Attention-Based Deep Learning Model for Arabic Handwritten Text Recognition

Takwa Ben Aïcha Gader, Afef Kacem Echi
University of Tunis, ENSIT-LaTICE, Tunis, Tunisia
takwa.ben.aichaa@gmail.com, ff.kacem@gmail.com

Abstract. This work proposes a segmentation-free approach to Arabic Handwritten Text Recognition (AHTR): an attention-based Convolutional Neural Network – Recurrent Neural Network – Connectionist Temporal Classification (CNN-RNN-CTC) deep learning architecture. The model receives as input an image and provides, through a CNN, a sequence of essential features, which are transferred to an Attention-based Bidirectional Long Short-Term Memory Network (BLSTM). The BLSTM gives features sequence in order, and the attention mechanism allows the selection of relevant information from the features sequences. The selected information is then fed to the CTC, enabling the loss calculation and the transcription prediction. The contribution lies in extending the CNN by dropout layers, batch normalization, and dropout regularization parameters to prevent over-fitting. The output of the RNN block is passed through an attention mechanism to utilize the most relevant parts of the input sequence in a flexible manner. This solution enhances previous methods by improving the CNN speed and performance and controlling over model over-fitting. The proposed system achieves the best accuracy of 97.1% for the IFN-ENIT Arabic script database, which competes with the current state-of-the-art. It was also tested for the modern English handwriting of the IAM database, and the Character Error Rate of 2.9% is attained, which confirms the model's script independence.

Key words: Arabic handwriting recognition, attention mechanism, BLSTM, CNN, CTC, RNN.

1. Introduction

Handwriting recognition is one of the critical research areas of Optical Character Recognition (OCR)[1], which consists of converting text on images into machine-encoded text. This step is vital, considering the importance of digitization in nowadays life. Handwriting recognition is employed in various domains such as document processing [25], writer identification, office automation, signature verification [5] automatic cheque processing in banks [26], postal code recognition [9], etc.

In this work, we present a segmentation-free approach to address the problem of Arabic Handwritten Text Recognition (AHTR). It is crucial since the Arabic alphabet is the second most widely used alphabetic writing system globally, and 234 million people speak Arabic. Researchers obtained very encouraging results in machine-printed Arabic text recognition because of the text's homogeneity, such as within and across word spaces, spacing within successive text lines, and character sizes. Oppositely, handwritten Arabic recognition is classified as a complex task for many reasons: the multiple writing styles, the Arabic script's vast variability, the use of ligatures and diacritics, the cursive

[1] All the abbreviations are explained in Table 6 at the end of the paper, p. 67.

nature where alphabets are written in a joint flowing style which may cause touched and overlapped characters (see Figure 1, letters with the same shapes, such as ب, ت and ث, which can only be distinguished by dots figured under or above the base alphabet, the change of the same character's shape according to its position in the word (see Figure 2) and the absence of significant available Arabic databases.

Deep Learning-based models have been the basis of most Computer Vision tasks [7, 48], performing higher performances than other state-of-the-art approaches. More particularly, models based on neural networks have given exciting results in handwriting recognition. As it has proven its success in this area, many works based on the architecture Convolutional Neural Network – Recurrent Neural Network – Connectionist Temporal Classification (CNN-RNN-CTC) have been proposed to recognize cursive scripts such as Arabic, Parsi, and Latin to avoid segmenting words into characters or isolating merged characters.

In the mentioned architecture, the text image is sent through convolutional layers to get the features from the image. Later these features are fed to recurrent neural network architecture, which outputs softmax probabilities over the vocabulary. These outputs from different time steps are fed to the CTC decoder to get the raw text from images. Note that CTC is designed for tasks where we need alignment between sequences but where that alignment is difficult. We used it to align each character to its location in the text. By just mapping the image to text and not worrying about each character's alignment to the input image's location, one should calculate the loss and train the network.

With this in mind, we tried to shed new light on previous works and propose an efficient AHTR system that uses a CNN, a RNN and a CTC block. Of particular interest, our system has three vital points. First, the CNN is extended by dropout layers. Second, to extract features from input images, the CNN employs batch normalization and dropout regularization parameters to prevent over-fitting and to improve system performance. Third, the output of the RNN block is passed through an Attention mechanism. This allows the essential information to be selected from the feature sequence resulting from the RNN and then transmitted to the CTC block. Thus, to recognize a text, the system does not segment it into words or characters in advance; rather, it recognizes the input text by extracting a feature map from the input image using a CNN and transfers it to the RNN layer, where an attention-based Bidirectional Long Short-Term Memory Network (BLSTM) with CTC is applied for sequence labeling. This solution enhances previous methods by improving the model speed and performance and controlling model over-fitting. From the functional point of view, the user simply performs the training of the network by just mapping the image to text, without worrying about each character's alignment to the location in the input, and the system calculates the loss and presents the results. We will discuss each of the steps in the further sections of the paper.

Fig. 1. Some Arabic script writing characteristics. Created by the Authors of [6]; reused under the CC BY 4.0 license.

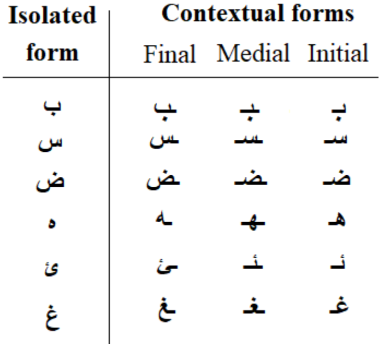


Fig. 2. Contextual forms of some Arabic letters.

The paper is organized as follows. Section 2 briefly reviews related works, and section 3 details the proposed models and the system's architecture. In section 4, we present the training process and discuss the obtained results.

2. Related Works

The recent progress in deep learning technology influenced the Arabic text recognition problem, where several deep learning-based approaches were proposed. The first deep learning-based approach for AHTR in images was introduced in 2008 by Graves and Schmidhuber [33]. The authors used a Multi-Dimensional Long Short Term Memory (MDLSTM) network and the CTC. The proposed model was evaluated on the IFN/ENIT dataset [57] and reached an accuracy of 91.4%. The same model was used in 2013 by Rashid et al. [60], where the MDLSTM was used on the input images to extract features fed to the CTC layer (120 units), which allows data labeling. The proposed method reached a recognition rate of 99% on the Arabic Printed Text Image database (APTI) [63]. In the same year, Chherawala et al. in [19] used the MDLSTM model on some handcrafted features and raw pixels extracted from IFN/ENIT, and they achieved an accuracy of 88.8%.

Continuing the study of the deep-learning-based approaches evolution, we mention the one proposed by Abandah et al. in 2014 [1]. It is based on the segmentation of cursive words into graphemes. A features vector is extracted and passed to a BLSTM, which transcript sequences by graphemes exploiting. In 2016, Ahmad et al. [2] used the three variants, LSTM, BLSTM, and MDLSTM, for Pashto (which uses the Arabic alphabet) handwritten text recognition. Their study offers the best performance using the MDLSTM model on the KPTI database with an error rate of 9.22%.

Meanwhile, Elleuch et al. [29] introduced a handwritten character recognition approach based on a CNN-SVM architecture. The CNN is used for feature extraction, and the SVM with Radial Basis Function (RBF) kernel for classification. The authors improved the results by adding the dropout technique, which temporarily removes some units from the network to prevent the system from over-fitting problems. The preformance of the system was evaluated on three datasets: HACDB [49] with 24 classes, HACDB with 66 classes, and IFN/ENIT with 56 classes, and error rates achieved were 2.09%, 5.83%, and 7.05%, respectively. During the same year, an RNL-based MDLSTM and dropout were introduced by Maalej et al. [52].

In 2017, Chen et al. [17] presented a segmentation-free approach of RNN with a four-layer bidirectional Gated Recurrent Unit (GRU) network with a CTC output layer and combined it with the dropout technique. The authors evaluated the system performance on the IFN/ENIT database with the "abcd-e" scenario. Accuracy of 86.4% was reached. El-Sawy et al. [27] proposed a CNN-based in-depth learning architecture for Arabic handwriting character recognition. The authors applied an optimization process that increased the model performance. However, this method's weak point was its incapacity to manage significant inputs and share their weights.

