OrigamiNet: Weakly-Supervised, Segmentation-Free, One-Step, Full Page Text Recognition by learning to unfold

Mohamed Yousef
Intuition Machines, Inc.
myb@imachines.com

Tom E. Bishop
Intuition Machines, Inc.
tom@imachines.com

Abstract

Text recognition is a major computer vision task with a big set of associated challenges. One of those traditional challenges is the coupled nature of text recognition and segmentation. This problem has been progressively solved over the past decades, going from segmentation based recognition to segmentation free approaches, which proved more accurate and much cheaper to annotate data for. We take a step from segmentation-free single line recognition towards segmentation-free multi-line / full page recognition. We propose a novel and simple neural network module, termed ***OrigamiNet****, that can augment any CTC-trained, fully convolutional single line text recognizer, to convert it into a multi-line version by providing the model with enough spatial capacity to be able to properly collapse a 2D input signal into 1D without losing information. Such modified networks can be trained using exactly their same simple original procedure, and using only* ***unsegmented*** *image and text pairs. We carry out a set of interpretability experiments that show that our trained models learn an accurate implicit line segmentation. We achieve state-of-the-art character error rate on both IAM & ICDAR 2017 HTR benchmarks for handwriting recognition, surpassing all other methods in the literature. On IAM we even surpass single line methods that use accurate localization information during training. Our code is available online at* https://github.com/IntuitionMachines/OrigamiNet.

1. Introduction

The ubiquity of text has made the automation of the processing of its various visual forms, an ever-increasing necessity. Over the years, one of the main driving themes for error rate reduction in text recognition systems has been reducing explicit segmentation proposals in favor of increasing full sequence recognition. In full sequence models, the recognition system learns to both simultaneously segment / align and recognize / classify an image representing a sequence of observations (i.e. characters). This trend progressed from the first systems that tried to segment each character alone then classify the character's image [6], to segmentation free approaches that tried to recognize all the characters in a word, without requiring / performing any explicit segmentation [21]. Today, state-of-the-art text recognition systems work on a whole input line image without requiring any prior explicit character / word segmentation [35, 18]. This removes the requirement for providing character localization annotations as part of ground-truth transcription. Also the recognition accuracy relies only on automatic line segmentation, a much easier process than automatic character segmentation.

Requirement	[4]	[3]	[30]	[7, 33, 19]	Ours
Full-page image	✓	✓	✓	✓	✓
Full-page text GT	✓	✓	✓	✓	✓
Seg. line images	✗	✗	✗	✓	✗
Seg. transcription	✗	✗	✗	✓	✗
Pre-train on seg. data	✓	✓	✓	✗	✗
Special curriculum	✓	✓	✗	✗	✗
# Iterations / image	500	10	10	10	1

Table 1: Comparison of what data is required to train a full page recognizer between various prior works and our proposed method. We can see that our method is the only that truly works at page level without requiring any segmented data at any stage. *# Iterations / image* is the average number of iterations required to transcribe a full paragraph image from the IAM dataset; we can note that while all other methods require multiple iterations per image (to recognize each segmented character or line), our method performs only one pass over the input full paragraph image.

However, line segmentation is still an error-prone process and can cause great deterioration in the performance of today's text recognition systems. This is especially true for documents with hard to segment text-lines such as handwritten documents [10, 24], with warped lines, uneven interline spacing, touching lines, and torn pages.

The main previous works that tried to address the problem of weakly supervised multi-line recognition were [3, 4, 30]. Besides these methods, other methods that work on full page recognition require the localization ground-truth of text lines during training. A detailed comparison between the training data required by our proposed method vs. other methods in literature is presented in Table 1.

In this work, we present a simple and novel neural network sub-module, termed OrigamiNet, that can be added to any existing convolutional neural network (CNN) text-line recognizer to convert it to a full page recognizer. It can transcribe full text pages in a weakly supervised manner without being given any localization ground-truth (either visual in the images or textual in the transcriptions) during training, and without performing any explicit segmentation. In contrast to previous work, this is done very efficiently using feed-forward connections only (no recurrent connections), essentially, in a single network forward pass.

Our main intuition in this work is, instead of the traditional two-step framework that first segments then recognizes extracted segments, to propose a novel integrated approach for learning to simultaneously implicitly segment and recognize. This works by learning a representation transformation that *transforms the input into a representation where both segmentation and recognition is trivial*.

