A Hybrid Approach for Deep Generative Handwritten Arabic Text Recognition

Hicham Lamtougui [1,*], Hicham El Moubtahij [2], Hassan Fouadi [1], and Khalid Satori [1]

[1] Computer Science Department, Faculty of Sciences Dhar-Mahraz,
Sidi Mohamed Ben Abdellah University, Fez, Morocco
[2] Systems and Technologies of Information Team, High School of Technology,
University of Ibn Zohr, Agadir, Morocco
Email: hicham.lamtougui@usmba.ac.ma (H.L.); h.elmoubtahij@uiz.ac.ma (H.E.M.);
hassan.fouadi@usmba.ac.ma (H.F.); khalid.satori@usmba.ac.ma (K.S.)
*Corresponding author

***Abstract*—Automatic offline recognition of handwritten Arabic characters poses a significant challenge in various application domains. While notable progress has been made in this area, challenges remain, particularly regarding children's handwriting. Indeed, the latter often has less legible characters, which complicates the task of automatic recognition. In this paper, we propose two innovative deep learning-based approaches to effectively identify children's Arabic handwriting. The developed models are based on the promising architectures of Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs). They were trained on the Hijja dataset, a rich collection of handwritten Arabic letters by Arabic-speaking schoolchildren in Saudi Arabia in 2019. The rise of generative adversarial learning has sparked particular interest in Variational Autoencoders (VAEs) and GAN networks. However, the intrinsic limitations of GANs in terms of inference have led to the adoption of hybrid models combining the strengths of VAEs and GANs. The results obtained are particularly encouraging, with our hybrid models achieving a remarkable accuracy rate of 97%, surpassing existing models in the literature.**

***Keywords*—handwritten Arabic, Variational Autoencoders (VAE), Generative Adversarial Networks (GAN), Hijja database**

I. Introduction

Arabic, spoken by over 500 million people worldwide, is a major language. Its alphabet, consisting of 28 letters, is characterized by right-to-left writing. The shape of each letter varies depending on its position within a word, whether it is at the beginning, middle, end, or in isolation, as demonstrated in Table I. The arrangement of each letter in the Arabic alphabet is thus influenced by its context within the word [1].

Handwriting identification is a challenge in the field of computer vision. This task involves the automation of handwritten text recognition on computers and converting texts from sources such as documents or touchscreens into a machine-readable format. The image can be captured offline from physical media such as paper or photographs, or online if the source is digital, such as with touchscreens [2].

TABLE I. Arabic Alphabet Characters with Different Positions

Character	Name	End	Middle	Begin
ا	Alif	ـا	ـا	ا
ب	Ba	ـب	ـبـ	بـ
ت	Ta	ـت	ـتـ	تـ
ث	Tha	ـث	ـثـ	ثـ
ج	Jim	ـج	ـجـ	جـ
ح	Ha	ـح	ـحـ	حـ
خ	Kha	ـخ	ـخـ	خـ
د	Dal	ـد	ـد	دـ
ذ	Dhal	ـذ	ـذ	ذـ
ر	Ra	ـر	ـر	رـ
ز	Zay	ـز	ـز	زـ
س	Sin	ـس	ـسـ	سـ
ص	Sad	ـص	ـصـ	صـ
ض	Dad	ـض	ـضـ	ضـ
ط	Ta	ـط	ـطـ	طـ
ظ	Zae	ـظ	ـظـ	ظـ
ع	Ayn	ـع	ـعـ	عـ
غ	Ghayn	ـغ	ـغـ	غـ
ف	Fae	ـف	ـفـ	فـ
ق	Qaf	ـق	ـقـ	قـ
ك	Kaf	ـك	ـكـ	كـ
ل	Lam	ـل	ـلـ	لـ
م	Mim	ـم	ـمـ	مـ
ن	Nun	ـن	ـنـ	نـ
ه	Ha	ـه	ـهـ	هـ
و	Waw	ـو	ـو	و
ي	Yae	ـي	ـيـ	يـ
ء	Hamza	ـئ	ـؤ	أ

In recent years, interest in the field of handwritten document recognition, especially in Arabic, has increased significantly. In the category of handwritten and printed documents, the task of automatic handwritten document recognition is becoming increasingly complex [3].

The diversity of handwritten characters, varying in size and shape depending on the individuals who write them, poses a significant challenge for recognition. Similarities between different character shapes, overlaps, ligatures, and connections between adjacent characters add to the complexity of this task [4].

Manuscript received February 16, 2024; revised May 20, 2024; accepted June 12, 2024; published October 21, 2024.

Unlike Latin and Chinese, which have seen significant advances in character recognition, research progress in Arabic character recognition has remained limited. This makes Arabic character recognition a field of research that is still largely open.

The challenges associated with Arabic text recognition are complex and can be summarized as follows [5]:

- Arabic text is written in cursive, with interconnected characters forming blocks.
- The cursive nature and varied shapes of Arabic characters make accurate classification and recognition difficult.
- The formation of Arabic words from one or more blocks necessitates adaptive segmentation algorithms.
- Arabic characters vary in shape depending on their position in the word: initial, medial, final, or isolated.
- Some characters can be oriented vertically or horizontally, with multiple baselines, making existing baseline detection methods less applicable.
- Diacritics and external elements such as dots, the "Hamza," and the "Madda", must be taken into account.
- Length variations of the same character in different Arabic texts are common.
- Handwritten Arabic characters vary in size and shape, and some combinations of characters create unique shapes.
- The lack of public and accessible datasets in Arabic script limits research compared to other languages such as Latin.

Various machine learning methods and techniques are used for handwritten character classification and recognition [6]. These methods include support vector machines [7] and the K-Nearest Neighbor (KNN) method [8]. On the other hand, Deep Learning (DL) [9], a branch of machine learning, offers a variety of algorithms and architectures such as Convolutional Neural Networks (CNNs) [10], Deep Belief Networks (DBNs) [11], Autoencoders (AEs) [12], Recurrent Neural Networks (RNNs) [13], and Generative Adversarial Networks (GANs) [14]. These techniques have shown promising results in Arabic handwritten character recognition, especially due to their ability to learn complex data features [15].

