EMITTER International Journal of Engineering Technology
Vol. 13, No. 2, December 2025, pp. 253~267
DOI : 10.24003/emitter.v13i2.982

Towards Robust Recognition of Handwritten Arabic Characters with Diacritics Using an Incremental Learning Approach Based on CNNs

Fatima Aliyu Shugaba[1], Usman Ullah Sheikh[1], Mohd Afzan Othman[1], Nurulaqilla Khamis[1], Muhammad Habibullah Abdulfattah[1]

[1]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM, Johor Bahru, Malaysia
Corresponding author: usman@utm.my.

Received July 12, 2025; Revised August 23, 2025; Accepted October 25, 2025

Abstract

Handwritten Arabic text recognition (HATR) presents unique challenges due to complex character shapes, contextual variations, cursive connections, and the presence of diacritical marks. This study introduces AHAD (Arabic Handwritten Alphabet with Diacritics), a novel benchmark dataset of 71,061 handwritten Arabic character images annotated with five primary vowel diacritics; Fathah, Kasrah, Dammah, Shaddah, and Sukoon, covering 492 distinct classes that combine character identity, contextual form, and diacritic. Leveraging this dataset, we propose an incremental learning framework based on Convolutional Neural Networks (CNNs) to address fine-grained recognition of handwritten Arabic characters with its corresponding diacritics. The model was initially trained on a 114-class dataset of handwritten Arabic characters (in all contextual forms) of non-diacritic characters and fine-tuned in two phases using the AHAD dataset. The two-phase strategy includes output layer expansion, learning rate adjustment, and gradual unfreezing of deeper layers to enhance knowledge retention and prevent catastrophic forgetting. The proposed method achieved a validation accuracy of 92.96% and a test accuracy of 93.26%. Our findings demonstrate the effectiveness of incremental learning for diacritic-aware Arabic handwriting recognition and establish AHAD as a strong baseline for future research in this field.

Keywords: AHAD, Contextual form, Diacritic-aware, Handwritten Arabic Character Recognition, Incremental Learning.

1. INTRODUCTION

Arabic handwriting recognition is a critical research area within the broader field of document image analysis and pattern recognition [1]. It has numerous applications in educational technologies, digital archiving, assistive technologies, and automated language tutoring systems [1, 2]. However, the Arabic script presents unique challenges that make recognition tasks particularly demanding [3, 4]. The Arabic alphabet consists of 28 characters and has sixteen (16) main strokes. Although identical strokes are used for different characters, the number of dots associated with the character distinguishes each character from the other [5, 6]. Unlike Latin-based scripts, Arabic is cursive in both printed and handwritten forms, and its characters adopt different shapes depending on their position in a word (isolated, initial, medial, or final) which increases the characters to as many as 100 [7, 8]. This variability significantly increases the complexity of developing robust recognition models.

An equally important but often underrepresented component in Arabic text is the use of diacritical marks, which are essential for accurate pronunciation and disambiguation [1, 9, 10]. Diacritics, which are small marks drawn either above or below the base character, are especially significant in educational contexts, where learners rely on them for correct pronunciation and comprehension of words [11-14]. Despite their importance, many existing recognition systems either ignore diacritics or treat them as optional features, limiting their effectiveness in real-world and educational applications [15, 16].

This paper addresses the integration of diacritics into handwritten Arabic character recognition. We propose an incremental learning framework that adapts a pretrained CNN model, initially trained on non-diacritic characters, to recognize 492 diacritic-rich classes. We also introduce AHAD (Arabic Handwritten Alphabet with Diacritics), a novel dataset of handwritten Arabic characters with five diacritics, aiming to improve recognition accuracy in diacritic-dependent contexts.

2. RELATED WORKS

Arabic handwriting recognition has advanced significantly in the past decade, mainly due to deep learning and larger annotated datasets. Traditionally, approaches relied on handcrafted features with classifiers like Hidden Markov Models (HMMs) [17], Support Vector Machines (SVMs) [18], or k-Nearest Neighbors (k-NN) [19].

In recent years, CNNs have become the standard for handwriting recognition due to their ability to learn spatial hierarchies of features. Models like VGGNet, ResNet, and custom CNNs have been successful on various Arabic datasets, particularly those without diacritics[20-22]. High recognition accuracies have been achieved on benchmark datasets such as AHCD (Arabic Handwritten Characters Dataset) [23], AlexU Isolated Alphabet (AIA9K) dataset [24], IFN/ENIT (Institute for Communications Technology/Ecole Nationale d'Ingénieurs de Tunis) [25], Hijja [26], HACDB (Handwritten Arabic Characters DataBase) [15], and HMBD [27]. However, most studies focus on isolated characters or words without diacritics, which limits their use in applications like language learning, text interpretation, manuscript digitization, and speech-driven processing, where diacritics are crucial for accurate meaning [14].

