Iraqi Journal for Computer Science and Mathematics

Manuscript 1246

Liberated Arabic Handwritten Text Recognition using Convolutional Recurrent Neural Networks

Ahmad AbdulQadir AlRababah

Mohammed Khalid Aljahdali

Abdulrahim Abdulhamid Al jahdali

Mohammed Saleh AlGhanmi

Israa Ibraheem Al_Barazanchi

IRAQI JOURNAL FOR COMPUTER SCIENCE AND MATHEMATICS 2025;6:72–81

ORIGINAL STUDY

Liberated Arabic Handwritten Text Recognition using Convolutional Recurrent Neural Networks

Ahmad AbdulQadir AlRababah [a,*], **Mohammed Khalid Aljahdali** [a], **Abdulrahim Abdulhamid Al jahdali** [a], **Mohammed Saleh AlGhanmi** [a], **Israa Ibraheem Al_Barazanchi** [b]

[a] Department of Computer Science, Faculty of Computing and Information Technology in Rabigh, King Abdulaziz University, KSA
[b] College of Engineering, University of Warith Al-Anbiyaa, Karbala, Iraq

ABSTRACT

Arabic script is exhibited in a cursive style, which is a departure from the norm in many common languages, and the shapes of letters are contingent on their positions within words. The form of the first letter is influenced by the subsequent letter, middle letters are shaped by both preceding and succeeding letters, and the shape of the final letter is determined by the preceding letter. Additionally, certain letters are found to have strikingly similar shapes, making Arabic text recognition a formidable challenge in computer vision. The challenge of detecting and recognizing Arabic handwritten text is addressed in this paper by proposing a novel system that integrates two Deep Learning models without the constraints of a predefined dictionary or language model. Object Detection techniques are employed in the first model to accurately identify lines, words, and punctuation marks, providing a robust foundation for text recognition. In the second model, a powerful recognition system is developed, trained on the IFN/ENIT dataset to predict character sequences from handwritten text images. The architecture is comprised of a convolutional neural network (CNN) with residual connections, followed by a Bi-directional Long Short-Term Memory (BLSTM) layer, and culminates in a fully connected layer. State-of-the-art results in unconstrained Arabic text recognition tasks are achieved by this approach.

Keywords: Deep learning, Text recognition, Convolutional neural network, Recurrent neural network, Computer vision

1. Introduction

The majority of the text we interact with daily is in digital form, facilitated by advancements in technology. However, handwritten text continues to hold significant importance across various domains, including personal notes, official documents, letters, and historical manuscripts. These handwritten materials play a vital role in preserving cultural heritage, enabling communication, and recording information. The digitalization of handwritten text is, therefore, crucial to integrate this valuable content into modern technological ecosystems, making it more accessible, searchable, and reusable.

Handwritten text recognition remains an open and challenging research problem in the field of Computer Vision. Despite extensive research on the topic, recognizing handwritten text accurately involves addressing several complexities. For Arabic handwritten text, these challenges are further magnified. Unlike Latin-based scripts, Arabic is inherently cursive, with characters varying in shape depending on their position in a word. Furthermore, the script includes characters that are visually similar, adding another layer of complexity. The diverse styles of handwriting among individuals exacerbate the problem. Fig. 1 illustrates these challenges, emphasizing the intricate nature of Arabic handwritten text. Compounding

Received 17 October 2024; revised 12 January 2025; accepted 26 February 2025.
Available online 26 April 2025

* Corresponding author.
E-mail address: ahd_68@yahoo.com (A. A. AlRababah).

https://doi.org/10.52866/2788-7421.1246

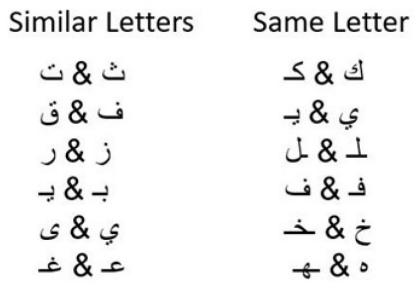

Fig. 1. Difficulties of the Arabic Language Different letters have very similar shapes, and the same letter could have very different shapes.

these difficulties is the limited availability of high-quality Arabic handwriting datasets, which hampers the development of robust models.

The advent of deep learning techniques has revolutionized the field of Computer Vision, particularly following the groundbreaking performance of convolutional neural networks (CNNs) on the ImageNet dataset. These advances have enabled researchers to tackle complex problems with remarkable success. Leveraging the power of deep learning for Arabic handwritten text recognition offers promising opportunities to overcome its unique challenges, paving the way for more effective solutions in this domain. Since then, deep learning-based models have revolutionized Computer Vision tasks, consistently achieving superior performance compared to other state-of-the-art methods. This dominance extends to text recognition, where models based on Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have set new benchmarks, surpassing traditional approaches. However, in the domain of Arabic handwritten text recognition, many of the current state-of-the-art methods face notable limitations. These models often operate under constraints, such as being trained on a limited vocabulary and relying heavily on decoders that restrict outputs to a predefined dictionary of words [1, 3]. Such constraints hinder their ability to generalize to out-of-vocabulary words or novel text styles, highlighting the need for more flexible approaches.