Generative Adversarial Networks (GANs) [16] are models capable of creating synthetic data based on the examples seen during training. Samples of noise from a prior distribution are used as inputs. In the context of Variational Autoencoders (VAEs) [17], the input is associated with a distribution rather than a fixed vector. The compression vector is replaced by a mean and standard deviation. We propose to use the latent spatial distribution of VAE to extract samples for the generator. We plan to analyze the effects resulting from the fusion of VAE and GAN for recognizing children's handwriting using the Hijja dataset [18].

The rest of the article is divided as follows: Section II discusses related research. Section III presents the proposed architecture in detail. Section IV reports on the experiments conducted, the results obtained, and the discussions that follow. Finally, Section V concludes the article.

II. Literature Review

In this section, we delve into several related works concerning the recognition of handwritten Arabic characters from the Hijja database. We establish connections between these works and provide detailed descriptions of the limitations associated with each method.

Alkhateeb *et al.* [19] developed a custom Convolutional Neural Networks (CNNs) model achieving a high accuracy of 92.5% on the Hijja database. However, the model's architectural complexity could pose challenges in implementation and comprehension. Additionally, the reported performance may be specific to the training datasets used and may not generalize well to other datasets. Altwaijry *et al.* [18] also employed a CNN model, achieving impressive results of 97% and 88% accuracy on the Arabic Handwritten Characters Dataset (AHCD) [20] and Hijja datasets, respectively. However, their method may be limited by its sensitivity to data variations and its focus on recognizing handwritten Arabic characters. Alrobah and Albahli [21] proposed a new approach for recognizing and classifying handwritten Arabic characters. This approach involves a hybrid model using Convolutional Neural Networks (CNN) alongside Support Vector Machine (SVM) and eXtreme Gradient Boosting (XGBoost). The CNN extracts feature from Arabic character images, which are then fed into machine learning classifiers. Their approach achieved a recognition rate of up to 96.3% in 29 classes, surpassing previously reported results. Nevertheless, the increased complexity associated with this hybrid model could prolong the training phase and make it more susceptible to overfitting, necessitating meticulous hyperparameter optimization. To address the data scarcity issue, Wagaa *et al.* [22] introduced an improved version of the previous model by adopting the dropout regularization technique. This technique aims to counteract the risks of overfitting associated with limited data. Additionally, relevant adjustments were made in both the selection of optimization algorithms and data augmentation approaches, all aimed at guaranteeing optimal performance. This new formulation of the model was trained on two separate datasets, AHCD and Hijja, both comprising Arabic handwritten characters. However, this method may still be limited by its sensitivity to hyperparameter choices, which could impact the model's robustness and performance across different datasets. Alheraki *et al.* [23] developed a custom CNN model with impressive accuracies of 97% and 91% on the AHCD and Hijja datasets, respectively. However, its specificity in recognizing the Arabic characters of children may constrain its applicability to other writing styles. Alwagdani and Jaha [24] conducted a detailed study in 2023 on handwritten Arabic character recognition and handwriting discrimination using custom-developed CNN models and hybrid approaches. Their study utilized datasets from both

adults (AHCD) and children (Hijja) to explore performance, focusing on the impact of different training datasets. They investigated classification problems using a standard machine learning pipeline, taking out visual features, and putting things into groups using well-known Machine Learning models like SVM, KNN, and Random Forest (RF). Results revealed that training the model on a combination of children and adult datasets yielded the best performance, achieving an impressive average accuracy of 92.87% in recognizing children's handwritten characters. Additionally, they extended their investigation to writer classification into two groups (children and adults) using the proposed CNN model. Initial results showed an average accuracy of 89.28%, indicating confusing similarities in writing styles between adults and children. The study suggested adding more features based on the Histogram of Oriented Gradients (HOG) and statistical measures to improve discrimination performance. These features, when combined with CNN features, led to a much higher accuracy of 92.29%. Khudeyer and Almoosawi [25] combined ResNet50 with Random Forest, producing precise results. However, modifying the last layer of ResNet50 can increase the architecture's complexity and make the model more sensitive to hyperparameters. This highlights the importance of carefully considering architectural modifications in complex models. While their approach demonstrated high precision, future research could focus on optimizing model complexity without compromising performance, potentially through alternative model architectures or regularization techniques. Further research into their model's ability to generalize across different datasets could also give them useful information about how robust it is and how it can be used in real life. Durayhim *et al.* [26] introduced two models leveraging the architecture of a CNN as well as a pre-trained CNN Visual Geometry Group-16 (VGG-16). The performance of this innovative model was evaluated by comparing it to similar models from previous work. The findings reveal that the developed model excels not only compared to the pre-trained CNN (VGG-16) but also compared to other models from the literature. However, their specificity in recognizing children's characters may limit their applicability to other types of writing, such as adult characters.

In the face of these challenges, our approach, based on a hybrid Variational Autoencoder and Generative Adversarial Network (VAE-GAN) model, aims to surpass the limitations of accurately representing Arabic characters. However, this method may require iterative adjustments to achieve optimal results and can be more complex to implement than other state-of-the-art approaches.

III. Proposed Methodology

We use advanced generative models, such as Generative Adversarial Networks (GANs) [27] and Variational Autoencoders (VAEs), to look at how to recognize handwritten Arabic characters. After a thorough analysis of the Hijja dataset, we perform data preprocessing and propose a GAN-VAE architecture for character generation. Finally, we describe an algorithm specific to our approach.

A. Hijja Dataset

The Hijja dataset, presented by Altwaijry *et al.* [18], is a recent compilation of distinct Arabic characters collected from Arabic-speaking schoolchildren aged 7 to 12 in Riyadh, Saudi Arabia. You can access this dataset at (https://github.com/israksu/Hijja2). It consists of 108 categories, with each category representing an Arabic letter in four different forms for positions at the beginning, middle, and end of a word, as well as when isolated. This dataset contains a total of 47,434 images, distributed across 29 files, each corresponding to an Arabic letter, with a separate file reserved for the Hamza character. Fig. 1 illustrates a sample of the Hijja dataset.

