International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Boosting Handwritten Arabic Text Recognition using Deep Autoencoders and Data Augmentation Techniques

Hicham Lamtougui*[1], Hicham El Moubtahij[2], Hassan Fouadi[3], and Khalid Satori[4]

Submitted: 05/05/2023 **Revised**: 12/07/2023 **Accepted**: 08/08/2023

Abstract: The recognition of handwritten characters and numbers is a complex challenge in the field of pattern recognition, especially for the Arabic language. While significant progress has been made for the automatic recognition of Latin handwritten characters, methods and approaches for the Arabic language remain insufficient. Deep learning technologies, in particular auto-encoders (AE), offer new perspectives for handwriting recognition. In this article, we introduce the different types of most popular AE, such as Convolutional AE (CAE), Sparse AE (SAE), Denoising AE (DAE), and Variational AE (VAE), and evaluate their performance on two reference databases: the Modified Arabic Digits dataBase (MADBase) and Arabic Handwritten Character Dataset (AHCD). Using data augmentation to improve results, the VAE algorithm showed higher accuracy than other Deep Learning algorithms on both databases, with very encouraging results of 98.77% for MADBase and 98.42% for AHCD.

Keywords: *handwritten characters, Deep Learning, data augmentation, Variational AE*

1. Introduction

Pattern recognition has become significantly important due to the growing demand for artificial intelligence in practical applications[1].

Character recognition is a subfield of pattern recognition that allows you to automatically recognize and understand characters in documents, images or videos. It is divided into online and offline recognition.

Offline recognition processes scanned images of documents, while online recognition recognizes characters as they are written or drawn on a tablet or touchscreen [2].

Recognition of Arabic handwritten characters are particularly difficult due to the variability in shape, slant, and cursive nature of Arabic writing [3]. This complexity has prompted the field of artificial intelligence to evolve in extraordinary ways in recent years, with the emergence of machine learning as a powerful force in various industries, including image recognition, speech recognition and autonomous driving. The key to machine learning is its ability to autonomously extract essential features from data through a training process. There are generally three main approaches to machine learning: supervised, unsupervised, and reinforcement learning.

Supervised learning, particularly in the field of deep learning, requires that training data be annotated, a crucial requirement for techniques such as convolutional neural networks (CNN) [4] recurrent neural networks (RNN) [5]and deep neural networks. This method involves the use of optimization algorithms, such as backpropagation, aimed at minimizing the discrepancy between the model predictions and the actual labels. However, manually annotating large amounts of data can be a time-consuming and tedious task.

Unsupervised learning, on the other hand, relies on unlabeled training data. Common methods include restricted Boltzmann machines and autoencoders.

Convolutional neural networks are widely used in practical applications due to their efficiency. Autoencoders, on the other hand, aim to acquire representations of data and generate models of that data. They consist of an encoder, which compresses the data by extracting relevant features, and a decoder, which restores the compressed data to its original form. Autoencoders are particularly useful for feature selection and extraction.

However, autoencoders have some disadvantages, including prolonged training times for deep models, difficulty in interpreting the extracted features, and sometimes unsatisfactory accuracy. To address these limitations, research has been conducted into improvements to autoencoders [6].

Autoencoders are commonly employed in handwritten character recognition due to their ability to extract meaningful features from input data and use them for

[1] *Computer Science Department, Faculty of Sciences Dhar EL Mahraz, Sidi Mohamed Ben Abdellah University, Fez, Morocco*
ORCID ID: 0000-0002-7655-9383
[2] *Higher School of Technology, University of Ibn Zohr, Agadir, Morocco*
ORCID ID: 0000-0003-3486-4950
[3] *Computer Science Department, Faculty of Sciences Dhar EL Mahraz, Sidi Mohamed Ben Abdellah University, Fez, Morocco*
ORCID ID: 0009-0009-5273-5352
[4] *Computer Science Department, Faculty of Sciences Dhar EL Mahraz, Sidi Mohamed Ben Abdellah University, Fez, Morocco*
ORCID ID: 0000-0001-6055-4169
** Corresponding Author Email: hicham.lamtougui@usmba.ac.ma*

classification.

In our study, we implemented and compared different types of autoencoders using two benchmark databases: the Modified Arabic Numerals Database (MADBase) and the Arabic Handwritten Characters Dataset (AHCD), in with particular emphasis on CAE which uses a ConvNet encoder to produce a low-dimensional representation in the latent space, followed by a ConvNet decoder to reconstruct the original image [7]. DAE is designed to remove noise from input data [8], while SAE uses a sparse representation of data [9]. Another model, VAE, uses a Gaussian distribution to compress data and generate an image [10]. We compared the performance of different types of autoencoders with and without data augmentation, which is a technique used to improve model performance by creating new data from existing data. The objective is to reduce overfitting on training data.

