Intelligent Arabic Handwriting Recognition Using Different Standalone and Hybrid CNN Architectures

Waleed Albattah * and Saleh Albahli

Department of Information Technology, College of Computer, Qassim University, Buraydah 52571, Saudi Arabia
* Correspondence: w.albattah@qu.edu.sa

Abstract: Handwritten character recognition is a computer-vision-system problem that is still critical and challenging in many computer-vision tasks. With the increased interest in handwriting recognition as well as the developments in machine-learning and deep-learning algorithms, researchers have made significant improvements and advances in developing English-handwriting-recognition methodologies; however, Arabic handwriting recognition has not yet received enough interest. In this work, several deep-learning and hybrid models were created. The methodology of the current study took advantage of machine learning in classification and deep learning in feature extraction to create hybrid models. Among the standalone deep-learning models trained on the two datasets used in the experiments performed, the best results were obtained with the transfer-learning model on the MNIST dataset, with 0.9967 accuracy achieved. The results for the hybrid models using the MNIST dataset were good, with accuracy measures exceeding 0.9 for all the hybrid models; however, the results for the hybrid models using the Arabic character dataset were inferior.

Keywords: classification; convolutional neural network; recurrent neural networks; MNIST; model selection

1. Introduction

Despite massive technological advances, many people’s textual compositions are still handwritten. Using pen and paper for writing is essential to people’s work. Handwriting has different sizes and styles, making the creation of automatic techniques for recognizing texts a challenging task in computer vision [1]. Text-recognition systems utilize automated techniques for text recognition by converting text included in images into matching digital formats. Such systems can discover typed or handwritten characters and are used in different application domains.

A handwritten-character-recognition system is a computer-vision system that is intended to classify and recognize handwritten characters [2]. Character recognition is still critical and challenging in many computer-vision tasks [3]. With the increased interest in handwriting recognition and the developments in machine-learning and deep-learning algorithms, researchers have made significant improvements and advances in this field. English-handwriting-recognition methodologies have received significant interest from researchers [4]; however, Arabic has not yet received enough interest. With more than 315 million native Arabic speakers [5], the need for Arabic-handwriting-recognition systems is critical. Arabic is one of the most popular spoken languages in the world, with twenty-eight alphabets and different letter styles, based on geography.

In general, handwriting is a pattern-recognition research area with different applications. For each application domain, specific constraints should be considered [6], otherwise the recognition process will be complicated due to the wide range of handwriting styles and sizes. For example, recognizing characters on car license plates is more straightforward than recognizing Arabic handwriting due to the different styles of handwriting. Therefore, researchers are making great efforts to improve recognition systems using various techniques, deep-learning algorithms being at the top of the list.
For a while, Arabic handwritten character recognition (AHCR) has been an area of research in pattern recognition and computer vision. Several machine-learning (ML) algorithms, such as support vector machines (SVMs), have improved AHCR. Such models are still limited and cannot outperform convolutional neural networks (CNNs) on different Arabic handwriting datasets.

Several Arabic handwriting datasets have been used in the literature to create convolutional neural network (CNN) models [1,7,8] that automatically extract features from images and outperform classical machine-learning techniques (such as [9,10]), especially when large datasets with a large number of classes are used. As Niu and Suen claim [11], better classification results can be achieved when an SVM is replaced with an MLP in deep learning. This is because MLPs are based on empirical risk minimization, which tries to minimize errors in the training set. As a result, the training procedure is terminated when the back-propagation algorithm finds the first separating hyperplane.

Several deep-learning architectures have been used in the literature in different applications (deep neural networks (DNNs), convolutional neural networks (CNNs), deep belief network (DBNs), recurrent neural networks (RNNs), and generative adversarial networks (GANs)) [12]. In this work, a CNN, a commonly used deep-learning algorithm, was used to recognize Arabic handwritten characters.

Despite all the progress that has been made, there are still some challenges and a demand for new methods to overcome these limitations [13]. Compared to the English language, the quantity of available datasets of Arabic handwritten characters is relatively small. Additionally, some of the published datasets include few records. There is a requirement for a vast dataset containing a variety of font sizes, styles, illuminations, users, and texts [14]. Some of the collected records (pictures) may have unnecessary data noise that must be eliminated, or misclassification might occur. Especially for massive datasets, it is necessary to identify a simple and rapid method for removing noise automatically rather than manually [15,16].

