [27]</b> ‡	6.4	23.2
<b>Voigtlaender et al. [16] – LM</b> *‡	8.3	27.5
<b>Puigcerver [1]</b> ‡	6.2	20.2
<b>Bluche [21]</b> †	7.9	24.6
<b>Pham et al. [15]</b> †	10.8	35.1

TABLE II: Results on the IAM dataset. Methods with † use 6482 lines for training, 976 for validation, and 2915 for test. Methods with ‡ use 6161 lines for training, 976 for validation and 2915 for test. Note that \* indicates results of re-implemented method as from [1].

	CER	WER
<b>Full-DefConv</b>	4.6	14.8
<b>Shi et al. [14]</b>	5.3	17.5
<b>Wigington et al. [27]</b>	2.1	9.3
<b>Voigtlaender et al. [16] – LM*</b>	4.0	17.7
<b>Puigcerver [1]</b>	2.6	10.7
<b>Bluche [21]</b>	2.9	12.6
<b>Pham et al. [15]</b>	6.8	28.5

TABLE III: Results on the RIMES dataset. Note that \* indicates results of re-implemented method as from [1].

effects of using DefConvs compared to standard convolutions. In particular, it can be noticed that background irregularities do not affect the transcription produced by Full-DefConv. Moreover, stroked-out words are handled with the ‘#’ character, which is preferable to the (erroneous) transcription of the stroked-out text.

	<i>No noise</i>		$\mathcal{G}(0, 10)$		$\mathcal{G}(0, 20)$		$\mathcal{G}(0, 30)$	
	CER	WER	CER	WER	CER	WER	CER	WER
<b>Full-DefConv</b>	4.6	19.3	4.7	19.5	5.5	22.2	18.3	49.0
<b>Shi et al. [14]</b>	5.7	23.2	5.8	23.7	6.9	26.5	24.4	62.8

TABLE IV: Results of the proposed approach and the baseline on IAM images with additive Gaussian noise  $\mathcal{G}(\mu, \sigma)$ .

	<i>No noise</i>		$\mathcal{G}(0, 10)$		$\mathcal{G}(0, 20)$		$\mathcal{G}(0, 30)$	
	CER	WER	CER	WER	CER	WER	CER	WER
<b>Full-DefConv</b>	4.6	14.8	4.6	14.8	4.7	15.4	5.1	17.0
<b>Shi et al. [14]</b>	5.3	17.5	5.3	17.3	5.4	18.2	6.0	20.2

TABLE V: Results of the proposed approach and the baseline on RIMES images with additive Gaussian noise  $\mathcal{G}(\mu, \sigma)$ .

did not act as though he found it necessary

**Groud Truth:** did not act as though he found it necessary  
**Full-DefConv:** did not act as though he found it necessary  
**Shi et al. [14]:** dd n act as thaugh kefanod it necarseay

earth had lost its life-tempo, as the heart

**Groud Truth:** earth had lost its life-tempo, as the heart  
**Full-DefConv:** earth had lost its lefe-tempo, as the heart  
**Shi et al. in [14]:** earthled bost its eferteupo, as the beat

liked ~~to go where~~ during his off-duty periods

**Groud Truth:** liked during his off-duty periods  
**Full-DefConv:** liked # during his off-duty periods  
**Shi et al. [14]:** liked tegotere during his off-duty periots

(a)

M<sup>d</sup> Dubois. je souhaiterais être couvert au titre de la responsabilité

**Groud Truth:** Md Dubois je souhaiterais être couvert au titre de la responsabilité  
**Full-DefConv:** Ma Duois je souhaiterais être coupet au. Titre de l ressonabilité  
**Shi et al. in [14]:** ma Buboiss. Je souhaterisr être lea mert u titre de b ressonssbilité

JE ME PERMET DE VOUS ECRIRE POUR AVOIR

**Groud Truth:** JE ME PERMETS DE VOUS ECRIRE POUR AVOIR  
**Full-DefConv:** JE ME MERMETS DE VOUS ECPUR POUP QNOIR  
**Shi et al. in [14]:** JEe PERMET DE voUS FcAlrRe Pour avoin

merci de votre collaboration

**Groud Truth:** merci de votre collaboration  
**Full-DefConv:** erci de votre collaloration  
**Shi et al. in [14]:** A'acexio de votre collaloration

(b)

Fig. 6: Qualitative results on some test lines of the IAM dataset (6a) and the RIMES dataset (6b).

	No noise		$\mathcal{P}(10)$		$\mathcal{P}(20)$		$\mathcal{P}(30)$	
	CER	WER	CER	WER	CER	WER	CER	WER
<b>Full-DefConv</b>	4.6	19.3	4.8	19.8	5.5	22.0	10.6	33.3
<b>Shi et al. [14]</b>	5.7	23.2	5.9	24.0	6.7	26.0	13.6	41.2

TABLE VI: Results of the proposed approach and the baseline on IAM images with additive Poisson noise  $\mathcal{P}(\lambda)$ .