Motivated by these limitations, we propose an unconstrained approach to Arabic handwritten text recognition. Unlike traditional methods, our model eliminates the dependency on decoders tied to fixed dictionaries, enabling it to handle a broader range of text inputs. The foundation of our method lies in treating input images as sequences of pixel widths. The process begins with feature extraction using a CNN, producing a feature map where the number of filters corresponds to the final convolutional layer, and the width is down sampled by a factor of 4. This feature map represents a sequential encoding of the input image, which is then processed by a Bidirectional Long Short-Term Memory (BLSTM) layer to capture contextual dependencies in both forward and backward directions.

The output of the BLSTM layer is decoded through a fully connected layer that predicts the probability distribution of all possible classes for each point in the sequence. To train the model, we employ the Connectionist Temporal Classification (CTC) loss function, which is particularly suited for sequence-to-sequence problems where the input and output lengths may differ. The CTC loss allows the model to align predicted probabilities with the ground-truth targets, even when the alignment between input and output sequences is not explicitly defined. This architecture enables our model to handle Arabic handwritten text recognition in a more flexible and unconstrained manner, offering promising results for broader real-world applications. This approach allows us to handle input images of arbitrary sizes. However, the limitation of our method is that the down sampled width length must be equal to or greater than the target sequence length. This paper introduces a convolutional recurrent neural network (CRNN) architecture designed to achieve state-of-the-art performance in the challenging task of unconstrained Arabic handwritten text recognition. Unlike traditional approaches that rely on constrained decoders or predefined dictionaries, the proposed method operates in a fully unconstrained setting, demonstrating its capability to generalize across a variety of handwriting styles and vocabulary.

The paper is organized as follows: Section II provides an overview of related work in the domain of handwritten text recognition, highlighting key advancements and limitations in existing approaches. This section underscores the need for a more flexible and robust model for Arabic handwritten text recognition, setting the stage for the proposed methodology. Section III details the dataset used for training and evaluation, emphasizing its relevance to the problem domain. It also discusses the preprocessing techniques employed to standardize the input data and the augmentation strategies designed to improve the model's robustness to variations in handwriting styles, orientations, and noise. Section IV focuses on the architecture of the proposed model. It describes how the convolutional layers extract spatial features from the input images, how the recurrent layers model sequential dependencies, and how the final fully connected layers predict character probabilities. The design choices that contribute to the model's performance in an unconstrained setting are also discussed in detail. Finally, Section V outlines

the experimental setup, including the system specifications, training strategies, and hyper parameter choices. It presents a comprehensive evaluation of the model's performance, comparing it against existing state-of-the-art methods and highlighting the key factors contributing to its success. The results demonstrate the model's ability to handle unconstrained Arabic handwritten text effectively, showcasing its potential for real-world applications.

2. Related works

Recent advancements in text recognition have focused on leveraging sophisticated architectures and innovative techniques to achieve superior performance across various tasks. One notable approach employed a Bidirectional Long Short-Term Memory (BLSTM) network combined with the Connectionist Temporal Classification (CTC) loss function, along with the Token Passing and Word Beam Search (WBS) decoders [2, 4]. The study also introduced an adaptive data augmentation (ADA) algorithm, enhancing the robustness of the training process. Through extensive experimentation, they identified the optimal combination of BLSTM, CTC loss, WBS decoder, and the ADA algorithm, achieving impressive results: 95.19% Character Accuracy Rate (CAR) and 96.19% Word Accuracy Rate (WAR) using character-based models. When the ADA algorithm was excluded, performance decreased to 86.70% CAR and 83.90% WAR. However, these results relied on search decoders constrained by a predefined dictionary, limiting their applicability in unconstrained scenarios [5, 21].

Another prominent approach focused on a Convolutional Neural Network (CNN) architecture inspired by the VGG style. This architecture comprised nine convolutional layers, three fully connected layers, and max-out activation layers. Batch normalization was applied after each convolution and before each max-out activation layer, improving model generalization and stability [6, 7]. A unique feature of this architecture was the use of multiple separate and parallel fully connected layers, each responsible for distinct groups of predictions. On the IFN/ENIT dataset, this approach achieved remarkable word accuracy rates of 99.29% for the abc-d configuration and 97.07% for the abcd-e configuration [25, 31].

For unconstrained text recognition, a simple yet effective neural network architecture was proposed, employing depth-wise convolutions instead of traditional convolutions and incorporating gate blocks with attention mechanisms. These gates filtered out insignificant signals and controlled inter-layer information flow[]. Evaluated on datasets from various languages, including the Arabic KHATT dataset, this model achieved an impressive Character Error Rate (CER) of 8.7% [8, 9]. This demonstrated the potential of lightweight architectures combined with attention mechanisms for challenging unconstrained recognition tasks [21].

In the domain of holistic word classification, a CNN-based architecture was proposed specifically for classifying Arabic handwritten names. Using the SUST-ARG dataset, which consists of handwritten Arabic names, this architecture employed convolutional layers, the RuLE activation function, batch normalization, and max-pooling layers [10]. The model achieved an outstanding accuracy of 99% across 20 name classes, highlighting the efficacy of specialized CNN architectures in focused classification tasks [28].