The rest of the paper is organized as follows. Section 2 explains the background of the previous research work on handwriting recognition. Section 3 presents the materials and methods of the study. Section 4 describes the experimental results and analysis, while Section 5 presents the discussion and comparison. Finally, the conclusions are drawn and future works are considered.

2. Related Works and Motivation

Handwriting recognition using CNNs has received attention in terms of research work in different languages, such as English [17–21], Arabic [1,3,7–10,22], Bangla [2], and Chinese [23–28]. Recently, the focus on Arabic handwriting recognition has increased [29]. Researchers have developed different techniques to enhance recognition outcomes.

According to El-Sawy et al. [3], CNN techniques outperform other feature-extraction and classification methods, especially with big datasets. This is not applicable for most of the studies on Arabic language recognition. Therefore, the authors created the Arabic Handwritten Characters Dataset (AHCD) and proposed a CNN model that achieved an accuracy of 94.9. Similarly, Altwaijry et al. [1] released Arabic handwritten alphabets in what is called “the Hijja” dataset. The dataset consists of samples written by children aged 7 to 12 years old. The researchers conducted handwriting-recognition experiments using a CNN trained on two datasets, the Hijja dataset and the Arabic Handwritten Character Dataset (AHCD). The proposed model achieved accuracies of 97% on the AHCD dataset and 88% on the Hijja dataset. These results indicate that they have outperformed the achieved results of El-Sawy et al. [3].

Using a different approach, Alrobah and Albahli [13] merged an ML model with a deep-learning model to create a hybrid, taking advantage of CNN models in feature extraction and ML models in classification. The study achieved an accuracy of 96.3, proving the hybrid model’s effectiveness. The result was better than those obtained in the original experiment performed on the same dataset (the Hijja) by Altwaijry and Al-Turaiki [1].
A CNN architecture known as Alexnet was utilized by Boufenar et al. [7], which includes three layers of max pooling followed by three layers of fully connected convolutions. They investigated the impact of preprocessing on model improvement. The Alexnet model was trained and evaluated using two datasets, OIHACDB-40 and AHCD, and three learning strategies: training the CNN model from scratch, utilizing a transfer-learning technique, and fine-tuning the weights of the CNN architecture. The experimental outcomes demonstrated that the first technique outperformed the others, with 100 percent and 99.98 percent accuracy for the OIHACDB-40 and AHCD datasets, respectively.

Balaha et al. [30] established a vast, complicated dataset of Arabic handwritten characters (HMBD). They implemented a deep-learning (DL) system with two convolutional-neural-network (CNN) architectures (called HMB1 and HMB2), using optimization, regularization, and dropout techniques. They employed Elsawy et al. [3] as a controlled study throughout their 16 experiments with the HMBD, CMATER, and AIA9k datasets. The study suggested that data augmentation helped increase testing accuracy and reduce overfitting. Data augmentation increased the volume of input data; as a result, the architectures were learned and trained using more data. The top results for HMBD, CMATER, and AIA9k were 90.7%, 97.3%, and 98.4%, respectively. Younis [31] and Najadat et al. [32] built CNN models that were trained and tested on AHCD to improve AHCD performance. Three convolutional layers and one fully connected layer comprised the model of [30]. In addition, two regularization methods, dropout and batch normalization, with distinct data-augmentation techniques, were employed to enhance the model’s performance. Using the AIA9k and AHCD datasets to train and test the model, the accuracies were 94.8 and 97.6 percent, respectively. Similarly, the CNN design suggested in [31] comprised four convolutional layers, two max-pooling layers, and three fully connected layers. The authors examined various epochs and batch sizes and found that 40 epochs with a batch size of 16 produced the best results for training of the model. Based on empirical findings, the model accuracy achieved was 97.2%.