	No noise		$\mathcal{P}(10)$		$\mathcal{P}(20)$		$\mathcal{P}(30)$	
	CER	WER	CER	WER	CER	WER	CER	WER
<b>Full-DefConv</b>	4.6	14.8	4.6	14.8	4.6	15.1	4.7	15.1
<b>Shi et al. [14]</b>	5.3	17.5	5.3	17.4	5.4	17.7	5.5	18.2

TABLE VII: Results of the proposed approach and the baseline on RIMES images with additive Poisson noise  $\mathcal{P}(\lambda)$ .

## V. CONCLUSION

In this paper, we showed that deformable convolutions are more suitable than standard convolutions for the task of HTR. The performance of the proposed approach has been evaluated on benchmark datasets of modern English and French handwritten text. Arguably, these could be improved by adding a language model for each specific text language.

The ability to adapt to highly distorted handwritten strokes makes DefConv-based HTR models promising for dealing with free-layout historic manuscripts. The robustness to noise is another advantage of using DefConvs.

These aspects will be explored in future work, by using both benchmark datasets of historic documents, and a new dataset we are currently collecting and aim to make available, which features manuscripts of XVI, XVII and XVIII Century Italian Historians and Writers, including letters by Lodovico Antonio Muratori and Giacomo Leopardi.

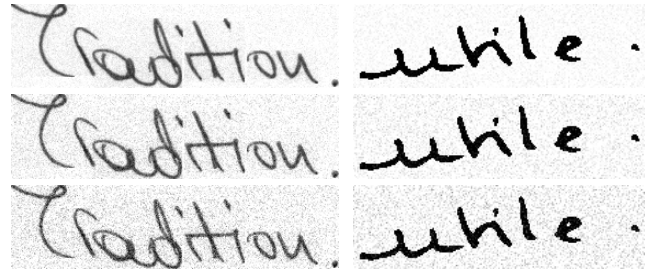


Fig. 7: Example images from the test set of the IAM dataset (left) and the RIMES dataset (right) when it is added White Gaussian Noise with variance ranging from 10 (top) to 30 (bottom).

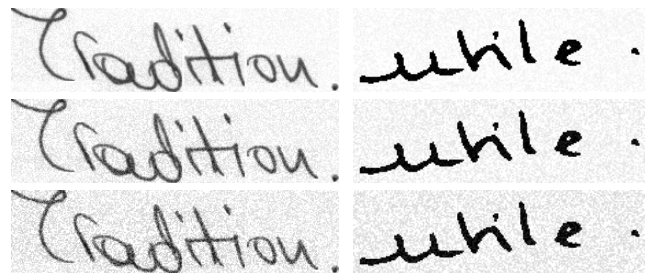


Fig. 8: Example images from the test set of the IAM dataset (left) and the RIMES dataset (right) when it is added Poisson Noise with variance ranging from 10 (top) to 30 (bottom).

## VI. ACKNOWLEDGMENTS

This work is financially supported by the Fondazione Cassa di Risparmio di Modena in the ambit of the project “AI for Digital Humanities” (Prot. n. 505.18.8b del 18/10/2018 - Pratica Sime n. 2018.0390).