While the HACDB and HMBD datasets are valuable for Arabic character recognition, they have notable limitations. HACDB omits dots to reduce class complexity, while HMBD lacks full coverage of all 28 characters in their positional forms. Neither dataset includes vowel diacritics, limiting their use in real-world applications where full character representation, including diacritics, is essential.

Some efforts to address diacritic-aware recognition use multi-stage pipelines or post-processing, treating diacritics separately from the main character [28, 29]. However, these methods often struggle with generalizability and integration. In [29], a segmentation method for offline Arabic words was proposed, but it may struggle with cursive handwriting and diacritics.

Other studies, like [30] have explored multi-task learning for diacritic restoration, but these approaches still rely on segmentation quality and face challenges with cursive handwriting.

In parallel, incremental and transfer learning strategies have gained traction in other domains for improving model adaptation without catastrophic forgetting, especially when transitioning from coarse to fine-grained tasks, as they retain and adapt previously learned features to new data. In [31] the authors explored the challenges of fine-grained image retrieval in an incremental learning setting. They proposed a method that mitigates catastrophic forgetting by employing a regularization function based on Maximum Mean Discrepancy, enabling the model to retain previously learned features while adapting to new classes. Likewise, [32] proposed exemplar-free strategies to reduce semantic drift in incremental learning. Similarly, [33] combined continual and transfer learning using sibling network for knowledge retention and adaptation. These approaches underscore the utility of incremental learning in preserving knowledge during fine-grained classification tasks. However, their application to Arabic handwriting recognition especially in the context of diacritics remains relatively underexplored.

Incremental learning strategies have shown promise in adapting models to new classes without forgetting previously learned knowledge. This study builds upon this idea by combining the strength of CNN-based recognition with an incremental learning paradigm. Whereas we incrementally fine-tune a CNN pretrained on non-diacritic characters to recognize diacritic-enriched characters.

3. ORIGINALITY

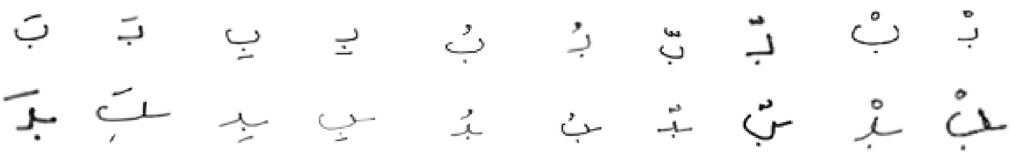
This study introduces a novel approach to Arabic handwritten character recognition by incorporating diacritical marks, essential for accurate interpretation in educational, linguistic, and historical contexts. We developed AHAD, the first dataset of handwritten Arabic characters with diacritics (Fathah, Kasra, Dammah, Shaddah, and Sukoon) across 100 contextual forms, resulting in 492 classes. The originality of this work lies in the incremental learning strategy applied to a custom CNN, fine-tuned on this diacritic-enriched dataset. This method transfers knowledge from non-diacritic to diacritic-rich characters, avoiding catastrophic forgetting. Compared to traditional methods, this approach provides a more robust and scalable solution for real-world Arabic handwriting recognition applications.

4. SYSTEM DESIGN

This section outlines the methodology, centered on an incremental learning strategy using a custom CNN. The process includes three stages: (i) initial training on a base dataset of Arabic characters without diacritics, (ii) architectural modifications for a more complex task, and (iii) two-phase fine-tuning on a diacritic-enriched dataset. Details of each component are provided below.

4.1 Dataset Description and Preprocessing

To support this study, we developed a new benchmark dataset named AHAD (Arabic Handwritten Alphabet with Diacritics) dataset. AHAD



(a) (b) (c) (d) (e)

Figure 1. Samples from AHAD showing the handwritten Arabic character Baa (ب) in all contextual forms with each of the five primary vowel diacritics: (a) Fathah (بَ), (b) Kasrah (بِ), (c) Dammah (بُ), (d) Shaddah (بّ), and (e) Sukoon (بْ). Each image demonstrates the combination of character form and diacritic used in classification

comprises 71,061 grayscale images of handwritten Arabic characters annotated with five primary diacritics: Fathah ◌َ, Kasrah ◌ِ, Dammah ◌ُ, Shaddah ◌ّ, and Sukoon ◌ْ. The dataset was collected from 20 volunteers, ensuring diversity in handwriting styles. As illustrated in Figure 1, each character appears in all its contextual forms (isolated, initial, medial, and final), resulting in a total of 492 distinct classes that combine base character identity, position, and diacritic, reflecting realistic writing conditions. All images are normalized to a resolution of 64×64 pixels.