Considering various model architectures, subsequent investigations attempted to identify more successful instances. Alyahya et al. [33] examined the performance of the ResNet-18 architecture when an FCL and dropout were added to the original architecture for recognizing handwritten Arabic characters. Two models utilized a fully connected layer with/without a dropout layer following all convolutional layers. The other two models used two fully connected layers with/without a dropout layer. They used the AHCD dataset to train and evaluate the CNN-based ResNet-18 model, with the original ResNet-18 achieving the best test result of 98.30 percent. Almansari et al. [34] examined the performance of a CNN and a multilayer perceptron (MLP) in detecting Arabic characters from the AHCD dataset. To reduce model overfitting, they examined various dropout levels, batch sizes, neuron counts in the MLP model, and filter sizes in the CNN model. According to the experimental results, the CNN model and the MLP model achieved the highest test accuracies of 95.3% and 72.08%, respectively, demonstrating that CNN models are more suitable for Arabic handwritten character recognition.

Similarly, to improve the detection of Arabic digits, Das et al. [35] provided a collection of 88 features of handwritten Arabic number samples. An MLP classifier was constructed with three layers (input, single hidden layer, and output). Back-propagation was performed to train the multi-layer perceptron, which was subsequently used to classify Arabic numerals from the CMATERDB 3.3.1 dataset. According to testing results, the model achieved an average accuracy of 94.93 percent on a database of 3000 samples.

Musa [36] introduced datasets that include Arabic numerals, isolated Arabic letters, and Arabic names. The vast majority of these datasets are offline. In addition, the report described published results and an upcoming study. Noubigh et al. [37] examined the issue of Arabic handwriting recognition. They offered a new architecture based on a character-model approach and a CTC decoder that combined a CNN and a BLSTM. For experimentation, the handwriting Arabic database KHATT was used. The results demonstrated a net perfor-
mance benefit for the CNN–BLSTM combined method compared to the methods employed in the literature.

De Sousa [8] suggested two deep CNN-based models for Arabic handwritten character and digit recognition: VGG-12 and REGU. The VGG-16 model was derived from the VGG-12 model by removing the fifth convolutional block and adding a dropout layer prior to the SoftMax FCL classifier. In contrast, the REGU model was created from scratch by adding dropout and batch-normalization layers to both the CNN and fully connected layers. The two models were trained, one with and the other without data augmentation. The predictions of each of the four models were then averaged to construct an ensemble of the four models. The ensemble model’s best test accuracy was 99.47 percent. Mudhsh et al. [10] proposed a model for recognizing Arabic handwritten numerals and characters. The model consisted of thirteen convoluted layers, followed by two max-pooling layers and three completely connected layers. To reduce model complexity and training time, the suggested model employed only one-eighth of the filters in each layer of the original VGG-16. The model was trained and evaluated using two distinct datasets: ADBase for the digit-recognition task and HACDB for the character-recognition task. To prevent overfitting, they utilized dropout and data augmentation. The model’s attained accuracy was 99.66% when using the ADBase dataset and 97.32% when using the HACDB dataset.

Al-Taani et al. [38] built a ResNet architecture to recognize handwritten Arabic characters. The suggested method included pre-processing, training ResNets on the training set, and testing trained ResNets on the datasets. Using MADBase, AIA9K, and AHCD, this method achieved 99.8 percent, 99.05 percent, and 99.55 percent accuracies, respectively.

AlJarrah et al. [39] established a CNN model to detect printed Arabic letters and numbers. Using an AHCD dataset, the model was trained. This study used data-augmentation techniques to improve model performance and detection outcomes. The experiment demonstrated that the proposed strategy might achieve a success rate of 97.2 percent. The model’s accuracy increased to 97.7 percent once data augmentation was implemented. Elkhayati et al. [40] created a method for segmenting Arabic words for recognition purposes using a convolutional neural network (CNN) and mathematical morphology operations (MMOs). In their study, the authors offered a directed CNN and achieved better performance than a standard CNN. Elleuch et al. [41] presented a deep-belief neural network (DBNN) for identifying handwritten Arabic characters/words. The proposed model began with the row data before proceeding to the unsupervised learning technique. This model’s performance on the HACDB dataset was 97.9 percent accurate. Kef et al. [42] developed a fuzzy classifier with structural properties for Arabic-handwritten-word-recognition offline systems based on segmentation procedures. Before extracting features using invariant pseudo-Zernike moments, the model splits characters into five distinct categories. According to the study’s findings, the proposed model achieved a high level of accuracy for the IFN/ENIT database of 93.8%.

