OCRoRACT: A Sequence Learning OCR System Trained on Isolated Characters

Conference Paper · April 2016

DOI: 10.1109/DAS.2016.51

Andreas Dengel
Deutsches Forschungszentrum für Künstliche I…
489 PUBLICATIONS **2,578** CITATIONS

Adnan Ul-Hasan
Technische Universität Kaiserslautern
16 PUBLICATIONS **66** CITATIONS

OCRoRACT: A Sequence Learning OCR System Trained on Isolated Characters

Adnan Ul-Hasan*
University of Kaiserslautern,
Kaiserslautern, Germany.
Email: adnan@cs.uni-kl.de

Syed Saqib Bukhari*
German Research Center for
Artificial Intelligence (DFKI)
Kaiserslautern, Germany.
Email: saqib.bukhari@dfki.de

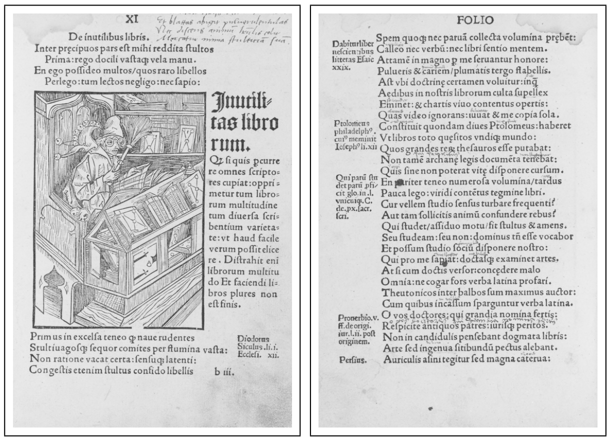
Andreas Dengel
University of Kaiserslautern
German Research Center for
Artificial Intelligence (DFKI)
Kaiserslautern, Germany.
Email: andreas.dengel@dfki.de

***Abstract*—Digitizing historical documents is crucial in preserving the literary heritage. With the availability of low cost capturing devices, libraries and institutes all over the world have old literature preserved in the form of scanned documents. However, searching through these scanned images is still a tedious job as one is unable to search through them. Contemporary machine learning approaches have been applied successfully to recognize text in both printed and handwriting form; however, these approaches require a lot of transcribed training data in order to obtain satisfactory performance. Transcribing the documents manually is a laborious and costly task, requiring many man-hours and language-specific expertise. This paper presents a generic iterative training framework to address this issue. The proposed framework is not only applicable to historical documents, but for present-day documents as well, where manually transcribed training data is unavailable. Starting with the minimal information available, the proposed approach iteratively corrects the training and generalization errors. Specifically, we have used a segmentation-based OCR method to train on individual symbols and then use the semi-corrected recognized text lines as the ground-truth data for segmentation-free sequence learning, which learns to correct the errors in the ground-truth by incorporating context-aware processing. The proposed approach is applied to a collection of 15th century Latin documents. The iterative procedure using segmentation-free OCR was able to reduce the initial character error of about 23% (obtained from segmentation-based OCR) to less than 7% in few iterations.**

Keywords*—*OCR, Historical Documents, Tesseract, OCRopus, LSTM Networks

I. Introduction

Preserving the literary heritage is important to gain valuable insights into the human history and to gain tremendous amount of knowledge about many aspect of our ancestors' daily lives. Documents, whether written on leaves, stones, textile, animal's skin or paper or recorded as speech or videos, have remained the eyes to the history of mankind. The most dominant and the oldest form among these media is written documents. Old documents are very delicate and they need extreme care. With the advancement in the capturing technology, the cost of preserving old documents has decreased many folds. Libraries and institutes around the globe have made valuable efforts in making digital copies of historical documents, such as shown in Figure 1.


XI
[illegible]
De inutilibus libris.
Inter pręcipuos pars eſt mihi reddita ſtultos
Prima: rego docili vaſtaqꝫ vela manu.
En ego poſſideo multos/quos raro libellos
Perlego: tum lectos negligo: nec ſapio:

Inutilitas librorum.

Qꝫ ſi quis ꝑcurrere omnes ſcriptores cupiat: opprimetur tum librorum multitudine tum diuerſa ſcribentium varietate: vt haud facile verum poſſit elicere. Diſtrahit eni librorum multitudo Et faciendi libros plures non eſt finis.