We implicitly unfold an input multi-line image into a single line image (i.e. from a 2D arrangement of characters to 1D), where all lines in the original image are stitched together into one long line, so no text-line segmentation is actually needed. Both segmentation and recognition are done in the same single step (single network forward pass) instead of being carried out iteratively (on each line), and thus all computations are shared between recognition and implicit segmentation, and the whole process is a lot faster.

The main ingredients to achieving this are: Using the idea of a spatial bottleneck followed by up-sampling, used widely in pixel-wise prediction tasks (e.g. [16, 23]); and using the CTC loss function [11] which strongly induces / encourages a linear 1D target. We construct a simple neural network sub-module that applies these novel ideas, and demonstrate both its effectiveness and generality by attaching it to a number of state-of-the-art text recognition neural network architectures. We show that it can successfully convert them from single line into multi-line text recognizers with exactly the same training procedure (i.e. without resorting to complex and fragile training recipes, like a special training curriculum or special pre-training strategies).

On the challenging ICDAR 2017 HTR [24] full page benchmark we achieve state-of-the-art Character Error Rate (CER) without any localization data. On full paragraphs of the IAM [17] dataset, we were able to achieve state-of-the-art CER surpassing models that work on carefully pre-segmented text-lines, without using any localization information during training or testing.

To summarize, we address the problem of weakly supervised full-page text recognition. In particular, we make the following contributions:

- We conceptually propose a new approach for weakly-supervised simultaneous object segmentation and recognition, and apply it to text.
- We propose a simple and generic neural network sub-module that can be added to any CNN-based text line recognizer to convert it into a multi-line recognizer that utilizes the same simple training procedure.
- We carry an extensive set of experiments on a number of state-of-the-art text recognizers that demonstrate our claims. The resultant architectures demonstrate state-of-the art performance on ICDAR2017 HTR and the full paragraph IAM datasets.

2. Related Work

There is not much prior work in the literature regarding full page recognition. Segmentation-free multi-line recognition has been mainly considered in [3, 4]. The idea of both is using selective attention to focus only on a specific part of the input image, either characters in [4] or lines in [3]. These works have two major drawbacks. First, both are difficult to train, and need to pre-train their encoder sub-network on single-line images before training on multi-line versions, which defeats the objective of the task. Second, though [3] is much faster than [4], both are very slow compared to current methods that work on segmented text lines.

Besides these two segmentation-free methods, other methods that work on full page recognition either require the localization ground-truth of text lines for all [5, 7, 19] or part [33] of the training data to train either a separate network or a sub-module (of a large, multi-task network) for text-line localization. Also, all these methods require line breaks to be annotated on all the provided textual ground-truth transcriptions (i.e. text lines must be segmented both visually in the image and textually in the transcription). [30] presented the idea of adapting [33] in a weakly supervised manner without requiring line breaks in the transcription by setting the alignment between the predicted line transcriptions and the ground truth as a combinatorial optimization problem, and greedily solving it. However [30] still requires the same pre-training as [33] and performs worse.

3. Methodology

Figure 1 presents the core idea of our proposed OrigamiNet module, and how it can be attached to any fully convolutional text recognizer. Both before and after versions are shown for easy comparison.

The Connectionist Temporal Classification (CTC) loss function allows the training of neural text recognizers on

unsegmented inputs by considering all possible alignments between two 1D sequences. The sequence of predictions produced by the network is denoted P, and the sequence of labels associated with the input image L, where $|L| < |P|$. The strict requirement of having P as a 1D sequence, introduces a problem, given that the original input signal (the image I) is a 2D signal. This problem has typically been dealt with by unfolding the 2D signal into 1D, using a simple reduction operation (e.g. summation) along one of the dimensions (usually the vertical one), giving:

$$P_i = \sum_{j=1}^{H} F(I_{i,j}) \tag{1}$$

Where F is a learned 2D representation transformation. This is the paradigm shown in Fig. 1a. As noted in [3, 4] this simple, *blind* collapse from 2D to 1D gives equal importance / contribution (and therefore gradients) to all the rows of the 2D input feature-map $F(I)$, and thus prevents the recognition of any 2D arrangement of characters in the input image. If two characters cover the same columns, only one can be possibly recognized after the collapse operation.

To tackle this problem, i.e. satisfy the 1D input requirement of CTC without sacrificing the ability of recognizing 2D arrangements of characters, we propose the idea of learning the proper 2D→1D unfolding through a CNN, motivated by the success of CNNs in pixel-wise prediction and image-to-image translation tasks.