The rest of the paper is organized as follows: Section 2 presents an overview of recent similar work carried out in the field. The different types of autoencoders used in our study are presented in Section 3. Section 4 describes in detail the databases used for this study, as well as the methodology adopted to conduct the experiments and present the results obtained. Finally, conclusions and future work are presented in Section 5.

2. Related Work

In the field of handwritten character recognition, various methods have been developed for Latin, Chinese and Japanese languages. On the other hand, the recognition of Arabic characters has long faced difficulties due to the complexity of handwritten forms [11].

In order to solve this challenge, various databases have been created, among which MADBase [12] and AHCD [4] are references in the recognition of Arabic numerals and characters. To overcome this complexity, several methods and approaches have been proposed for the recognition of these characters.

2.1. Works Based on the MADBase Database

The MNIST (Modified National Institute of Standards and Technology) database is widely used as a reference in the field of Latin handwritten digit recognition [13]. Similarly, MADBase is a reference database for recognizing Arabic numerals, which is similar to the MNIST database. Several methods and approaches have been proposed to develop reliable Arabic numeral recognition systems.

Alkhateeb et al (2014) [14] proposed a multi-class digit classification system that uses a three-step Dynamic Bayesian Network (DBN). The first step is to preprocess the image to obtain a normalized image that eliminates variations in the images. Then the features are extracted using the DCT coefficients. Then the features are extracted using the DCT coefficients. Finally, the DBN is used to classify an unknown cipher by determining the class to which it belongs. However, one of the main disadvantages of this system is the training time required for some applications. The model has a performance accuracy of 85.26%.

El-Sawy et al [15] proposed the LeNet-5 architecture, which is a CNN used to solve image recognition tasks. This architecture consists of 7 layers, including 3 convolution layers, 2 downsampling layers and 2 fully conected layers. Although this method managed to achieve 88% accuracy on training data, it suffers from an overfitting problem and does not generalize well to new data.

Mudhsh and Almodfer [16] proposed a VGG architecture for Arabic character. The model includes 13 convolutional layers, 2 max-pooling and 3 fully connected layers. They compared the performance of their model with other approaches such as DBN and RNN, and showed that their VGG model was able to achieve a recognition rate of 97.32 %, which is a significant improvement over other approach.

Techniques based on stacked autoencoders have been proposed recently by Loey et al. [17] to manage large entries. Although the model achieved a classification accuracy of 98.5%, it suffers from a sparse dimensionality issue that affects the prediction of dense data. Additionally, using only two layers of stacked autoencoders might not guarantee accurate formation.

In 2020, Alkhawaldeh [18] developed a hybrid deep transfer model using two pre-trained convolutional neural networks, to which LSTM layers were added. This model allows both to learn the relevant features from the CNN model, and to extract the long-term dependency features thanks to the LSTM layers. Experimental results showed high performance, with accuracy reaching 97.4%.

2.2. Works Based on the AHCD Database

Several deep learning methods and approaches are available for handwritten Arabic character recognition from the Arabic Handwritten Character Dataset (AHCD).

El-Sawy et al. [4] demonstrated the efficiency of CNN compared to other approaches and methods for the classification of handwritten Arabic characters. Their CNN model featured two layers of convolution along with normalization and dropout techniques, resulting in an accuracy of 94.9%.

Similarly, Younis [19] constructed a deep CNN with regularization and normalization parameters to prevent overfitting. Its approach consists of three convolutional layers followed by a fully connected layer. The system achieved a recognition rate of 94.7%.

According to Boufenar et al. [20] reported promising

experimental results using the popular CNN architecture Alexnet, achieving an accuracy of 96.78%. However, they also highlighted the limitations of this approach, which include the requirement for deeper pre-trained models and the difficulties in collecting large amounts of data to effectively train a DCNN from scratch, in order to enhance the model's ability to generalize to new data.

Alyahya et al. [21] sought to improve Arabic letter recognition performance by exploring modifications of the ResNet-18 architecture. They added a dropout layer after each convolutional layer to avoid overfitting. They built two models, one with a single fully connected layer and the other with two fully connected layers. The latter yielded superior results, achieving a recognition rate of 98.30%.

According to Shams et al. [22] suggested a method combining deep convolutional neural networks (DCNN) and support vector machines (SVM) to improve handwritten Arabic character recognition. To handle similar Arabic characters, a clustering algorithm based on the K-means approach is also used. This approach resulted in a recognition accuracy of 95.07% for the system.

Altwaijry et al. [23] also proposed a handwriting recognition model based on CNN, which included three convolutional layers, followed by batch normalization and ReLu activation. The authors evaluated the performance of the model using two optimizers, Stochastic Gradient Descent and Adam, and achieved an average accuracy of 97%.