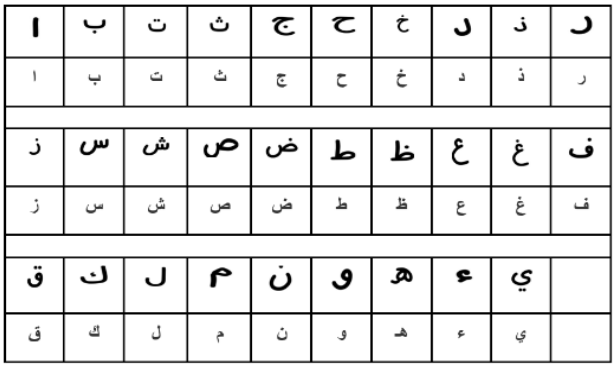

Fig. 1. Sample of the Hijja dataset [18].

The distinctive characteristics of the Hijja dataset make it an invaluable resource for the study of handwritten Arabic characters. The images, uniformly sized at 32×32 pixels, are collected from children, providing a diversity of writing styles and character forms. With contributions from 591 authors, each character is represented by a unique sample of 28 characters per author, totaling an impressive quantity of samples. This variety ensures a comprehensive representation of the intrinsic variations in each character.

On average, each character is represented by approximately 400 to 500 samples, providing significant scope for analysis and training of character recognition models. In addition, there is a segment with 12,355 samples of single characters. This segment lets you look more closely at the unique features of each letter in a context-free setting, which makes it easier to train recognition models.

The authors' category, exclusively composed of children, introduces additional diversity in writing styles and character forms, reflecting the nuances inherent in children's writing. This diversity, combined with the richness of available samples, makes the Hijja dataset an invaluable resource for training and evaluating Arabic character recognition models.

The Hijja dataset represents much more than a mere compilation of Arabic character images. It embodies a valuable data source, offering a wide variety of samples and a detailed representation of letter characteristics, thereby contributing to its utility and relevance in the field of character recognition. Table II provides a detailed description of the characteristics of the Hijja dataset.

TABLE II. CHARACTERISTICS OF THE HIJJA DATASET: A DETAILED OVERVIEW

Characteristics	Description
Source	Arabic-speaking schoolchildren
Type	Handwritten Arabic characters
Number of Categories	108 (one for each Arabic letter)
Variations per Category	4 (beginning, middle, end of word, isolated)
Total Number of Images	47,434
Image Distribution	29 files for Arabic letters, 1 file for Hamza
Image Size	32×32 pixels
Authors	591
Samples per Author	28
Average Samples per Character	400–500
Isolated Character Samples	12,355

B. Data Preprocessing

The preprocessing process of character images in the Hijja database aims to optimize the accuracy of the proposed system. Images are converted to grayscale and undergo inversion to enhance foreground pixels while darkening the background. Subsequently, contrast is adjusted to enhance the intensity of foreground components and reduce background pixel values, which corresponds to image normalization within a range of values between 0 and 1. Following an empirical exploration of image thresholds, thresholding is applied, where values below 90 are considered background pixels and reset to zero. Finally, images are centered around character pixels and resized to a uniform size of 32×32, by the input image dimension specified in the parameter table.

In parallel, a fusion of the training and testing datasets from the Hijja database is performed to create a new dataset. This new dataset comprises 26,880 characters for training and 5,831 characters for testing and is used to train the model on both character recognition and writer group classification tasks. It is noteworthy that, for writer group classification, all images are converted to binary images, unlike character recognition, which uses grayscale images, as specified in Table III.

TABLE III. PREPROCESSING PARAMETERS

Parameter	Value
Input image size	(32, 32, 1)
Image normalization	Division by 255 (between 0 and 1)
Latent space dimension	64

C. Proposed Architecture

In this section, we elaborate in detail on the architecture of our model.

1) Variationnal Autoencoder (VAE)

An autoencoder is a neural network design characterized by an encoder-decoder framework designed to derive a condensed representation of input data. There are many types of autoencoders, each one better at a certain job. These include Convolutional Autoencoders (CAE) [28], Sparse Autoencoders (SAE) [29], Denoising Autoencoders (DAE) [30], and Variational Autoencoders (VAE) [31]. Each variant offers an architecture uniquely suited to its intended application [32].

The Variational Autoencoder (VAE) is a deep learning technique that utilizes both a generative neural network and an inference neural network to address the challenges of variational inference. Its primary objective is to enhance the likelihood of faithfully representing the entire dataset [17]. This methodology yields a generative network adept at producing synthetic data that reflects the characteristics of the original training data. However, given the complexity involved in precisely determining data likelihood, VAE approximates the optimization of the Evidence Lower Bound (ELBO). This optimization translates to gradient ascent on the ensuing objective function, as delineated by the following equations:

$$L_{vae}=D_{kl}(q(z|x)\,\|p(z)) - E_{q(z|x)}[log\ p(x|z)] \quad (1)$$

$$\text{ELBO}=E_{q(z|x)}[\,log\,p(x,z)\,] - E_{q(z|x)}[log\ q(z|x)] \quad (2)$$

In Eq. (1), $p(z)$ represents the distribution of the latent variable z, which is modeled as a Gaussian distribution. The mean and covariance of this distribution are derived by passing noise through the generator. On the other hand, $q(z|x)$ represents the approximate posterior distribution of z, conditioned on the input data instance x. This distribution is also modeled as a Gaussian, with its mean and covariance being determined by processing the data through the inference network.

The overarching objective of the VAE is to minimize the Kullback-Leibler (KL) divergence between these two distributions, $p(z)$ and $q(z|x)$. This minimization encourages the generated data to closely resemble the patterns observed in the training data, as depicted in Fig. 2 of the context. However, it's worth noting that the second term in the objective function has a different effect. This term promotes the generative distribution $p(x|z)$ to exhibit a high level of dispersion or variability. Consequently, this dispersion can lead to synthetic images with a significant degree of blurriness, which is generally considered undesirable in many practical applications.