For Arabic character classification, a CNN architecture was proposed by M. E. Mustafa and M. K. Elbashir, accompanied by the introduction of a novel dataset named Hijja. The architecture featured convolutional layers, max-pooling layers, and fully connected layers for final predictions. This model achieved an accuracy of 88%, along with a precision of 87.88%, recall of 87.81%, and an F1 score of 87.8% [11]. The Hijja dataset provided a valuable resource for advancing Arabic character recognition research, offering a benchmark for future work.

These diverse approaches illustrate significant progress and innovation in the field of Arabic handwritten text recognition. On one hand, constrained methods that leverage powerful decoders, such as BLSTM networks combined with CTC loss and Word Beam Search (WBS) have set impressive benchmarks by utilizing predefined dictionaries and vocabulary to improve accuracy. These methods excel in controlled settings but face limitations when dealing with out-of-vocabulary words or novel handwriting styles, highlighting the need for more flexible models. On the other hand, unconstrained architectures that incorporate lightweight designs and advanced techniques such as attention mechanisms and depth-wise convolutions offer greater adaptability and scalability [24, 29]. These approaches are more suited for real-world applications, as they can handle diverse writing styles and text variations without relying on fixed vocabularies.

Collectively, these methods represent a broad spectrum of solutions to the Arabic handwritten text recognition problem, each with its strengths and weaknesses. The constrained methods offer high accuracy but lack generalization, while unconstrained models emphasize flexibility and robustness. Despite

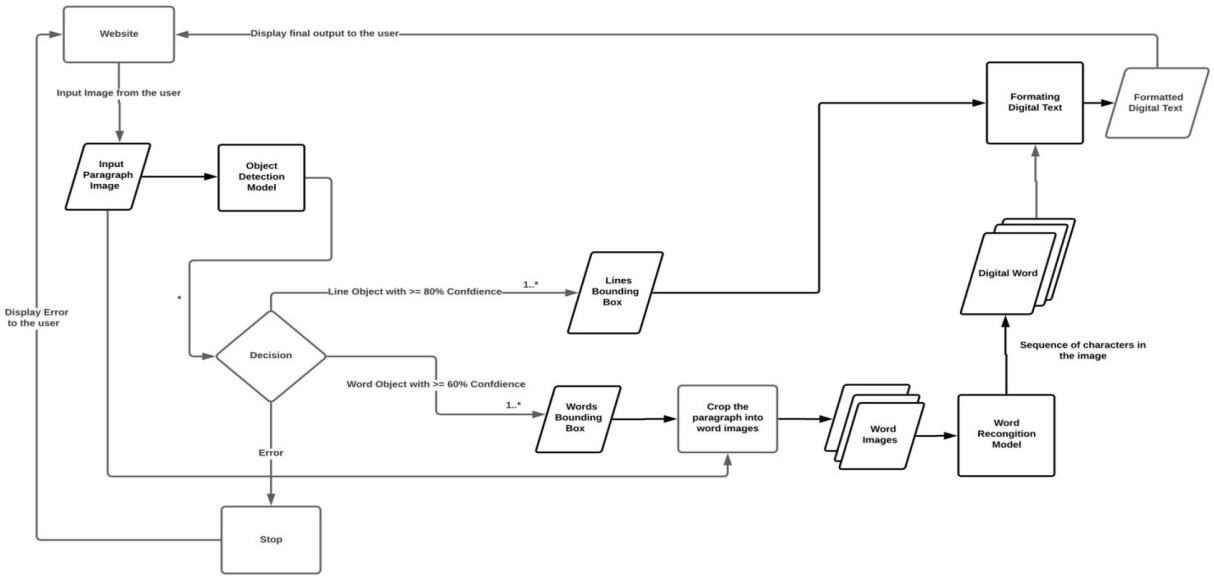

Fig. 2. System diagram.

the advances made, several challenges remain, including improving model efficiency, handling a wider range of text types, enhancing real-time performance, and ensuring robustness to noise and distortions. As research progresses, addressing these challenges will be crucial in advancing the field and creating practical, scalable solutions for real-world applications [30]. The evolution from rigid, dictionary-dependent systems to more dynamic and adaptive models signals the potential for further breakthroughs in Arabic handwritten text recognition, but it also emphasizes the complexity and multi-faceted nature of the problem.

The weaknesses of related works in Arabic handwritten text recognition primarily stem from their reliance on constrained decoders tied to predefined dictionaries, limiting their generalizability to out-of-vocabulary words and diverse handwriting styles. Many models are trained and evaluated on limited datasets that fail to capture the full variability of Arabic handwriting, such as differences in style, orientation, and noise. Some architecture is overly complex, making them unsuitable for resource-constrained environments, while others lack robust data augmentation techniques to enhance their adaptability. Additionally, there is a narrow focus on specific tasks or constrained settings, leaving unconstrained recognition and broader applications underexplored. Many methods also fail to leverage advanced techniques, such as transformers or self-supervised learning, and do not adequately address practical deployment challenges, reducing their impact on real-world applications.

3. Proposed system

A system is presented that leverages two Deep Learning models to detect and recognize Arabic handwritten text. For the detection task, an Object Detection model is employed, which is specifically designed to identify lines, words, and punctuation marks within the handwritten text. This model is trained using paragraph images from the KHATT dataset, where each image is manually labeled with bounding box coordinates corresponding to the various classes (lines, words, and punctuation marks) present in the image (Fig. 2). A custom function is then implemented to crop word images from the paragraph images based on the predicted bounding boxes for the words. These cropped images are then prepared for the subsequent recognition step.