3. Autoencoders AND Vae

3.1. The architecture of Autoencoders

An autoencoder is a type of neural network architecture that learns a compressed representation of input data through an encoder-decoder structure [24]. This approach is typically used for unsupervised learning and reducing data dimensionality [25]. The encoder in an autoencoder uses an encoding function g(x) parameterized by φ to reduce the input's dimensionality to a latent space (Bottleneck), while the decoder reconstructs the original input from the latent space. The encoder's output for input x is represented by h = gφ(x). The output of the decoder for the latent space h is represented by x'= fϴ(gφ(x)), where f is the decoder function and ϴ the decoder parameters, as depicted in Figure 1. The autoencoder model is designed to learn its parameters in order to minimize the discrepancy between the original input and the output produced by the model. The loss of the autoencoder is expressed in equation (1) :

$$L(x, x') = \|x - x'\|^2 = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)} - f_\theta(g_\varphi(x^{(i)})))^2 \quad (1)$$

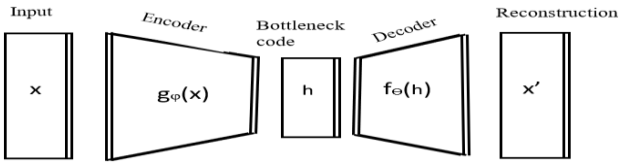

Fig. 1. Illustration of the autoencoder model architecture.

Various types of autoencoders have been developed to learn image representations that are appropriate for classification tasks, such as convolutional autoencoders (CAE), sparse autoencoders (SAE), denoising autoencoders (DAE), and variational autoencoders (VAE). Each of these autoencoder types has an architecture that is tailored and effective for the task at hand.

3.1.1. Convolutional Autoencoder

CAEs are a commonly used technique to extract features from images prior to classification. Their structure is similar to that of classical autoencoders, but they use convolutional layers rather than fully connected layers to minimize errors between input and output [26].

CNNs are used in the encoding and decoding layers to extract features from the input data using convolution filters. Pooling layers are used to eliminate feature noise and to reduce the size of the input data. Activation functions are also applied in convolutional layers [27].

CAE are a type of neural network that can learn to extract features from unlabeled images [28]. In MADBase database, a CAE is used to extract features from the input image and transform them into a 10-dimensional space. The model is then trained in an unsupervised way to reconstruct the original image from the extracted features. Figure 2 illustrates this process.

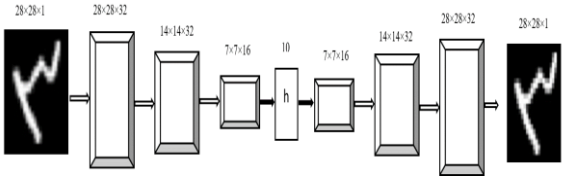

Fig. 2. The structure of the CAE proposed for the MADBase database

3.1.2. Sparse Autoencoder

SAEs, which are a well-studied class of autoencoders, can have a number of neurons in their hidden layer equal to or greater than the number of nodes in their input layer [29]. To limit the activation of nodes in the hidden layer of SAEs, a regularization is applied, as illustrated in Figure 3. Nonlinearity is introduced using a sigmoid activation function. where an active node a_j^h will output a value close to 1, while an inactive node will output a value close to 0 for a given input. The average activation of the i-th neuron in

the hidden layer is also taken into account. The average activation of the j-th neuron of the hidden layer $\hat{\rho}_j$

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m} a_j^h(x_i) \qquad (2)$$

x_i : i-th input vector.

a_j^h : the activation of j-th node in the hidden layer h for x_i.

m: number of input vectors.

We impose a constraint so that the average activation is equal:

$$\hat{\rho}_j = \rho$$

We introduce a regularization term which is the divergence KL between two distributions, one defined by $\hat{\rho}_j$ and the other by ρ:

Regularizer:

$$\sum_{j=1}^{N_h}\left[\rho \log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}\right] = \sum_{j=1}^{N_h} KL(\rho \,\|\hat{\rho}_j) \quad (3)$$

The Kullback-Leibler KL divergence measures the distance between two different distributions.

The complete loss function for SAE defined by:

$$J_{sparse}(W) = \mathrm{L}(X, \hat{X}) + \lambda \sum_{j=1}^{N_h} KL(\rho \,\|\hat{\rho}_j) \qquad (4)$$

W : weight matrix

$\mathrm{L}(X, \hat{X})$: data loss component.

X: input vector.

$\hat{X}$: output vector.

λ : the relative weight between the data loss component and the regularization loss component.

$\sum_{j=1}^{N_h} KL(\rho \,\|\hat{\rho}_j)$: Regularization loss

L1 regularization and KL divergence are used to learn useful features from the input layer.