The main idea of our work (presented in Fig. 1b) is augmenting the traditional paradigm with a series of up-scaling operations that transforms the input feature-map into the shape of a single line, that is long enough to hold all the lines (2D character arrangements) from the input image. Up-scaling operations are followed by convolutional computational blocks as our learned resize operations (as done by many researchers, e.g. [8]). The changed direction of up-scaling encourages each line of the input image to be mapped into a distinct part of the output vertical dimension.

After such changes, we proceed with the traditional paradigm as-is, perform the simple sum reduction (Eq. 1) along the vertical dimension w of the resulting line (which is perpendicular to the original input multi-line image's vertical dimension). The model is trained with CTC.

Moreover, we argue that the main bottleneck preventing all previous works from learning proper 2D→1D mappings directly as we do, is *spatial* constraints (i.e. not overall capacity or architectural constraints). Providing enough spatial capacity to the model allows it to easily learn such transformations (even for simple limited capacity models, as we will show in the experiments section). Given the spatial capacity and the strong linear prior induced by CTC, the model is able to learn strong 2D→1D unfolding function with the same simple training procedure used for training single line recognizers, and without any special pre-training or curriculum applied to any sub-module of the network (both of which are used exclusively in the literature).

One natural question here is how to choose the final line length L_2 (see definition in Fig. 1b)? To gather space for the whole paragraph / page, L_2 must be at least as long as the largest number of characters in any transcription in the training set. Longer still is better, given that (i) CTC needs to insert blanks to separate repeated labels; (ii) characters vary greatly in spatial extent, and mapping each to multiple target frames in the final vector is an easier task than transforming to exactly one frame.

4. Experiments

We carry out an extensive set of experiments to answer the following set of questions:

- Does the module actually work as expected?
- Is it tied to a specific CNN architecture?
- Is it tied to a specific model capacity?
- How does final spatial size affect model performance?

4.1. Implementation Details

All experiments use an initial learning rate of 0.01, exponentially decayed to 0.001 over 9×10^4 batches. We implement in PyTorch [20], with the Adam [15] optimizer.

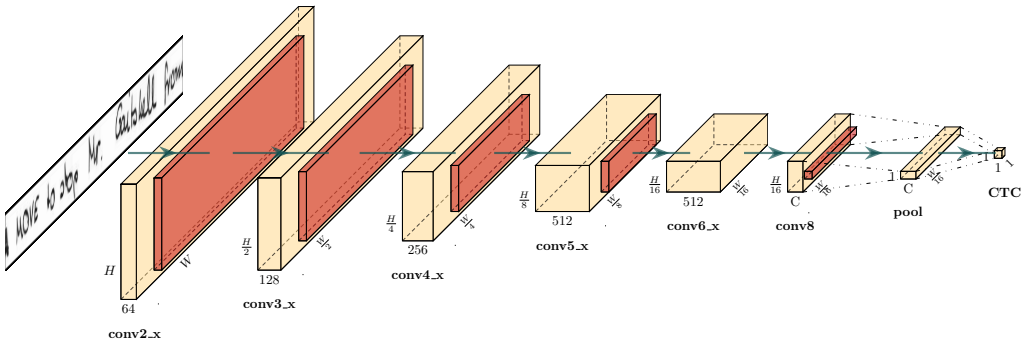
4.2. Datasets

IAM [17] (modern English) is a famous offline handwriting benchmark dataset. It is composed of 1539 scanned text pages handwritten by 657 different writers, corresponding to English texts extracted from the LOB corpus [14]. IAM has 747 documents (6,482 lines) in the training set, 116 documents (976 lines) in the validation set and 336 documents (2,915 lines) in the test set.