For the recognition task, a separate recognition model is trained on the IFN/ENIT dataset to predict the sequence of characters within each word image. The model uses the Connectionist Temporal Classification (CTC) objective function to optimize the character sequence predictions without the need for explicit alignment between the input image and the output text [11]. This approach enables the model to efficiently handle the complex nature of handwriting, where letter shapes can vary depending on their position within a word. By using CTC, the model can learn to output a sequence of characters from the image without relying on a fixed sequence length or explicit character segmentation.

The system's modular approach allows for significant improvements in performance, achieving higher

accuracy on both tasks while reducing memory usage and computational demands. Once the word images are cropped and processed by the recognition model, the predicted digital text is formatted based on the location of the bounding boxes that define the words and lines in the original paragraph image. This ensures that the spatial structure of the text is preserved in the final output.

A user-friendly website is developed to allow users to upload Arabic handwritten text images. Upon uploading, the system processes the images, performs detection and recognition, and then displays the formatted output, which contains the recognized Arabic text arranged according to the original layout. This web-based solution provides an accessible interface for users to interact with the system and obtain their recognized text with minimal effort.

4. Dataset and preprocessing

4.1. Dataset

It was used the IFN/ENIT dataset, which is considered a benchmark in the field of Arabic handwritten text recognition. It is composed of 946 Tunisian town/village names, written by more than 400 people. The dataset is split into 5 subsets: a, b, c, d, and e, and there are 3 train/test configurations which are: abc/d, bcd/a, and abcd/e [12].

4.2. Preprocessing

The ground-truth preprocessing is done to reformat the ground-truth files. In the case of the IFN/ENIT dataset, the ground truth is provided in terms of character shapes as modeling units and not in terms of character as modeling units, also in their labeling they do not consider space between two words as a character or a modeling unit. For instance, in an image of a town name that has more than one word, there is no indication that a word ended and another has started. Moreover, there is an addition of 'llL' on any ground truth character shape that has a 'shadda' see Fig. 3. It was changed this format into character as a modeling unit, add a space between words, and remove the 'shadda' indicators. As for input images preprocessing, also normalize (Normalization of images is making their pixel values between 1 and 0) images by dividing their pixel values by 255, this helps the model learn faster and better; because, neural networks process inputs using small weight values, and inputs with large values can disrupt or slow down the learning process [14]. Moreover the images resized to a fixed height and dynamic width to preserve their aspect ratio. When feed a batch of images to the

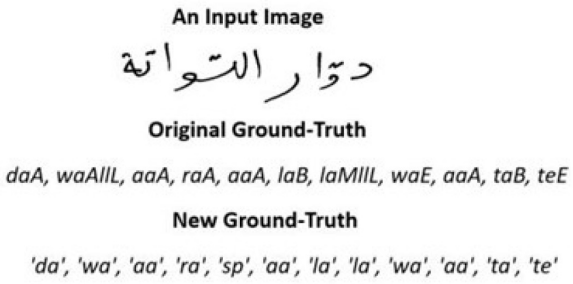

Fig. 3. The difference between the original provided ground-truth and our new ground-truth.

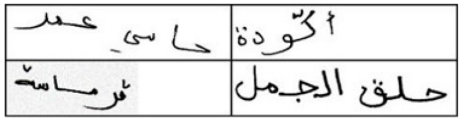

Fig. 4. An example of a batch of images, with different widths. All the images are padded with white space on the right to have the same width as the widest image.

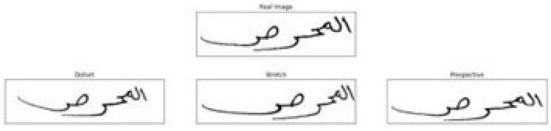

Fig. 5. Real image vs augmented images by general geometric augmentation for text images.

word recognition model, done also changes in width of all the images in the batch to have the same width as the widest image by adding white padding to the smaller images see Fig. 4.

4.3. Augmentation

As for input augmentation, it was used a general geometric augmentation for text images, this augmentation helps to create more images that realistically appear as written by a different writer see Fig. 5. Furthermore, we use arithmetic image augmentations, by adding a random value to each pixel, inverting the pixel values, or multiplying each pixel with a random value, these arithmetic operations result in changing the background color, text color, or both of them see Fig. 6. Also, used standard image augmentations Gaussian noise, Poisson noise, rotating, and shearing see Fig. 7. Overall, we use these augmentations to improve the generalization capability of the model and reduce overfitting [13, 15, 16].

5. Methods

The proposed architecture integrates Convolutional Neural Networks (CNNs) with residual connections

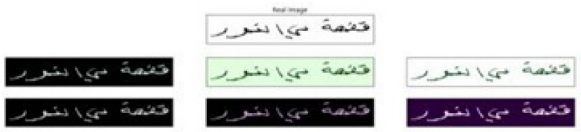

Fig. 6. Real image vs augmented images by arithmetic augmentation.

Fig. 7. Real image vs augmented images by Gaussian noise, Poisson noise, rotation, and shear augmentations.