Based on the generic-feature–independent-pyramid multilevel model (GFIPML), Korichi et al. [43] developed a method for recognizing Arabic handwriting. To evaluate their system’s performance, the authors utilized the AHDB dataset and obtained better outcomes. They combined local phase quantization (LPQ) with multiple binarized statistical image features (BSIs) to enhance the recognition. The proposed system achieved an accuracy of 98.39%. However, in a previous study, Korichi et al. [44] compared the performance of multiple CNN networks to statistical descriptors derived from the PML model. The highest recognition rate attained by the LPQ descriptor was 91.52%. Another common application of handwriting recognition is Arabic handwritten literal amount recognition. Korichi et al. [45] performed numerous experiments with convolutional neural networks (CNNs), such as basic CNN, VGG-16, and ResNet, which were developed using regularization approaches, such as dropout and data augmentation. The results demonstrated that CNN architectures are more effective than previous approaches based on handmade characteristics.
Since pre-trained models are trained on a general dataset and may not be suitable for Arabic handwriting classification, it is evident from the articles mentioned above that models created from scratch tend to produce better results. Models that have been pre-trained include machine-learning and deep-learning models that have been created and trained on a wide range of data. For example, the ImageNet dataset is used to train models and comprises thousands of images; however, models constructed utilizing the image dataset from the beginning of their training make them far more fit for the classification task due to their comprehensive understanding of the dataset. Later, adjusting the hidden layers of the deep-learning models makes them more suitable for the classification challenge. In this work, we tried to overcome this limitation by building the model specifically for Arabic handwritten character classification. Table 1 summarizes the CNN models developed in the literature for Arabic handwriting recognition. Table 2 presents a summary of the Arabic handwriting datasets.

Table 1. Summary of the CNN models for Arabic handwriting recognition.

| Paper                  | Year | Model                          | Dataset       | Accuracy (%) |
|------------------------|------|--------------------------------|---------------|--------------|
| El-Sawy et al. [3]     | 2017 | CNN                            | AHCD          | 94.9         |
| Altwajry et al. [1]    | 2017 | CNN                            | Hijja         | 97           |
| Alrobah and Albahli [29]| 2021 | Hybrid CNN models: CNN + FCL, CNN + SVM, CNN + XGBoost CNN from scratch, CNN as features extractor, Fine-tuned CNN | AHCD | 94.5 |
|                        |      |                                | Hijja         | 96.3         |
|                        |      |                                | OIHACDB-40    | 100          |
| Boufenar et al. [7]    | 2018 | CNN                            | AHCD          | 98.86        |
|                        |      | AHCD from scratch              | Hijja         | 99.98        |
|                        |      | AHCD as feature extractor      | AHCD          | 82.53        |
|                        |      | Fine-tuned CNN                 |               | 96.78        |
| Balaha et al. [30]     | 2021 | CNN                            | HMBD          | 90.7         |
|                        |      | AHCD                           |               | 97.3         |
|                        |      | AHCD                           | AIA9k         | 98.4         |
| Younis et al. [31]     | 2017 | CNN                            | AIA9k         | 97.6         |
| Najadat et al. [32]    | 2019 | CNN                            | AHCD          | 97.2         |
| Alyahya et al. [33]    | 2020 | CNN                            | AHCD          | 98.3         |
| Almansari et al. [34]  | 2019 | CNN                            | AHCD          | 95.27        |
|                        |      | MVL                            |               | 72.08        |
| Das et al. [35]        | 2010 | CNN                            | CMATERDB      | 94.93        |
| Musa [36]              | 2011 | CNN                            | SUST-ALT      |              |
| Nouhig et al. [37]     | 2021 | CNN and BLSTM                  | KHAHT         |              |
| De Sousa [8]           | 2017 | VG16-based CNN                 | AHCD          | 97.32        |
| Mudshb et al. [10]     | 2018 | CNN                            | AHCD          | 98.42        |
| Al-Taani et al. [38]   | 2021 | CNN                            | AHCD          | 99.5         |
| AlJarrah et al. [39]   | 2021 | CNN                            | AHCD          | 97.7         |
| Elleuch et al. [43]    | 2015 | CNN                            | HACDB         | 97.9         |
| Kef et al. [42]        | 2016 | 5 Fuzzy ARTMAP                 | IFN/ENIT      | 93.8         |