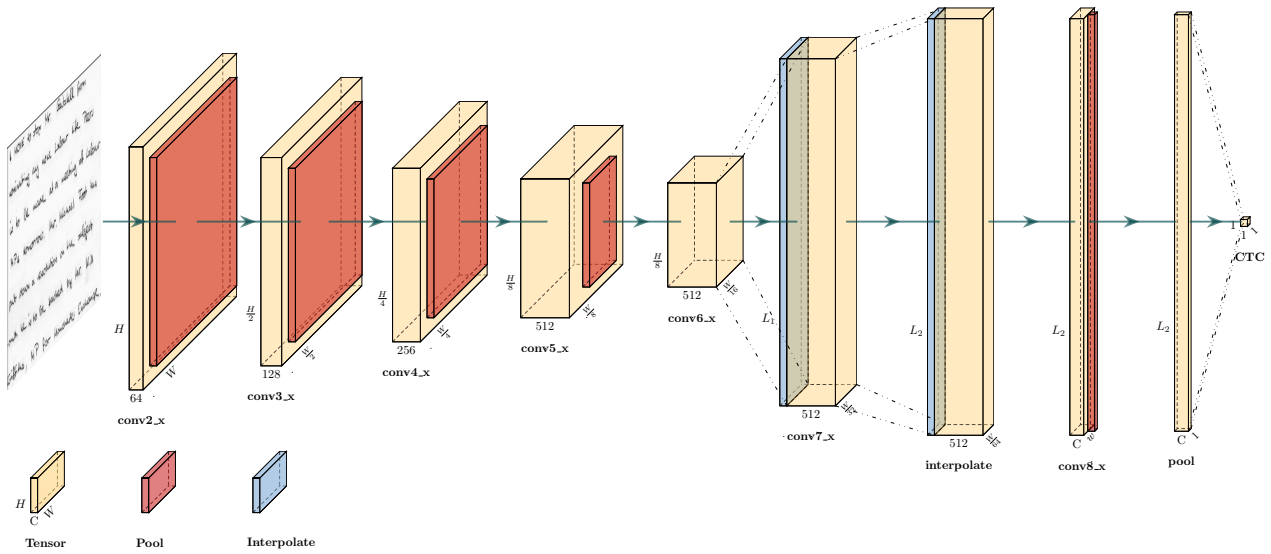
The ICDAR2017 full page HTR competition [24] consists of two training sets. The first contains 50 fully annotated images with line-level localization and transcription ground-truth. The second set contains 10,000 images with only transcriptions (with annotated line breaks). Most of the dataset was taken from the Alfred Escher Letter Collection (AEC) which is written in German but it also has pages in French and Italian. In all our experiments on this dataset, *we don't make any use of* either the 50-page training set or the annotated line-breaks on the 10,000-page training set

4.3. CNN Backbones

To emphasize the generality of our proposed module, we evaluate it on a number of popular CNN architectures that achieved strong performance in the text recognition literature. Inspired by the benchmark work [2], we evaluate VGG and ResNet-26 (the specific variants explored in [2]), as well as deeper and much more expressive variants (ResNet-66 and ResNet-74). We also evaluate a newly proposed



(a) A generic four stage fully convolutional single line recognizer, input is a singe line image, training is done using the CTC loss function. Backbone CNN can be any of the ones presented in Table 2. Input gets progressively down-sampled, then converted into 1D by average pooling along the vertical dimension right before the loss calculation. (Figures created via PlotNeuralNet [13])



(b) Here we convert the fully convolutional single-line recognizer into an OrigamiNet multi-line recognizer; comparing the two figures shows that the main change introduced is up-scaling *vertically* in two stages, and at the same time, down-scaling horizontally. We obtain a feature-map that is *tall* and narrow (the shape of one very long vertical line, length L_2). After that we proceed exactly as above, average pooling over the short dimension, w (of the new line *not* the original image) then using the CTC loss function to drive the training process.

Figure 1: Converting a fully-convolutional single line recognizer into a multi-line recognizer using our OrigamiNet module.

gated, fully convolutional architecture for text recognition [35], named Gated Text Recognizer (GTR). The detailed structure of the CNN backbones we evaluate our proposed model on is presented in Table 2. More details on the basic building blocks of these architectures can be found in their respective papers, VGG [25], ResNet [12], and GTR [35].

4.4. Final Length, L_2

For IAM, the final length should be at least 625, since the longest paragraph in the training set contains 624 characters. We have two questions here: what value can balance running time and recognition accuracy? And how does the relation between L_1 and L_2 affect the final CER?

Table 3 presents some experiments on this. First, we can see that generally, even a very simple model like VGG can successfully learn to recognise multiple lines (at a relatively bad CER = 30%) at various configurations, yet, the deeper ResNet-26 achieves a much better performance on the task reaching 7.2%. Second, it is evident that wider generally gives better performance (but at diminishing returns), which is evident for VGG more than ResNet-26. We see that for reasonable values (>800) the network is fairly robust to the choice of L_2. We can also note that both L_1 and L_2 should be relatively close to each other.