## REFERENCES

- [1] J. Puigcerver, “Are multidimensional recurrent layers really necessary for handwritten text recognition?” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 67–72.
- [2] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable convolutional networks,” in *CVPR*, 2017, pp. 764–773.
- [3] U.-V. Marti and H. Bunke, “Handwritten sentence recognition,” in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 3. IEEE, 2000, pp. 463–466.
- [4] E. Krevat and E. Cuzzillo, “Improving off-line handwritten character recognition with hidden markov models,” *IEEE Trans. PAMI*, vol. 33, 2006.
- [5] T. Plötz and G. A. Fink, “Markov models for offline handwriting recognition: a survey,” *IJDAR*, vol. 12, no. 4, p. 269, 2009.
- [6] L. Quirós, V. Bosch, L. Serrano, A. H. Toselli, and E. Vidal, “From hmms to rnns: computer-assisted transcription of a handwritten notarial records collection,” in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 116–121.
- [7] N. D. Cilia, C. De Stefano, F. Fontanella, and A. S. di Freca, “A ranking-based feature selection approach for handwritten character recognition,” *Pattern Recognition Letters*, vol. 121, pp. 77–86, 2019.
- [8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [9] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, “Spatial transformer networks,” in *NeurIPS*, 2015, pp. 2017–2025.
- [10] T. Clanuwat, A. Lamb, and A. Kitamoto, “Kuronet: Pre-modern japanese kuzushiji character recognition with deep learning,” *arXiv preprint arXiv:1910.09433*, 2019.
- [11] A. Graves and J. Schmidhuber, “Offline handwriting recognition with multidimensional recurrent neural networks,” in *NeurIPS*, 2009, pp. 545–552.
- [12] A. K. Bhunia, A. Das, A. K. Bhunia, P. S. R. Kishore, and P. P. Roy, “Handwriting recognition in low-resource scripts using adversarial learning,” in *CVPR*, 2019, pp. 4767–4776.
- [13] F. P. Such, D. Peri, F. Brockler, H. Paul, and R. Ptucha, “Fully convolutional networks for handwriting recognition,” in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 86–91.
- [14] 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 Trans. PAMI*, vol. 39, no. 11, pp. 2298–2304, 2016.
- [15] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, “Dropout improves recurrent neural networks for handwriting recognition,” in *2014 14th International Conference on Frontiers in Handwriting Recognition*. IEEE, 2014, pp. 285–290.
- [16] P. Voigtlaender, P. Doetsch, and H. Ney, “Handwriting recognition with large multidimensional long short-term memory recurrent neural networks,” in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2016, pp. 228–233.
- [17] T. Bluche and R. Messina, “Gated convolutional recurrent neural networks for multilingual handwriting recognition,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 646–651.
- [18] A. Chowdhury and L. Vig, “An efficient end-to-end neural model for handwritten text recognition,” *arXiv preprint arXiv:1807.07965*, 2018.
- [19] B. Moysset, C. Kermorvant, and C. Wolf, “Full-page text recognition: Learning where to start and when to stop,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 871–876.
- [20] T. Bluche, J. Louradour, and R. Messina, “Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1050–1055.
- [21] T. Bluche, “Joint line segmentation and transcription for end-to-end handwritten paragraph recognition,” in *NeurIPS*, 2016, pp. 838–846.
- [22] M. Alberti, L. Vöglin, V. Pondenkandath, M. Seuret, R. Ingold, and M. Liwicki, “Labeling, cutting, grouping: an efficient text line segmentation method for medieval manuscripts,” *arXiv preprint arXiv:1906.11894*, 2019.
- [23] V. B. Campos, V. R. Gómez, A. H. T. Rossi, and E. V. Ruiz, “Text line extraction based on distance map features and dynamic programming,” in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 357–362.
- [24] V. Khare, P. Shivakumara, B. Navya, G. Swetha, D. Guru, U. Pal, and T. Lu, “Weighted-gradient features for handwritten line segmentation,” in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 3651–3656.
- [25] R. R. Nair, B. U. Kota, I. Nwogu, and V. Govindaraju, “Segmentation of highly unstructured handwritten documents using a neural network technique,” in *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 2016, pp. 1291–1296.
- [26] N. Arvanitopoulos and S. Stsitrunk, “Seam carving for text line extraction on color and grayscale historical manuscripts,” in *2014 14th International Conference on Frontiers in Handwriting Recognition*. IEEE, 2014, pp. 726–731.
- [27] C. Wigington, C. Tensmeyer, B. Davis, W. Barrett, B. Price, and S. Cohen, “Start, follow, read: End-to-end full-page handwriting recognition,” in *ECCV*, 2018, pp. 367–383.
- [28] V. Jayasundara, S. Jayasekara, H. Jayasekara, J. Rajasegaran, S. Seneviratne, and R. Rodrigo, “Textcaps: Handwritten character recognition with very small datasets,” in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 254–262.
- [29] C. Wigington, S. Stewart, B. Davis, B. Barrett, B. Price, and S. Cohen, “Data augmentation for recognition of handwritten words and lines using a cnn-lstm network,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 639–645.
- [30] A. Poznanski and L. Wolf, “Cnn-n-gram for handwriting word recognition,” in *CVPR*, 2016, pp. 2305–2314.
- [31] C.-L. Liu and K. Marukawa, “Pseudo two-dimensional shape normalization methods for handwritten chinese character recognition,” *Pattern Recognition*, vol. 38, no. 12, pp. 2242–2255, 2005.
- [32] Z. Zhong, X.-Y. Zhang, F. Yin, and C.-L. Liu, “Handwritten chinese character recognition with spatial transformer and deep residual networks,” in *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 2016, pp. 3440–3445.
- [33] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [34] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *arXiv preprint arXiv:1312.6034*, 2013.
- [35] U.-V. Marti and H. Bunke, “The iam-database: an english sentence database for offline handwriting recognition,” *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, 2002.
- [36] E. Augustin, M. Carré, E. Grosicki, J.-M. Brodin, E. Geoffrois, and F. Preteux, “RIMES evaluation campaign for handwritten mail processing,” in *International Workshop on Frontiers in Handwriting Recognition (IWFHR’06)*, La Baule, France, Oct. 2006, pp. 231–235. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-00271963>
- [37] S. Johansson, G. N. Leech, and H. Goodluck, *Manual of information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital computer*. Department of English, University of Oslo, 1978.
- [38] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.