A Deep CNN-BiLSTM Framework for Arabic Handwritten Text Recognition

Muhammad Ramzan
Department of Computer Science, College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia
m.ramzan@seu.edu.sa (corresponding author)

Received: 21 December 2025 | Revised: 2 February 2026 | Accepted: 13 February 2026

 DOI: https://doi.org/10.48084/etasr.17092

ABSTRACT

Text recognition is one of the most essential aspects of pattern recognition and Natural Language Processing (NLP). Although there is much research on handwritten text recognition, Arabic text recognition is a challenging task not only due to linguistic diversity and complexity, but also due to handwritten text variations, including shape, skew, cursive style, fonts, and formats. The KHATT dataset includes a wide variety of handwritten text from people across different countries. This study presents a framework that combines Bidirectional Long Short-Term Memory (BiLSTM) and Convolutional Neural Networks (CNN). The input data was preprocessed by segmenting it into lines and applying adaptive thresholding for binarization. Data augmentation was also performed to improve model training. A Character Error Rate (CER) of 13% was obtained, demonstrating that the proposed system outperformed state-of-the-art techniques.

Keywords-Arabic handwritten text recognition; bidirectional long short-term memory; connectionist temporal classification; optical character recognition; CNN

I. INTRODUCTION

Optical Character Recognition (OCR) is used to transform text images into machine-readable formats. OCR is a classical pattern recognition issue that has been studied with the advancement of computer vision [1, 2], and has a wide range of applications, including number plate recognition, traffic sign recognition, classification of different data types, and language translation [3, 4]. OCR systems using machine learning techniques have been used to support linguistic rules for automatic language translation with high accuracy and precision. Arabic Handwriting Recognition (AHR) involves the conversion of handwritten Arabic text from images to machine-readable form, which is crucial for various applications such as handling forms, automatically processing bank checks, and digitizing historical documents [2]. More than 400 million people are estimated to benefit from this effort of translation and digitization [5].

Working with Arabic script, specifically cursive Arabic handwritten text, is challenging because it has many peculiarities compared to Latin alphabet scripts, such as the shapes of letters depending on what letters are next to when writing [6]. In addition, heterogeneous handwriting and non-standard fonts further decrease the accuracy rates compared with Latin scripts. Research findings have estimated that current Arabic OCR systems can achieve recognition rates of only approximately 80% on handwritten texts [4]. Addressing these challenges is critical for enhancing the efficiency of document digitization efforts in Arabic-speaking regions, where there is a pressing need for reliable data extraction tools to support educational, governmental, and commercial activities [5, 6].

Recent advances in deep learning have led to hybrid CNN-BiLSTM-CTC architectures becoming a typical baseline for handwritten text recognition tasks. Although such frameworks are well established, they rely heavily on preprocessing strategies, data augmentation decisions, and evaluation procedures in Arabic handwriting. This study investigates a CNN-BiLSTM-CTC-based system on the KHATT dataset, with a particular focus on systematic preprocessing, controlled training, and statistically based performance analysis. By developing an effective architecture, this study offers a transparent and reusable empirical evaluation of a typical deep learning pipeline for Arabic handwritten text recognition.

II. RELATED WORK

In [7], an Arabic handwritten text recognition system used the KHATT dataset, which was preprocessed using median filtering to remove salt-and-pepper noise. First, paragraphs were extracted from the pages, and then lines were extracted from the paragraphs. Derivatives of both vertical and horizontal edges were used, along with statistical and gradient properties. Two classifiers were used to evaluate images: HMM and a novel syntactic classifier, achieving an accuracy of 51.5%. In [8], an RNN-CNN-based model was used for Arabic handwritten text recognition. Multiple versions of Efficient Net were explored, involving BiLSTM with and without self-attention. EfficientNetB3 was chosen for feature extraction. Sequential features were utilized by applying three BiLSTM

layers, while a Connectionist Temporal Classification (CTC) layer aligned the input sequence with the output sequence. Experiments on the KHATT and AHAWP datasets achieved Character Error Rate (CER) of 17.26% and 0.2%, respectively. In [9], a transformer-based architecture was proposed for Arabic handwritten text recognition. This study used two models, a standard Sequence-to-Sequence and a Transformer Transducer, to overcome the sequential-nature limitations of RNN models by employing pretrained transformers for language modeling and visual comprehension. This technique was tested on the KHATT dataset and attained a CER of 18.45 for the transformer with a cross-attention architecture.