part	layer name	output size	ResNet-26	ResNet-66	ResNet-74	VGG	GTR-8	GTR-12
Encoder	Input	$H \times W$						
	ln1	$H \times W$	static layer normalization					
	conv1	$H \times W$	7×7, 64				13×13, 16	
	conv2_x	$\frac{H}{2} \times \frac{W}{2}$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	$\begin{bmatrix} 3\times3, 64 \end{bmatrix}\times1$	$[GateBlock(512)]\times1$	$[GateBlock(512)]\times1$
			2×2 max pool, stride 2					
	conv3_x	$\frac{H}{4} \times \frac{W}{4}$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times6$	$\begin{bmatrix} 3\times3, 128 \end{bmatrix}\times1$	$[GateBlock(512)]\times1$	$[GateBlock(512)]\times1$
			2×2 max pool, stride 2					
	conv4_x	$\frac{H}{8} \times \frac{W}{8}$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times5$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times25$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times25$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times1$	$[GateBlock(512)]\times1$	$[GateBlock(512)]\times2$
			2×2 max pool, stride 2					
	conv5_x	$\frac{H}{8} \times \frac{W}{16}$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	$[GateBlock(1024)]\times1$	$[GateBlock(1024)]\times3$
			2×2 max pool, stride 1×2					
	conv6_x	$\frac{H}{8} \times \frac{W}{16}$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	$[GateBlock(1024)]\times3$	$[GateBlock(1024)]\times4$
Decoder			interpolate bilinearly to $L_1 \times \frac{W}{32}$					
	conv7_x	$L_1 \times \frac{W}{32}$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$	$[GateBlock(512)]\times1$	$[GateBlock(512)]\times1$
		$L_2 \times \frac{W}{64}$	interpolate bilinearly to $L_2 \times \frac{W}{64}$					
	conv8	$L_2 \times w$	1×1, C					
		L_2	average pool over short dimension w					
	ln2	L_2	static layer normalization					
		1	CTC					
# Parameters $\times10^6$			38.2	61.9	63.05	10.6	9.9	16.4

Table 2: Architectural details of our evaluated CNN backbones (*Encoder* part), and how our module (*Decoder* part) is attached to them. The table tries to abstract the architectures to their most common details. Although there is subtle difference in the components of the basic building block (in brackets []) of every architecture, the overall organization of the network, and how our module fits, is the same.

4.5. Final Width

Does the final shape need to have the largest possible aspect ratio? How would the final width, w (shorter output dimension) affect the learning system? Table 4 presents experiments using VGG and ResNet-26 on this regard. It is clear that a large value like 62 deteriorates training significantly for ResNet-26, but small and medium values (<31) are comparable in performance. On the other hand, a model with limited receptive field and complexity like VGG can generally make a lot of use from the added width.

4.6. End-to-end Layer Normalization

The idea of using parameter-less layer normalization as the first and last layer of a model was proposed in [35], and shown to increase performance and facilitate optimization. The same idea was very effective for our module, as initially some deep models that converged for single line recognition completely diverged here. This is most probably due to the large number of time-steps CTC works on for our case.

As can be seen in Table 5, end-to-end layer normalization can bring significant increases in accuracy for models that already worked well; more importantly, it makes it possible to train very deep models that were constantly diverging before, leading to state-of-art performance on the task.

4.7. Hard-to-segment text-lines

Due to the way IAM was collected [17], its lines are generally easy to segment. To study how our model would handle harder cases, we carried out two separate experiments, artificially modifying IAM to produce new variants with hard-to-segment lines. Firstly, interline spacing is massively reduced via seam carving [1], resizing to 50% height, creating heavily touching text lines, Fig. 4(b). GTR-12 achieved 6.5% CER on this dataset. Secondly, each paragraph has random projective (rotating/resizing lines), and random elastic transforms (like [32] but at the page level) applied, creating wave-like non-straight lines, Fig. 4(c). GTR-12 achieved 6.2% CER on this dataset.

4.8. Comparison to state-of-the-art

For all the previous experiments, IAM paragraph images were scaled down to 500 × 500 pixels before training, and although we were already achieving state-of-the-art results, we wanted to explore whether we can break even with single line recognizers. As shown in Table 6, by increasing image / model sizes, we were for the first time able to exceed the performance of state-of-the-art single line recognizers using a segmentation free full page recognizer that trains without any visual or textual localization groundtruth. Note that we don't include in the comparison methods that use additional data, either in the form of training images as in [34, 9] or language modeling as in [31].

For the ICDAR2017 HTR dataset we follow [30] and report CER on the validation set proposed in [33] (the last 1000 pages of the 10,000 image training set), as the evaluation server doesn't provide CER or other character based metrics. Results are in Table 7. Note that both [33, 30] report results using CER normalized by GT length (*nCER* in the table). We used author released pre-trained models from [33] to compute their results without a language model. It is very evident our method can get far superior performance using weaker training signals.