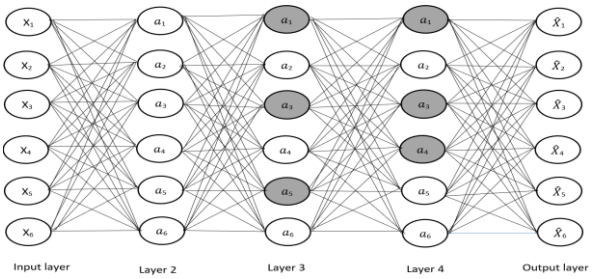

Fig. 3. Illustration of SAE model architecture

3.1.3. Denoising Autoencoder

DAEs are based on auto-encoders (AEs), but they introduce a noise stage to alter the input data before encoding it into a hidden representation. The objective of the DAE is to reconstruct the original data from the noisy data [30]. This means that instead of just memorizing the input data, DAEs learn to extract features from the noisy data $\tilde{x}$ and use them to retrieve the original data x.

$$\tilde{\mathrm{x}} = \mathrm{x} + \mathrm{Noisy} \qquad (5)$$

DAE is a model that extracts features from noisy data and retrieves input data to solve the denoising problem. It aims to minimize the reconstruction error between the unaltered input data and the decoder output using an optimization algorithm such as stochastic gradient descent. This approach involves corrupting the initial input x into an altered version x' using stochastic mapping, a function that assigns a probability distribution qD to indicate that x' came from a distribution conditioned on the value of x, so as to ensure that the reconstructed output is closer to x than to x' (Figure.4).

$$x' \sim \mathrm{q_D}(x' | x) \qquad (6)$$

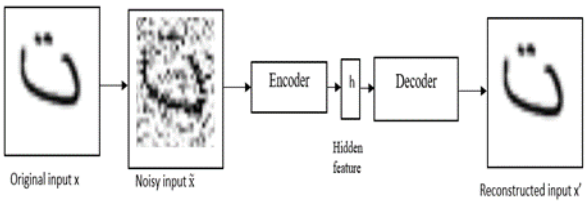

Fig. 4. The structure of the DAE proposed for the AHCD database

3.2. Variational Autoencoder

VAE models are used to generate data by representing a distribution of data as Gaussian latent variables [31]. Their main purpose is to encode each element of the data distribution into a latent vector and decode it to generate an instance of the original distribution [32]. Unlike other types of autoencoders such as SAEs and DAEs, VAEs use the Kullback-Leibler D_{KL} divergence to regulate the compact data representation, which allows VAEs to produce higher quality reconstructions and generate data in a more consistent and controlled manner.

VAEs have a different mathematical basis than other types of autoencoders, seeking to maximize the likelihood of data using a probabilistic approach and to represent data efficiently using fewer dimensions.

$$z = \mu + \sigma.\varepsilon \qquad (\varepsilon \sim N(0,1)) \quad (7)$$

The VAE model consists of an array of encoders that transforms the input data x into a latent representation $z \sim q(z|x)$ and a decoder network to decode this vector into a data reconstruction of input: $z \sim p(x|z)$.

When training a VAE model, the reconstruction loss $\mathcal{L}_{rec}$ reconstruction loss measures the amount of information lost when transforming an instance of the data distribution into a latent vector and back into an instance of the data distribution, while the divergence loss measures the

difference between the distribution $q(z|x)$ and the reference distribution $p(z)$ [10]. By optimizing the loss function combining these two terms, the model learns to encode data efficiently while maintaining a latent space distribution that is close to the reference distribution as shown in Figure 5.

$$\mathcal{L}_{vae} = \mathcal{L}_{rec} + \mathcal{L}_{kl} \quad (8)$$

with $\mathcal{L}_{kl} = D_{kl}(q(z|x) \| p(z))$

and $\mathcal{L}_{rec} = -E_{q(z|x)}[\log p(x|z)]$

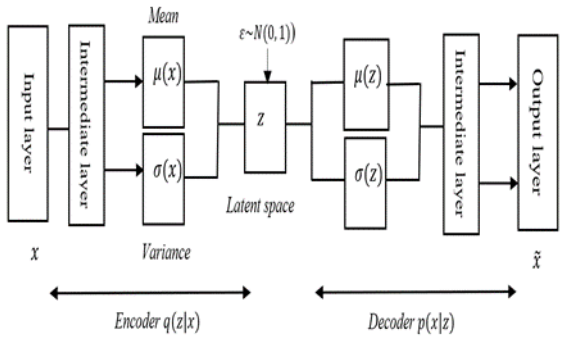

Fig. 5. The schematic structure of VAE

4. Experimental Results and Discussion

4.1. Dataset

There are several offline recognition studies in the literature that used different Arabic text bases. In this section, we will present MADBase and AHCD, two benchmark databases, to compare the performance of Arabic character recognition systems.