In [10], a segmentation-free framework was proposed for Arabic handwritten text recognition, combining a deep CNN for feature extraction, a BiLSTM for sequence recognition, and a CTC loss function. This model was tested on the KHATT dataset, achieving an accuracy of 71% at the word level and 84% at the character level. In [11], two deep NNs were combined for Arabic handwritten text recognition. A residual network model was used for feature extraction, and BiLSTM and CTC were used for sequential modeling. The KHATT and AHTID/MW datasets were used to evaluate performance, achieving a CER of 13.2% and a WER of 27.31. In [12], a DL-based approach was presented for Arabic text recognition on the KHATT dataset. This study performed three preprocessing steps: Cutting extra spaces, de-skewing the text lines, and using rotation, flipping, and shifting to augment the data. This model employed a Multi-Dimension LSTM and CTC, achieving a CER of 19.98%. In [13], a transformer-based architecture was proposed for recognizing online handwritten Arabic texts. This model combined two different decoding techniques: A CTC decoder and a Self-Attention Decoder (SAD) with a Self-Attention Encoder (SAE). The model was evaluated on the KHATT and CHAW datasets, achieving a CER of 22% on the former. In [14], Arabic handwritten character segmentation was performed using a CNN-graph theory method. This approach overcame a major challenge, the inaccurate character recognition on the segmentation of linked and overlapping Arabic cursive letters. This model was evaluated on the IESK-ArdB dataset. To address the challenges of Arabic script, an optimized approach for handwritten Arabic character recognition was proposed in [15], using a Support Vector Machine (SVM) classifier. This was a segmentation-based approach to increase the accuracy and efficiency of the recognition process.

III. PROPOSED METHODOLOGY

Figure 1 shows the proposed architecture for Arabic handwritten text recognition, which involves advanced techniques for image preprocessing, data augmentation, training, and testing. Initially, to enhance the dataset, the original images were subjected to contrast enhancement, thresholding, and segmentation, followed by data augmentation to enrich the dataset. The enhanced and augmented dataset was used to train a custom CNN-BiLSTM model. The model was developed from scratch, and features were extracted during training. This approach ensures a robust and interpretable recognition system for handwritten Arabic text recognition.

Although the CNN-BiLSTM-CTC model has been proven to be effective for handwritten text recognition, this study focused on a reproducible and strictly validated pipeline specific to the Arabic script, making the following contributions: (i) a dedicated preprocessing pipeline that solves the problem of line overlapping in Arabic cursive text using adaptive morphological operations, (ii) a direction-aware augmentation protocol that does not alter the semantics of the Arabic script using flipping and shifting transformations, and (iii) a statistically validated evaluation framework that offers transparent and reproducible benchmarks on the KHATT dataset.

A. Dataset Description

KHATT is a handwritten Arabic text database that is publicly available, taken from numerous writers from many Arab countries, such as Jordan, Morocco, KSA, and Egypt. All users have their own writing styles. This dataset is based on 1000 handwritten forms with varying resolutions ranging from 200 to 600 dpi, and the text is in the form of scanned documents. The ground-truth images were manually annotated. Numerous paragraphs and lines of text were used as samples. Large variations in text, such as shape, orientation, style, font, and format, exhibit natural writing and make it more challenging [7, 16]. Figure 2 shows a sample of the dataset.

The data were divided into an 80:20 ratio at the writer level so that no lines of the same writer were present in both the training and testing sets. This writer-independent split protocol preserves the realism of the recognition task and is consistent with handwritten text recognition practice.