Primus in excelſa teneo ꝙ naue rudentes
Stultiuagoſqꝫ ſequor comites per flumina vaſta:
Non ratione vacat certa: ſenſuqꝫ latenti:
Congeſtis etenim ſtultus confido libellis

Diodorus Siculus .li.i. Ecclesi. xii.

b iii.

FOLIO
Spem quoqꝫ nec paruā collecta volumina prębēt:
Dabitur liber neſcientibus litteras Eſaie xxix. Calleo nec verbū: nec libri ſentio mentem.
Attamē in magno ꝑ me ſeruantur honore:
Pulueris & cariem/plumatis tergo flabellis.
Aſt vbi doctrinę certamen voluitur: inq̄
Aedibus in noſtris librorum culta ſupellex
Eminet: & chartis viuo contentus opertis:
Quas video ignorans: iuuat & me copia ſola.
Ptolomeus philadelph⁹. cui⁹ meminit Ioſeph⁹ li.xii Conſtituit quondam diues Ptolomeus: haberet
Vt libros toto quęſitos vndiqꝫ mundo:
Quos grandes rerꝫ theſauros eſſe putabat:
Non tamē archanę legis documēta tenebat:
Quis ſine non poterat vitę diſponere curſum.
Qui parū ſtudet parū ꝓficit glo. in .l. vnicuiqꝫ. C. de. px. ſacr. ſcri. En pariter teneo numeroſa volumina/tardus
Pauca lego: viridi contētus tegmine libri.
Cur vellem ſtudio ſenſus turbare frequenti?
Aut tam ſollicitis animū confundere rebus?
Qui ſtudet/aſſiduo motu/fit ſtultus & amens.
Seu ſtudeam: ſeu non: dominus tñ eſſe vocabor
Et poſſum ſtudio ſociū diſponere noſtro:
Qui pro me ſapiat: doctaſqꝫ examinet artes.
At ſi cum doctis verſor: concędere malo
Omnia: ne cogar fors verba latina profari.
Theutonicos inter balbos ſum maximus auctor:
Cum quibus incaſſum ſparguntur verba latina.
Prouerbio. v. ff. de origi. iur. l. ii. poſt originem. O vos doctores: qui grandia nomina fertis:
Reſpicite antiquos patres: iuriſqꝫ peritos.
Non in candidulis penſebant dogmata libris:
Arte ſed ingenua ſitibundū pectus alebant.
Perſius. Auriculis aſini tegitur ſed magna caterua:

Figure 1. Samples of "Narrenschiff" in medieval Latin script.These documents are taken from the image database of the German government funded project, *Kallimachos*[2]. This project aims to digitize 15th century documents in Latin and Fraktur scripts.

However, searching through these documents is still very difficult as the scanned images are not suitable for searching and indexing. Automatic recognition can help a Paleographer by indexing the old documents in a digital library. But, one must contemplate that automatic recognition of historical documents is very challenging due to following reasons:

- These documents are usually found in very bad condition, so the quality of text is not very high, it is often highly degraded with torn pages.
- The document may had been written in an ancient script with archaic orthography.
- Many modern OCR algorithms are based on supervised machine learning that require a lot of transcribed training data. Transcribing documents by a human is very laborious and costly.