4.1.1. MADBase

MADBase (Modified Arabic Digits dataBase), an offline Arabic numeral recognition database, is an adaptation of ADBase (Arabic Digits dataBase). It follows the format of the famous MNIST database, which contains handwritten digits ranging from 0 to 9. MADBase and ADBase contain the same Arabic numerals, but the images have different sizes [12]. Both databases consist of 70,000 Arabic numerals written by 700 people, with each person writing the Arabic numerals 0 to 9 ten times each as shown in Table 1. Image numbers are normalized and centered on a 28x28 pixel still image. The database is partitioned into two sets: a training set comprising 6,000 digital images per label, totaling 60,000 digits, and a test set comprising 1,000 digital images and 10,000 digits. Both databases, ADBase and MADBase, are available online for free at (https://datacenter.aucegypt.edu/shazeem/).

Table 1. Arabic digits in printed and handwritten

Latin Numbers	0	1	2	3	4	5	6	7	8	9
Handwritten Arabic Numbers	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Printed Numbers	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Numbers										

4.1.2. AHCD

Another dataset called Arabic Handwritten Characters Dataset (AHCD) was created to process Arabic handwritten characters [4]. This database contains 16,800 letters written by 60 participants, each of whom wrote every character from 'alif' to 'Yae' ten times. The images were scanned at a resolution of 300 dpi and the data set is divided into two parts: a training set comprising 13,440 characters spread over 480 images per class and a test set comprising 3,360 characters spread over 120 pictures per class. This database is freely available at (https://www.kaggle.com/mloey1/ahcd1). Creating this database presents additional challenges due to variability in writing style, thickness, number and position of dots. To improve the performance of character recognition models, it is recommended to train them with a large number of sample images (Table.2).

4.2. Data Augmentation

In this study, data augmentation techniques were used to improve the performance of MADBase and AHCD classifiers and to avoid overfitting. Data augmentation is the process of adding additional information to an existing dataset to make it more efficient and accurate for machine learning models [33].

For this, the study used ImageDataGenerator from Keras, a pre-existing tool that uses common techniques to augment image data [34].

When training a model, several techniques can be used to increase the size of datasets and improve its generalizability. These techniques include feature normalization, rotations, ZCA whitening, horizontal and vertical flips, zoom and shift range, and shear intensity. For each transformation, specific parameters are defined. The rotation parameter is set to 90 degrees and the ZCA whitening algorithm is used to even out the brightness and variance of the images. The horizontal and vertical flip settings are enabled to invert images, while the shear setting is set to 0.2 to allow random image distortion up to 20 degrees. Similarly, the zoom parameter is set to 0.1 to allow random enlargement or shrinkage of images up to 20% as shown in Figure 6.

Table 2. Arabic Characters in Printed and Handwritten Forms

Printed Arabic Character	ا	ب	ت	ث	ج	ح	خ	د	ذ	ر	ز	س	ص
Name	Alif	Ba	Ta	Tha	Jim	Ha	Kha	Dal	Dhal	Ra	Zay	Sin	Ṣad
Handwritten Arabic Character	أ	ب	ت	ث	ج	ح	خ	د	ذ	ر	ز	س	ص
Printed Arabic Character	ض	ط	ع	غ	ف	ق	ك	ل	م	ن	ه	و	ي
Name	Dad	Ta	Ayn	Ghayn	Fae	Qaf	Kaf	Lam	Mim	Nun	Ha	Waw	Yae
Handwritten Arabic Character	ض	ط	ع	غ	ف	ق	ك	ل	م	ن	ه	و	ي

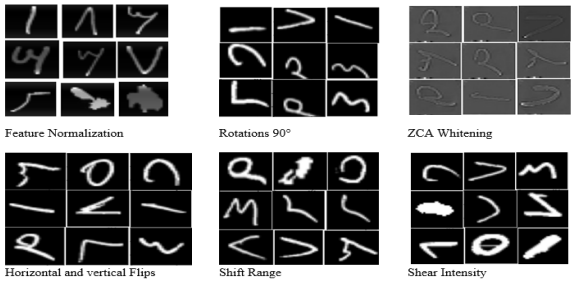

Fig. 6. Examples of Data Augmentation Techniques Used for MADBase

4.3. Implementation Details

In this study, we present a method to train character recognition models using different types of autoencoders on MADBase and AHCD datasets, with and without data augmentation. To ensure that the comparison was fair, we utilized an identical architecture design for each of the various types of autoencoders that were used in our approach.

To implement the model, we use the Keras library, which includes convolution layers, batch normalization, and LeakyReLU activation functions for the encoder, as well as convolution and upsampling layers for the decoder. The reconstruction loss is used to measure the difference between the original input and the predicted output, and the model is trained on the training dataset using the training dataset as the expected output.