Table 2. Summary of the Arabic handwriting datasets.

| Dataset      | Type      | Amount |
|--------------|-----------|--------|
| AHCD         | Characters| 16,800 |
| Hijja        | Characters| 47,434 |
| OIHACDB-40   | Characters| 30,000 |
| HMBD         | Characters| 54,115 |
| AIA9k        | Alphabet  | 8737   |
| CMATERDB     | Digits    | 3000   |
Table 2. Cont.

| Dataset   | Type   | Amount  |
|-----------|--------|---------|
| SUST-ALT  | Digits | 47,988  |
|           | Letters|         |
|           | Words  |         |
| KHATT     | Characters | 4000  |
| HACDB     | Characters | 6600  |
| IFN/ENIT  | Characters | 212.211 |

3. Materials and Methods

The convolutional neural network (CNN) is the leading technique applied in automatic character and digit detection using computer systems. Various deep-learning models are being tested for multiple languages. As one of the most spoken languages in the world, Arabic is no exception. This section discusses the methods and techniques used to create the system for detecting handwritten Arabic characters.

There have been many approaches used for handwritten character recognition in different languages, but proper techniques have not yet been developed for the Arabic language. Arabic handwritten character recognition is now needed. The CNN approach was best suited for other language datasets, so the proposed system tried this technique with the complete setup.

3.1. Brief Overview of Convolutional Neural Networks

A typical convolutional neural network (CNN) is an artificial neural network that tries to mimic the way the human brain detects, recognizes, and interprets images. It does so by processing pixel data to find features that stand out and serve as identification points. It works by assigning importance (learnable biases and weights) to certain parts of inputted images to differentiate them from one another, which ultimately leads to recognition of what the images contain. The various parts of a typical CNN are further elaborated below.

A CNN automatically detects the available features in a dataset. These features may be statistical, texton, curvature, along with many others. The features used depends on the problem that needs to be addressed, but this process was performed automatically in this model. According to the proposed system, the data were required to know these character types. Here, the CNN model also detected the curvature features or image contours.

3.1.1. Convolutional Layer

A convolutional layer constitutes the foundation and main building block of a CNN. It works by converting an input image into a feature map, also known as an activation map. The convolutional layer has a kernel, or filter, a two-dimensional array of weights that is responsible for carrying out the task of feature extraction, which leads to the creation of a feature map. The filter works by moving from one image stride to another while performing a dot product and feeding the result to an output array. This output is the feature map. The filter needs to be configured before the operation begins and maintained throughout. The parameters to be configured are:

- **The number of filters**: This affects the number of feature maps to be obtained, as the number of feature maps increases with the number of filters.
- **Stride**: This is the number of pixels the filter travels for each operation. Usually, a stride of one or two is used because a larger number of strides leads to a smaller output and missing key features. The stride and feature map are shown in Figure 1.
- **Zero padding**: This is crucial, since most input images have elements that fall outside the input matrix, which might be ignored. Zero padding covers boundary pixels, thereby producing larger outputs of high quality.
weights. Instead, it applies an aggregation function to the image to populate the output layer. The application of max pooling on the input.

In Figure 1, the stride image shows the input image being turned into a matrix. The filter is then applied to the green section, and a dot product is computed between the input pixels and the filter. After this, the filter is then moved by one stride, repeating the initial process until the kernel has covered the entire matrix formed by the image. The final result is a new, smaller matrix called the feature map or activation map.

3.1.2. Pooling Layer

The pooling layer, or down sampling, performs the task of reducing the parameters in the input image, thereby resulting in dimensionality reduction. It also sweeps through the entire input just like the filter; however, unlike the convolutional layer, it does not carry any weights. Instead, it applies an aggregation function to the image to populate the output array. Two types of pooling are usually used:

Max pooling selects the pixel with the maximum value as the filter moves across the input image and sends it to the output array.

Average pooling: Here, the average value within the receptive field is calculated as the filter moves through the image and sends it to the output array.