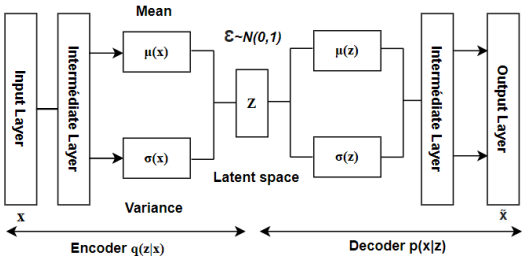

Fig. 2. The schematic structure of VAE.

2) Generative Adversarial Network (GAN)

Goodfellow's GAN model, as described in [14], employs a unique architecture comprising two deep neural networks: the generator and the discriminator, with the primary objective of training an image generator. The generator typically employs a Deconvolutional Neural (DCN) structure, while the discriminator is constructed using Convolutional Neural Networks (CNNs).

In the training process, the generator takes input in the form of fixed-dimensional noise vectors, referred to as latent variables, to generate synthetic images, as visualized

in Fig. 3. These synthetic images are then combined with real images from a dataset and subjected to evaluation by the discriminator. The discriminator's task is to distinguish between real and synthetic images, and its classification accuracy is used to provide feedback to the generator.

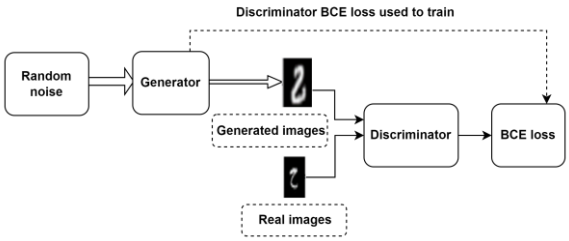

Fig. 3. Generate handwritten characters with GAN.

Consequently, the training objective of the generator is twofold: first, to increase the classification error of the discriminator, and second, to enhance the quality of the synthetic images it generates. Conversely, the discriminator's goal is to reduce its classification error, becoming more adept at differentiating between real and generated data. This training dynamic can be written in mathematical terms as a minimax problem, where G is the mapping from the latent space to the data space and D is the discriminator loss, which measures how well the discriminator sorts the real and fake data together.

The optimization formulation expressed in Eq. (3) can be viewed as a zero-sum game possessing a single equilibrium point. This equilibrium point represents the ideal distribution of generated images, determined by the generator, which effectively resolves the optimization challenge. This formulation offers a comprehensive framework for training deep generative models. However, it's important to note that difficulties arise in training this model when the discriminator becomes overly proficient at its classification task, making it challenging for the generator to improve further.

$$\min_G \max_D V(D,G) = E_{x\sim P_{data}(x)}[logD(x)] + E_{z\sim P_{z(z)}}[1 - log(G(z))] \quad (3)$$

The GAN training process is a balancing act where the generator strives to create realistic data, and the discriminator tries to get better at telling real and fake data apart. When they reach equilibrium, the generator produces high-quality synthetic data that is similar to real data.

3) VAE and GAN hybrid models

An advantage of VAE models over GAN models lies in their ability to establish a correspondence between initial data and latent factors and then reconstruct an image using the approximate generator. However, images generated by VAEs tend to have blurriness and lower quality compared to those produced by GAN models. To leverage the strengths of both approaches, we proposed a hybrid VAE-GAN model, an architecture illustrated in Fig. 4. This architecture, in addition to the original VAE models, proposes the VAE-GAN model. The latter integrates a discriminator on top of the generated images. The discriminator loss function is similar to that used in GANs. The loss function for the decoder and encoder encompasses two distinct components. The first component is analogous to Eq. (1) of VAE. The second component is minimized when the generator manages to induce confusion in the discriminator (Eq. (3)). Thanks to this architecture, the VAE-GAN manages to generate images in the style of GANs while preserving the essential function of establishing a correspondence between an image sample and its latent variables.

When it comes to loss functions, VAE uses Kullback-Leibler (KL divergence) to find the difference between the target distribution and the distribution that the model made. GAN, on the other hand, uses two separate loss functions. The generator has a loss function that encourages it to generate more realistic images, while the discriminator has its loss function to evaluate its ability to discern between real and generated images. VAE aims to generate images by maximizing the probability of matching the original input, while GAN seeks to create a competitive balance between the generator and discriminator to produce realistic images.

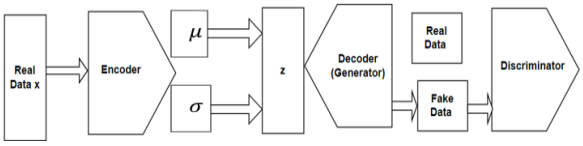

Fig. 4. The architecture of the VAE-GAN model.

The integration of VAE and GAN architectures in a hybrid model addresses the challenges of Arabic handwritten text recognition by combining the benefits of both methods:

- Correspondence and Generation: VAEs establish a correspondence between the initial data and latent factors, enabling precise reconstruction. GANs, on the other hand, excel at generating high-quality images. The VAE-GAN hybrid model leverages the VAEs' ability to encode data into a latent representation while harnessing the power of GANs to enhance the visual quality of generated images.
- Image Quality: Images generated by VAEs may be blurry, while those generated by GANs are sharper and more realistic. The hybrid model capitalizes on the clarity of GANs to produce images that are faithful to the original data and aesthetically superior.
- Loss Functions: VAEs use KL divergence to measure the difference between target and generated distributions, aiding in data reconstruction. GANs employ a loss function for the generator that encourages the creation of realistic images and another for the discriminator that improves its ability to distinguish real from fake images. The hybrid model combines these functions to optimize both reconstruction and generation.
- Competitive Balance: GANs operate on a principle of competitive balance between the generator and discriminator, resulting in highly realistic image

production. The VAE-GAN hybrid model maintains this balance while ensuring that image generation is consistent with latent representations, crucial for character recognition.

The VAE-GAN hybrid model combines the accuracy of VAEs for reconstruction with the generation quality of GANs. This makes it a powerful way to deal with the unique problems of Arabic handwritten text recognition. This integration allows for the generation of Arabic character images that are not only visually compelling but also faithful to the complex variations in handwritten scripts.