[17, 18], followed by a Bidirectional Long Short-Term Memory (BLSTM) layer, a fully connected layer for decoding, and a SoftMax activation to convert the output into probabilities, all optimized with the Connectionist Temporal Classification (CTC) loss function [19, 21]. While most existing methods directly employ Recurrent Neural Networks (RNNs) to address the task of handwritten text recognition [4], we take a more holistic approach by considering the task as both a visual and sequential problem. This dual nature of the problem leads us to combine the strengths of CNNs and RNNs. The CNNs are essential for extracting meaningful features from the input images, while the BLSTM effectively captures the contextual dependencies inherent in Arabic handwritten text, where the meaning of letters often depends on their surrounding characters.

It was firstly employ CNNs to extract hierarchical features from the input image, utilizing residual connections to improve training by allowing gradients to flow more easily through the network. These features are then processed by the convolutional layers, followed by a down sampling step where the height dimension is reduced to 1, and the width is down sampled by a factor of 4. This down sampling results in a feature map with dimensions corresponding to the reduced width and the number of output channels. By treating the width as a sequence, we transform the problem into a sequence-to-sequence learning task. We then pass this sequence through the BLSTM, which allows the network to capture both forward and backward dependencies between letters, essential for understanding the context of Arabic handwriting, where the form of a letter can change depending on its position within a word. After the BLSTM layer, the output sequence is decoded by a fully connected layer followed by a SoftMax activation function [13]. This transformation converts the output into a set of probability distributions, one for each character class at each width position. For training, the predicted probabilities are passed through the CTC loss function, which aligns the predicted sequence with the ground-truth label and calculates the loss necessary for optimizing the model. During inference, we apply a greedy decoding approach, where the most probable character at each width is selected. Since the CTC loss function includes a blank class in addition to the character classes, we must decode the output sequence by applying a two-step CTC decoding process. First, we remove any consecutive duplicate characters to prevent repetition, and second, we eliminate all blank classes, leaving only the predicted sequence of letters.

This approach enables the model to perform unconstrained Arabic handwritten text recognition by handling variable-length sequences and leveraging both visual and contextual information. The combination of CNN for feature extraction and BLSTM for sequential modeling is a powerful approach for this task, allowing our system to achieve high accuracy even in the presence of complex handwriting styles and diverse input variations. By incorporating CTC loss, we ensure that the model can handle misalignments between input and output sequences, making it more robust in real-world scenarios where precise alignment is often challenging.

To delve deeper into the details of our proposed architecture, we begin by examining the Convolutional Neural Network (CNN) component, which is structured using two key building blocks: residual blocks and standard convolutional layers. The residual block is central to the CNN design and consists of two convolutional layers, each followed by a batch normalization layer to stabilize training and improve convergence. After each convolutional layer and batch normalization, we apply a ReLU activation function to introduce non-linearity into the model. The critical feature of the residual block is the addition of a residual connection, defined by the equation $y = f(x) + x$, where $f(x)$ represents the transformation learned by the convolutional layers, and x is the original input to the block. This residual connection allows the network to bypass certain transformations, helping to mitigate the vanishing gradient problem and enabling deeper networks by facilitating gradient flow during backpropagation.

In the case that the dimensions of x and $f(x)$ do not match, we apply a 1×1 convolution to x to adjust its dimensions, ensuring compatibility for the residual connection. This adjustment allows for more flexibility in the architecture, enabling the model to handle feature maps of different sizes while maintaining the benefits of residual learning. The CNN is constructed

with four stacked residual blocks, with each pair of residual blocks followed by a dropout layer to mitigate overfitting and encourage generalization. These layers are designed to capture increasingly complex features as they propagate through the network, with dropout helping to prevent the model from overly relying on any single feature. The CNN architecture includes three main convolutional layers, each followed by a max-pooling layer, except for the final layer, which is followed by an adaptive average pooling layer. This adaptive pooling layer serves a critical function in reducing the height of the feature map to 1 while maintaining the width, allowing the model to focus on the most important spatial features. However, if the width of the feature map exceeds 256 pixels, the adaptive average pooling layer reduces it to 256, ensuring that the network does not produce excessively large feature maps that would be computationally expensive and difficult to process. This adaptive approach to pooling helps the network maintain important spatial information while also ensuring that the dimensions are suitable for input to the subsequent layers.

The next stage in our architecture involves the Bidirectional Long Short-Term Memory (BLSTM) layer, which is crucial for capturing sequential dependencies within the input text. The BLSTM layer is composed of two stacked LSTM layers, which allow the model to process the input sequence in both forward and backward directions. This bidirectional processing is particularly important for Arabic handwriting, as the context of a letter can depend on both its preceding and following letters. The input dimensions of the BLSTM layer are the same as the output dimensions of the final convolutional layer in the CNN, ensuring seamless integration between the two components. The hidden dimensions of the BLSTM are a hyper parameter that we set during model configuration, allowing flexibility in controlling the model's capacity to capture long-range dependencies.