4.2 Base CNN Architecture

The original CNN was designed to handle a 114-class classification problem involving handwritten Arabic characters in their various contextual forms (isolated, initial, medial, and final), but without diacritical marks. The model consists of the following layers:

- **Input Layer**: Accepts 64×64 grayscale images of handwritten characters.
- **Convolutional Block 1**: Comprising a 2D convolutional layer with 32 filters of size 3×3, followed by ReLU activation.

- **Max Pooling 1**: A 2×2 pooling layer to reduce spatial dimensions and overfitting.
- **Convolutional Block 2**: A second convolutional layer with 64 filters of size 3×3, again followed by ReLU activation.
- **Max Pooling 2**: Another 2×2 pooling operation.
- **Flatten Layer**: Converts the 2D feature maps into a 1D vector.
- **Fully Connected Layer 1**: A dense layer with 128 units and ReLU activation.
- **Output Layer**: A final softmax layer that outputs a 114-dimensional probability vector corresponding to the character classes.

The training objective is to minimize the categorical cross-entropy loss:

$$\mathcal{L}_{base} = -\sum_{i=1}^{N} y_i log(\hat{y}_i) \tag{1}$$

Where $\mathcal{L}$ is the cross-entropy loss, N is the number of classes (114 for the pretrained model), y_i is the true label for class i (1 if class i is correct, 0 otherwise), and $\hat{y}_i$ is the predicted probability for class i (output from softmax).

4.3 Architecture Modification for Diacritics Recognition

To adapt the pretrained model for recognizing characters with diacritics, the architecture was modified to accommodate 492 output classes, each representing a unique combination of character form and diacritic. The primary changes involved:

i. **Loading the pretrained model:** We loaded the CNN model pretrained on a character-only dataset and froze all initial layers (except the final classification layer) to preserve learned features.

$$\text{For all layers; } l \in L_{frozen},\ \theta_l = constant \tag{2}$$

Where l is the index layer, L_{frozen} denotes the frozen layers, and θ_l represents the parameters (i.e. weights and biases) of layer l.

ii. **Replacing the Output Layer**: The original 114-unit softmax layer was replaced with a 492-unit softmax layer to match the number of classes in the new dataset.

$$y_{new} = softmax(W_{new} \cdot x + b_{new}) \tag{3}$$

Where x is the input vector, W_{new} is the weight matrix of the new layer, b_{new} is the bias vector of the new layer, and y_{new} is the output of the new layer (after applying softmax activation).]

iii. **Weight Initialization**: The new output layer was initialized randomly, while the rest of the network retained the pretrained weights.

iv. **Layer Unfreezing Strategy**: A subset of deeper layers was unfrozen to allow gradient updates during fine-tuning, promoting feature adaptation without erasing the previously learned representations.

This setup enables the model to leverage existing knowledge while gradually learning the nuanced differences introduced by diacritics.

4.4 Two-Phase Fine-Tuning Strategy

Fine-tuning was conducted in two phases to ensure stable convergence and prevent overfitting:

4.4.1 Phase 1: Partial Fine-Tuning

In the first phase of fine-tuning, the training was selectively applied to the newly added output layer and a subset of the upper (more abstract) layers of the CNN, while keeping the earlier convolutional layers frozen. This approach was adopted to preserve the foundational features learned from the Arabic characters (without diacritics), and to prevent destabilizing the well-internalized low-level representations such as stroke patterns, curvature, and edge orientations. These fundamental visual features are generally shared across both diacritic and non-diacritic characters and therefore, do not require retraining.

A moderately low learning rate (0.0001) was employed during this phase to enable gradual adaptation of the model to the new target domain comprising 492 diacritic-enriched classes. The learning rate was deliberately kept small to avoid abrupt changes in the network weights, which could otherwise lead to the phenomenon of catastrophic forgetting. Catastrophic forgetting is a common issue in incremental learning where previously learned knowledge is overwritten by new information. By constraining the updates to only a limited portion of the network, the model was able to specialize its output space for fine-grained class discrimination, while still leveraging the general feature hierarchies formed during initial training. This partial fine-tuning strategy not only ensures stable convergence but also significantly accelerates the adaptation process compared to training from scratch.