D. The Proposed Algorithm

A VAE-GAN hybrid model designed for Arabic handwritten character recognition pursues two primary objectives: generating Arabic characters and enhancing their realism. This algorithm employs the Variational Autoencoder (VAE) to acquire a latent representation of characters. Subsequently, the Generative Adversarial Network (GAN) generator utilizes this learned representation to generate new characters resembling those present in the training set. The model's training unfolds through an iterative process, alternating between VAE and GAN training steps, as outlined in Algorithm 1. In each iteration, a set of M images is sampled from the database, and corresponding latent codes are generated using the encoder. These codes are then used to reconstruct images through the decoder. Additionally, a prior set of latent codes is sampled, and images are generated based on these codes. The training process involves updating the encoder to minimize differences between reconstructed and original images, decreasing the Kullback-Leibler divergence, and refining the discriminator's ability to distinguish between real and generated images. This comprehensive approach results in the generation of more realistic and diverse Arabic characters while retaining a valuable latent representation beneficial for tasks such as character recognition and related applications.

Algorithm 1. Iterative Training for VAE-GAN

1. Initialize Enc, Dec, Disc
2. In each iteration:
3. Sample M images $x^1, x^2, x^3, \ldots\ldots x^M$ from database
4. Generate M codes $\tilde{z}^1, \tilde{z}^2, \tilde{z}^3, \ldots\ldots \tilde{z}^M$ from encoder
5. $\tilde{z}^i$=Enc(x^i)
6. Generate M images $\tilde{x}^1, \tilde{x}^2, \tilde{x}^3, \ldots\ldots \tilde{x}^M$ from decoder
7. $\tilde{x}^i = \text{Dec}(\tilde{z}^i)$
8. Sample M codes $z^1, z^2, z^3, \ldots\ldots z^M$ from prior P(z)
9. Generate M images $\hat{x}^1, \hat{x}^2, \ldots\ldots\ldots \hat{x}^M$ from encoder
10. $\hat{x}^i$=Enc(z^i)
11. Update Enc to descrease $\|\tilde{x}^i - x^i\|$, descrease KL($P(\tilde{z}^i|x^i)\|P(z)$)
12. Update Dec to descrease $\|\tilde{x}^i - x^i\|$, increase $\text{Disc}(\tilde{x}^i)$ and $\text{Disc}(\hat{x}^i)$
13. Update Disc to increase , $\text{Disc}(x^i)$ decrease $\text{Disc}(\tilde{x}^i)$ and $\text{Disc}(\hat{x}^i)$

The VAE-GAN hybrid algorithm transforms the way handwritten Arabic characters are recognized by combining the strengths of the Variational Autoencoder (VAE) and the Generative Adversarial Network (GAN). This process involves three main phases:

- Random sampling: Images of handwritten Arabic characters are randomly selected from the training database.
- Extraction of latent codes: The VAE encoder analyzes these images and extracts compressed latent codes, robustly capturing their essential characteristics.
- Reconstruction and image generation: The VAE decoder uses latent codes to reconstruct the original images, thereby improving its ability to capture the nuances of Arabic characters. Additionally, the algorithm generates new realistic images from predefined latent codes, increasing the diversity of the training data and strengthening the model's robustness.

Furthermore, the process includes discriminative learning, where the GAN discriminator is trained to distinguish between real and artificially generated images. By using adversarial learning, this process pushes the VAE-GAN to make Arabic characters that are more and more like the training data. This makes the model work better overall.

IV. Experimental Results and Discussion

In this section, we present and discuss the results of our experiments. We begin by describing the environment used, implementation details, hyperparameter tuning, then we examine the experimental setup and experiment design. We also address the results and discussion and finally compare them with other state-of-the-art work on the Hijja database.

A. Experimental Setup

Our model was trained using the RMSprop optimizer, with a learning rate of 0.001, over 30 epochs, determined empirically. Python language and the Keras framework were employed for implementation, and all experiments were conducted on Google Colab with GPU acceleration enabled. During training, a learning rate scheduler was utilized: RMSprop initially starts with a learning rate of 0.001 and dynamically optimizes the learning rate during training. However, through experimentation, we observed that incorporating a callback scheduler, which adjusts the learning rate after each epoch following RMSprop optimization, led to improved overall model accuracy. The parameters and techniques utilized for training are summarized in Table IV.

TABLE IV. Summary of Parameter

Hyperparameters	Setting
Input Size	32×32
Batch size	32
learning Rate	0.001
Epochs	30
Optimizer	RMSprop
Loss function	Binary Cross-Entropy (BCE)

B. Evaluation Measures

The results of the proposed deep learning model for multi-category classification were analyzed using metrics such as accuracy, precision, recall, and F1-Score.

consequently, there are four potential outcomes: True Positive (TP) represents positive data correctly identified as positive, True Negative (TN) refers to negative data accurately categorized as negative, False Positive (FP) occurs when negative data is mistakenly labeled as positive.

Accuracy represents the proportion of correctly predicted instances to the total number of predictions:

$$\text{Accuracy}=\frac{\text{TP+TN}}{\text{TP+FP+FN+TN}} \quad (4)$$

Precision (P) signifies the ratio of correctly predicted positive instances to the total number of instances predicted as positive:

$$\text{recision} = \frac{\text{TP}}{\text{TP+FP}} \quad (5)$$

Recall is the ratio between the number of correctly classified images and the total images belonging to the same class.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (6)$$

The F1-Score combines both recall and precision in the following manner:

$$\text{F1} - \text{Score} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision+Recall}} \quad (7)$$