Finally, the output of the BLSTM layer is passed through a fully connected layer, which serves as the decoder for the network. This fully connected layer takes as input the hidden dimensions of the BLSTM and produces an output of size equal to the number of character classes (including the blank class for CTC). The output is then passed through a SoftMax activation function to convert the raw output values into probabilities, which represent the likelihood of each character at each sequence position. This architecture combines the strengths of CNNs for feature extraction and BLSTMs for sequential modeling, making it well-suited for handling the complexities of Arabic handwritten text recognition. The residual blocks help prevent overfitting and facilitate deeper learning, while the adaptive pooling layer ensures efficient handling of varying input sizes. The combination of CNN and BLSTM enables the model to capture both spatial features and temporal dependencies, leading to robust recognition performance even in the presence of complex handwriting variations..

$$y = f(x) + x \tag{1}$$

6. Experiments and results

As for experiments we built a whole system for training, testing, and inference. This system helped us conduct many experiments. In following two sections we will discuss the system, and our results.

6.1. Experiments preparations

This system uses PyTorch for the neural networks, and PyTorch Lightning to integrate callbacks, (which allow us to add features like changing the images size during training), training loggers, and 16 bit precision training. Our system has many features that allowed us to experiment easily, which are: Multiple weight initializers for increasing the size of images during training different learning rate schedulers.

Different optimizers for changing learning rate schedulers during training and support multiple datasets, by defining a class for that dataset. Support multiple architectures. Also it was many hyper parameters choices via CLI. Logging experiment name, hyper parameters, and results in Weights and Biases.

6.2. Experiments

In these experiments used 1 GPU, which is the RTX 2080; also, it used 16 bit precision for training, which allowed us to train faster and use less GPU memory. As for abc/d configuration of dataset, we used 64 as the initial number of channels in our CNN and increased the number by factor of 2 in the following layers, we had two dropout layers with 0.55 probabilities in each layer as seen in Fig. 8, and we used BLSTM with 2 layers, 256 hidden size, and 0.2 dropout probability. Additionally used the Stochastic Gradient Descent (SGD) optimizer, with a learning rate of 0.025, momentum of 0.9, and weight decay of 1e-4 (0.0001); furthermore, we started with constant learning rate, then used a callback to use exponential learning rate decay policy from the 40 epochs with a factor of 0.965. Also was used the default weight initialization provided by PyTorch. As for the data related hyper parameters, a batch size of 8 was used, and the initial images height was 32, and used a callback to increase the height of the images by 8, and 16 respectively, which leads in an increase to the width to preserve

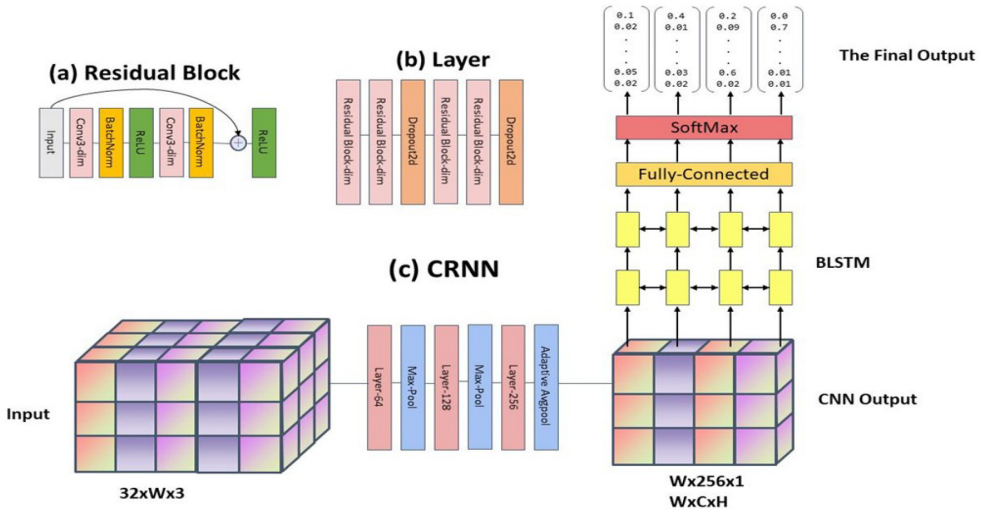

Fig. 8. This figurer describes our full CRNN architecture. (a) Is the smallest block in the CNN. (b) Layer is composed of 4 residual blocks and two dropout layers. (c) Is the whole architecture, which consists of a CNN that have 4 Layers, then followed by BLSTM which takes the output feature map of the CNN as a sequence, then the fully connected layer will output a probability distribution of all the classes for each point in the sequence.

Undecoded Sequence:
['ja', '-', 'ma', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'ya', 'ya',
'ya', 'ya', 'ya', 'ya', 'ya', 'ya', 'la', 'la', 'la', 'te', 'te', '-']
Step 1:
['ja', '-', 'ma', '-', 'ya', 'la', 'te,'-']
Step 2:
['ja', 'ma', 'ya', 'la', 'te]
Decoded Sequence:
['ja', 'ma', 'ya', 'la', 'te']

Fig. 9. Example of CTC decoding. Which first removes all the repetitions, then removes all the blank labels, resulting in a decoded sequence.