*These two authors contributed equally

[2]http://www.kallimachos.de/project/doku.php

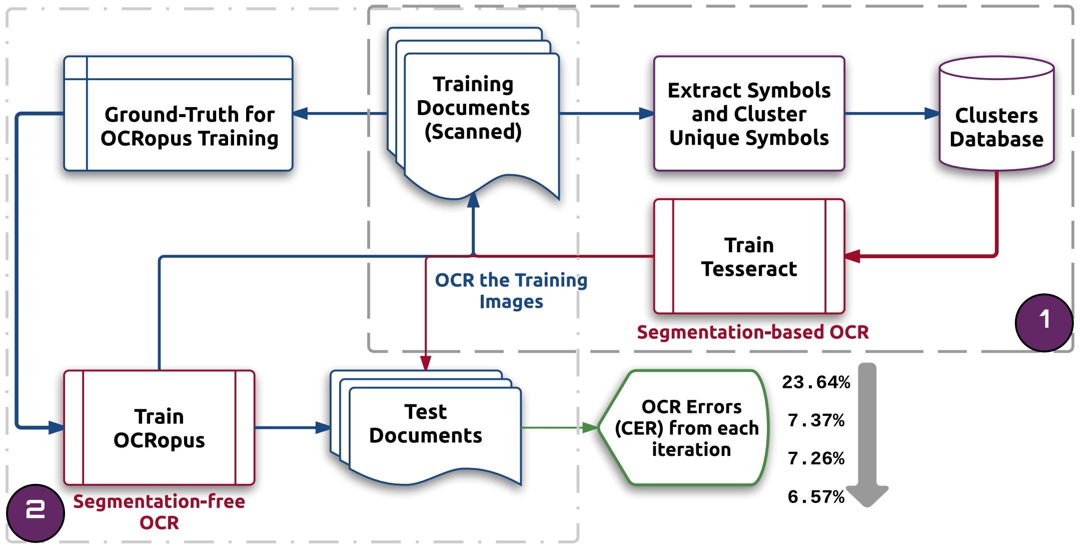


Figure 2. The complete pipeline for **OCRoRACT**. In the first phase of process (shown in dashed box marked with a 1), unique symbols are extracted from the training corpus of scanned pages. These symbols are then used to train the Tesseract OCR system. The trained models obtained from the Tesseract are used to generate semi-corrected ground-truth information for the training of OCRopus OCR system. In the second phase (shown in dashed-dotted box marked with a 2), the OCRopus is used to train LSTM-based OCR models. The best model is applied first to the Test data and then to improve the ground-truth data, that is used in the next iteration.

There could be multiple approaches to OCR documents (either historical or modern), for which the training data is not available. The first intuitive approach is to use a segmentation-based OCR approach (that rely on character segmentation from a scanned image), which does not require much training data. The only requirement is to extract the unique characters from the whole document and train a shape-based classifier [1], [2]. The second approach is to manually transcribe a part of data that needs to be OCR, train a classifier based on segmentation-free OCR approach (that are better for context-aware recognition), and then use this model to OCR the rest of the corpus.

However, both of the above-mentioned solutions carry their own demerits. The first approach, in practice, does not generalize well for new documents [3]. The second approach requires the creation of transcribed data, which is tedious and costly, and would require the manual efforts to transcribe every type of document that is to be digitized. The use of artificial data is also getting popular in many computer vision related tasks; however, generating such a data also requires some already transcribed data.

This paper reports our contribution to minimize the requirement of a language expert in manually transcribing the documents, esp. the historical documents. The only requirement in the proposed framework, named as ***OCRoRACT***[3], from the language expert is to provide the unique set of characters in a document along with their Unicode representation. The reported approach is capable of reducing the recognition errors by iterative training.

Our hypothesis is that the segmentation-based approaches can be used in tandem with the segmentation-free approaches to OCR documents where no or very limited training data is available. Specifically, the proposed approach is designed for 15th century printed Latin documents, for which very small amount of training data is available. However, the proposed framework can be equally applied to other situations, where no training data is available.

The proposed framework successfully combines the benefits of both approaches. The first issue in using a *segmentation-free* approach is to have a manually transcribed data. This problem is solved by using a segmentation-based approach to generate semi-correct ground-truth (GT) data. The second issue in using a *segmentation-based* approach is the poor generalization on the new data. LSTM networks nicely solve this problem, as they have been shown to yield excellent results on the unseen data [4].

This paper is further organized as follows. Section II describes the proposed approach in detail. Section III explicates the experiments performed to validate our hypothesis, while Section IV discusses various types of errors and results obtained. Section V finally concludes the paper with lessons learnt and some future directions in which the current work can be extended.

II. Methodology

The novelty of our concept is to use both segmentation-based and segmentation-free OCR approaches in tandem to design a high performance OCR system for documents having no or very limited GT data. The complete pipeline of our procedure is shown in Figure 2. The idea is to use the segmentation-based OCR to start the training on

[3] *OCRo* refers to **OCRo**pus and *RACT* refers to Tessa**RACT**

individual symbols, as not much training data is usually required for such systems. The OCR models obtained can be used to get a semi-corrected text, which can be used subsequently to train a segmentation-free OCR system.