The pooling used for this project is illustrated in Figure 2.

![Stride and feature map](image)

**Figure 1.** Stride and feature map.

As shown in Figure 2, we applied max pooling, which returns the maximum value in the filter, then moves by one stride and repeats the process. This is repeated until the entire image has been covered. In the above figure, we can see how 6 is chosen from the first stride. The same applies to 8, 3, and 4.

3.1.3. Fully Connected Layer

The convolutional and pooling layers perform the task of feature extraction, using the filter to create the feature map [46]. The fully connected layer performs the detection and recognition tasks by matching patterns in the feature maps of the images being operated on. The fully connected layer is arranged so that each node in the output layer connects directly to a node in the previous layer. Figure 3 shows a typical convolutional neural network and how all the parts are interconnected.
10 classes for 0–9 numerical digits; the other dataset contains handwritten Arabic characters.

Our research is a classification problem, so the extracted data will be used to classify the handwritten Arabic characters when we get to the final stage.

From the literature review, it can be seen that machine-learning models have not been used as extensively as deep-learning models. Most of the papers examined were generally focused on feature ANN and CNN approaches. This research explores advanced ensemble methods of classification which have not previously been used in related experiments.

The two datasets used for all the models were split into training and test sets at a ratio of 70% to 30%. The first dataset used was the Arabic MNIST dataset, which contains 10 classes for 0–9 numerical digits; the other dataset contains handwritten Arabic characters and has 28 classes resembling the 20 Arabic characters.

Two different strategies were used to conduct a series of experiments. The first strategy involved standalone models, while the second strategy utilized hybrid models, which were designed by combining two models. The two strategies are discussed below:

Standalone Models

- XGBoost stands for extreme gradient boosting. It is an ensemble machine that uses trees for boosting. It makes use of gradient boosting for the decision tree, thereby increasing speed and performance. These trees are built sequentially to reduce errors from the preceding tree, and each new tree learns from the previous one. Hence, as new trees grow, more knowledge is passed on. To enhance accuracy, the model, after the iterations, tries to minimize the following objective function, which comprises a loss function and regularization. There are three main forms of boosting:
  1. A gradient-boosting algorithm uses a gradient algorithm and the learning rate;
  2. A stochastic gradient-boosting algorithm uses sampling at the row and column per split levels;
  3. A regularized gradient-boosting algorithm uses L1 and L2 regularization.

- Random forest is a meta-estimator that fits several different decision trees on various subsamples, and the output is averaged so as to control overfitting. Random forest was used for this experiment because the error generated is always lesser than the decision tree due to out-of-bag error. Decision trees are a popular method for machine-learning tasks. Tree learning derives from the shell method for data mining because of its invariant behavior when it comes to scaling and other transformation methods for feature values, which are robust given the inclusion of feature values. The error generated by a random forest is always lesser than that generated by a decision tree because of the out-of-bag error, which is also called the out-of-bag estimate. This error-estimation technique is a method of estimating the prediction errors of random forests, which involves boosted decision trees that utilize bootstrap aggregation to

Figure 3. A typical convolutional neural network.

3.2. Architecture of the Applied CNN Models

This section outlines the machine-learning and deep-learning algorithms used in this experiment. All artificial intelligence experiments follow the same procedure. First, relevant data are collected and preprocessed to ensure that the raw data are suitable for training and testing. Second, certain features are extracted from the data and used to train and test the models. Finally, the extracted features are used for prediction, depending on the purpose of the research. Our research is a classification problem, so the extracted data will be used to classify the handwritten Arabic characters when we get to the final stage.

From the literature review, it can be seen that machine-learning models have not been used as extensively as deep-learning models. Most of the papers examined were generally focused on feature ANN and CNN approaches. This research explores advanced ensemble methods of classification which have not previously been used in related experiments.

The two datasets used for all the models were split into training and test sets at a ratio of 70% to 30%. The first dataset used was the Arabic MNIST dataset, which contains 10 classes for 0–9 numerical digits; the other dataset contains handwritten Arabic characters and has 28 classes resembling the 20 Arabic characters.