4.9. Model Interpretability

Here we consider an important question: what does the model actually learn? We can see that the model works well in practice and we have a hypothesis of what it *might* be doing, but it would very interesting if we can have a peek at how our model is able to make its predictions.

To gain an understanding of what parts of the input biases the model towards a specific prediction, we utilize the framework of Path-Integrated Gradients [29] ensembled using SmoothGrad [26]. Note that unlike typical classification tasks, we predict L_2 labels per image. Of those we discard blanks and repeated consecutive labels (in CTC, representing continuation of the same state; we found their attribution maps to be global and uninformative for these purposes).

For integrated gradients (IG), we change the baseline to use an empty white image to designate no-signal, rather than an empty black one (which would be an all-signal image in our case) - as our data is black text over a white background. Using white baselines produced much sharper attribution maps than black ones, showing how sensitive IG is to the choice of the baseline (studied more in [28]). We used 50 steps to approximate the integral in our tests.

Standard SmoothGrad produces attribution maps that are very noisy (see [27]), but the SmoothGrad-Squared variant often suppresses most of the signal (a direct consequence of squaring fractions). After analysing the results of both, we suggest the root cause of SmoothGrad problems is averaging positive and negative signals together. The squaring in SmoothGrad-Squared solves this problem, but at the cost of suppressing some important parts of the signal. So we propose *SmoothGrad-Abs*, which simply averages the absolute value of the attribution maps. SmoothGrad-Abs strikes a good balance between SmoothGrad and SmoothGrad-Squared. For our experiments, we used 5 noisy images.

Fig. 2 shows the attribution maps of a single random character from each line of the input image (computed from the attribution of the corresponding output neuron in the 1D prediction map fed to CTC). We see that the model does indeed implicitly learn good character-level localization from the input 2D image to the output 1D prediction map.

Fig. 3 provides a holistic view that gathers all the maps into one image. We took the one-character attribution map from the previous step, apply Otsu thresholding to it (to keep only the most important parts) then add a marker at the position of the center of mass of the resulting binary image. The marker is colored according to the transcription text line it belongs to. As can be seen, the result represents a very good implicit line segmentation of the original input.

4.10. Limitations

We also trained our network on a variant of IAM with horizontally flipped images and line-level flipped groundtruth transcription, where it managed to achieve

Final length (L_2)	700	800	950	1100	1500
First stage length L_1 = 450					
VGG	43.14	34.32	34.55	34.55	30.34
ResNet-26	8.121	7.675	7.602	7.238	7.449
First stage length L_1 = 225					
VGG	37.5	39.6	37.5	36.46	34.75

Table 3: The IAM test set CER of VGG and ResNet-26 for various values of L_1 and L_2.

Final width	62	31	15	8	3
VGG	25.98	17.41	37.4	34.55	24.21
ResNet-26	19.9	9.128	8.64	7.238	8.34

Table 4: The IAM test set CER of VGG and ResNet-26 for various final widths. Here $L_1 = 450$ and $L_2 = 1100$

LN	VGG	ResNet-26	ResNet-66	ResNet-74	GTR-8
w/o	51.37	10.03	8.925	76.9	72.4
w	34.55	7.238	6.373	6.128	5.639

Table 5: The IAM test set CER for various models, with and without layer-normalization

Method	Input Scale	Test CER(%)	Remarks
Single-line methods			
[22]	128 × W	5.8	CNN+BLSTM+CTC
[18]	64 × W	5.24	Seq2Seq (CNN+BLSTM encoder)
[35]	32 × W	4.9	CNN+CTC
Multi-line methods			
[4]	150 dpi	16.2	Requires pre-training the encoder (MDLSTM) on segmented text lines
[3]	150 dpi	10.1	
[3]	300 dpi	7.9	
[5]	150 dpi	15.6	Requires fully segmented training data
[7]		8.5	
[33]		6.4	Requires full line-break annotation and partial visual localization
ResNet-74 OrigamiNet	500 × 500	6.1	
GTR-8 OrigamiNet	500 × 500	5.6	
GTR-8 OrigamiNet	750 × 750	5.5	
GTR-12 OrigamiNet	750 × 750	**4.7**	

Table 6: Comparison with the state-of-the-art on the IAM paragraph images, best result is highlighted.

Figure 2: Results of the interpretability experiment. For each of these 8 images (from left-right, top-down) we show the attribution heat-map for a single character output (for each line in the image) overlaid over a faint version of the original input image. The randomly chosen character is highlighted in green in the transcription below the image.

nearly the same CER. This verifies that the proposed method is robust and can learn the reading order from data.