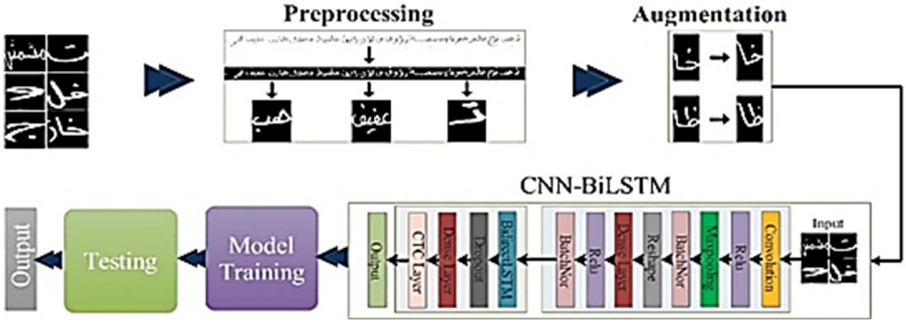

Fig. 1. The overview of the proposed CNN-BiLSTM-CTC text-recognition pipeline.

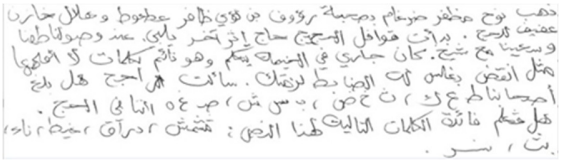

Fig. 2. Sample image from the KHATT handwritten Arabic dataset.

B. Preprocessing

As a preprocessing step, the lines from the sample images taken from the dataset were segmented. Since Arabic text is cursive, it exhibits variations, and line extraction serves as a Region of Interest (ROI), making the layout of the extracted text consistent. This will ultimately result in an improved accuracy rate. In most cases, the lines of the dataset were well-written and aligned. In addition, line spacing is almost the same. However, in a few cases, de-skewing was used so that accurate segmentation can be done with horizontally aligned text and lines of segmented images.

The quality of the segmented textual images was improved by applying segmentation across all data. Colored input images were converted to grayscale before processing, which can help capture variations in grayscale intensity. The images were converted to binary format using the optimal thresholding technique. Due to this preprocessing step, there was a considerable change in the accuracy and computational efficiency of the proposed system. Background and contrast variations significantly affect the results. Significant improvement was observed by using binarization and applying an adaptive global threshold.

It has been observed that peaks and valleys correspond to locations where there is a majority of text and spaces, respectively. These peaks and valleys help improve differentiation and eventually segmentation using a threshold. A gradient is calculated using a histogram, which further refines the process. An ideal set of cut points was identified to accurately reduce interference from closed lines by determining the minimum gradient values adjacent to each valley.

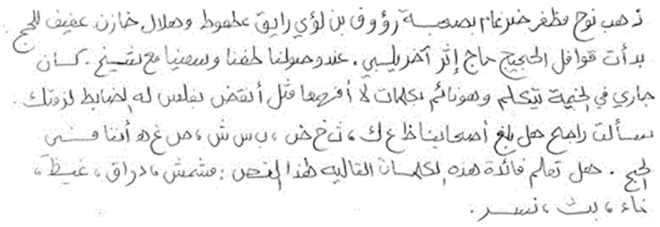

Fig. 3. A sample input before preprocessing.

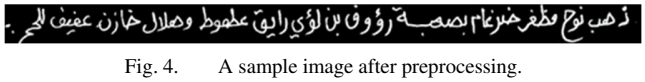

Fig. 4. A sample image after preprocessing.

The Arabic handwritten challenge addressed in this work is overlapping ascenders and descenders between lines using morphological processing techniques on the images after applying the threshold. Line separation is reduced while retaining the integrity of each line's content by using erosion and dilation. In conditions when freestyle handwriting showed connections within lines, better line boundaries can be extracted due to the mentioned morphological processing, which helped manage overlapping characters.