The same year, Ahmad et al. in [3] presented an MDLST-CTC architecture for recognizing Arabic handwritten text. They used data augmentation to improve the model performances and reached a CRR level of 80.02% on the KHATT dataset. Among the recent works, we cite the one proposed in 2018 by M. Amrouch et al. [8]. It is a CNN-based HMM model, where CNN is a prominent feature extractor, and the Hidden Markov Model (HMM) baseline system is a recognizer. The model was validated using the two IFN/ENIT database scenarios, "abc-d" and "abcd-e". It reached a recognition accuracy of 88.95% and 89.23%, respectively.

Continuing with the progress of Handwritten Arabic recognition, we mention two works presented in 2019. The first one [28] is a Convolutional Deep Belief Network (CDBN) framework proposed to recognize low/high-level dimensional data. The authors used data augmentation and a dropout regularization to increase the model's performance and avoid over-fitting. The model was first evaluated on the HACDB characters database and achieved an accuracy rate of 98.86%. Moreover, second, on the IFN/ENIT words database, it reached an accuracy of 92.9%. The second one presented in [56] was designed to recognize an image of Arabic text/characters. The introduced model takes a single line of Arabic text and segments it into words and letters. The trained model recognizes these image fragments as characters. The model evaluation was done on a custom dataset, and reached a CRR of 83%. In 2020, authors in [4] proposed a deep learning-based approach for Arabic text extraction. They used preprocessing, including pruning of extra white spaces plus de-skewing the skewed text lines. Furthermore, they trained the proposed MDLSTM-CTC model on the KHATT database with data augmentation. They achieved a Character Recognition rate (CRR) of 80.02%. In the same year, authors in [61] introduced a CNN-RNN-based model for word recognition. They worked on Persian handwritten text, which has the same properties as Arabic. Their main improvement was not using the segmentation step. In [6], a supervised Deep CNN (DCNN) model was used to address the challenges of recognizing offline handwritten Arabic text, including isolated digits, characters, and words. The model reached an accuracy of 99.95% on the SUST-ALT database.

To clotureclose our related works section, we mention the work [10], where authors proposed an approach based on sequentially transferring the mid-level word image representations through two consecutive phases strategy based on transfer learning. They used the ResNet18 model that has been pre-trained on the ImageNet dataset. They achieved a recognition accuracy of 96.11% on the IFN/ENIT database.

The related works have been synthetically compared in Tab. 1.

3. Proposed System

This section will introduce the proposed deep neural network for AHTR inspired by the work in [62]. It comprises three principal end-to-end parts: a CNN, and an RNN, followed by a CTC. This combination is the best choice as it currently outperforms all other approaches. The CNN is used for sequence feature extraction from the input images. Furthermore, the RNN is used to propagate information within this sequence. For each sequence element, it outputs a matrix of character scores. The CTC operation is set up to calculate the loss value to train the proposed model and to perform the inference at this stage. The CTC decodes the RNN's output matrix to infer the text contained in the input image. These two associated networks with the CTC make word-level recognition possible without character-level segmentation. Figure 3 shows an overview of the AHTR system and Figure 4 details it. The proposed neural network (NN) can be described by a mathematical formula (1) that performs the mapping between an image I and a character sequence S:

$$\mathrm{NN} : I_{W\times H} \to (c_1, c_2, ..., c_n)_{0\leq n\leq L} \ , \tag{1}$$

Tab. 1. Comparison of the state-of-the-art methods.

Year	Method	Advantages	Limits
2008	[33]	The general system applied for Arabic as English. The system works directly on raw pixel data and requires minimal changes for languages with different alphabets.	When applying the model to other languages, the dimensionality of the networks should be modified to match the data.
2013	[60]	The method performs very well on printed Arabic text recognition, even for very low resolution and small font size images.	It is not evaluated for handwriting text.
2013	[19]	The MDLSTM can automatically learn features from the input image (automatically learned features).	Despite their ability to learn features, the architecture of MDLSTM networks can limit their performance, especially the amount of horizontal sub-sampling.
2014	[1]	We describe a robust rule-based segmentation algorithm that uses particular feature points identified in the word skeleton to segment the cursive words into graphemes.	Improper segmentation of the graphemes leads to a lousy extraction of the features, therefore an inadequate recognition.
2016	[2]	It is a comparison, not an approach that showed that MDLSTM achieved a good performance on their KPTI (Pashto language) database.	-
2016	[29]	They used a CNN-based-SVM model, which automatically extracts features from the raw images and performs classification.	A non-generic system. The proposed architecture must be extended to deal with handwritten words in different languages and enhance the recognition rate.
2016	[52]	Dropout.	-
2017	[17]	The use of GRU units and dropout.	-
2017	[27]	The use of an optimization process.	The method is incapable of managing significant inputs and sharing their weights.
2018	[8]	The ability to extract automatically salient features directly from raw pixels.	Low Recognition Rate compared with recent models.
2019	[56]	Using a preprocessing step to improve the quality of the images and to segment text lines to words and characters.	The recognition rate is low for letters with loops and those including dots.
2020	[4]	MDLSTM has the advantage of scanning the Arabic text lines in all directions. A pre-processing step includes pruning of extra white spaces plus de-skewing the skewed text lines.	Limited dataset.
2020	[61]	Using the advantages of both CNN and RNN for the word recognition purpose. The method benefits from CTC for eliminating the segmentation procedure.	-
2021	[6]	Using the Transfer Learning (TL)-based feature extraction. The approach can effectively deal with high-dimensional data by automatically and contextually extracting the best features. PGeneral model.	A non-generic system. The database has an insufficient number of training samples.
2021	[10]	Using the transfer learning	The model misclassifies words with similarities in shape and number of characters.

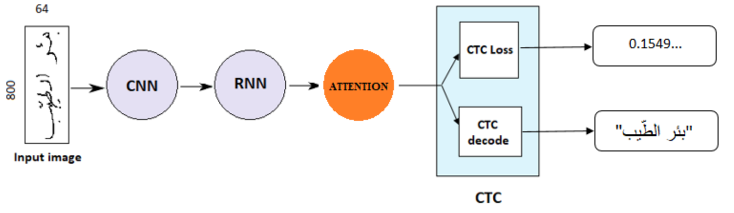


Fig. 3. The proposed AHTR pipeline.

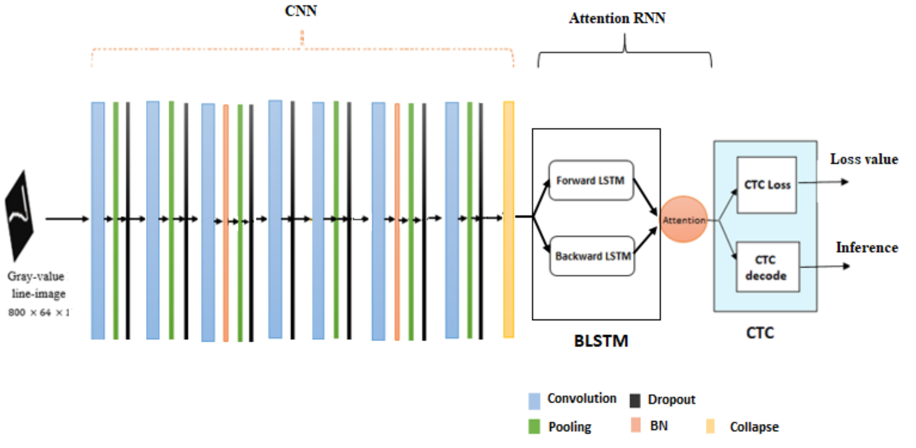


Fig. 4. The used Neural Network architecture for the text-line recognition.

where L is the max sequence length and $c_i, i \in \{1..L\}$ are the predicted characters. It transforms an image $I_{W \times H}$ to a sequence of characters $(c_1, c_2, ..., c_n)$ with a length L. As the recognition is done on a character level, the model can recognize text that does not belong to the training data. This is a strong point of the proposed model. We describe the model and the used dataset in the rest of this section.