4.4.2 Phase 2: Full Fine-Tuning

Following the initial phase of partial fine-tuning, the second phase involved unfreezing all layers of the CNN and performing end-to-end training across the entire network. This stage was crucial for aligning both low-level and high-level feature representations with the new, fine-grained classification task involving diacritic-aware Arabic characters. By allowing weight updates across all layers, the network could refine earlier learned features, such as edges, curves, and stroke transitions to become more sensitive to subtle visual distinction introduced by diacritical marks.

To ensure stable optimization and avoid overfitting, the learning rate was reduced by a factor of 10 between phases (from 0.0001 to 0.00001). This cautious reduction allowed the model to make smaller and more deliberate updates to the network parameters, thereby, minimizing the risk of disrupting the hierarchical feature structure formed during earlier training. A smaller learning rate in this context serves as a form of fine-grained calibration that enables the network to adjust its internal representations more precisely, especially, for ambiguous or visually similar character-diacritic combinations.

$$\eta_{phase\ 2} = 0.1 \times \eta_{phase\ 1} \quad (4)$$

Where η denotes the learning rate.

Furthermore, early stopping and model checkpointing strategies were employed based on the validation accuracy to avoid unnecessary training cycles and to retain the best-performing model. The combination of a fully trainable architecture and conservative learning dynamics allowed the model to converge more effectively, improving generalization on the test set. This two-phase approach ultimately led to better performance than direct full fine-tuning or training from scratch, highlighting the value of incremental transfer learning in domains with dense class granularity, such as diacritic-rich handwriting recognition.

4.5 Loss Function and Optimization

The training loss during both fine-tuning phases remained categorical cross-entropy, whereas the number of classes (N) in equation 1, is updated to 492. The model optimization was carried out using the Adam optimizer, a widely adopted adaptive learning rate algorithm known for its computational efficiency and low memory requirements. Adam combines the advantages of two other extensions of stochastic gradient descent: Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp), enabling it to handle sparse gradients and noisy updates effectively. The optimizer was configured with default hyperparameters, specifically $\beta_1 = 0.9$ and $\beta_2 = 0.999$, as originally proposed by [34]. These values control the exponential decay rates of the first and second moment estimates of the gradients, respectively, and have been empirically validated across a wide range of deep learning tasks to offer a stable convergence behavior without the need for extensive tuning. A mini-batch size of 32 was chosen as a balance between computational efficiency and generalization performance. Smaller batch sizes tend to introduce gradient noise, which can help escape shallow local minima and improve generalization, whereas excessively large batches may lead to faster convergence but poorer generalization. Batch size 32 is commonly reported in literature as an effective trade-off, especially for image classification tasks involving medium-sized datasets [35]. This configuration ensured that the training was stable and memory-efficient, while also supporting convergence in a reasonable number of epochs. The entire training pipeline is summarized in Figure 2.

4.6 Evaluation Metrics

The metrics used to evaluate the proposed model are defined in the following equations:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