While the proposed method works well on paragraphs or full pages of text, learning the flow of multiple columns is not addressed directly. However, given that region / paragraph segmentation is trivial compared to text line segmen-

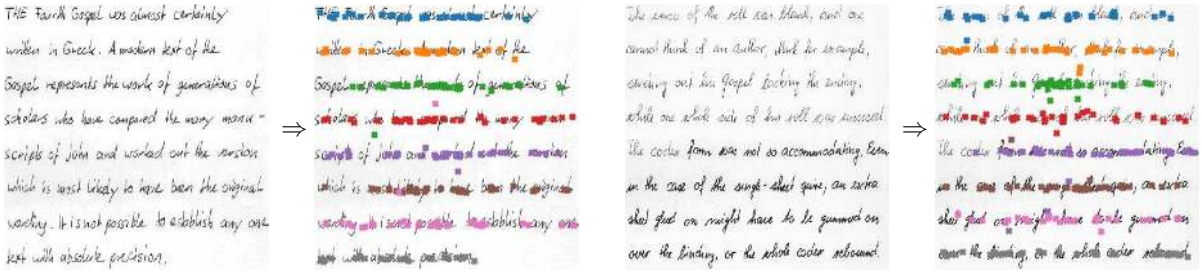
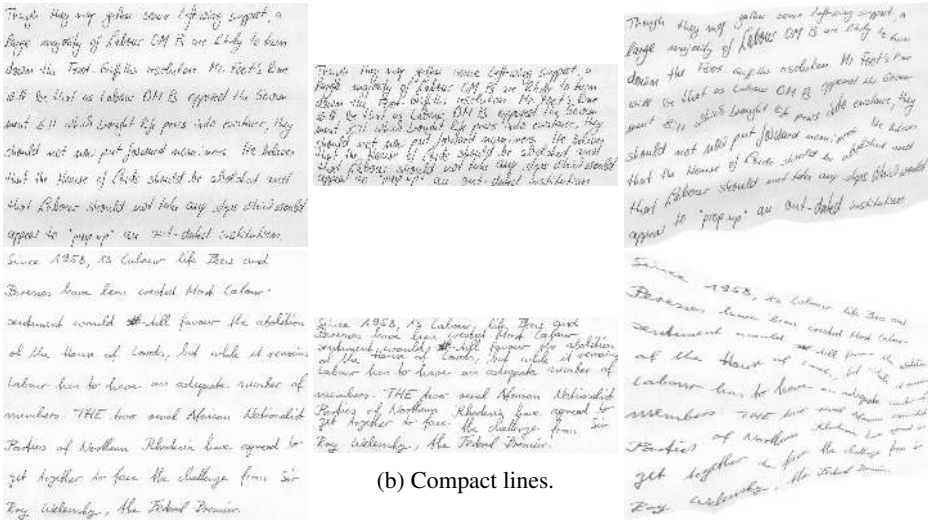


Figure 3: The first and third columns represent two input images. The second and fourth columns are the corresponding color coded scatter plot, where, for each character, the position of the center of mass for the attribution map associated with that character is marked. Character markers belonging to the same line are given the same color. We can see that the model learns a very good implicit segmentation of the input image into lines without any localization signal.



(a) Original Image.

(b) Compact lines.

(c) Rotated and warped.

Figure 4: Synthetic distortions applied to the IAM dataset to study the how our model handles hard to segment text-lines. (a) original paragraph image. (b) touching text-lines. (c) rotated and wavy text-lines

Method	CER	nCER	linebreaks	Pre-train
SFR [30]	8.18	8.68	✓	50 fully annotated pgs
SFR-align [33]	-	11.05	✗	
GTR-12 OrigamiNet	**6.80**	**5.87**	✗	-

Table 7: Comparison on ICDAR2017 HTR, best result is highlighted. *nCER* is CER normalized by GT length. *linebreaks* indicates their presence or removal from the GT.

tation we think this is not a serious practical limitation.

5. Conclusion

In this paper we tackled the problem of multi-line / full page text recognition without any visual or textual localization ground-truth provided to the model during training. We proposed a simple neural network sub-module, OrigamiNet, that can be added to any existing fully convolutional single-line recognizer and convert it into a multi-line recognizer by providing the model with enough spatial capacity to be able to properly unfold 2D input signals into 1D without losing information.