3.1. CNN

CNNs are specific neural networks that apply convolution in place of general matrix multiplication in at least one of their layers. These networks have succeeded in several fields, such as automatic image classification, multi-object detection, object localization, handwritten digit recognition, and object classification. With this success, the application of CNNs in machine learning projects has increased drastically. Over time, several approaches have been used to improve the performance of CNNs. We cite, for example, the development of computational systems, the design of regularization techniques such as the batch normalization and the dropout method, adding hidden layers, and the abundance of the training data. With all these stratagems, CNNs performance increased over time.

Recall that a CNN consists of an input layer, hidden layers, and an output layer sequentially connected. Each convolutional layer has input from the preceding layer convolved with trained filters. Hidden layers' inputs and outputs are masked by the activation function (generally the *Relu*) and the final layer's convolution. The convolution output can be followed by other layers, such as fully connected, pooling, and normalization layers. Every neuron in one layer is connected to another in a fully connected layer. The pooling operation reduces the risk of over-fitting. It minimizes the data size by combining the neuron clusters' outputs at one layer into a single neuron in the next layer. There are two well-used types of pooling: max pooling and average pooling. Taking the maximum value of each local neuron cluster in the feature map is max-pooling, and taking the average value is average pooling. While the Batch Normalization (BN) layer is added to a sequential model to standardize the input or the outputs, it provides each network layer to learn more independently. In normalization, the input layer is scaled by the activations. The normalization layer is usually set just after the convolution and pooling layers.

The first stage of our model is a CNN with seven layers (see Table 2). These layers are trained to select essential features from the input image. The CNN takes an image of size (800×64) and returns a features sequence of size (100×512). Each model's layer consists of a convolution (with 5×5 or 3×3 kernel) followed or not by one of the pooling (a 2×2 or 1×2 pooling) or BN operations. A dropout is added at the end of each layer to prevent over-fitting in the model. They are added to randomly and temporarily remove some percentage (at a given rate, see Table 2) of neurons and their connections.

3.2. Att-BLSTM

A Bidirectional Long Short-Term Memory Network (BLSTM) is a neural network model for training sequential data. It uses two isolated LSTMs, a forward LSTM reading the input sequence from left to right and a backward LSTM reading the sequence from right to left. The LSTM model was first proposed in [37] to defeat the gradient vanishing problem. The BLSTM model was introduced to get high-level features from the input features sequence. Further, the BLSTM networks extend the LSTMs by including a second layer, where the hidden-to-hidden connections flow in an opposite temporal order. The model is then able to manipulate past and future information.

In the proposed AHTR, we used an Attention-based BLSTM (see Figure 5) with 512 hidden cells for each LSTM. It employs a neural attention mechanism to extract relevant information from an input features sequence and explicitly includes the potential of

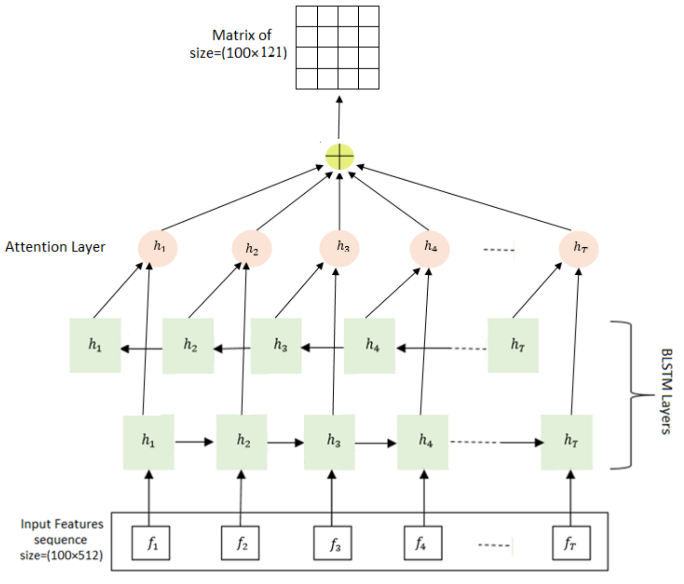


Fig. 5. An overview of the used Att-BLSTM architecture.

handling different writing styles. Attention-based networks have shown striking success in various deep learning domains. The attention mechanism is applied to an image to search in specific regions like a human when looking for a particular pattern, orientating himself to specific zones in the image.

The BLSTM includes two LSTMs, which are forward and backward, respectively. It takes as input a feature map with a size of (100×512), 512 features per time-step, and outputs a matrix consisting of combined LSTM output vectors of size (100×1024); 1024 for each time step. The attention block produces a weighted vector and multiplies features from each step by the latter. It gives an output sequence, which is mapped to a matrix of size (100×121), where 100 is the text line max length and 121 is the number of characters contained in the IFN/ENIT dataset considering the black character ('-'). The number 121 is also the number of classes C, so the length of the labeled axis in the CTC layer will be 121. Therefore there are 121 entries for each of the 100 time steps.

3.3. CTC

As mentioned before, our model comprises a CNN, an RNN, and a CTC. The CNN allows feature sequence extraction from input images, and the RNN propagates information through this sequence and produces a matrix of character scores for each sequence element. The CTC is set up for sequence labeling without input segmentation. In other words, the CTC is a softmax layer, which produces probabilities corresponding to all the possible label alignments of the input sequence with length T, where T is the length of the probabilities sequence fed to the CTC function. for all steps. It interferes in two steps: 1) during training, it takes the Att-BLSTM output matrix and the ground truth text to calculate the loss value, and 2) while inference, where it takes just the output matrix and provides the predicted text with a max length of 100 characters. The CTC was first introduced in [32]. Let us detail how it works in the rest of this section. It is essential to know that while training, the CTC takes the output matrix and the ground truth text and does not try to learn each ground truth text's character position. Still, it tries all possible alignments of the ground truth text in the image and calculates the sum of the scores. The ground truth text's score is high if only the sum of the alignment scores is high.

There are three CTC fundamental functions to detail:

1. **Text encoding**: To defeat the significant problem of the database annotation when characters take more than one time-step in the image and duplicated characters are provided (an example is given in Figure 6). The CTC is set up to overcome this problem by representing redundant characters with a single one. It uses alphabet labels (composed of all characters that occurred in the training data). The CTC adds a blank label (indicated by '-') to specify no label at a particular time position in the output sequence. For example, in Figure 6b, the CNN result (annotation) for the input image is '-ننققة-', which will be decoded to 'نقة'.
2. **Loss calculation**: The CTC calculates the loss function and backpropagates it to the NN to restart the end-to-end learning process (training). As mentioned before, the Att-BLSTM produces a matrix of scores for each character at each time step. The loss is calculated by adding all scores of all possible alignments of the ground truth text. Figure 7 gives a simple example for two-time steps with a reduced matrix (where the alphabet is composed of three characters {أ, ي, –}. Supposing that the current ground truth is "ي," then all feasible paths are: "يي", "–ي" and "ي–." The probability of the ground truth occurring is calculated by: $P = 0.12 + 0.24 + 0.18 = 0.54$, where corresponding character scores are multiplied to get the score for one path. The CTC applies the negative logarithm to the obtained probability to calculate the loss value L: $L = -\log(P)$.
3. **Text decoding**: For CTC decoding (inference), we used the best path decoding method to decode the output probability matrix as follows:
   (a) For each time step, it determines the character with the maximum probability, and at the final time step, the best path is calculated.
   (b) To determine the resulting text, it removes blanks and duplicated characters.

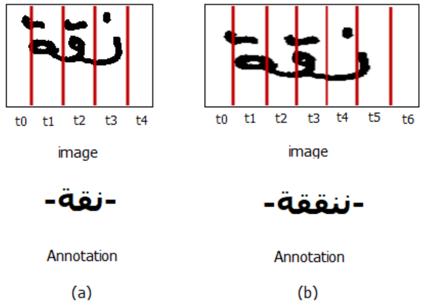


Fig. 6. (**a**) Image annotation where each character takes up one time-step. (**b**) Image annotation where a few characters take up more than one time-step.

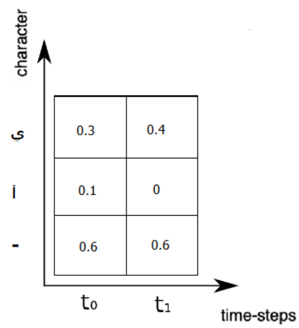


Fig. 7. The Att-BLSTM output matrix representing the character probability at each time step.