C. Data Augmentation

Data augmentation is a widely used technique that increases the volume of datasets. By augmenting the first 5,160 samples to 15,480 with rotation-based augmentation, the model is exposed to greater variation in handwriting styles. This practice allows for increased diversity in the training set and can help the model to generalize, thus reducing overfitting and improving performance on out-of-sample data. Augmenting a training dataset can significantly affect the performance of DL models. The KHATT dataset, which contains 1000 samples, was augmented using character-level augmentations, exclusively rotation. In addition, supplementary 5000-line images were extracted from a source not previously used in the KHATT dataset to be used only in the training set, as shown in Figure 5. The initial step in processing these pages was to divide them into single text lines with the preprocessing pipeline, which produced 5,160 line-level images. Rotation-based data augmentation was only used in the training set to improve model generalization and reduce overfitting. All training lines were rotated by 5 and 10 degrees, resulting in three versions of each original line.

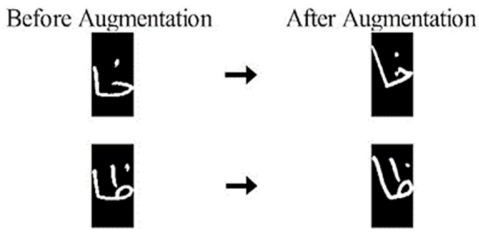

Fig. 5. Illustration of sample images after the augmentation step.

D. Proposed Architecture

The model employs a series of CNN layers for feature extraction, followed by a Fully Connected dense layer and regularization techniques, optimized for handling handwritten Arabic text. Recognition is performed at the line level, where each input image is associated with a complete text line (not a single word). A batch size of 64 images was applied, each image having a size of 64×32. The feature extraction step starts with the first CNN layer, which has a depth of 32, each sized 3×3. After the CNN layer, a Max Pooling layer is added with a depth size of 2×2, resulting in 32 feature maps, each of size 32 pixels wide and 16 pixels tall. To remove negative values, ReLU activation is applied after the CNN layer to enhance non-linearity. A Batch Normalization layer follows to manage the feature distribution and minimize internal covariance shift, stabilizing the input distribution of the network and speeding up learning.

A second CNN layer with 64×64 depth and 3×3 kernel size was added, connected to another 2×2 Max Pooling layer. The Max Pooling layer outputs 64 feature maps, each sized 16×8, to further improve feature stability and facilitate more effective learning. Similarly to the first layer, the second layer's output was subjected to Batch Normalization and ReLU activation.

This framework takes advantage of Batch Normalization, which reduces sensitivity to initialization and helps the model converge by mitigating the differences in input distributions between batches. This allows for efficient training with fewer mini-batches. In order to prepare the feature maps for classification by the following RNN, they are first passed through a Fully Connected dense layer that flattens them into a single sequence. In particular, the spatial feature maps that CNN generates were redesigned by projecting the horizontal axis as the temporal axis to create a sequence of feature vectors to be processed recurrently. With 16 output nodes, this thick layer was designed to minimize the risk of overfitting. The CNN output feature maps have a dimension of 64×16×8, which is reshaped so that the width dimension becomes the time axis, thus producing a sequence of 16. Each time step is characterized by a 512-dimensional feature map (8×64). Each feature vector in a time-distributed Fully Connected layer is projected to a lower-dimensional embedding, and then the BiLSTM works with it. This sequence-based reshaping maintains the spatial order of the characters in the direction of writing and adheres to the principles of conventional CNN-BiLSTM-CTC designs of systems with OCR.

The Fully Connected layer was followed by a Dropout layer, which deactivates a certain number of nodes at random during each training iteration. By reducing overfitting, this improvement strengthens the model's resilience and boosts its ability to generalize to new data. Figure 6 shows the flow of the proposed Bi-LSTM. Bidirectional LSTM layers capture the relationship between words in both the forward and reverse states throughout a given line of text, allowing for better recognition of characters within their respective sequences. The CTC layer provides a mechanism for aligning the input features with character labels without providing explicit segmentation of the character. For example, in handwriting recognition, it allows for multiple alignments of labeled data in time (many-to-one) or opposite (one-to-many) using a blank character as a placeholder for characters or ligatures with very little space between them when they are on top of each other. In addition, multiple predictions are combined during the decoding stage. A standard CTC greedy decoding technique was utilized to provide inference, collapsing repeated labels and dropping out any blank labels to create the final transcription. Another feature of CTC is that it supports contextual information from the BiLSTM, as it can look at contextual temporal information to help determine what character(s) a person may have written in these situations, where visually they might look compressed together or on top of each other. All architectural parameters and training setups were maintained constant throughout the experiments to ensure consistency and reproducibility.