The process starts with the extraction of individual symbols from the scanned pages or text-lines. A language expert can provide a list of unique symbols in a given document along with their Unicode representation. Alternatively, a clustering process can follow the symbol extraction step to find the unique symbols. In the present work, we chose the former path as some language experts were accessible for our project. Tesseract [5], a well-known segmentation-based open-source OCR system is trained to recognize text based on the given symbols. Tesseract OCR model is then used to generate the semi-corrected GT information, which is then used to train OCRopus [6], a segmentation-free open-source OCR system based on Long Short-Term Memory (LSTM) neural networks. The trained LSTM model is then used again to improve the GT information. This iterative process can be repeated for any number of iterations; however, in the current work, we chose to stop the training after seeing a small improvement (less than 1%) in the GT data from the previous iteration.

Before proceeding further, it is appropriate to understand the internal working of two OCR systems that have been used to validate our hypothesis.

A. Tesseract: A Segmentation-based OCR System

Tesseract [5] is an open-source system for document analysis and recognition. The Tesseract OCR engine is a segmentation-based OCR system which mainly requires a file, referred here as *char-box-file*, that contains multiple instances of each character class with corresponding Unicode representation for training. However, for extracting better features for each character class, Tesseract uses the context information. For that purpose, Tesseract training engine also needs a single/multi-page document image containing text paragraphs and their corresponding *char-box-file*.

As mentioned above, the 15th century novels, which we are dealing here, do not have any representative text in any form. Therefore, a randomized "meaningless" text generation procedure has been adopted to produce the training data. This means that we randomly select a word length in a fixed interval, and then print random characters from our available character set to our Tesseract training *char-box-file*. This *char-box-file* is then used to train Tesseract OCR model. Additionally, for making meaningless text looks close to real text, we manually investigate the interword- and intraword-spacing from sample pages of original novel and incorporated this information in generating the meaningless text. An example of a used training input image is given in Figure 3.

This basic method by itself is not sufficient to generate reliable models because in the *char-box-file*, all types of characters are placed at the baseline and therefore it loses the contextual information related to the baseline, ascender-line, middle-line and descender-line. Therefore, we need to incorporate such information in the *char-box-file*. To do this, each character in our character set has been

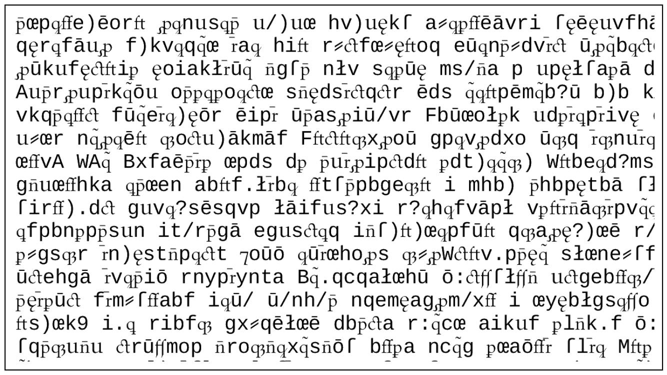


Figure 3. An example of the training input image of "meaningless" text (that was produced using the procedure mentioned in Section II-A) for training Tesseract OCR model.

assigned to a specific "character-placement" class. The different character-placement classes help to discern where the character should be placed on a line. This process also enable us to have primitive control over the frequency of each character in the training text. For example, it is easy to see that upper-case characters occur less frequently than lower-case characters, and so on. A visualization of the line offsets can be found in Figure 4.

Another way to improve our "meaningless" text generation process, to contain more realistic contextual information, is to apply simple rules to character classes. The rules we implemented ensure that upper-case characters only occur at the beginning of a word, or throughout the entire word, or not at all. Similar rules are applied for lower-case characters. The primary goal of this approach is to make the training text more realistic, and in that way improve model accuracy. Additionally, future rules might handle the placement of punctuation, or even vowels and consonants. However, it should be noted that this would result in a loss of generality regarding the languages and texts that could be handled.