$$F1\ Score = 2.\frac{Precision \cdot Recall}{Precision + Recall} \tag{8}$$

Figure 2. Training Pipeline and Flow

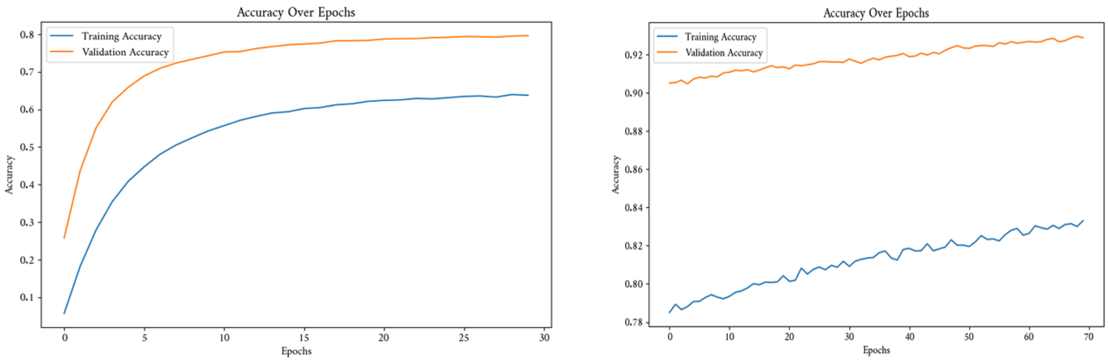
5. EXPERIMENT AND ANALYSIS

The experimental evaluation assessed the proposed incremental learning strategy for recognizing handwritten Arabic characters with diacritics. A CNN model was pretrained in 114 non-diacritic Arabic classes, then fine-tuned into two phases on a new dataset of 492 classes, including five diacritical marks. The AHAD dataset, with 71,061 images from 20 individuals, was split into training (70%), validation (15%), and test (15%) sets. Model performance was evaluated using accuracy, precision, recall, and F1-score.

During the first phase of fine-tuning, the validation accuracy improved significantly from an initial 25.79% to 79.62% over 30 epochs, accompanied by a consistent decline in validation loss from 3.86 to 0.84. This confirmed that the pretrained model was effectively adapting to the more complex, diacritic-aware classification task. Further fine-tuning in the second phase, where all layers were unfrozen and a reduced learning rate was applied, validation accuracy further improved to 92.96% after additional 70 epochs, while the loss decreased to 0.24, indicating enhanced generalization.

Figure 3(a and b) shows the training and validation accuracy curves during the first and second phases of fine-tuning. In both (a) and (b), a noticeable gap exists between the two curves, with validation accuracy consistently outperforming training accuracy throughout the epochs. This behavior is expected, as data augmentation was applied solely to the training set, increasing its complexity and effectively regularizing the model. In contrast, the validation set remained unaltered, serving as a cleaner benchmark for evaluating generalization. The steadily improving and high validation accuracy suggests that the model retained useful features from the pretraining phase and successfully adapted to the more granular diacritic-rich classification task without overfitting.

The performance progression across the training phases as shown in Table 1, demonstrates the effectiveness of the proposed incremental learning approach. During pretraining on 114 classes without diacritics, the model achieved a strong test accuracy of 924% and F1-score of 92.24%. In Fine-tuning Phase 1, where the model was exposed to the expanded 492-class diacritic-aware dataset, performance initially dropped to 79.89% F1-score due to the increased complexity and granularity of the task. However, after full fine-tuning in Phase 2, the model recovered and surpassed its earlier performance, reaching a validation accuracy of 92.96%, test accuracy of 93.26%, and F1-score of 93.24%. This improvement underscores the model's capacity to retain foundational features while adapting effectively to the diacritic-rich character space.



(a) (b)

Figure 3. Training and validation accuracy during fine-tuning: (a) Phase 1 (partial fine-tuning of the output and upper layers); (b) Phase 2 (full fine-tuning of all network layers).

Table 1. Performance of the Proposed Method Across Phases and Test Set

Phase	Classes	Validation Loss	Metric (%)				
			Validation Accuracy	Test Accuracy	Precision	Recall	F1-Score
Pretraining	114	0.27	92.20	92.24	92.48	92.24	92.24
Fine-tuning Phase 1	492	0.84	79.62	80.18	81.61	80.18	79.89
Fine-tuning Phase 2	492	0.24	92.96	93.26	93.61	93.26	93.24

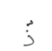
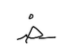
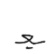
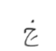
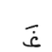
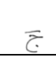
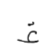
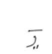
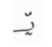
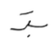
Table 2 presents a comparative analysis of recent AHCR methods. El-Sawy et al. [23], who first introduced the AHCD dataset, achieved 94.9% accuracy using a CNN with dropout-based regularization. Altwaijry and Al-Turaiki [26] later improved upon this with a CNN model which they evaluated on both the AHCD dataset and their new dataset named Hijja that includes handwritten characters written by children aged 7-12. The model achieved 97% accuracy on AHCD, though performance dropped to 88% on the Hijja dataset, reflecting reduced robustness to more challenging handwriting such as children's. Balaha et al. [36] identifies 14 native CNN architectures, with varying layer hierarchies, through extensive experimentation and reported the best model achieving a test accuracy of 91.96% on their HMBD dataset. They further applied optimization techniques such as transfer learning and a genetic algorithm using VGG16 to improve the accuracy to 92.88%. Ullah and Jamjoom [7] further enhanced AHCD recognition using dropout regularization, and reported 96.78% test accuracy. In comparison, our proposed method achieved 93.26% accuracy on the significantly more complex AHAD dataset with 492 classes, demonstrating strong performance through an incremental fine-tuning approach and validating its scalability to diacritic-rich character classification.