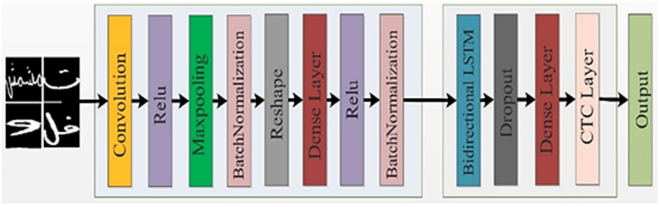

Fig. 6. The architecture of the CNN-BiLSTM feature extraction and recognition block.

IV. EXPERIMENTAL RESULTS

To evaluate the model's performance, the original dataset of 5,160 images was augmented to 15,480 images, as described above, increasing the model's exposure to data variances and improving its generalization capabilities.

The KHATT dataset was divided into training and validation sets in an 80-20 ratio, with 80% of the images for model training and the remaining 20% for testing and validation. A total of 450 epochs was used to train the model; this number was selected to maximize convergence without causing appreciable overfitting, taking into account the optimizer and learning rate dynamics. Early stopping was not applied. The average CER of the proposed framework was 13, achieving an average performance of 13.2±0.15, which was consistent over multiple runs.

Figure 7 illustrates the training and validation losses, which decreased steadily and stabilized after approximately 400 epochs, indicating convergence. The model's smooth loss curves reflect the effectiveness of the exponential decay learning rate scheduler and confirm the stability of the Adam optimizer. The validation loss of 1.1795 and training loss of 1.5730 after 450 epochs suggest that the model maintained a good balance between underfitting and overfitting. The experiment was repeated three times with different random seeds to assess model stability. The validation loss was also slightly smaller than the training loss, which is explained by dropout and batch normalization applied during training but turned off during inference, making the regularization noise smaller during validation.

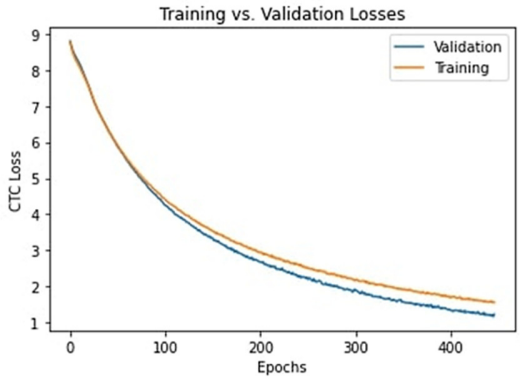

Fig. 7. Training and validation loss across 450 epochs.

A. *Experimental Setup*

The ADAM optimizer employed in the training process ensured consistent performance across epochs with its adaptive learning rate capabilities. The learning rate was applied to an exponential decay scheduler with a starting maximum of 0.0001, epochs of 450, and a batch size of 64. A small initial learning rate was intentionally chosen to avoid overfitting. Training was stabilized, and the performance of the final model was improved by gradually lowering the learning rate. The model was developed using Python on a workstation equipped with 32 GB RAM and an NVIDIA GPU with 16 GB.

B. Performance Evaluation Metrics

CER was employed to evaluate the performance of the proposed framework, as it is a widely used evaluation metric for OCR systems. The Levenshtein distance between the predicted text and the ground truth, normalized by the total number of characters in the ground truth, measures the differences between two strings by determining the smallest number of character deletions, substitutions, and insertions required to transform one string into another. CER is mathematically expressed as:

$$CER = \frac{X+Y+Z}{m} \quad (1)$$

where X stands for the number of substitutions, Y for insertions, Z for deletions, and m for the total number of characters in the ground truth [9]. CER identifies certain typologies of character-level or word-level text, thus allowing another level of examination with respect to transcription model accuracy. The results from the test dataset show a CER of approximately 13%, which indicates that 87% of the characters in each of the ground-truth transcriptions evaluated by the model (including punctuation and spaces) were predicted accurately. This result indicates the model's ability to appropriately recognize Arabic handwriting text, which grew in complexity with encoding, resulting in an acceptable understanding. Figure 8 presents prediction examples.