TABLE V. CLASS-WISE ANALYSIS ON THE HIJJA DATASET

Class Number	Class Label	Precision	Recall	F1-Score
0	ا	1.00	1.00	1.00
1	ب	1.00	0.99	1.00
2	ت	0.93	0.97	0.95
3	ث	0.97	0.96	0.97
4	ج	0.99	0.99	0.99
5	ح	0.98	0.98	0.98
6	خ	1.00	0.99	1.00
7	د	0.95	0.99	0.97
8	ذ	0.96	0.90	0.93
9	ر	0.94	0.97	0.96
10	ز	0.95	0.94	0.95
11	س	1.00	0.97	0.98
12	ش	0.99	1.00	1.00
13	ص	0.93	1.00	1.00
14	ض	0.98	0.95	0.97
15	ط	0.97	0.99	0.98
16	ظ	0.98	0.97	0.97
17	ع	0.95	0.99	0.97
18	غ	1.00	0.99	1.00
19	ف	0.94	0.98	0.96
20	ق	0.98	0.93	0.95
21	ك	0.99	0.98	0.99
22	ل	0.98	0.99	0.99
23	م	0.99	1.00	1.00
24	ن	0.98	0.94	0.96
25	ه	0.99	0.97	0.98
26	و	0.94	0.97	0.96
27	ي	1.00	0.96	0.98
Accuracy				**0.97**
macro avg		**0.97**	**0.97**	**0.97**
weighted avg		**0.97**	**0.97**	**0.97**

The Hijja database presents a comprehensive collection of 28 handwritten Arabic characters, ranging from أ (alef) to ي (ya). A detailed analysis of the model's performance, as depicted in Table V, reveals remarkable precision and recall scores for many characters. Notably, classes like أ (alef), خ (kha), and ش (sheen) exhibit near-perfect precision, underscoring the models. proficiency in identifying these characters without significant errors. Similarly, characters such as ش (sheen), ص (sad), and م (meem) demonstrate exceptional recall, indicating the model's effectiveness in capturing these characters whenever they appear. However, a few characters like ذ (dhal) and ق (qaf) display slightly diminished scores, suggesting potential areas for refinement. Overall, with an impressive accuracy of 0.97 and consistent performance metrics across classes, the model showcases robust capabilities in recognizing handwritten Arabic characters from the Hijja database.

C. Implementation Details

During the optimization of the hyperparameters of our hybrid GAN-VAE model for handwritten Arabic character recognition, we provide a comprehensive breakdown of key implementation aspects. We conducted a meticulous hyperparameter tuning process to maximize model performance while avoiding overfitting. The model architecture consists of an encoder, a decoder, and a discriminator. The encoder converts input images into a 64-dimensional latent representation, while the decoder reconstructs images from this latent representation. The discriminator is used to differentiate between real and generated images. The encoder architecture comprises 2D convolutional layers, batch normalization, LeakyReLU activation, and dense layers to encode the image into a 64-dimensional latent vector. The decoder architecture is responsible for reconstructing the image from the latent vector and utilizes dense layers, reshaping, 2D transposed convolution, batch normalization, LeakyReLU, and a final 2D convolution with sigmoid activation. The discriminator architecture includes 2D convolutional layers, LeakyReLU activation, batch normalization, a dense layer with 1024 units, and a final dense layer with sigmoid activation to determine the authenticity of generated images compared to real ones. Specific architecture details, including layer types, parameters, and activation functions, are summarized in Table VI.

D. Results and Discussion

The results of our experiments, presented in Table VII, offer a comprehensive comparison of three distinct methods for Arabic character generation and recognition: VAE, GAN, and our proposed hybrid VAE-GAN model. The performance of each method is evaluated using key metrics such as accuracy, precision, recall, and F1-Score. Our approach, utilizing the VAE-GAN model, demonstrated superior results across all metrics. The model achieved an outstanding accuracy of 97%, highlighting its ability to generate and recognize Arabic characters with remarkable precision. This substantial improvement is evident compared to the individual performances of VAE (89.15%) and GAN (87.11%). Furthermore, the precision,

recall, and F1-Score metrics further validate the effectiveness of the VAE-GAN approach, surpassing both VAE and GAN in terms of recognition accuracy and overall performance.

TABLE VI. GAN-VAE MODEL ARCHITECTURE AND HYPERPARAMETER SETTINGS

	Layer	Parameters
Encoder	Conv2D	32 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Conv2D	64 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Conv2D	128 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Flatten	-
	Dense	64 units (mean)
	Dense	64 units (log-variance)
Decoder	Dense	4 × 4 × 128 units
	Reshape	(4, 4, 128)
	Conv2DTranspose	128 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Conv2DTranspose	64 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Conv2DTranspose	32 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Conv2D	1 filter, kernel 5×5, padding = 'same', activation = 'sigmoid'
Discriminator	Conv2D	32 filters, kernel 5×5, stride = 2, padding = 'same'
	LeakyReLU	alpha = 0.2
	Conv2D	64 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Conv2D	128 filters, kernel 5×5, stride = 2, padding = 'same'
	BatchNormalization	-
	LeakyReLU	alpha = 0.2
	Flatten	-
	Dense	1,024 units
	BatchNormalization	-
	Dense	1 units, activation = 'sigmoid'

TABLE VII. COMPARISON OF VAE, GAN, AND VAE-GAN METHODS

Model	Accuracy	Precision	Recall	F1-Score
VAE	89.15%	89.91%	89%	89.01%
GAN	87.11%	88.84%	88.95%	88.15%
VAE-GAN	97%	97%	97%	97%

The success of the hybrid VAE-GAN model can be attributed to its unique integration of VAE's latent representation learning and GAN's generative capabilities. The iterative training process described in Algorithm 1 was crucial in refining the model's key components: the encoder, decoder, and discriminator. Particularly, the 97% accuracy achieved by the VAE-GAN model highlights its ability to minimize false positives, indicating its proficiency in generating characters closely resembling authentic Arabic characters from the dataset. A remarkable recall rate of 97% underscores the model's effectiveness in capturing a significant portion of real positive instances, further reinforcing its practicality. With an F1-Score of 97%, reconciling precision and recall, the balanced performance of the VAE-GAN model is accentuated. Overall, these results suggest that the hybrid approach not only enhances realism in character generation but also refines recognition accuracy, highlighting its potential for various applications in handwritten Arabic character recognition. Carefully chosen hyperparameters reinforced the model's stability and convergence throughout the 30 training epochs.