After training the Tesseract OCR model, we apply the OCR model to each text-line in our dataset. While examining the output we find that the Tesseract OCR model produces non-consistent errors, such that for each of the character class some samples are recognized correctly at some places and other sample are recognized incorrectly at other places. We refer this text-line data, containing images with their corresponding text outputs from the Tesseract OCR engine, as semi-corrected ground-truth for training the segmentation-free OCR engine, which is capable to use its powerful context-aware processing capability

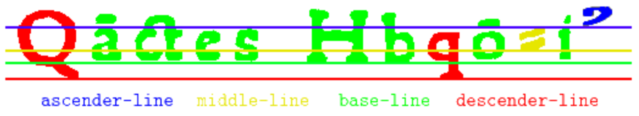


Figure 4. Information about the line offsets that was used in producing training data (Figure 3) for training Tesseract OCR model. Placement classes (baseline, ascender-line, middle-line and descender-line) and the corresponding characters belong to these placement classes are mentioned in the figure.

over the semi-corrected ground-truth data to reduces OCR errors.

B. OCRopus: A Segmentation-free OCR System

OCRopus [6] is an open-source document analysis and recognition system. The recognition module of this system is based on contemporary Recurrent Neural Networks (RNNs). Specifically, 1D LSTM architecture has been employed for printed OCR purpose. This architecture has shown outstanding results on many printed OCR tasks [3], [4], [7], [8]. This variant of LSTM network is simple-to-use, yet it has yielded excellent OCR results for a variety of scripts. In this architecture, a window of width equal to $1-pixel$ and height equal to that of the input image is traversed over the whole text-line. The $1-column$ slice of the image is called a "frame". The height of this frame is termed as the "depth of the sequence". These 1D sequences (frames) are then fed to the input layer of the LSTM network. It is very important to make the depth of sequence equal for all input text-line images. The process of making text-line image heights equal is called "normalization". We can simply rescale the text-line image for normalization, but it does not preserved the baseline and x-height information of all characters consistently. This information is very important in distinguishing many Latin characters. OCRopus framework employs a different normalization method which is based on Gaussian filtering and affine transformation. The details of this method are given in [9].

A simplified Figure of a typical LSTM cell (without the peephole connections) is shown in Figure 5 and the following set of corresponding equations represents the forward path of the training.

$$\begin{aligned}
f_t &= \sigma(W_{xf}.x_t + W_{hf}.h_{t-1} + b_f)\\
i_t &= \sigma(W_{xi}.x_t + W_{hi}.h_{t-1} + b_i)\\
v_t &= tanh(W_{xv}.x_t + W_{hv}.h_{t-1} + b_v)\\
o_t &= \sigma(W_{xo}.x_t + W_{ho}.h_{t-1} + b_o)\\
C_t &= f_t * C_{t-1} + i_t * v_t\\
h_t &= o_t * tanh(C_t)
\end{aligned}$$

where

- f_t is the **forget gate layer**, i_t is the **input gate layer**, and o_t is the **output gate layer**,
- C_t is the **Cell State**, h_t is the **cell output** and h_{t-1} is the cell output at previous timestep,
- $b_y, y \in \{f, i, v, o\}$ is the bias unit for forget gate, input gate, input squashing and output gate respectively,
- W_{ij} is the weight connection between i^{th} and j^{th} node, e.g., W_{xf} is the weight between the external input and the forget gate,
- σ is the *logistic sigmoid* function given by, $\sigma = \frac{1}{1+exp(-x)}$ and *tanh* is the tangent hyperbolic function defined a,s $tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$.

During the training process, the input image (randomly chosen from the training database) is given at the input