C. Validation

To test the statistical significance of the results, additional analyses on the standard CER were also performed. A bootstrap resampling scheme, which consists of 1,000 steps, was applied to estimate the 95% Confidence Interval (CI) of CER and both the training and validation losses. Random samples of the test data were selected with replacement at every iteration, and CER was recomputed on each resample. Lower and upper bounds of the CI were then taken to be the 2.5-th and 97.5-th percentiles of the resulting bootstrap distribution, respectively. This is a non-parametric technique that provides a quantification of uncertainty in a data-driven way that does not require any presumptions about the underlying distribution. The proposed architecture had a CER of 13.0% (95% CI: 12.8-13.3%), a validation loss of 1.1795 (95% CI: 1.160-1.198), and a training loss of 1.5730 (95% CI: 1.552-1.590). The comparatively small CIs can be taken as a testament to the low degree of variability, which once again justifies the similarity in the performance of the model in a variety of resampled data subsets.

In addition, a paired McNemar test was performed to directly compare the predictions of the proposed model with the most competitive benchmark quoted as reference [12]. The test provided a statistically significant difference ($p < 0.05$), which implies that the improvement in CER was due to architectural and preprocessing decisions and not by chance. Table I shows the results of the statistical validation.

TABLE I. BOOTSTRAP CONFIDENCE INTERVALS FOR PERFORMANCE METRICS

Metric	Mean	Confidence level
CER	13.0%	[12.8-13.3]
Val Loss	1.1795	[1.160-1.198]
Train Loss	1.5730	[1.552-1.590]

D. Error Distribution and WER Contribution

Table II summarizes the error distribution of the proposed model. Substitutions are the most common in both the CER and Word Error Rate (WER), and they are often due to visually similar characters or ligatures. The error type distribution of the predictions of the model offers a broad view of what recognition failures are. As summarized in Table II, substitutions represent the highest percentage of CER and WER, 7.2 and 15.1% of the total errors, respectively. This means that the model most often mixes visually similar characters or ligatures, which is a typical problem with Arabic cursive handwriting.

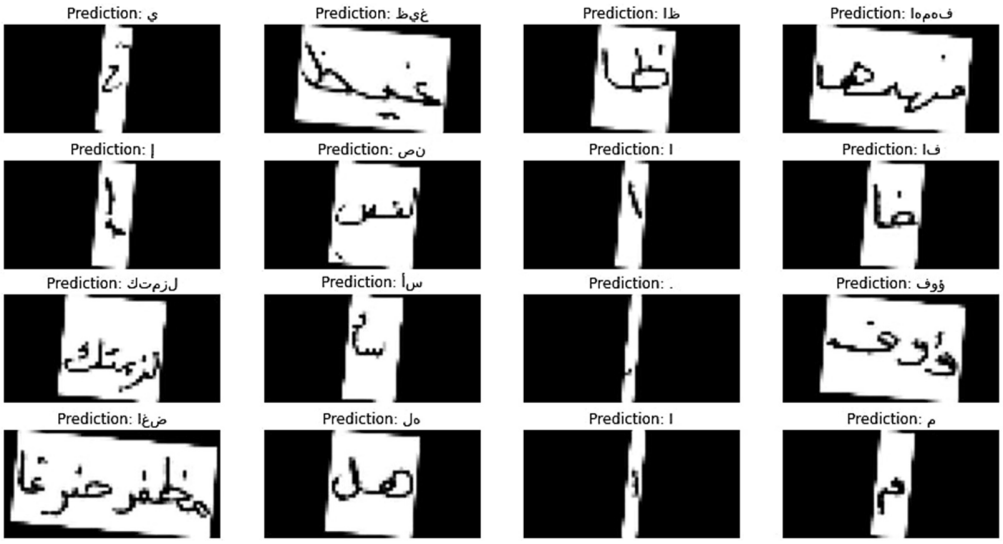

Fig. 8. Model's predictions on test images.