Table 2. Comparison with Existing CNN Methods for AHCR

Study	Dataset Used	Classes	Model / Strategy	Accuracy (%)	Remark
[23]	AHCD	28	CNN + Regularization	94.9	Contains only 28 characters
[26]	AHCD & Hijja	28 (AHCD) 29 (Hijja)	CNN + Dropout + BN	97 (AHCD) 88 (Hijja)	Although Hijja is said to contain different character forms, classification task was only for 28 characters.
[36]	HMBD	105 (Characters) 10 (Digits)	14 Native CNNs + TL + GA (HMB-AHCR-DLGA)	92.88	Class complexity extends to 105 characters. Incomplete coverage of character forms, and no diacritics.
[7]	AHCD	28	CNN + Dropout Regularization	96.78	Classifcation conducted for 28 characters only.
Proposed Method	AHAD Proposed Dataset	492	CNN (Incremental Approach)	93.26	Complex dataset. 492 character classes including five diacritics (Fathah, Kasrah, Dammah, Shaddah, and Sukoon)

The class-wise performance analysis showed that the majority of character-diacritic combinations achieved F1-scores above 90%, reflecting strong recognition capabilities. Perfect or near-perfect performance was observed in majority of the classes. As shown in Table 3, several classes including ـعِ, ـقِ, ـطِ, ذّ, and ـضْـ, attained an F1-score of 100%, indicating the model’s ability to generalize well and distinguish highly discriminative features.

Table 3. Selected Class-wise Performance Showing Highest and Lowest F1-Scores

Class	Sample Image	Per-Class Evaluation (%)			Class	Sample Image	Per-Class Evaluation (%)		
		Precision	Recall	F1-score			Precision	Recall	F1-score
ذّ		100	100	100	ـصْـ		87.50	73.68	80.00
ـطِ		100	100	100	ـطْ		76.19	84.21	80.00
ـقِ		100	100	100	ـكّ		92.31	70.59	80.00
ـعِ		100	100	100	ـقْ		92.31	70.59	80.00
ـضْـ		100	100	100	ـفْـ		81.25	76.47	78.79
ـصِـ		100	100	100	حِ		80.00	76.19	78.05
صِ		100	100	100	خُ		69.57	88.89	78.05
ـصِ		100	100	100	خِ		82.35	73.68	77.78
ـسّـ		100	100	100	غْ		68.18	88.24	76.92
ـشِ		100	100	100	ـخِ		82.35	70.00	75.68
ـشَـ		100	100	100	ـطْ		86.67	68.42	76.47
طَ		100	100	100	ـغْ		76.47	76.47	76.47
غَ		100	100	100	ـفّ		81.25	72.22	76.47
عْ		100	100	100	جَ		70.37	82.61	76.00
عّ		100	100	100	ظَ		70.00	82.35	75.68
ـزِ		100	100	100	ـفّـ		72.22	76.47	74.29
ـزّ		100	100	100	ـثْـ		77.78	70.00	73.68
يَـ		100	100	100	ظّ		84.62	64.71	73.33
يّـ		100	100	100	ـبَـ		83.33	55.56	66.67

However, a small subset of classes, such as ظّـ ,ـبَـ, and ثْـ, showed relatively lower performance, with F1-scores falling between approximately 66% and 74%. These underperforming combinations may stem from subtle handwriting variations or overlapping visual similarities between characters. In summary, the model correctly classified approximately 9,940 instances, out of a total of 10,660 test samples, resulting in only around 720 misclassifications. This reflects a strong recognition capability, with an overall classification accuracy exceeding 93%, across the large and diverse set of 492 handwritten Arabic character–diacritic combinations.

The experimental results confirm the effectiveness of the proposed approach in distinguishing visually similar characters, even with subtle diacritical variations. Transferring learned representations to finer classification tasks reduced training time while maintaining high accuracy. These results emphasize the importance of diacritic-aware modeling in Arabic handwriting recognition, particularly for educational and linguistic applications.

To contextualize the proposed CNN-based incremental learning framework, it was briefly compared with other diacritic-aware methods. Qaroush et al. [14] applied a hierarchical divide-and-conquer method using baseline detection and projection profiles for printed Arabic texts, but its suitability for handwritten data is limited. Maghraby and Samkari [12] used a CNN-GRU hybrid with CTC loss, achieving good accuracy on isolated words, though scalability remains uncertain. Transformer-based models such as Ocformer [21] show promise for contextual dependencies, yet character-level diacritic recognition remains underexplored. In contrast, the proposed approach offers robust performance and scalability for fine-grained diacritic-rich classification.