The parameters of the autoencoder model include the dimension of the input, which is 28x28 for MADBase and 32x32 for AHCD, as well as 10 convolution layers used to compress and decompress the input. Each convolution layer uses 512 filters and "same" padding is used for each layer. The loss function used is the root mean (MSE). The Adam optimizer is used to update the model weights during training, with a learning rate set at 0.001 and a batch size of 256. The step between the different positions of the convolution kernels is (2 ,2). In other words, to avoid the gradient vanishing problem, the LeakyReLU activation function is used in the hidden layers, while the sigmoid activation function is used in the output layer. Strides size 2x2 are used to extract features from the image (Table.3).

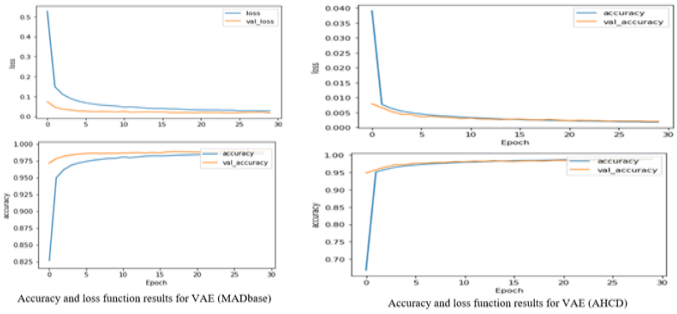

Fig. 7. Accuracy and loss function results using MADBase and AHCD dataset

Table 3. implementation details

Parameters	***Values***
Table 3. implementation details	
Input dimension	Shape MADBase = (28,28,1) Shape AHCD = (32,32,1)
Number of convolution layers	10
Number of epochs	30
Number of filters	512
Padding	Same
Loss function	mean_squared_error
Optimizer	Adam
Learning rate	0.001
Batch size	256
Strides size	(2,2)
Activation function of the last layer	Sigmoid
Activation function for hidden layers	LeakyReLU
Size of convolutional kernels	(2,2)

4.4. Experimental Results

We used four networks for classification, each trained

with two different approaches from MADBase and AHCD databases, with and without data augmentation. Adding data augmentation techniques improved the performance of the MADBase database, increasing accuracy by several percentage points for all AE types (Table 4). VAE performed best on this database, with 98.77% accuracy using data augmentation, while CAE also performed very well with 92.16% accuracy using data augmentation. data increase. Similar results were obtained for the AHCD database, with a marked improvement in accuracy when using data augmentation. The VAE was also the best performing algorithm on this database, with an accuracy of 98.42% using data augmentation (Table 5). Furthermore, using the augmented data, our approach managed to reduce training/validation loss faster, which is demonstrated in Figure 7.

Table 4. Experimental results of the compared algorithms on the MADBase dataset

Table 4. Experimental results of the compared algorithms on the MADBase dataset

Methods	***Without Data Augmentation***		***With Data Augmentation***	
	Accuracy %	***Loss***	***Accuracy %***	***Loss***
SAE	88.16	0.0191	88.7	0.004
DAE	90.8	0.0185	89.9	0.0016
CAE	91.13	1.0733e-05	92.16	1.02e-05
VAE	**98.33**	0.031	**98.77**	0.017

Table 5. Experimental results of the compared algorithms on the AHCD dataset

Methods	***Accuracy %***	***Loss***	***Accuracy %***	***Loss***
SAE	84.45	0.0185	84.92	0.016
DAE	88.14	0.0685	87.11	0.060
CAE	88.4	0.0031	89.3	0.0022
VAE	**98.11**	0.002	**98.42**	0.038

4.5. Comparison with State-of-the-Art

Recognizing Arabic handwritten characters is a significant challenge due to their complex and variable form. The methods used for recognition can be divided into two categories: artisanal methods and deep learning-based methods. We evaluated the performance of our deep learning-based model against existing methods using the MADBase and AHCD databases. The results showed that our VAE model with data augmentation was superior to other deep learning methods and achieved the best published results on both databases (Tables 6-7). Although Loey et al. [17] have recently obtained good results on the MADBase database using ensemble methods, such as SAE, our model is simple and generic, which makes it applicable to other languages such as Latin and Chinese. Alyahya [21] also demonstrated the efficiency of ResNet-18 with Dropout for Arabic character recognition. Our model's performance in different contexts indicates its efficacy in recognizing digits and handwritten characters.