Two different strategies were used to conduct a series of experiments. The first strategy involved standalone models, while the second strategy utilized hybrid models, which were designed by combining two models. The two strategies are discussed below:

Standalone Models

- XGBoost stands for extreme gradient boosting. It is an ensemble machine that uses trees for boosting. It makes use of gradient boosting for the decision tree, thereby increasing speed and performance. These trees are built sequentially to reduce errors from the preceding tree, and each new tree learns from the previous one. Hence, as new trees grow, more knowledge is passed on. To enhance accuracy, the model, after the iterations, tries to minimize the following objective function, which comprises a loss function and regularization. There are three main forms of boosting:
  1. A gradient-boosting algorithm uses a gradient algorithm and the learning rate;
  2. A stochastic gradient-boosting algorithm uses sampling at the row and column per split levels;
  3. A regularized gradient-boosting algorithm uses L1 and L2 regularization.

- Random forest is a meta-estimator that fits several different decision trees on various subsamples, and the output is averaged so as to control overfitting. Random forest was used for this experiment because the error generated is always lesser than the decision tree due to out-of-bag error. Decision trees are a popular method for machine-learning tasks. Tree learning derives from the shell method for data mining because of its invariant behavior when it comes to scaling and other transformation methods for feature values, which are robust given the inclusion of feature values. The error generated by a random forest is always lesser than that generated by a decision tree because of the out-of-bag error, which is also called the out-of-bag estimate. This error-estimation technique is a method of estimating the prediction errors of random forests, which involves boosted decision trees that utilize bootstrap aggregation to
subsample data samples required for training. The parameters used for training were 
\text{max\_depth = 12, random\_state = 0, and n\_estimators = 100.}

- **CatBoost** stands for category boosting, which was developed by Yandex [47]. It uses 
the gradient-boosting technique and does not require conversion of the dataset to a 
specific format, unlike other machine-learning algorithms, making it more reliable and 
easier to implement. The parameters used for this experiment were iterations = 100, 
deepth = 4, and learning\_rate = 0.1.

- **Logistic regression** uses L1, L2, and ElasticNet as regularization techniques and then 
calculates the probability of a particular set of data points belonging to either of those 
classes. For this experiment, the log was used as the cost function.

- **A support vector machine (SVM)** is a supervised machine-learning algorithm for 
classification. It takes data points as inputs and outputs a hyperplane that separates 
the classes with the aim of achieving a hyperplane that maximizes the margin between 
the classes. This is the best hyperplane. For this experiment, two kernels were tested, 
the RBF and the linear kernel.

- **A feed-forward neural network** is an artificial neural network wherein connections 
between the nodes do not form a cycle. This means that information moves only 
in the forward direction, from the input nodes, through the hidden nodes, to the 
output nodes [48]. Four optimization methods were experimented on, including 
Adam Optimizer, RMSprop, Adagrad, and stochastic gradient descent. Table 3 shows 
the feed-forward network architecture and all the parameters.

Table 3. The feed-forward architecture design.

| Feed-Forward Network | Architecture Design |
|----------------------|---------------------|
| Layer (type)         | Output Shape        | Param #  |
| Dense_30 (Dense)     | (None, 512)         | 401,920  |
| Dense_31 (Dense)     | (None, 512)         | 262,656  |
| Dense_32 (Dense)     | (None, 10)          | 5130     |
| Total params: 669,706 | Trainable params: 669,706 | Non-trainable params: 0 |

- All these proposed algorithms were used for the recognition of handwritten Arabic 
characters, where statistical type features, shape-based features (curvatures), and 
categorical features with indexes were used.

- A convolutional neural network is a deep-learning algorithm that can take in an input 
image, assign importance (learnable weights and biases) to various aspects/objects in 
the image, and differentiate one from another. The convolutional layer is first applied 
to create a feature map with the right stride and padding. Next, pooling is performed 
to lessen the dimensionality and properly adjust the quality of parameters used for the 
training, thereby reducing preparation time and battling overfitting. This experiment 
used max pooling, which takes the maximum incentive in the pooling window. Trans-
fer learning was also carried out. This focuses on storing knowledge gained while 
solving one problem and applying it to a different, related problem; it is particularly 
useful in our case, since there is a limited number of data. A sequential model was 
used while using different optimization algorithms, including RMSProp, Adagrad, 
stochastic gradient descent, and Adam Optimizer. Table 4 shows the architecture of 
the convolutional neural network, with all the layers, the shapes expected, and the 
number of parameters.
Table 4. The CNN architecture design.