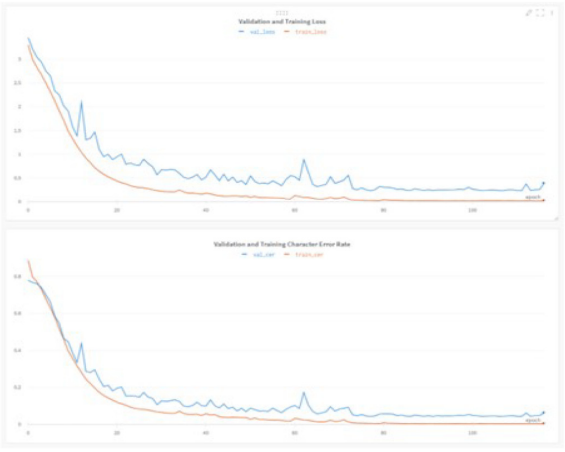

Fig. 10. Plot of the training loss vs validation loss over epochs, as well as the training CER vs validation CER for the abc/d configuration.

the aspect ratio, the height increases was in the 123, and 137 epochs respectively. It was trained for 149 epochs, which took 1 day and 15 hours. In Fig. 9 and Fig. 10 we see the loss and the Character Error Rate (CER) plotted over epochs during the training process. As for the abcd/e configuration, we trained with almost the same hyper parameters as the abc/d configurations expect the height increase was by 8 and 16 in the 120 and 132 epochs respectively, finally the model was trained for 142 epochs.

For the final configuration bcd/a, It had to be trained on a new model, where used also the same hyperparameters as the last two configurations, expect the height increase by 8 and 16 in the 120 and 145 epochs respectively, finally the model was trained for 155 epochs.

6.3. Results

The proposed system has been developed and rigorously tested, demonstrating high accuracy on the test dataset. The workflow begins with users uploading input images through a web interface. These images are preprocessed and passed to the Detection Module (DM), which generates a set of word-containing images for each input, along with indices marking the end of each line. Subsequently, the extracted word images are organized into lists and processed by the Word Recognition Module (WRM), which predicts the corresponding text for each word image. Finally, the system reconstructs the text by aligning the recognized words with their respective lines based on the indices provided by the DM, ensuring an accurate and structured output. In general, we have achieved cutting-edge results in the recognition task on the IFN/ENIT dataset without employing language

Table 1. CRNN Model results compared with other results.

CER				
		ConFigureurations		
Methods		abc/d	abcd/e	bcd/a
Ours		1.99	7.27	2.61
BLSTM[1]	[4]	6.9	11.84	8.59
BLSTM[2]	[4]	4.81	8.79	6.67

models, dictionaries, or search-based decoders. You can find our results in Table 1. Unfortunately, as far as we know, there is no published work that refrains from using language models, dictionaries, and search-based decoders on the IFN/ENIT dataset. Consequently, we had to compare our results to methods that utilize language models, dictionaries, or search-based decoders. This comparison may not be entirely fair, but it underscores that we can achieve top-notch results without restricting the output of the recognition models to a limited dictionary. The finalized, formatted text is seamlessly delivered back to the website, where users can view, edit, and download the output with ease.

Arabic handwritten text recognition models have broad real-world applications across various sectors. These models can be used for automating the digitization of documents in areas such as banking, healthcare, postal services, education, and government, enabling faster and more accurate data processing. They can also assist in creating accessible technologies for the visually impaired, improve e-commerce operations, and streamline customer service. Moreover, they support research and data collection efforts, and enhance translation and language learning tools. By eliminating reliance on dictionaries and search-based decoders, our model provides a flexible, scalable solution that can handle diverse handwriting styles, making it a valuable tool for many industries.

7. Conclusion

In summary, this study introduces a neural network architecture incorporating Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) to address the Arabic text recognition task without imposing constraints. The proposed work modifies the ground-truth format to operate at the character level and apply three types of augmentations. The methodology aims to recognize letters irrespective of their shapes, and our model is designed to accommodate inputs of varying sizes. Remarkably, it was achieved the state-of-the-art results without relying on language models, dictionaries, or search-based decoders. This approach comes close to surpassing the current state-of-the-art results achieved by models that rely on language models, dictionaries, or search-based decoders. Traditional methods in Arabic handwritten text recognition often incorporate powerful language models or decoders, such as Word Beam Search (WBS), that utilize predefined dictionaries to improve recognition accuracy by constraining the output to valid word sequences. While these techniques have shown impressive results, they have inherent limitations. The reliance on dictionaries can lead to errors when encountering out-of-vocabulary words, novel handwriting styles, or domain-specific terms that are not covered in the dictionary. Additionally, the use of search-based decoders adds computational complexity and often requires manual tuning or domain-specific adaptations to perform optimally.

In contrast, this proposed method, which does not depend on such linguistic constraints or external knowledge sources, focuses on directly learning from the raw visual input and modeling the sequential dependencies of handwritten Arabic text through a combination of Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (BLSTM) layers. By using the Connectionist Temporal Classification (CTC) loss function, our model is able to handle variable-length sequences and make predictions without being restricted by a predefined set of words or language models. This allows our approach to recognize text in a more flexible and generalized manner, free from the limitations imposed by dictionary-based constraints.

While this proposed model does not explicitly rely on external language resources, it is still able to achieve competitive or near state-of-the-art performance by effectively capturing the spatial and temporal dependencies inherent in handwritten Arabic text. The ability to perform well without the reliance on dictionaries or complex decoders signals the potential of our approach to handle more unconstrained recognition tasks, where flexibility and generalization to diverse handwriting styles are crucial. Moreover, the absence of search-based decoders and language model dependencies allows our method to be more computationally efficient, making it suitable for real-time applications and scenarios with limited resources. By coming close to surpassing the best-performing models that rely on these traditional techniques, our approach represents a promising step forward in the development of more robust, scalable, and versatile handwriting recognition systems.