Table 6. Comparison between the proposed approach and the other approach

Authors & publication year	***Techniques***	***Training Database***	***Testing Database***	***Classification Accuracy***
AlKhateeb (2014) [14]	DBN	60000	10000	85.26%
El-Sawy (2016) [15]	LeNet-5 (7 layer)	60000	10000	88%
Mudhsh (2017) [16]	VGGNet (13 layer)	60000	10000	97,32%
Loey (2017) [17]	SAE + Softmax	60000	10000	98.5%
Alkhawaldeh (2020) [18]	LeNet + LSTM	60000	10000	97.4%
Our approaches	**VAE with Data Augmentation**	**60000**	**10000**	**98.77%**

Table 7. Comparison between the proposed approach and the other approach for the AHCD database

Authors& publication year	*Approaches*	*Training Database*	*Testing Database*	*Classification Accuracy*
El-Sawy (2017) [4]	CNN	13440	3360	94.9%
Younis (2017) [19]	Deep CNN	13440	3360	94,7 %
Boufenar (2018) [20]	CNN (Alexnet)	13440	3360	96,78%
Alyahya (2020) [21]	ResNet-18 + Dropout	10080 (training) 3360 (validation)	3360	98,30%
Shams (2020) [22]	DCNN + SVM	13440	3360	95,07%
Altwaijry (2021) [23]	CNN (ResNet)	10080 (training) 3360 (validation)	3360	97%
Our Approach	**VAE with Data Augmentation**	**13440**	**3360**	**98, 42%**

5. Conclusions and Future Works

Arabic handwriting recognition is a complex task that has many applications in different fields. This scientific article presents a study on the automatic recognition of handwritten Arabic numerals and characters, using different types of autoencoders (AE). AEs are unsupervised learning algorithms based on artificial neural networks, which have grown in popularity in recent years due to their high performance in pattern recognition. The system was trained and tested on the two largest Arabic numeral and character datasets available, MADBase and AHCD, using the Keras library to implement the models. Our data augmentation approach is generic and can be applied to other datasets. The results showed that the VAE algorithm achieved the best performance on both databases, with an accuracy of 98.77% on MADBase and 98.42% on AHCD, using the augmentation data to improve model performance. These results are very encouraging and may have practical applications in Arabic handwriting recognition.

More data will need to be gathered in the future in order to evaluate our strategy more precisely. By utilizing VAE in combination with other data augmentation techniques, we might enhance our model. The use of Generative Adversarial Networks (GANs) to amplify the data would also be intriguing to investigate, and the outcomes would be fascinating to compare to those of our existing methodology.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] P.-Y. Yin, Pattern recognition, BoD–Books on Demand, 2009.

[2] C.L. Liu, F. Yin, D.H. Wang, Q.F. Wang, Online and offline handwritten Chinese character recognition: Benchmarking on new databases, Pattern Recognit. 46 (2013) 155–162.

[3] O.J. ONI, F.O. ASAHIAH, Computational modelling of an optical character recognition system for Yorùbá printed text images, Sci Afr. 9 (2020) e00415. https://doi.org/10.1016/j.sciaf.2020.e00415.

[4] Elsawy, M. Loey, H.M. El-Bakry, A. El-Sawy, H. El-Bakry, Arabic Handwritten Characters Recognition using Convolutional Neural Network Master researchers View project Strategic Business Analytics and Alternative View project Arabic Handwritten Characters Recognition using Convolutional Neural Network, n.d. https://www.researchgate.net/publication/313891953.

[5] Sherstinsky, Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network, Physica D. 404 (2020) 132306.

[6] Majumdar, R. Singh, M. Vatsa, Face Verification via Class Sparsity Based Supervised Encoding, IEEE Trans Pattern Anal Mach Intell. 39 (2017) 1273–1280.

[7] M. Chen, X. Shi, Y. Zhang, D. Wu, M. Guizani, Deep Feature Learning for Medical Image Analysis with Convolutional Autoencoder Neural Network, IEEE Trans Big Data. 7 (2017) 750–758. https://doi.org/10.1109/tbdata.2017.2717439.

[8] P. Vincent, A Connection Between Score Matching and Denoising Autoencoders, n.d.

[9] L. Zhang, Y. Lu, B. Wang, F. Li, Z. Zhang, Sparse Auto-encoder with Smoothed l1 Regularization, Neural Process Lett. 47 (2018) 829–839. https://doi.org/10.1007/s11063-017-9668-5.

[10] D.P. Kingma, M. Welling, An Introduction to Variational Autoencoders, (2019). https://doi.org/10.1561/2200000056.

[11] X.X. Niu, C.Y. Suen, A novel hybrid CNN-SVM classifier for recognizing handwritten digits, Pattern Recognit. 45 (2012) 1318–1325. https://doi.org/10.1016/j.patcog.2011.09.021.

[12] S. Abdleazeem, E. El-Sherif, Arabic handwritten digit recognition, International Journal on Document Analysis and Recognition. 11 (2008) 127–141

[13] Y. Lecun, E. Bottou, Y. Bengio, P. Haffner, Gradient-Based Learning Applied to Document Recognition, 1998.