3.4. Configuration

We implemented the proposed offline Arabic handwritten text recognition model on a Hp Z-440 workstation with 16 GB RAM using Keras [34] (a Python deep learning library created by Google).

This section gives a detailed description of the proposed model architecture's configuration: the general hyperparameters, the CNN hyperparameters, the RNN hyperparameters, the attention blocks, and the CTC hyperparameters. The general hyperparameters provide those for the proposed model and the configuration network, which enclose the optimizer, learning rate, batch size, number of convolution layers, number of LSTM layers, the attention mechanism, and the CTC configuration. These hyperparameters are illustrated as follows:

- **Optimizer**: The adaptive learning rate optimization algorithm incrementally updates the CNN's weights after each epoch passes over the training dataset. The optimization algorithm used in our experiments is the Adam optimizer, presented by Diederik Kingma from OpenAI and Jimmy Ba from the University of Toronto in 2014 [44]. We choose it because it is a popular algorithm in the deep learning area and reaches good results fast. We set the Adam parameters as follows:
  - **Alpha:** is the learning rate or step size. The proportion that weights are updated was set to 10^{-4}.
  - **Beta1:** is the exponential decay rate for the first moment estimates is set to 0.9, the default value in the Keras deep learning library.
  - **Beta2:** is the exponential decay rate for the second-moment estimates is set to 0.999, the default value in the Keras library.
  - **Epsilon:** is a tiny number to prevent any division by zero in the implementation, and it was set to its default value, which is 10^{-7}.
- **Learning rate:** the learning rate controls how much to modify the model in reply to the calculated error each time the model weights are updated. Selecting a reasonable learning rate is a difficult task, where a too-small value can lead to a lengthy training process that might get stuck. At the same time, a too-large value can lead to learning a set of sub-optimal weights too fast or to an inconsistent training process. In our experiments, the learning rate is set to 10^{-4}.
- **Batch size:** it presents the total number of training samples presented in a single batch; in our training, we set it to 260.
- **The CNN configuration:**
  - **The number of convolution layers:** since handwriting recognition is a complex task, our CNN must be a deep architecture; with more than three layers to guarantee a good features extraction from the input images. We limited our model to 7 layers since an enormous number of layers can increase the number of weights and the complexity of the model.
  - **Regularization**: more precisely, the dropout; to handle the problem of overfitting. It is a simple method that randomly drops nodes out of the network, and it has a regularizing impact as the remaining nodes should adjust to pick up the slack of the removed nodes. So a dropout is added at the end of each layer (see Table 2 for the dropout rates setting).
  - **Convolution batch normalization**: is used to improve a neural network's performance and is designed to automatically standardize the inputs to a deep learning neural network layer. In our CNN architecture, we added batch normalization to the third and sixth layers between the convolution and max-pooling operations, and we used it in a binary range $(0, 1)$. The layer will transform inputs to be standardized, meaning they will have a mean of zero and a standard deviation of one. The momentum parameter is set to its default value, 0.99.

  - **Convolution activation function**: In a network layer, the activation function specifies how the weighted sum of the input is transformed into an output of one single or multiple nodes. The activation function controls how sufficiently the network model learns the training dataset in the hidden layers. In the output layer, the activation function defines the type of predictions the model should make. In our CNN, we used the *ReLU* function for the hidden layers and the last layer since the last layer is not making predictions. It produces $100 \times 1 \times 512$ vectors that will be first passed to a collapse layer to remove dimension to 100×512; this feature matrix will be passed as input to the RNN block.
  - **Convolution kernel size:** Each convolution kernels' number belongs to the set $\{1, 64, 128, 256, 512\}$.
  - **Convolution kernels:** Each convolution kernel's size belongs to the set $\{1, 2, 4, 8, 16, 32, 64\}$.
- **The number of LSTM layers:** We used two LSTMs with 512 hidden cells for each one in the proposed architecture.
- **Attention mechanism:** Integrating attention mechanisms into deep-learning applications has proven to be a notable improvement in many applications, such as image recognition and machine translation. It is added to deep learning models to select which information to reserve to achieve the best use of the limited resources. More precisely, it is the power to dynamically select and use the essential parts of the available information as a human brain does. We added an Attention layer after the BLSTM using Keras in the proposed architecture. We set the `return_sequences` parameter to `True` when creating the BLSTM model to return the hidden units' output for all the previous time steps. The attention is calculated by a weighted sum of the value vectors resulting from the RNN.
- **CTC configuration:** The CTC translates a prediction into a label sequence. Its input is a sequence of observations, and the outputs are a sequence of labels, including blank outputs. In our architecture, the sequence max length is 100, the class number is 121, and the used decoder is the `bestpath` decoder.

The global structure of the used model is shown in Figure 4, and a detailed architecture is presented in Table 2.

4. Experimental Results

4.1. Training

The parameters used in the training process are listed in Tab. 3.

4.1.1. Databases

We used the IFN/ENIT [57] database for the model training. The Institute of Communications Technology in Germany has created IFN in association with the National

Tab. 2. Architecture for the used CNN. Abbreviations: Pooling (Pool), batch normalization (BN), convolutional layer (Conv).

Type	**Description**	**Output size**
Input	gray-value text line image	$800 \times 64 \times 1$
Conv + Pool + Dropout	kernel 5×5, pool 2×2, rate(0.1)	$400 \times 32 \times 64$
Conv + Pool + Dropout	kernel 5×5, pool 1×2, rate(0.2)	$400 \times 16 \times 128$
Conv + BN + Pool + Dropout	kernel 3×3, pool 2×2, rate(0.25)	$200 \times 8 \times 128$
Conv + Dropout	kernel 3×3, rate(0.3)	$200 \times 8 \times 256$
Conv + Pool + Dropout	kernel 3×3, pool 2×2, rate(0.35)	$100 \times 4 \times 256$
Conv + BN + Pool + Dropout	kernel 3×3, pool 1×2, rate(0.4)	$100 \times 2 \times 512$
Conv + Pool + Dropout	kernel 3×3, pool 1×2, rate(0.45)	$100 \times 1 \times 512$
Collapse	remove dimension	100×512
Att-BLSTM	bidir., 2 layers, 512 hidden cells	100×1024
Projection	Projection into C classes	$100 \times C$
CTC	Loss calculation and decoding	≤ 100

Engineers School of Tunis (ENIT) in Tunisia. It consists of five sets: a, b, c, d, e, containing 32492 images of handwritten names of 937 Tunisian towns written by more than 1000 different writers. Furthermore, two new sets, f and s, are added for word recognition evaluation. The set f was created in Tunisia by new writers who did not build the old five sets. The set s was collected in the United Arab Emirates (UAE) by students at Sharjah.

Furthermore, we trained the model on a Latin database called IAM [54]. It contains 1066 forms of handwritten English text produced by approximately 400 different writers. They were scanned at 300 dpi resolution and saved as PNG images with 256 gray levels. A total of 82227-word instances out of 10841 words occur in this database. The dataset can be used to train and test English handwritten text recognition. We kept the same parameters except for the number of classes, fixed at 54, which is the number of different characters in the IAM database.

Tab. 3. Parameter settings

Image pre-processing	Segmenting the IFN/ENIT database to three sets: set_1 composed of sets a, b, c, and s for training, set_2 is d for validation, and set e for testing. All images are resized to ($800 \times 64 \times 1$). No other processing.
Training setting	`initial_epoch` = 0, `no_improvement` = 30, `learning_rate` = 10^{-4}, `initial_weights` = 0.01, `batch_size` = 260, Evaluation metrics: `Accuracy` and `Loss`,
CTC setting	`max_sequence_length` = 100, `Decoder_Type` = `bestpath`, and the number of classes $C = 121$

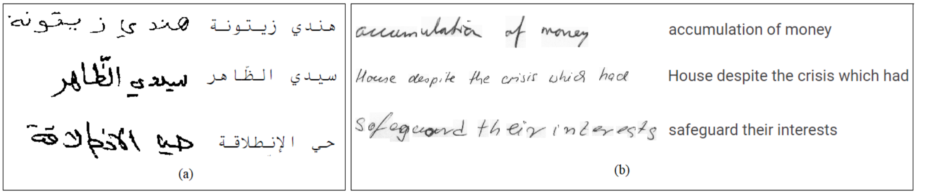


Fig. 8. (**a**) Samples from the IFN/ENIT database and their transcriptions. (**b**) Samples from the IAM database and their transcriptions.