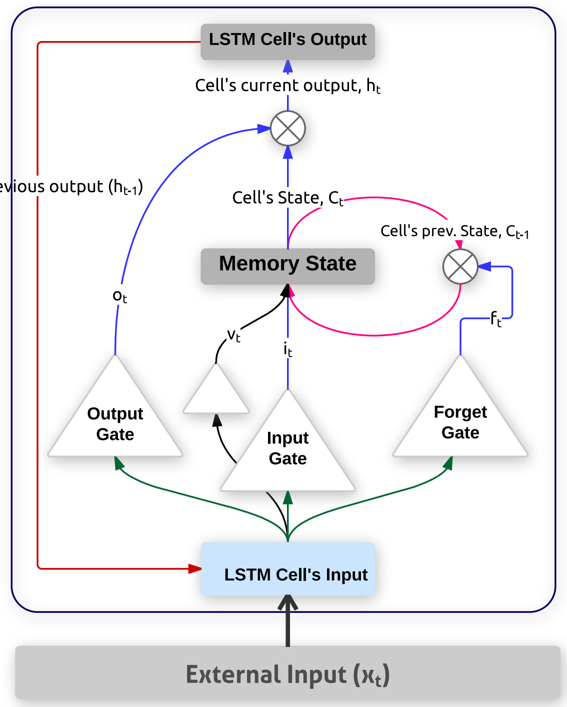


Figure 5. A simplified LSTM cell without peephole connections. The forget gate is used to decide which input are to be retained at a given timestep. The input gate decides which are the input that are to be processed by the cell, and the output gate decides which outputs are to be taken out of the cell. Please refer to the text for further explanation of the LSTM networks.

layer. The input consists of h number of cells, where h is equal to the normalized height of the input image. The hidden layer consists of LSTM cells (a tunable parameter), that process the input activations in the forward pass. Each unit receives one input from the previous state (the feedback loop) and one from the current time-step. The output activations of the hidden layer are given to a special layer – The Connectionist Temporal Classification – before the output layer. This layer aligns the output activations with the target labels using forward-backward algorithm [10]. The output layer consists of k units, where k is equal to the number of character classes in the data. During the backpropagation stage, the deltas at the output layer are propagated back to update the network's weights using the standard backpropagation through time algorithm [11].

III. Experimental Evaluation

The basic concept behind the proposed framework is to jointly use segmentation-based and segmentation-free approaches to OCR documents for which GT data is unavailable to train machine learning algorithms. There are three ways in which the proposed methodology has been evaluated. This is done to compare the results of various alternatives mentioned in Section I. Firstly, the Tesseract is used to OCR the database described in Section III-A. Secondly, an LSTM network is trained directly on a subset of dataset that has been transcribed manually. Thirdly, the proposed approach is used to iteratively improve the semi-corrected GT data to train LSTM networks.

The performance is estimated in term of the *Character Error Rate (CER)* which is measured using the "edit distance" metric. This distance is defined as the ratio of insertion, deletion and substitution to the total length of the text-line.

A. The Database

As mentioned before, 15th century novels are available under the German government funded project, Kallimachos. Some of the sample document images are shown in Figure 1. There are hundreds of novels written in Latin and Fraktur that are to be digitized. Firstly, it is also important to remember that we have no text (ground-truth) data available for any of the novel. Secondly, the pages in these novels not only contain degradations because of aging but also contain annotations, which make them more challenging for document analysis.

In order to validate our research hypothesis, we have selected a subset of 100 images from one novel of Latin script for training, as well as we have also selected around 8 images from another Latin novel for testing. For performance evaluation, ground-truth for 100 scanned pages with 3329 text-lines have been manually generated. It took us more than a month to generate this ground-truth data.

For Tesseract training, we just need all unique symbols along with the corresponding Unicode to generate "meaningless" training data. To train the proposed OCRoRACT OCR model we also needed text-line images, that we have manually cropped from the training dataset. To train OCRopus model, which we have done to show the effectiveness of our OCRoRACT model as compared to OCRopus model, we needed not only text line images but also their manually labeled ground-truth text.

B. The System Parameters

Tesseract system was trained with the default settings that comes with the open-source version. To train LSTM-based OCR models, the OCRopus system uses some free parameters. The first important parameter is the number of hidden layers. Only single hidden layer is available in OCRopus. The second important parameter is the number of LSTM cells at the hidden layer. For the experimental results reported in this paper, the number of LSTM at the hidden layer is fixed to 100. Normalizing text-line images to a fixed height is an important preprocessing step. In our experience, the image height of 48 pixels is a reasonable choice and has worked satisfactorily for a variety of scripts. The learning rate and momentum are the remaining parameters and they are set to $1e-4$ and 0.9 in the current work.

C. Results

This section reports the results of three evaluations that have been performed to test the performance of the proposed framework. There are two test sets used in the evaluation process. The first dataset consists of two randomly selected pages from the same book that is used for training the Tesseract and OCRopus OCR systems; however, these pages were not part of any training. Total number of text-lines are 104 in these two pages and total number of characters, $N1$, are 2877. This dataset is termed as 'T1'. The second dataset consists of 8 pages randomly selected from a book that has not been used in training at all. There are a total of 270 text-lines in these pages and the total number of character, $N2$, are 7017 This dataset is termed as 'T2'.