6. CONCLUSION

This study presented a diacritic-aware recognition framework using incremental fine-tuning of CNN and the new AHAD dataset, consisting of 492 Arabic character-diacritic classes. The method showed strong performance, highlighting its effectiveness for fine-grained classification. These results have practical applications in educational tools and language learning systems where accurate diacritic interpretation is crucial. Diacritic-aware modeling extends beyond handwriting recognition to NLP tasks like speech recognition, text-to-speech, and machine translation. Accurate diacritic interpretation enhances pronunciation and word sense differentiation, especially in morphologically rich languages like Arabic. Advances in this area can lead to more precise, human-aligned NLP systems. Future work will explore sequence-based models for connected Arabic scripts, extend the AHAD dataset with full words and phrases, and investigate Transformer models for long-range dependencies. Additionally, multilingual handwriting extensions will be explored to assess the approach's generalizability across scripts.

Acknowledgments

The authors thank Universiti Teknologi Malaysia (UTM) for their support and resources in making this research possible.

REFERENCES

[1] A. A. A. Ali, M. Suresha, and H. A. M. Ahmed, **A survey on arabic handwritten character recognition**, *SN Computer Science,* vol. 1, no. 3, p. 152, 2020.

[2] R. Najam and S. Faizullah, **Analysis of Recent Deep Learning Techniques for Arabic Handwritten-Text OCR and Post-OCR Correction**, *Applied Sciences,* vol. 13, no. 13, p. 7568, 2023.

[3] N. Rahal, M. Tounsi, A. Hussain, and A. M. Alimi, **Deep Sparse Auto-Encoder Features Learning for Arabic Text Recognition**, *IEEE Access,* vol. 9, pp. 18569-18584, 2021.

[4] M. Alghamdi and W. Teahan, **Printed Arabic script recognition: A survey**, *International Journal of Advanced Computer Science and Applications,* vol. 9, no. 9, 2018.

[5] H. M. Balaha, H. A. Ali, and M. Badawy, **Automatic recognition of handwritten Arabic characters: a comprehensive review**, *Neural Computing and Applications,* vol. 33, pp. 3011-3034, 2021.

[6] R. Ahmed *et al.*, **Offline arabic handwriting recognition using deep machine learning: A review of recent advances**, in *Advances in Brain Inspired Cognitive Systems: 10th International Conference, BICS 2019, Guangzhou, China, July 13–14, 2019, Proceedings 10*, 2020: Springer, pp. 457-468, 2020.

[7] Z. Ullah and M. Jamjoom, **An intelligent approach for Arabic handwritten letter recognition using convolutional neural network**, *PeerJ Computer Science,* vol. 8, p. e995, 2022.

[8] A. A. A. Ali and M. Suresha, **Survey on segmentation and recognition of handwritten arabic script**, *SN Computer Science,* vol. 1, no. 4, p. 192, 2020.

[9] N. Alrobah and S. Albahli, **Arabic handwritten recognition using deep learning: A Survey**, *Arabian Journal for Science and Engineering,* vol. 47, no. 8, pp. 9943-9963, 2022.

[10] M. Tanvir Parvez and S. A. Mahmoud, **Offline Arabic Handwritten Text Recognition: A Survey**, *ACM computing surveys,* vol. 45, no. 2, 2013.

[11] H. Almuallim and S. Yamaguchi, **A method of recognition of Arabic cursive handwriting**, *IEEE transactions on pattern analysis and machine intelligence,* no. 5, pp. 715-722, 1987.

[12] A. Maghraby and E. Samkari, **Arabic Text Recognition with Harakat Using Deep Learning**, *IJCSNS,* vol. 23, no. 1, p. 41, 2023.

[13] Y. M. Alginahi, **A survey on Arabic character segmentation**, *International Journal on Document Analysis and Recognition (IJDAR),* vol. 16, no. 2, pp. 105-126, 2013.

[14] A. Qaroush, A. Awad, A. Hanani, K. Mohammad, B. Jaber, and A. Hasheesh, **Learning-free, divide and conquer text-line extraction algorithm for printed Arabic text with diacritics**, *Journal of King Saud University-Computer and Information Sciences,* vol. 34, no. 9, pp. 7699-7709, 2022.