The second most common error is deletions, which adds 3.5% to CER and 8.3% to WER. Deletions usually appear in situations where the ink is faint, there are overlapping strokes, or where the writer has not included the diacritical marks. Although less common (2.3% CER, 3.4% WER), insertions are typically caused by over-segmentation of adjacent glyphs or in the input image.

The increased WER contributions of all types of errors are indicative of the compounding effect of character-level errors. Since a single character substitution, deletion, or insertion can result in a completely incorrect word, WER is disproportionately higher. This discussion highlights that the model is very effective in character recognition (13.0% CER), but it is more difficult at the word level since these errors accumulate in cursive word units.

TABLE II. ERROR DISTRIBUTION OF THE PROPOSED MODEL

Error Type	CER contribution	WER contribution
Substitution	7.2%	15.1%
Deletions	3.5%	8.3%
Insertions	2.3%	3.4%
Total	**13.0%**	**26.8%**

E. Comparison with State-of-the-Art Models

Table III compares the proposed technique with the most advanced previous methods. When tested on the KHATT dataset, the proposed framework outperformed other recent approaches, achieving a CER of 13%, outperforming [11], and being significantly better than other approaches that recorded CERs ranging from 17.26% to 22%. The performance of the model can be attributed to the combination of CNN layers to extract important features from the images and BiLSTM layers to understand the order of the characters. In addition, the special loss function (CTC) helps to match the input and output without needing exact character positions. Adding batch normalization and dropout made the model more stable and prevented overfitting.

Although the model gave satisfactory results on the KHATT dataset, it still needs to be tested on other types of handwriting and different datasets. Attention mechanisms, transformer models, and training with data from different Arabic writing styles or other related languages could be used to further improve the model.

TABLE III. COMPARISON WITH EXISTING TECHNIQUES

	Reference	Dataset	CER (%)
1	[8]	KHATT	17.26%
2	[9]	KHATT	18.45%
3	[11]	KHATT	13.2%
4	[12]	KHATT	19.98%
5	[13]	KHATT	22%
6	**Proposed method**	**KHATT**	**13%**

V. CONCLUSION

This study presented a deep-learning model that incorporates CNN, BiLSTM, and CTC layers to address the problem of handwritten text recognition in Arabic. The model was trained on the KHATT dataset for 450 epochs and achieved a CER of only 13%, with high accuracy and consistent convergence. The architecture used learned both the spatial and the sequential patterns of the Arabic text because it combined convolutional and recurrent layers. This synthesis contributed to the higher performance compared to most of the currently existing methods. However, this study is limited to a single dataset. In addition, attention mechanisms and transformer-based components were not incorporated, although they may enhance contextual understanding. Future work includes extending the approach to additional languages, exploring cross-dataset scenarios, integrating attention or transformer modules, and developing lightweight variants to support real-time or mobile OCR applications.

In general, the proposed CNN-BiLSTM model presents a reliable and generalized initial framework to further advance Arabic handwriting recognition and the associated document analysis activities. The KHATT dataset consists of handwritten papers that have been scanned at various resolutions (between 200 and 600 dpi), which means that there are certain discrepancies in the quality of the images. Although no single analysis was performed on artificially degraded images, the training and testing data contain natural variations such as contrast differences and small scanning artifacts. Proposed preprocessing pipelines, such as adaptive binarization and morphological processing, could help to resist moderate noise and resolution distortion. An extensive assessment of harsh noise conditions, e.g., heavy stains or extremely low-contrast images, is regarded as a significant area of further development.

REFERENCES

[1] A. Shetty and S. Sharma, "Ensemble deep learning model for optical character recognition," *Multimedia Tools and Applications*, vol. 83, no. 4, pp. 11411–11431, Jan. 2024, https://doi.org/10.1007/s11042-023-16018-0.

[2] M. El Khayati, I. Kich, and Y. Taouil, "CNN-based Methods for Offline Arabic Handwriting Recognition: A Review," *Neural Processing Letters*, vol. 56, no. 2, Mar. 2024, Art. no. 115, https://doi.org/10.1007/s11063-024-11544-w.