We conducted an extensive set of experiments on the IAM handwriting dataset to show the applicability and generality of our proposed module. We achieve state-of-the-art CER on the ICDAR2017 HTR and IAM datasets surpassing models that explicitly made use of line segmentation information during training. We then concluded with a set of interpretability experiments to investigate what the model actually learns and demonstrated its implicit ability to localize characters on each line.

References

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. In *ACM SIGGRAPH 2007 papers*, pages 10–es. 2007. 5

[2] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. *arXiv preprint arXiv:1904.01906*, 2019. 3

[3] T. Bluche. Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. In *Advances in Neural Information Processing Systems*, pages 838–846, 2016. 1, 2, 3, 7

[4] 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)*, volume 1, pages 1050–1055. IEEE, 2017. 1, 2, 3, 7

[5] M. Carbonell, J. mas romeu, M. Villegas, A. Fornés, and J. Lladós. End-to-end handwritten text detection and transcription in full pages. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, 07 2019. 2, 7

[6] R. G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 18(7):690–706, 1996. 1

[7] J. Chung and T. Delteil. A computationally efficient pipeline approach to full page offline handwritten text recognition. *arXiv preprint arXiv:1910.00663*, 2019. 1, 2, 7

[8] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 3

[9] K. Dutta, P. Krishnan, M. Mathew, and C. Jawahar. Improving cnn-rnn hybrid networks for handwriting recognition. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 80–85. IEEE, 2018. 6

[10] B. Gatos, G. Louloudis, T. Causer, K. Grint, V. Romero, J. A. Sánchez, A. H. Toselli, and E. Vidal. Ground-truth production in the transcriptorium project. In *2014 11th IAPR International Workshop on Document Analysis Systems*, pages 237–241. IEEE, 2014. 1

[11] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006. 2

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[13] H. Iqbal. Harisiqbal88/plotneuralnet v1.0.0, Dec. 2018. 4

[14] S. Johansson. The lob corpus of british english texts: Presentation and comments. 1980. 3

[15] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. Dec. 2014. 3

[16] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2

[17] U.-V. Marti and H. Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46, 2002. 2, 3, 5

[18] J. Michael, R. Labahn, T. Grüning, and J. Zöllner. Evaluating sequence-to-sequence models for handwritten text recognition. *arXiv preprint arXiv:1903.07377*, 2019. 1, 7

[19] B. Moysset, C. Kermorvant, and C. Wolf. Learning to detect, localize and recognize many text objects in document images from few examples. *International Journal on Document Analysis and Recognition (IJDAR)*, 21(3):161–175, 2018. 1, 2

[20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 3

[21] T. Plötz and G. A. Fink. Markov models for offline handwriting recognition: a survey. *International Journal on Document Analysis and Recognition (IJDAR)*, 12(4):269, 2009. 1

[22] J. Puigcerver. Are multidimensional recurrent layers really necessary for handwritten text recognition? In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 67–72. IEEE, 2017. 7

[23] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[24] J. A. Sanchez, V. Romero, A. H. Toselli, M. Villegas, and E. Vidal. Icdar2017 competition on handwritten text recognition on the read dataset. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 1383–1388. IEEE, 2017. 1, 2, 3

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 4

[26] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 6

[27] I. Sturm, S. Lapuschkin, W. Samek, and K.-R. Müller. Interpretable deep neural networks for single-trial eeg classification. *Journal of neuroscience methods*, 274:141–145, 2016. 6

[28] P. Sturmfels, S. Lundberg, and S.-I. Lee. Visualizing the impact of feature attribution baselines. *Distill*, 2020. https://distill.pub/2020/attribution-baselines. 6

[29] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org, 2017. 6

[30] C. Tensmeyer and C. Wigington. Training full-page handwritten text recognition models without annotated line breaks. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1–8. IEEE, 2019. 1, 2, 6, 8

[31] 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)*, pages 228–233. IEEE, 2016. 6

[32] 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)*, volume 1, pages 639–645. IEEE, 2017. 5

[33] C. Wigington, C. Tensmeyer, B. Davis, W. Barrett, B. Price, and S. Cohen. Start, follow, read: End-to-end full-page handwriting recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 367–383, 2018. 1, 2, 6, 7, 8

[34] S. Xiao, L. Peng, R. Yan, and S. Wang. Deep network with pixel-level rectification and robust training for handwriting recognition. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 9–16. IEEE, 2019. 6

[35] M. Yousef, K. F. Hussain, and U. S. Mohammed. Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. *arXiv preprint arXiv:1812.11894*, 2018. 1, 4, 5, 7