[15] A. Lawgali, M. Angelova, and A. Bouridane, **HACDB: Handwritten Arabic characters database for automatic character recognition**, in *European workshop on visual information processing (EUVIP)*, 2013: IEEE, pp. 255-259, 2013.

[16] M. Elkhayati and Y. Elkettani, **Arabic handwritten text line segmentation using a multi-agent system and a directed CNN**, presented at the 2021 Fifth International Conference On Intelligent Computing in Data Sciences (ICDS), 2021.

[17] H. M. Eraqi and S. Abdelazeem, **HMM-based offline Arabic handwriting recognition: Using new feature extraction and lexicon ranking techniques**, in *2012 International Conference on Frontiers in Handwriting Recognition*, 2012: IEEE, pp. 554-559, 2012.

[18] A. A. A. Ali and S. Mallaiah, **Intelligent handwritten recognition using hybrid CNN architectures based-SVM classifier with dropout**, *Journal of King Saud University-Computer and Information Sciences,* vol. 34, no. 6, pp. 3294-3300, 2022.

[19] S. Kanoun, A. M. Alimi, and Y. Lecourtier, **Natural Language Morphology Integration in Off-Line Arabic Optical Text Recognition**, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 41, no. 2, pp. 579-590, 2011.

[20] N. A.-A. Nahla Ibrahim Youssef, **A REVIEW ON ARABIC HANDWRITING RECOGNITION**, *Journal of Southwest Jiaotong University,* vol. 57, no. 6, 2022.

[21] A. Mostafa *et al.*, **Ocformer: A transformer-based model for arabic handwritten text recognition**, in *2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, 2021: IEEE, pp. 182-186, 2021.

[22] W. Albattah and S. Albahli, **Intelligent arabic handwriting recognition using different standalone and hybrid CNN architectures**, *Applied Sciences,* vol. 12, no. 19, p. 10155, 2022.

[23] A. El-Sawy, M. Loey, and H. El-Bakry, **Arabic handwritten characters recognition using convolutional neural network**, *WSEAS Transactions on Computer Research,* vol. 5, no. 1, pp. 11-19, 2017.

[24] M. Torki, M. E. Hussein, A. Elsallamy, M. Fayyaz, and S. Yaser, **Window-based descriptors for Arabic handwritten alphabet recognition: a comparative study on a novel dataset**, *arXiv preprint arXiv:1411.3519,* 2014.

[25] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri, **IFN/ENIT-database of handwritten Arabic words**, in *Proc. of CIFED*, 2002, vol. 2: Citeseer, pp. 127-136, 2002.

[26] N. Altwaijry and I. Al-Turaiki, **Arabic handwriting recognition system using convolutional neural network**, *Neural Computing and Applications,* vol. 33, no. 7, pp. 2249-2261, 2020.

[27] H. M. Balaha, H. A. Ali, M. Saraya, and M. Badawy, **A new Arabic handwritten character recognition deep learning system (AHCR-DLS)**, *Neural Computing and Applications,* vol. 33, pp. 6325-6367, 2021.

[28] M. K. Dhamad, **Arabic Hand Written Segmentation and Recognition**, Baghdad University, 2014.

[29] N. AbdAllah and S. Viriri, **Off-Line arabic handwritten words segmentation using morphological operators**, *arXiv preprint arXiv:2101.02797,* 2021.

[30] S. Alqahtani, A. Mishra, and M. Diab, **A multitask learning approach for diacritic restoration**, *arXiv preprint arXiv:2006.04016,* 2020.

[31] W. Chen, Y. Liu, W. Wang, T. Tuytelaars, E. M. Bakker, and M. Lew, **On the exploration of incremental learning for fine-grained image retrieval**, *arXiv preprint arXiv:2010.08020,* 2020.

[32] L. Yu *et al.*, **Semantic drift compensation for class-incremental learning.**, in *CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 2020 IEEE, pp. 6980-6989, 2020.

[33] M. Boschini *et al.*, **Transfer without forgetting**, in *European conference on computer vision*, 2022: Springer, pp. 692-709, 2022.

[34] D. P. Kingma and J. Ba, **Adam: A method for stochastic optimization**, in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014: Banff, 2014.

[35] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, **On large-batch training for deep learning: Generalization gap and sharp minima**, *arXiv preprint arXiv:1609.04836,* 2017.

[36] H. M. Balaha *et al.*, **Recognizing arabic handwritten characters using deep learning and genetic algorithms**, *Multimedia Tools and Applications,* vol. 80, pp. 32473-32509, 2021.