The proposed VAE-GAN model represents a promising solution for Arabic character generation and recognition. Its superior performance compared to traditional VAE and GAN approaches demonstrates the effectiveness of the hybrid architecture in capturing both the latent representation and generative capabilities required for accurate character recognition. The model's ability to minimize false positives and capture a significant portion of real positive instances highlights its robustness and generalizability. Furthermore, the carefully chosen hyperparameters and the iterative training process contribute to the model's stability and convergence.

Fig. 5 illustrates the evolution of accuracy and loss over 30 training epochs. Accuracy steadily improves, while loss decreases, reaching remarkable values that attest to effective learning.

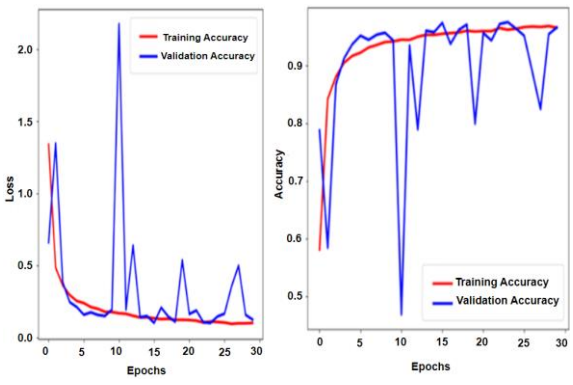

Fig. 5. Accuracy and loss plot of VAE-GAN training.

E. Comparative Analysis to Other Approaches

Comparing our hybrid GAN-VAE model with other methods for recognizing handwritten Arabic characters provides significant insights into its effectiveness and superiority. Recognizing Arabic characters presents challenges due to their complex shapes and considerable variability. Current techniques can be broadly categorized

into conventional methods and those based on deep learning. In our evaluation, we focused on the latter, utilizing the Hijja database as a reliable reference.

Table VIII offers a concise overview of the performance of different models, along with their respective limitations. Alkhateeb *et al.* [19] proposed a deep learning-based system for recognizing handwritten Arabic letters using CNN and three separate datasets, Arabic Handwriting Character Recognition (AHCR), Arabic Handwriting Character Dataset (AHCD), and Hijja, to validate the proposed system. According to their experimental results, the suggested approach achieved accuracies of 89.8%, 95.4%, and 92.5% on the AHCR, AHCD, and Hijja datasets, respectively. Altwaijry *et al.* [18] achieved an accuracy of 88% using CNN on a specific dataset. However, their method faces challenges in capturing the variability of Arabic characters. Alrobah *et al.* [21] combined CNN with SVM, eXtreme Gradient Boosting (XGBoost), achieving an impressive accuracy of 96.3% on another dataset the same year. Despite this, the use of multiple classifiers can make their approach complex to implement and interpret. Wagga *et al.* [22] integrated Data Augmentation (DA) with CNN, achieving an accuracy of 91.24% on a larger dataset. However, the effectiveness of data augmentation may depend on the quality and diversity of available data, which can limit the generalization of their model. Meanwhile, Khudeyer *et al.* [25] utilized ResNet50, recording an accuracy of 91.64% on that dataset. Although ResNet50 is known for its depth and ability to extract complex features, its use may be limited by computational and memory requirements. Additionally, Durayhim *et al.* [26] implemented VGG-16 and achieved a commendable accuracy of 94%. However, models based on older architectures like VGG-16 may lack the capacity to capture the richness of features in handwritten Arabic data.

Significantly, our hybrid GAN-VAE model surpassed its predecessors, registering an accuracy of 97% on the Hijja database. This achievement not only solidifies our model's position but also highlights its superiority over other advanced deep learning methodologies tailored for the intricate Hijja database. In essence, this comparative analysis underscores the power and promise of our GAN-VAE approach in the realm of recognizing handwritten Arabic characters.

TABLE VIII. COMPARISON BETWEEN OUR APPROACH AND THE VARIOUS METHODS FOR THE HIJJA DATABASE

References	Year	Feature Extractor	Classifier	Size	Accuracy
Alkhateeb *et al.* [19]	2020	CNN	SoftMax	47 434	92.5%
Altwaijry *et al.* [18]	2021	CNN	SoftMax	47 434	88 %
Alrobah *et al.* [21]	2021	CNN	SoftMax, SVM, XGBoost	47 434	89%, 96.3%, 95.7%
Wagaa *et al.* [22]	2022	CNN + DA	SoftMax	47 434	91.24%
Alheraki *et al.* [23]	2023	CNN	SoftMax	47 434	91,24%
Alwagdani [24]	2023	CNN	SoftMax, SVM, KNN, RF	47 434	91.78%, 91.95%, 91.50%, 91.87%
Khudeyer *et al.* [25]	2023	ResNet50	SoftMax, SVM	47 434	92,37%, 91,64 %
Durayhim [26]	2023	VGG-16	SoftMax	47 434	94%
Our Approach		**GAN-VAE**		**47 434**	**97 %**

V. CONCLUSION

In the field of Arabic handwritten character recognition, deep learning models have witnessed remarkable progress in recent years, particularly generative models based on adversarial learning. Our approach presents an innovative VAE-GAN hybrid model that overcomes the limitations of existing approaches for synthetic data generation and character recognition. The VAE-GAN hybrid model combines the advantages of Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) to capture complex latent representations of Arabic handwritten characters and generate new realistic characters. This composite architecture enables more accurate and reliable character recognition, as confirmed by rigorous empirical evaluations conducted on the Hijja dataset, renowned for its complexity. The obtained results reveal a significant improvement in precision metrics compared to state-of-the-art methods. This advancement demonstrates the effectiveness and innovation of the VAE-GAN hybrid model, positioning it as an innovative solution in the field of Arabic handwritten character recognition.