References

1. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
2. A. A. AlRababah, "Neural networks precision in technical vision sys-tems," *IJCSNS*, vol. 20, no. 3, p. 29, 2020.
3. G. A. Abandah, F. T. Jamour, and E. A. Qaralleh, "Recognizing hand-written arabic words using grapheme segmentation and recurrent neural networks," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 17, no. 3, pp. 275–291, 2014.
4. M. Eltay, A. Zidouri, and I. Ahmad, "Exploring deep learning ap-proaches to recognize handwritten arabic texts," *IEEE Access*, vol. 8, pp. 89 882–89 898, 2020.
5. A. Graves, "Connectionist temporal classification," in *Supervised Se-quence Labelling with Recurrent Neural Networks*. Springer, 2012, pp. 61–93.
6. A. Poznanski and L. Wolf, "Cnn-n-gram for handwriting word recog-nition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2305–2314.
7. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
8. A. A. AlRababah, and A. Alzahrani, "Software Maintenance Model through the Development Distinct Stages," *IJCSNS International Journal of Computer Science and Network Security*, 19 (2), 2019.
9. M. Yousef, K. F. Hussain, and U. S. Mohammed, "Accurate, data-efficient, unconstrained text recognition with convolutional neural net-works," *Pattern Recognition*, vol. 108, p. 107482, 2020.
10. S. A. Mahmoud, I. Ahmad, W. G. Al-Khatib, M. Alshayeb, M. T. Parvez, V. Margner, and G. A. Fink, "Khatt: An open arabic offline handwritten text database," *Pattern Recognition*, vol. 47, no. 3, pp. 1096–1112, 2014.
11. M. E. Mustafa and M. K. Elbashir, "A deep learning approach for handwritten arabic names recognition," 2020.
12. N. Altwaijry and I. Al-Turaiki, "Arabic handwriting recognition system using convolutional neural network," *Neural Computing and Applica-tions*, pp. 1–13, 2020.
13. M. Pechwitz, H. El Abed, and V. Margner, "Handwritten arabic word recognition using the ifn/enit-database," in *Guide to OCR for Arabic Scripts*. Springer, 2012, pp. 169–213.
14. A. A. Q. AlRababah, "On the associative memory utilization in english-arabic natural language processing," *International Journal of Advanced and Applied Sciences*, vol. 4, pp. 14–18, 2017.
15. T. Jayalakshmi and A. Santhakumaran, "Statistical normalization and back propagation for classification," *International Journal of Computer Theory and Engineering*, vol. 3, no. 1, pp. 1793–8201, 2011.
16. C. Luo, Y. Zhu, L. Jin, and Y. Wang, "Learn to augment: Joint data augmentation and network optimization for text recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 746–13 755.
17. A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, C. Reinders, S. Ya-dav, J. Banerjee, G. Vecsei, A. Kraft, Z. Rui, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernandez,´ F.-M. De Rainville, C.-H. Weng, A. Ayala-Acevedo, R. Meudec, M. Laporte *et al.*, "imgaug," https://github.com/aleju/imgaug, 2020, online; accessed 01-Feb-2020.
18. R. M. Mrayyan and A. A. Al Rababah, "Debugging of Parallel Programs using Distributed Cooperating Components", *International Journal of Computer Science & Network Security*, vol. 21, no. 12spc, pp. 570–578, 2021.
19. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
20. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
21. A. Graves, M. Liwicki, S. Fernandez,´ R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained hand-writing recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2008.
22. V. O. Nyangaresi, A. J. Rodrigues, and A. A. Al Rababah, "Secure Protocol for Resource-Constrained IoT Device Authentication", *International Journal of Interdisciplinary Telecommunications and Networking (IJITN)*, vol. 14, no. 1, pp. 1–15, 2022.
23. A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," *Advances in Neural Infor-Mation Processing Systems*, vol. 21, pp. 545–552, 2008.
24. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B.Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems* 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche´-Buc, E. Fox, and R.Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online].Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
25. W. Falcon, "Pytorch lightning," GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning, vol. 3, 2019.
26. L. Biewald, "Experiment tracking with weights and biases," 2020, software available from wandb.com. [Online]. Available: https://www.wandb.com/.
27. A. A. Al Rababah, "Assurance quality and efficiency in corporate information systems," *International Journal of Computer Science and Network Security*, vol. 19, no. 4, pp. 87–95, 2019.
28. J. Bharadiya, "Convolutional neural networks for image classification," *International Journal of Innovative Science and Research Technology*, vol. 8.5, pp. 673–677, 2023.
29. Ahmed M. Khedr, "Enhancing supply chain management with deep learning and machine learning techniques: A review," *Journal of Open Innovation: Technology, Market, and Complexity*, p. 100379, 2024.
30. Mienye, Ibomoiye Domor, Theo G. Swart, and George Obaido, "Recurrent neural networks: A comprehensive review of architectures, variants, and applications," *Information* vol. 15.9, p. 517, 2024.
31. Černevičienė, Jurgita, and Audrius Kabašinskas, "Explainable artificial intelligence (XAI) in finance: A systematic literature review," *Artificial Intelligence Review*, vol. 57.8, p. 216, 2024.