Samples of the used databases and their transcriptions are given in Figure 8.

4.1.2. Training and Validation

We used precision and loss to evaluate the model's training on the IFN/ENIT and the IAM datasets. Table 3 displays the parameter settings when training the models.

- Training on the IFN/ENIT database:
  The training is stopped after 525 epochs, and the resulting curves show excellent training. As shown in Figure 9, the learning loss curve decreases to the point of stability, as does the validation loss plot, revealing a small gap with the learning loss plot. We conclude that the model loss is lower on the training dataset than on the validation dataset. In parallel, the accuracy plots of training and validation increase to the point of stability with a small gap. To conclude, the accuracy and loss curves show a good fit.
- Training on the IAM database:
  Figure 10 presents the loss and accuracy plots during training and validation, and they show a good fit. Training loss curves decrease slowly with slight noisy movements during the training and validation. Likewise, the training and validation accuracy curves increase gradually with subtle noisy movements to reach a height accuracy during training and validation.

4.2. Test

We reported the same evaluation metrics used in the benchmarks to facilitate comparison to other systems to access the performance of the proposed architecture on the test databases. On IAM, we show our results using Character Error Rate (CER) measures. Whereas on IFN/ENIT, we used the string accuracy metric. Recall that:

$$\mathrm{CER}(\%) = \frac{\sum_{i=1}^{n}(\mathrm{dist}_c(P_i, \mathrm{True}_i))}{\sum_{i=1}^{n}\mathrm{len}_c(\mathrm{True}_i)}\,, \tag{2}$$

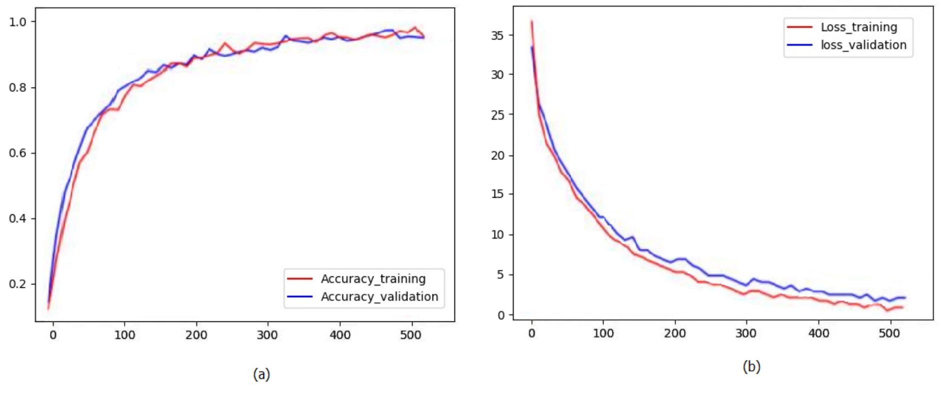


Fig. 9. Training curves: (**a**) Training and Validation Accuracy, and (**b**) Training and Validation Loss on IFN/ENIT database.

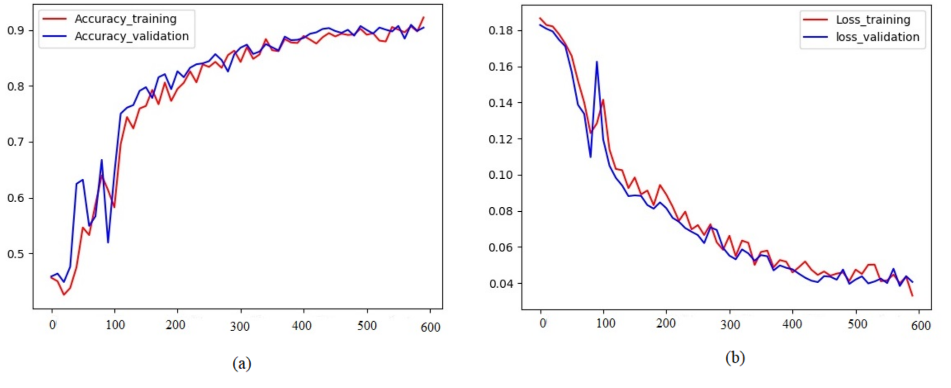


Fig. 10. Training curves: (**a**) Training and Validation Accuracy, and (**b**) Training and Validation Loss on IAM database.

where dist_c is the Levenshtein distance calculated at character level (including spaces), len_c is the number of characters in the input string, P_i is the string of characters to be recognized for the i^{th} input image and True_i is the true transcription of the i^{th} image.

$$\text{Accuracy} = \frac{\sum_{i=1}^{n}(P_i = \text{True}_i)}{n} \times 100\ [\%]\ , \tag{3}$$

where P_i is the string of characters that the model recognizes for the i^{th} input image, True_i is the true transcription of the i^{th} image, and n is the size of the test database.

Tab. 4. Performance comparisons with IFN/ENIT Database.

Methods	Architecture	Accuracy
[52]	MDLSTM-CTC	88.38%
[29]	CNN-SVM	92.95%
[39]	Dynamic Bayesian Network	82%
[8]	CNN-HMM	89.23%
[11]	HMM(128 Mixtures)	93%
[53]	CNN-BLSTM	92.21%
[36]	ANN	87.10%
[1]	RNN	94.45%
[30]	DBN + CDBN	96.23%
[52]	MDLSTM with dropout	94.65%
[27]	CNN	94.9%
[43]	Bayesian + CNN	95.2%
[61]	CNN + RNN	96.75%
Proposed Method	**CNN + Att-BLSTM + CTC + dropout**	**97.1%**

Table 4 gives recognition rates of other systems tested on the IFN/ENIT database. It is shown that our result is very prominent compared to other systems. Our trained model gives an impressive recognition rate of 97.1% for the testing set, whereas Table 5 shows CER rates of other systems tested on the IAM database. A CER of 2.9 % is attained, confirming the model's script independence. Interestingly, our system shows a clear advantage over the other systems tested on the IAM database. This result has further strengthened our conviction that the proposed model is script-independent.

Figure 11 presents examples of the model's inferences on English, French, and Italian handwritten text images.

5. Conclusion

Handwriting recognition is a dynamic field of research that regularly needs an accuracy increase. In this work, we proposed a deep learning approach based on a CNN-Att-BLSTM-CTC architecture with dropout and batch-normalization to recognize Arabic handwritten text accurately in images. Texts can be of various sizes and writing styles. We improved the used CNN with dropout, temporarily removing some units from the network to prevent the system from over-fitting problems. The CNN is used for sequence feature extraction and passes its output to the ATT-BLSTM to propagate information. For each sequence element, it outputs a matrix of character scores. Then the CTC operation is set up to calculate the loss value, train the proposed model, and to perform inference at this stage. The CTC decodes the Att-BLSTM's output matrix to infer the

Tab. 5. Performance comparisons with IAM Database.