[3] S. Amar"A Detailed study and recent research on OCR," *Vol. 19 No. 2 International Journal of Computer Science and Information Security (IJCSIS)*, vol. 19 no. 2, Jan. 2021, https://doi.org/10.5281/ZENODO.4578125.

[4] N. Sahu and M. Sonkusare, "A Study on Optical Character Recognition Techniques," *International Journal of Computational Science, Information Technology and Control Engineering*, vol. 4, no. 1, pp. 01–15, Jan. 2017, https://doi.org/10.5121/ijcsitce.2017.4101.

[5] K. Hamad and M. Kaya, "A Detailed Analysis of Optical Character Recognition Technology," *International Journal of Applied Mathematics Electronics and Computers*, no. Special Issue-1, pp. 244–249, Dec. 2016, https://doi.org/10.18100/ijamec.270374.

[6] M. G. Mahdi, A. Sleem, and I. El-henawy, "Deep Learning Algorithms for Arabic Optical Character Recognition: A Survey," *Multicriteria Algorithms with Applications*, vol. 2, pp. 65–79, Jan. 2024, https://doi.org/10.61356/j.mawa.2024.26861.

[7] S. A. Mahmoud *et al.*, "KHATT: An open Arabic offline handwritten text database," *Pattern Recognition*, vol. 47, no. 3, pp. 1096–1112, Mar. 2014, https://doi.org/10.1016/j.patcog.2013.08.009.

[8] M. Elsayed, A. Alnaggar, M. Abdeen, A. Wahdan, and W. Gomaa, "Arabic Handwritten Text Recognition using Advanced CNN-RNN Architecture," in *6th IAPR International Workshop on Document Analysis Systems (DAS) ICDAR*, 2023.

[9] S. Momeni and B. BabaAli, "A transformer-based approach for Arabic offline handwritten text recognition," *Signal, Image and Video Processing*, vol. 18, no. 4, pp. 3053–3062, June 2024, https://doi.org/10.1007/s11760-023-02970-9.

[10] S. Aabed and A. Khairaldin, "An End-to-End, Segmentation-Free, Arabic Handwritten Recognition Model on KHATT." arXiv, June 21, 2024, https://doi.org/10.48550/arXiv.2406.15329.

[11] A. M. Mutawa, M. Y. Allaho, and M. Al-Hajeri, "Machine Learning Approach for Arabic Handwritten Recognition," *Applied Sciences*, vol. 14, no. 19, Oct. 2024, https://doi.org/10.3390/app14199020.

[12] R. Ahmad, S. Naz, M. Afzal, S. Rashid, M. Liwicki, and A. Dengel, "A Deep Learning based Arabic Script Recognition System: Benchmark on KHAT," *The International Arab Journal of Information Technology*, vol. 17, no. 3, pp. 299–305, May 2020, https://doi.org/10.34028/iajit/17/3/3.

[13] F. Alwajih, E. Badr, and S. Abdou, "Transformer-based models for Arabic online handwriting recognition," *Inernational Journal of Advanced Computer Science and Applications,* vol. 13, no. 5, pp. 6, 2022.

[14] L. Berriche, A. Alqahtani, and S. RekikR, "Hybrid Arabic handwritten character segmentation using CNN and graph theory algorithm," *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 1, Jan. 2024, Art. no. 101872, https://doi.org/10.1016/j.jksuci.2023.101872.

[15] B. Kada, A. Mohammed, and B. Abdelmajid, "An Optimized Approach for Handwritten Arabic Character Recognition based on the SVM Classifier," *Engineering, Technology & Applied Science Research*, vol. 15, no. 2, pp. 22232–22238, Apr. 2025, https://doi.org/10.48084/etasr.9292.

[16] S. A. Mahmoud *et al.*, "KHATT: Arabic Offline Handwritten Text Database," in *2012 International Conference on Frontiers in Handwriting Recognition*, Sept. 2012, pp. 449–454, https://doi.org/10.1109/ICFHR.2012.224.