Table I. Comparison of applying Tesseract, OCRopus and OCRoRACT. The results obtained by the OCRopus seems the best, but it must be noted that this system was trained with the correct GT information, while the OCRoRACT is trained with semi-correct GT information. Noteworthy is the results obtained on 'T2' dataset where the difference between the OCRopus and OCRoRACT is relatively small.

	Character Error Rate, CER (%)		
Dataset	**Tesseract**	**OCRopus**	**OCRoRACT**
T1	23.64	1.8	6.57
T2	25.21	10	10.6

The intermediate results obtained by the OCRoRACT framework are presented first before comparing the final results with other two alternatives. During the first stage (*iteration-0*), a Tesseract OCR model is trained on the unique symbols extracted from the given training documents. Tesseract yields a CER of 23.64% on 'T1' dataset and a CER of 14.4% on the data collection that is later used as the training database for LSTM-based OCR. This means that the LSTM-based OCR has been trained with a 14.4% erroneous GT data (*iteration-1*). The LSTM model thus trained outputs a CER of 7.37% on 'T1' database. The same model improves the GT data from having an error of 14.4% to 7.154% (an improvement of 50.1%).

During the second iteration (*iteration-2*), the LSTM models are trained with the improved GT obtained from the first iteration. The model thus trained gives a CER of 7.26% on the 'T1' dataset, and improves the GT by further 9.56% (CER of 6.47%). The improved GT is used again (*iteration-3*) to train another LSTM model in third iteration. The LSTM model trained during the third iteration results in a CER of 6.57% on 'T1' data and improve the GT by 0.9%. This iterative process could go on further, but we decided to stop it at this stage as the GT improvement is now less than 1%.

The LSTM models obtained after the third iteration are used to OCR the 'T2' dataset along with Tesseract (that was trained at the beginning of the process) and the OCRopus model, trained with correct GT information. The results of three evaluations are listed in Table I.

IV. Error Analysis and Discussions

The top confusions in applying the three OCR systems on 'T1' dataset are shown in Table II. Tesseract has mostly made errors in recognizing between small and capital letters. This is understandable as these errors normally occur when contextual information is not present. It should also be noted that the usual application of Tesseract requires an associated language model. The same has not been done in the current work as no language model for medieval Latin is available.

Table II. Top errors generated by the Tesseract, OCRoRACT and OCRopus OCR systems when applied on 'T1' dataset. **GT** denotes the ground-truth, **Pred** denoted the predicted output and **Error** shows the number of errors. _ shows the *insertion* or *deletion* of the corresponding character. Total number of errors made by a particular OCR system are shown below the name of that system.

Tesseract			OCRoRACT			OCRopus		
Total Errors		680	Total Erros		189	Total Errors		53
Pred.	**GT**	**Errors**	**Pred.**	**GT**	**Errors**	**Pred.**	**GT**	**Errors**
	_	18	_	i	19	_		7
_		14	_		17		_	3
1		11		_	13	_	l	3
S	s	8	$\bar{a}$	a	7	_		2
$\bar{r}$	i_	7	$\bar{u}$	u	5	$\bar{u}$	u	2
f	i	6	l	f	4	.	_	1

Table III. Top errors generated by the Tesseract, OCRoRACT and OCRopus OCR systems when applied on 'T2' dataset. **GT** denotes the ground-truth, **Pred** denoted the predicted output and **Error** shows the number of errors. _ shows the *insertion* or *deletion* of the corresponding character. Total number of errors made by a particular OCR system are shown below the name of that system.

Tesseract			OCRoRACT			OCRopus		
Total Errors		1769	Total Erros		751	Total Errors		704
Pred.	**GT**	**Errors**	**Pred.**	**GT**	**Errors**	**Pred.**	**GT**	**Errors**
	_	72	_		72	_	f	142
_		48	_	i	38	_		29
t	r	25		_	36	_	l	20
u	n	23	t	r	16	.	_	16
l	_	21	.	_	13		_	13
.	_	14	$\bar{a}$	$\bar{e}$	10	.	:	7

The OCRoRACT system, which has been trained on the erroneous GT data corrects many of these errors. However, it confuses between similar shape characters like '$\bar{a}$' and 'a' or '$\bar{u}$' and 'u'. It is to be noted that both of these systems yield many *insertion* and *deletion* errors related to the 'space' character. One way to reduce the errors made by the OCRoRACT system is to retrain Tesseract system with more variety of symbols that are causing confusions.