Methods	Architecture	CER
[15]	CNN + BLSTM + CTC	3.2%
[65]	MDLSTM + CTC	3.5%
[16]	MDLSTM + MLP/HMM	3.6%
[12]	MDLSTM + CTC	4.4%
[59]	CNN + LSTM+CTC	4.4%
[13]	MDLSTM + Attention	4.4%
[40]	Transformer	4.6%
[22]	LSTM + HMM	4.7%
[66]	LSTM + HMM	4.8%
[55]	CNN + LSTM + Attention	4.8%
[67]	CNN + CTC	4.9%
[21]	CNN + LSTM + Attention	4.9%
[45]	LSTM + HMM	5.1%
[58]	MDLSTM + CTC	5.1%
[35]	CNN + BLSTM + Attention + CTC	5.1%
[24]	CNN + BLSTM	5.7%
[41]	CNN + BGRU + GRU + Attention	5.7%
[38]	CNN + CTC	6.1%
[14]	MDLSTM + CTC	6.6%
[42]	CNN + BGRU + GRU	6.8%
[20]	CNN + BLSTM + LSTM	8.1%
[46]	GMM/HMM	8.2%
[64]	CNN + LSTM + Attention	8.8%
[47]	CNN + LSTM + CTC	9.7%
[31]	MLP/HMM	9.8%
[18]	MDLSTM + CTC	11.1%
[23]	MLP/HMM	12.4%
[51]	MDLSTM + CTC	17.0%
[50]	BLSTM + CTC	18.2
Proposed method	**CNN + Attention-Blstm + CTC + dropout + BN**	**2.9%**

text contained in the input image. The proposed model is validated on the IFN/ENIT database. According to the experimental results, the accuracy reaches 97.1%. The feature extraction, training, and recognition components of the model are all designed to be script-independent. The model parameters are estimated automatically from the training data without the need for laborious handwritten rules. It requires no pre-segmentation of the data, either at the word level or at the character level. Thus, it can handle languages with cursive handwritten scripts straightforwardly. We tested it on the IAM database, and a CER of 2.9% was attained.

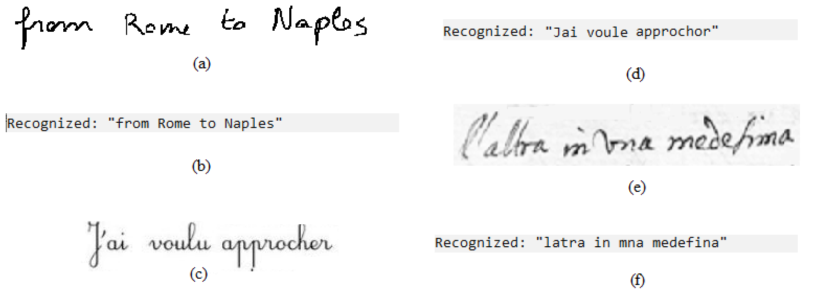


Fig. 11. Model's results on handwritten Latin text lines; where (**a**) and (**b**) are the handwritten English text image, and its recognition result, (**c**) and (**d**) are the handwritten French text image, and its recognition result, respectively, and (**e**) and (**f**) are handwritten Italian text image and its recognition result.

In future work, we intend to train the developed model on a complex database, such as HACDB, KHATT, etc., to improve our model's performance, employing deeper models. We also aim to extend the set of input images to recognize text lines with larger sizes and inclinations. We also plan to add a second Att-BLSTM before the CTC layer to improve the results.

Tab. 6. Table of Abbreviations.

AHTR	Arabic Handwritten Text Recognition
APTI	Arabic Printed Text Image database
Att	Attention
BLSTM	Bidirectional Long Short Term Memory
BN	Batch Normalization
CDBN	Convolutional Deep Belief Network
CER	Character Error Rate
CNN	Convolutional Neural Network
Conv	Convolutional layer
CRR	Character Recognition Rate
CTC	Connectionist Temporal Classification
DCNN	Deep CNN
GRU	Gated Recurrent Unit
HMM	Hidden Markov Model
LSTM	Long Short Term Memory
MDLSTM	Multi-Dimensional Long Short Term Memory
NN	Neural Network
OCR	Optical Character Recognition
Pool	Pooling
RNN	Recurrent Neural Network
RBF	Radial Basis Function

References

[1] G. A. Abandah, F. T. Jamour, and E. A. Qaralleh. Recognizing handwritten Arabic words using grapheme segmentation and recurrent neural networks. *International Journal on Document Analysis and Recognition*, 17(3):275–291, 2014. doi:10.1007/s10032-014-0218-7.

[2] R. Ahmad, M. Z. Afzal, S. F. Rashid, M. Liwicki, T. Breuel, and A. Dengel. KPTI: Katib's Pashto text imagebase and deep learning benchmark. In *Proc. 2016 15th Int. Conf. Frontiers in Handwriting Recognition (ICFHR)*, pages 453–458, Shenzhen, China, 23-26 Oct 2016. IEEE. doi:10.1109/ICFHR.2016.0090.

[3] R. Ahmad, S. Naz, M. Z. Afzal, et al. KHATT: A deep learning benchmark on Arabic script. In *Proc. 2017 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 10–14, Kyoto, Japan, 9-15 Nov 2017. IEEE. doi:10.1109/ICDAR.2017.358.

[4] R. Ahmad, S. Naz, M. Z. Afzal, et al. A deep learning based Arabic script recognition system: Benchmark on KHAT. *International Arab Journal of Information Technology*, 17(3):299–305, 2020. doi:10.34028/iajit/17/3/3.

[5] R. Ahmed, K. Dashtipour, M. Gogate, A. Raza, et al. Offline Arabic handwriting recognition using deep machine learning: A review of recent advances. In *Advances in Brain Inspired Cognitive Systems. Proc. Int. Conf. Brain Inspired Cognitive Systems (BICS) 2019*, pages 457–468, Guangzhou, China, 13-14 Jul 2019. 2020. Springer International Publishing. doi:10.1007/978-3-030-39431-8_44.

[6] R. Ahmed, M. Gogate, A. Tahir, et al. Novel deep convolutional neural network-based contextual recognition of Arabic handwritten scripts. *Entropy*, 23(3):340, 2021. doi:10.3390/e23030340.

[7] A. A. Al Rababah. Neural networks precision in technical vision systems. *International Journal of Computer Science and Network Security*, 20(3):29–36, 2020. `http://paper.ijcsns.org/07_book/202003/20200305.pdf`.

[8] M. Amrouch, M. Rabi, and Y. Es-Saady. Convolutional feature learning and CNN based HMM for Arabic handwriting recognition. In *Image and Signal Processing. Proc. Int. Conf. Image and Signal Processing (ICISP) 2018*, volume 9887 of *Lecture Notes in Computer Science*, pages 265–274, Cherbourg, France, 2-4 Jul 2018. Springer. doi:10.1007/978-3-319-94211-7_29.

[9] Z. Asebriy, S. Raghay, O. Bencharef, and Y. Chihab. Comparative systems of handwriting Arabic character recognition. In *Proc. 2014 2nd World Conf. Complex Systems (WCCS)*, pages 90–93, Agadir, Morocco, 10-12 Nov 2014. doi:10.1109/ICoCS.2014.7060923.

[10] M. Awni, M. I. Khalil, and H. M. Abbas. Offline Arabic handwritten word recognition: A transfer learning approach. *Journal of King Saud University – Computer and Information Sciences*, 34(10, Part B):9654–9661, 2022. doi:10.1016/j.jksuci.2021.11.018.

[11] S. A. Azeem and H. Ahmed. Effective technique for the recognition of offline Arabic handwritten words using hidden markov models. *International Journal on Document Analysis and Recognition*, 16(4):399–412, 2013. doi:10.1007/s10032-013-0201-8.

[12] T. Bluche. *Deep Neural Networks for Large Vocabulary Handwritten Text Recognition.* PhD thesis, Université Paris 11, 2015.

[13] T. Bluche. Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. In *Advances in Neural Information Processing Systems 29 – Proc. 30th Conf. NIPS 2016*, volume 29, pages 838–846, Barcelona, Spain, 5-10 Dec 2019. Curran Associates, Inc. `https://proceedings.neurips.cc/paper/2016/file/2bb232c0b13c774965ef8558f0fbd615-Paper.pdf`.

[14] T. Bluche, J. Louradour, and R. Messina. Scan, Attend and Read: End-to-end handwritten

paragraph recognition with MDLSTM attention. In *Proc. 2017 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 1050–1055, Kyoto, Japan, 9-15 Nov 2017. IEEE. doi:10.1109/ICDAR.2017.174.

[15] T. Bluche and R. Messina. Gated convolutional recurrent neural networks for multilingual handwriting recognition. In *Proc. 2017 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 646–651, Kyoto, Japan, 9-15 Nov 2017. IEEE. doi:10.1109/ICDAR.2017.111.