[14] J.H. and A.M. AlKhateeb, DBN-Based learning for Arabic handwritten digit recognition using DCT features, in: 2014: pp. 222–226.

[15] H.E.-B.& M.L. Ahmed El-Sawy, CNN for Handwritten Arabic Digits Recognition Based on LeNet-5, in: *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics* 2016, 2016: pp. 566–575.

[16] M.A. Mudhsh, R. Almodfer, Arabic handwritten alphanumeric character recognition using very deep neural network, Information (Switzerland). 8 (2017). https://doi.org/10.3390/info8030105.

[17] M. Loey, A. El-Sawy, H. El-Bakry, Deep Learning Autoencoder Approach for Handwritten Arabic Digits Recognition, n.d. http://datacenter.aucegypt.edu/shazeem/.

[18] R.S. Alkhawaldeh, Arabic (Indian) digit handwritten recognition using recurrent transfer deep architecture, Soft Comput. 25 (2021) 3131–3141. https://doi.org/10.1007/s00500-020-05368-8.

[19] Younis, Khaled S. Arabic hand-written character recognition based on deep convolutional neural networks. Jordanian Journal of Computers and Information Technology, 2017, vol. 3, no 3.

[20] Boufenar, A. Kerboua, M. Batouche, Investigation on deep learning for off-line handwritten Arabic character recognition, Cogn Syst Res. 50 (2018) 180–195.

[21] H. Alyahya, M.M. Ben Ismail, A. Al-Salman, Deep ensemble neural networks for recognizing isolated Arabic handwritten characters, ACCENTS Transactions on Image Processing and Computer Vision. 6 (2020) 68–79.

[22] M. Shams, A.A. Elsonbaty, W.Z. Elsawy, Arabic Handwritten Character Recognition based on Convolution Neural Networks and Support Vector Machine, 2020. www.ijacsa.thesai.org.

[23] N. Altwaijry, I. Al-Turaiki, Arabic handwriting recognition system using convolutional neural network, Neural Comput Appl. 33 (2021) 2249–2261. https://doi.org/10.1007/s00521-020-05070-8.

[24] Al Bataineh, A. Mairaj, D. Kaur, Autoencoder based semi-supervised anomaly detection in turbofan engines, International Journal of Advanced Computer Science and Applications. 11 (2020).

[25] G.E. Hinton, R.R. Salakhutdinov, Reducing the Dimensionality of Data with Neural Networks, 2006.

[26] Azarang, H.E. Manoochehri, N. Kehtarnavaz, Convolutional Autoencoder-Based Multispectral Image Fusion, IEEE Access. 7 (2019) : pp/ 35673–35683.

[27] X. Guo, X. Liu, E. Zhu, J. Yin, Deep Clustering with Convolutional Autoencoders, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, 2017: pp. 373–382. https://doi.org/10.1007/978-3-319-70096-0_39.

[28] A.B. Shinde, J. Bagade, R. Bhimanpallewar, Y.H. Dandawate, Image Compression of Handwritten Devanagari Text Documents Using a Convolutional Autoencoder, International Journal of Intelligent Systems and Applications in Engineering. 11 (2023) 449–457.

[29] Ng, CS294A Lecture notes Sparse autoencoder, n.d.

[30] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and Composing Robust Features with Denoising Autoencoders, n.d.

[31] O.O. Karadag, O.E. Cicek, Empirical evaluation of the effectiveness of variational autoencoders on data

augmentation for the image classification problem, International Journal of Intelligent Systems and Applications in Engineering. 8 (2020) 116–120.

[32] D.P. Kingma, M. Welling, Auto-Encoding Variational Bayes, (2013). http://arxiv.org/abs/1312.6114.

[33] Shorten, T.M. Khoshgoftaar, A survey on Image Data Augmentation for Deep Learning, J Big Data. 6 (2019). https://doi.org/10.1186/s40537-019-0197-0.

[34] H. Lamtougui, H. El Moubtahij, H. Fouadi, K. Satori, An Efficient Hybrid Model for Arabic Text Recognition, Computers, Materials and Continua. 74 (2023) 2871–2888.

[35] Lavanya, A. ., & Priya, N. S. . (2023). Enriched Model of Case Based Reasoning and Neutrosophic Intelligent System for DDoS Attack Defence in Software Defined Network based Cloud. International Journal on Recent and Innovation Trends in Computing and Communication, 11(4s), 141–148. https://doi.org/10.17762/ijritcc.v11i4s.6320

[36] Mr. Anish Dhabliya. (2013). Ultra Wide Band Pulse Generation Using Advanced Design System Software . International Journal of New Practices in Management and Engineering, 2(02), 01 - 07. Retrieved from http://ijnpme.org/index.php/IJNPME/article/view/14