However, despite its performance, the VAE-GAN hybrid model has limitations. Its performance is highly dependent on the quality and quantity of training data, and it is sensitive to the initial learning conditions. Hyperparameter selection is also complex. Additionally, the model still needs to demonstrate its generalizability in diverse real-world settings. By overcoming these limitations through future research, the VAE-GAN model could reach its full potential and become a valuable tool for Arabic handwritten character recognition in various practical applications.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Hicham Lamtougui led the research by defining the research framework, designing the research methodology, conducting analysis, modeling, and writing the article. Hicham El Moubtahij contributed to the research by reformulating and revising the research document to make it suitable for publication. Hassan Fouadi provided expertise and contributed to the development of the methodology, particularly in assessing forecast performance measures and statistical analysis. Khalid Satori provided support and offered necessary comments and ideas for the entire research document. All authors approved the final version.

REFERENCES

[1] F. Javed, "Arabic and English phonetics: A comparative study," *The Criterion: An International Journal in English*, vol. 4, no. 4, pp. 1–13, 2013.

[2] M. T. Parvez and S. A. Mahmoud, "Offline Arabic handwritten text recognition: A survey," *ACM Computing Surveys (CSUR)*, vol. 45, no. 2, pp. 1–35, 2013.

[3] A. I. El-Desouky, M. M. Salem, A. O. Abd El-Gwad, and H. Arafat, "A handwritten Arabic character recognition technique for machine reader," in *Proc. Third International Conference on Software Engineering for Real Time Systems*, 1991, pp. 212–216.

[4] Y. Elarian, I. Ahmad, S. Awaida, W. Al-Khatib, and A. Zidouri, "Arabic ligatures: Analysis and application in text recognition," in *Proc. 2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, 2015, pp. 896–900.

[5] H. M. Balaha, H. A. Ali, and M. Badawy, "Automatic recognition of handwritten Arabic characters: A comprehensive review," *Neural Comput Appl*, vol. 33, pp. 3011–3034, 2021.

[6] J. H. AlKhateeb, J. Ren, J. Jiang, and H. Al-Muhtaseb, "Offline handwritten Arabic cursive text recognition using Hidden Markov Models and re-ranking," *Pattern Recognit. Lett.*, vol. 32, no. 8, pp. 1081–1088, 2011.

[7] C. Saunders, M. O. Stitson, J. Weston, L. Bottou, and A. Smola. (1998). Support vector machine-reference manual. [Online]. Available: https://eprints.soton.ac.uk/258959/1/SVM_Reference.pdf

[8] A. S. A. Huque, M. Haque, H. A. Khan, A. Al Helal, and K. I. Ahmed, "Comparative study of KNN, SVM and SR classifiers in recognizing Arabic handwritten characters employing feature fusion," *Signal and Image Processing Letters*, vol. 1, no. 2, pp. 41–49, 2019.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[10] J. Wu. (2017). Introduction to convolutional neural networks. *National Key Lab for Novel Software Technology. Nanjing University. China*. [Online]. 5(23), 495. Available: https://cs.nju.edu.cn/wujx/paper/CNN.pdf

[11] G. E. Hinton, "Deep belief networks," *Scholarpedia*, vol. 4, no. 5, 5947, 2009.

[12] D. Bank, N. Koenigstein, and R. Giryes, *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*, 2023, pp. 353–374.

[13] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza *et al.*, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[15] M. Shams, A. A. Elsonbaty, and W. Z. Elsawy, "Arabic handwritten character recognition based on convolution neural networks and support vector machine," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 11, issue 8, 2020.

[16] I. Goodfellow J. Pouget-Abadie, M. Mirza *et al.*, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv preprint, arXiv:1312.6114, 2013.

[18] N. Altwaijry and I. Al-Turaiki, "Arabic handwriting recognition system using convolutional neural network," *Neural Comput. Appl.*, vol. 33, no. 7, pp. 2249–2261, 2021.

[19] J. H. Alkhateeb, "An effective deep learning approach for improving off-line Arabic handwritten character recognition," *International Journal of Software Engineering and Computer Systems*, vol. 6, no. 2, pp. 53–61, 2020.

[20] 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.

[21] N. Alrobah and S. Albahli, "A hybrid deep model for recognizing arabic handwritten characters," *IEEE Access*, vol. 9, pp. 87058–87069, 2021.

[22] N. Wagaa, H. Kallel, and N. Mellouli, "Improved Arabic alphabet characters classification using Convolutional Neural Networks (CNN)," *Comput. Intell. Neurosci.*, vol. 2022, 2022.

[23] M. Alheraki, R. Al-Matham, and H. Al-Khalifa, "Handwritten arabic character recognition for children writing using convolutional neural network and stroke identification," *Human-Centric Intelligent Systems*, vol. 3, no. 2, pp. 147–159, 2023.

[24] M. S. Alwagdani and E. S. Jaha, "Deep learning-based child handwritten Arabic character recognition and handwriting discrimination," *Sensors*, vol. 23, no. 15, 6774, 2023.

[25] R. S. Khudeyer and N. M. Almoosawi, "Combination of machine learning algorithms and ResNet50 for Arabic handwritten classification," *Informatica*, vol. 46, no. 9, 2023.

[26] A. Bin Durayhim, A. Al-Ajlan, I. Al-Turaiki, and N. Altwaijry, "Towards accurate children's Arabic handwriting recognition via deep learning," *Applied Sciences*, vol. 13, no. 3, 1692, 2023.

[27] S. Bunrit, N. Kerdprasop, and K. Kerdprasop, "Improving the representation of CNN based features by autoencoder for a task of construction material image classification," *Journal of Advances in Information Technology*, vol. 11, no. 4, 2020.

[28] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *CNeural Information Processing: 24th International Conference, ICONIP 2017*, Guangzhou, China, 2017, pp. 373–382. doi: 10.1007/978-3-319-70096-0_39

[29] A. Ng, "Sparse autoencoder," *CS294A Lecture Notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[30] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and Composing Robust Features with Denoising Autoencoders." in *Proc. the 25th International Conference on Machine Learning*, 2008, pp. 1096–1103.

[31] D. P. Kingma and M. Welling, "An Introduction to Variational Autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, Jun. 2019. doi: 10.1561/2200000056

[32] H. Lamtougui, H. El Moubtahij, H. Fouadi, and K. Satori, "Boosting handwritten Arabic text recognition using deep autoencoders and data augmentation techniques," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 11, no. 4, pp. 800–809, 2023.