[16] D. Castro, B. L. D. Bezerra, and M. Valença. Boosting the deep multidimensional long-short-term memory network for handwritten recognition systems. In *Proc. 2018 16th Int. Conf. Frontiers in Handwriting Recognition (ICFHR)*, pages 127–132, Niagara Falls, NY, USA, 5-8 Aug 2018. IEEE. doi:10.1109/ICFHR-2018.2018.00031.

[17] L. Chen, R. Yan, L. Peng, A. Furuhata, and X. Ding. Multi-layer recurrent neural network based offline Arabic handwriting recognition. In *Proc. 2017 1st Int. Workshop on Arabic Script Analysis and Recognition (ASAR)*, pages 6–10, Nancy, France, 3-5 Apr 2017. IEEE. doi:10.1109/ASAR.2017.8067749.

[18] Z. Chen, Y. Wu, F. Yin, and C.-L. Liu. Simultaneous script identification and handwriting recognition via multi-task learning of recurrent neural networks. In *Proc. 2017 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 525–530, Kyoto, Japan, 09-15 Nov 2017. IEEE. doi:10.1109/ICDAR.2017.92.

[19] Y. Chherawala, P. P. Roy, and M. Cheriet. Feature design for offline Arabic handwriting recognition: Handcrafted vs automated? In *Proc. 2013 12th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 290–294, Washington, DC, USA, 25-28 Aug 2013. IEEE. doi:10.1109/ICDAR.2013.65.

[20] A. Chowdhury and L. Vig. An efficient end-to-end neural model for handwritten text recognition. *arXiv*, 2018. arXiv:1807.07965v2. doi:10.48550/arXiv.1807.07965.

[21] D. Coquenet, C. Chatelain, and T. Paquet. End-to-end handwritten paragraph text recognition using a vertical attention network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):508–524, 2022. doi:10.1109/TPAMI.2022.3144899.

[22] P. Doetsch, M. Kozielski, and H. Ney. Fast and robust training of recurrent neural networks for offline handwriting recognition. In *Proc. 2014 14th Int. Conf. Frontiers in Handwriting Recognition (ICFHR)*, pages 279–284, Hersonissos, Greece, 01-04 Sep 2014. IEEE. doi:10.1109/ICFHR.2014.54.

[23] P. Dreuw, P. Doetsch, C. Plahl, and H. Ney. Hierarchical hybrid MLP/HMM or rather MLP features for a discriminatively trained gaussian HMM: A comparison for offline handwriting recognition. In *2011 18th IEEE Int. Conf. Image Processing (ICIP)*, pages 3541–3544, Brussels, Belgium, 11-14 Sep 2011. IEEE. doi:10.1109/ICIP.2011.6116480.

[24] K. Dutta, P. Krishnan, M. Mathew, and C.V. Jawahar. Improving CNN-RNN hybrid networks for handwriting recognition. In *Proc. 2018 16th Int. Conf. Frontiers in Handwriting Recognition (ICFHR)*, pages 80–85, Niagara Falls, NY, USA, 5-8 Aug 2018. IEEE. doi:10.1109/ICFHR-2018.2018.00023.

[25] B. El Qacimy, A. Hammouch, and M. A. Kerroum. A review of feature extraction techniques for handwritten Arabic text recognition. In *Proc. 2015 Int. Conf. Electrical and Information Technologies (ICEIT)*, pages 241–245, Marrakech, Morocco, 25-27 Mar 2015. IEEE. doi:10.1109/EITech.2015.7162979.

[26] B. El Qacimy, M. A. Kerroum, and A. Hammouch. Word-based Arabic handwritten recognition using SVM classifier with a reject option. In *Proc. 2015 15th Int. Conf. Intelligent Systems Design and Applications (ISDA)*, pages 64–68, Marrakech, Morocco, 14-16 Dec 2015. IEEE. doi:10.1109/ISDA.2015.7489190.

[27] A. El-Sawy, M. Loey, and H. El-Bakry. Arabic handwritten characters recognition using convolutional neural network. *WSEAS Transactions on Computer Research*, 5:11–19, 2017. `https://www.wseas.com/journals/articles.php?id=3300`.

[28] M. Elleuch and M. Kherallah. Convolutional deep learning network for handwritten arabic script recognition. In *Proc. Int. Conf. Hybrid Intelligent Systems (HIS 2019)*, volume 1179 of *Advances in Intelligent Systems and Computing*, pages 103–112, Sehore, India, 10-12 Dec 2019. Springer. doi:10.1007/978-3-030-49336-3_11.

[29] M. Elleuch, R. Maalej, and M. Kherallah. A new design based-SVM of the CNN classifier architecture with dropout for offline Arabic handwritten recognition. *Procedia Computer Science*, 80:1712–1723, 2016. doi:10.1016/j.procs.2016.05.512.

[30] M. Elleuch, N. Tagougui, and M. Kherallah. Deep learning for feature extraction of Arabic handwritten script. In *Computer Analysis of Images and Patterns. Proc. Int. Conf. Computer Analysis of Images and Patterns (CAIP) 2015*, volume 9257 of *Lecture Notes in Computer Science*, pages 371–382, Valletta, Malta, 2-4 Sep 2015. Springer. doi:10.1007/978-3-319-23117-4_32.

[31] S. España-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez. Improving offline handwritten text recognition with hybrid HMM/ANN models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):767–779, 2010. doi:10.1109/TPAMI.2010.141.

[32] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML '06: Proc. 23rd Int. Conf. Machine Learning*, pages 369–376, Pittsburgh, PA, USA, 25-29 Jun 2006. doi:10.1145/1143844.1143891.

[33] A. Graves and J. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in Neural Information Processing Systems 21 – Proc. 22nd Conf. NeurIPS 2008*, volume 21, pages 545–552. Curran Associates, Inc., 2008. `https://proceedings.neurips.cc/paper/2008/file/66368270ffd51418ec58bd793f2d9b1b-Paper.pdf`.

[34] Keras Special Interest Group. Keras. simple. flexible. powerful. `https://keras.io`.

[35] L. Gui, X. Liang, X. Chang, and A. G. Hauptmann. Adaptive context-aware reinforced agent for handwritten text recognition. In *Proc. 29th British Machine Vision Conference (BMVC) 2018*, volume 207, Newcastle, United Kingdom, 3-6 Sep 2018. British Machine Vision Association and Society for Pattern Recognition. `http://bmvc2018.org/contents/papers/0628.pdf`.

[36] S. Haboubi, S. Maddouri, N. Ellouze, and H. El-Abed. Invariant primitives for handwritten Arabic script: A contrastive study of four feature sets. In *Proc. 2009 10th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 691–697, Barcelona, Spain, 26-29 Jul 2009. IEEE. doi:10.1109/ICDAR.2009.281.

[37] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi:10.1162/neco.1997.9.8.1735.

[38] X. Huang, L. Qiao, W. Yu, et al. End-to-end sequence labeling via convolutional recurrent neural network with a connectionist temporal classification layer. *International Journal of Computational Intelligence Systems*, 13(1):341–351, 2020. doi:10.2991/ijcis.d.200316.001.

[39] K. Jayech, M. A. Mahjoub, and N. E. B. Amara. Arabic handwritten word recognition based on dynamic Bayesian network. *International Arab Journal of Information Technology*, 13(6B):1024–1031, 2016. doi:10.34028/iajit/16/13/6B. `https://iajit.org/PDF/Vol.13,No.3/7681.pdf`.

[40] L. Kang, P. Riba, M. Rusiñol, et al. Pay attention to what you read: Non-recurrent handwritten text-line recognition. *Pattern Recognition*, 129:108766, 2022. doi:10.1016/j.patcog.2022.108766.

[41] L. Kang, P. Riba, M. Villegas, et al. Candidate fusion: Integrating language modelling into a

sequence-to-sequence handwritten word recognition architecture. *Pattern Recognition*, 112:107790, 2021. doi:10.1016/j.patcog.2020.107790.

[42] L. Kang, J. I. Toledo, P. Riba, et al. Convolve, attend and spell: An attention-based sequence-to-sequence model for handwritten word recognition. In *Proc. 40th German Conf. Pattern Recognition (GCPR) 2018*, volume 11269 of *Lecture Notes in Computer Science*, pages 459–472, Stuttgart, Germany, 9-12 Oct 2018. Springer. doi:10.1007/978-3-030-12939-2_32.