The OCRopus system produces less errors; however, the top confusions are again 'space' *insertion* and *deletion*. Other notable errors are deletion errors. The *deletion* errors can generally be reduced by better training. It must be noted that the training data is not much and the performance of LSTM networks in still quite satisfactory.

One drawback of this system is its use for cursive scripts like Arabic and Devanagari, where character/ligature segmentation is non-trivial. Arabic-like scripts contain a huge amount of ligatures and the reliable segmentation of ligatures requires more research efforts. However, one possible solution for such scripts is to use the artificial data instead of using the segmentation-based approach to start the training. The LSTM models thus obtained could be used to OCR the scanned documents.

It is also important to see the errors obtained from applying these three systems on 'T2' dataset, which has not been used during the training. The top confusions are listed in Table III. By comparing the outputs of three systems on 'T2' dataset from Table III, it can seen that Tesseract and OCRoRACT make similar "shape confusion" errors. However, the top errors for OCRopus are due to the misrecognition of characters. This shows that it is difficult for OCRopus to generalize well to unseen variety of characters that it had seen during the training.

It is also evident from the results that OCRoRACT performs very well on unseen data. The iterative nature of this framework allows for the addition of further training data (albeit erroneous) into the system, thereby, resulting in better training. These options are currently being investigated and will be reported elsewhere.

V. Conclusion

This paper presents a novel framework, the *OCRoRACT*, to combine the benefits of both segmentation-based and segmentation-free OCR approaches to OCR documents where no training data is available. Moreover, the reliance on a human expert to generate the transcribed data is greatly reduced, if not eliminated by using the proposed methodology. The performance of this system is excellent when a document with similar script is evaluated. Moreover, since the system does not require accurate GT information, it can be retrained as new output becomes available. This framework can be improved further by incorporating the ability of the system to be used for cursive scripts having huge amount of ligatures.

Acknowledgment

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), project Kallimachos (01UG1415C).

References

[1] Q. U. A. Akram, S. Hussain, A. Niazi, U. Anjum, and F. Irfan, "Adpating Tesseract for Complex Scripts: An Example of Urdu Nastalique," in *DAS*, 2014, pp. 191–195.

[2] N. White, "Training Tesseract for Ancient Greek OCR," *Eutypon*, pp. 1–11, 2013.

[3] F. Simistira, A. Ul-Hasan, V. Papavassiliou, B. Gatos, V. Katsouros, and M. Liwicki, "Recognition of Historical Greek Polytonic Scripts Using LSTM Networks," in *ICDAR*, Tunisia, 2015.

[4] T. M. Breuel, A. Ul-Hasan, M. Al Azawi, F. Shafait, "High Performance OCR for Printed English and Fraktur using LSTM Networks," in *ICDAR*, Washington D.C. USA, aug 2013.

[5] R. Smith, "An Overview of the Tesseract OCR Engine," in *ICDAR*, 2007, pp. 629–633.

[6] "OCRopus – Open Source Document Analysis and OCR System." [Online]. Available: https://github.com/tmbdev/ocropy

[7] A. Ul-Hasan, S. B. Ahmed, S. F. Rashid, F. Shafait, and T. M. Breuel, "Offline Printed Urdu Nastaleeq Script Recognition with Bidirectional LSTM Networks." in *ICDAR'13*, USA.

[8] T. Karayil, A. Ul-Hasan, and T. M. Breuel, "A Segmentation-Free Approach for Printed Devanagari Script Recognition," in *ICDAR*, Tunisia, 2015.

[9] M. R. Yousefi, M. R. Soheili, T. M. Breuel, and D. Stricker, "A Comparison of 1D and 2D LSTM Architectures for Recognition of Handwritten Arabic," in *DRR–XXI*, USA, 2015.

[10] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks." in *ICML'06*.

[11] P. Werbos, "Backpropagation through time: what does it do and how to do it," in *Proceedings of IEEE*, vol. 78, no. 10, 1990.