[43] A. Khémiri, A. K. Echi, and M. Elloumi. Bayesian versus convolutional networks for Arabic handwriting recognition. *Arabian Journal for Science and Engineering*, 44(11):9301–9319, 2019. doi:10.1007/s13369-019-03939-y.

[44] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. 3rd Int. Conf. Learning Representations, ICLR 2015*, San Diego, CA, 7-9 May 2015. Accessible in arXiv. doi:10.48550/arXiv.1412.6980.

[45] M. Kozielski, P. Doetsch, and H. Ney. Improvements in RWTH's system for off-line handwriting recognition. In *Proc. 2013 IAPR 12th Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 935–939, Washington, DC, USA, 25-28 Aug 2013. IEEE. doi:10.1109/ICDAR.2013.190.

[46] M. Kozielski, D. Rybach, S. Hahn, et al. Open vocabulary handwriting recognition using combined word-level and character-level language models. In *Proc. 2013 IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, pages 8257–8261, Vancouver, Canada, 26-31 May 2013. IEEE. doi:10.1109/ICASSP.2013.6639275.

[47] P. Krishnan, K. Dutta, and C. V. Jawahar. Word spotting and recognition using deep embedding. In *Proc. 2018 13th IAPR Int. Workshop on Document Analysis Systems (DAS)*, pages 1–6, Vienna, Austria, 24-27 Apr 2018. IEEE. doi:10.1109/DAS.2018.70.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. doi:10.1145/3065386.

[49] A. Lawgali, M. Angelova, and A. Bouridane. HACDB: Handwritten Arabic characters database for automatic character recognition. In *Proc. European Workshop on Visual Information Processing (EUVIP)*, pages 255–259, Paris, France, 10-12 Jun 2013. `https://ieeexplore.ieee.org/abstract/document/6623974`.

[50] M. Liwicki, A. Graves, and H. Bunke. Neural networks for handwriting recognition. In *Computational Intelligence Paradigms in Advanced Pattern Classification*, volume 386 of *Studies in Computational Intelligence*, pages 5–24. Springer, 2012. doi:10.1007/978-3-642-24049-2_2.

[51] J. Louradour and C. Kermorvant. Curriculum learning for handwritten text line recognition. In *Proc. 2014 11th IAPR Int. Workshop on Document Analysis Systems (DAS)*, pages 56–60, Tours, France, 07-10 Apr 2014. IEEE. doi:10.1109/DAS.2014.38.

[52] R. Maalej and M. Kherallah. Improving MDLSTM for offline Arabic handwriting recognition using dropout at different positions. In *Artificial Neural Networks and Machine Learning. Proc. Int. Conf. on Artificial Neural Networks (ICANN) 2016*, volume 9887 of *Lecture Notes in Computer Science*, pages 431–438, Barcelona, Spain, 6-9 Sep 2016. Springer. doi:10.1007/978-3-319-44781-0_51.

[53] R. Maalej and M. Kherallah. Convolutional neural network and BLSTM for offline Arabic handwriting recognition. In *Proc. 2018 Int. Arab Conf. Information Technology (ACIT)*, pages 1–6, Werdanye, Lebanon, 28-30 Nov 2018. IEEE. doi:10.1109/ACIT.2018.8672667.

[54] U.-V. Marti and H. Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5:39–46, 2002. doi:10.1007/s100320200071.

[55] J. Michael, R. Labahn, T. Grüning, and J. Zöllner. Evaluating sequence-to-sequence models for handwritten text recognition. In *Proc. 2019 IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 1286–1293, Sydney, NSW, Australia, 20-25 Sep 2019. IEEE. doi:10.1109/ICDAR.2019.00208.

[56] A. Mohsin and M. Sadoon. Developing an Arabic handwritten recognition system by means of artificial neural network. *Journal of Engineering and Applied Sciences*, 15(1):1–3, 2019. doi:10.36478/jeasci.2020.1.3.

[57] V. Märgner and H. El Abed. IFN/ENIT-database. Database of handwritten Arabic words, 2002. `http://ifnenit.com`.

[58] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *Proc. 2014 14th Int. Conf. Frontiers in Handwriting Recognition (ICFHR)*, pages 285–290, Hersonissos, Greece, 01-04 Sep 2014. IEEE. doi:10.1109/ICFHR.2014.55.

[59] J. Puigcerver. Are multidimensional recurrent layers really necessary for handwritten text recognition? In *Proc. 2017 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 67–72, Kyoto, Japan, 9-15 Nov 2017. IEEE. doi:10.1109/ICDAR.2017.20.

[60] S. F. Rashid, M.-P. Schambach, J. Rottland, and S. von der Nüll. Low resolution Arabic recognition with multidimensional recurrent neural networks. In *Proc. 4th Int. Workshop on Multilingual OCR (MOCR '13)*, pages 1–5, Washington, DC, USA, 24 Aug 2013. doi:10.1145/2505377.2505385.

[61] V. M. Safarzadeh and P. Jafarzadeh. Offline Persian handwriting recognition with CNN and RNN-CTC. In *Proc. 2020 25th Int. Computer Conf., Computer Society of Iran (CSICC)*, pages 1–10, Tehran, Iran, 1-2 Jan 2020. IEEE. doi:10.1109/CSICC49403.2020.9050073.

[62] H. Scheidl. Build a handwritten text recognition system using TensorFlow. In B. Huberman et al., editors, *Towards Data Science*, 2015. [Accessed 4 May 2022]. `https://towardsdatascience.com/build-a-handwritten-text-recognition-system-using-tensorflow-2326a3487cd5`.

[63] F. Slimane, R. Ingold, S. Kanoun, et al. A new Arabic printed text image database and evaluation protocols. In *Proc. 2009 10th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, pages 946–950, Barcelona, Spain, 26-29 Jul 2009. IEEE. doi:10.1109/ICDAR.2009.155.

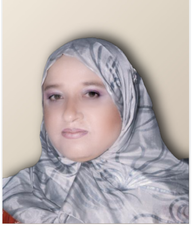
[64] J. Sueiras, V. Ruiz, A. Sanchez, and J. F. Velez. Offline continuous handwriting recognition using sequence to sequence neural networks. *Neurocomputing*, 289:119–128, 2018. doi:10.1016/j.neucom.2018.02.008.

[65] P. Voigtlaender, P. Doetsch, and H. Ney. Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In *Proc. 2016 15th Int. Conf. Frontiers in Handwriting Recognition (ICFHR)*, pages 228–233, Shenzhen, China, 23-26 Oct 2016. IEEE. doi:10.1109/ICFHR.2016.0052.

[66] P. Voigtlaender, P. Doetsch, S. Wiesler, et al. Sequence-discriminative training of recurrent neural networks. In *Proc. 2015 IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, pages 2100–2104, South Brisbane, QLD, Australia, 19-24 Apr 2015. IEEE. doi:10.1109/ICASSP.2015.7178341.

[67] M. Yousef, K. F. Hussain, and U. S. Mohammed. Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. *Pattern Recognition*, 108:107482, 2020. doi:10.1016/j.patcog.2020.107482.

Takwa Ben Aïcha Gader is a Ph.D. student and a member of the Laboratory for Technologies of Information and Communication – LaTICE, at the National High School of Engineers of Tunis. Received her engineering teleinformatics degree from the Higher Institute of Computer Science and Communication Technologies of Hammam Sousse, Sousse, Tunisia, in 2014. (ORCID: 0000-0002-3786-3649)



Afef Kacem Echi Dr. Afef Kacem Echi received M.Sc. and Ph.D. degrees in Computer Sciences from the National School of Computer Sciences of Tunis in 1997 and 2001, respectively. Since 2000, she has been an assistant in the computer science department at the Faculty of Sciences of Monastir and was appointed Assistant Professor there in 2002. Dr. Kacem is responsible for the research area of Analysis and Recognition of Handwriting and document at the Laboratory for Technologies of Information and Communication – LaTICE, at the National High School of Engineers of Tunis. She has authored over 50 papers in various national and international journals and conference proceedings. (ORCID: 0000-0001-9219-5228)