| Layer (type)       | Output Shape   | Param #  |
|--------------------|----------------|----------|
| Conv2d_7           | (None, 26, 26, 32) | 320      |
| Max_pooling2d_7    | (None, 13, 13, 32) | 0        |
| Batch_normalization_7 | (None, 13, 13, 32) | 128      |
| Conv2d_8           | (None, 11, 11, 64) | 18,496   |
| Max_pooling2d_8    | (None, 5, 5, 64)  | 0        |
| Batch_normalization_8 | (None, 5, 5, 64)  | 256      |
| Conv2d_9           | (None, 3, 3, 128) | 73,856   |
| Max_pooling2d_9    | (None, 1, 1, 128) | 0        |
| Batch_normalization_9 | (None, 1, 1, 128) | 512      |
| Flatten_3          | (None, 256)      | 0        |
| Dense_5            | (None, 256)      | 33,024   |
| Dense_6            | (None, 10)       | 2570     |

Total params: 129,162  
Trainable params: 128,714  
Non-trainable params: 448

CNN parameters were selected, and 129,162 parameters were chosen during the model training. Trainable parameters were set at 128,714, whereas the non-trainable parameters numbered 448 during the model training. The number of epochs was 20, the batch size was 32, and there were 3600 training samples and 2400 validation samples.

3.3. Hybrid Models

Hybrid models are models that combine two or more models. Those models integrate machine-learning models and other soft-computing, deep-learning, or optimization techniques.

In this research, we used CNN as a base-feature extractor, and these extracted features were fed into machine-learning models to see how the models performed, as shown in Figure 4. The architecture of the feature extractor was the same as that of the CNN model and the various machine-learning models mentioned above. The following are the hybrid models that were experimented on:

1. CNN + SVM;
2. CNN + Random Forest;
3. CNN + AdaBoost;
4. CNN + XGBoost;
5. CNN + CatBoost;
6. CNN + Logistic Regression.
Hybrid models are models that combine two or more models. Those models integrate machine-learning models and other soft-computing, deep-learning, or optimization techniques. In this research, we used CNN as a base-feature extractor, and these extracted features were passed to the model for classification. The proposed system used two approaches: deep learning and conventional machine learning. Both methods involve some signs; as with the conventional machine learning approach, where the model can be trained with an available dataset, this approach worked based on the extraction of various features, such as statistical, curvature, and multiple different features. The machine-learning approach must extract features manually and pass them to the model for training. The deep-learning approach has some other aspects compared with this approach. In deep-learning models, there is no need to extract features manually because the models can extract the required or available features automatically. These features are passed for the model training and classification. The CNN approach extracts features and gives them to the model for classification. The proposed system used both to check where the model can deliver the best results for the problem.

The proposed approaches for the system have their benefits and drawbacks. A CNN was proposed because this approach has been used for handwritten character recognition with many other languages, so adopting this method could achieve the best performance. The CNN approach involved two steps. The first was the extraction of the required features; the second was classification. The standalone approach was proposed to check the individual impact on the dataset and see whether it could perform the same procedure for handwriting recognition. The hybrid approach was designed because it helped to achieve more reliable results than the standalone approach. It works like the ensemble approach in solving the Arabic handwriting recognition problem. This was decided after reviewing the multiple approaches to handwriting recognition reported in the literature.

4. Experimental Results and Analysis

This section discusses the experimental setup for this work, including all the hardware and software requirements. Then, the results obtained from all the models are reported to show how they all performed. We compared the models to see which ones did well and which did not. Finally, we took the best performance we achieved and compared it with state-of-the-art models for Arabic handwriting recognition to see how well our model performed compared to the others.

The models were trained using both the Arabic MNIST dataset and the Arabic character dataset. The CNN’s only task is to extract relevant features from the handwritten images, which are its outputs. It then passes the features to the machine-learning models, which use these features to find patterns, thereby classifying the images.

The proposed system used two approaches: deep learning and conventional machine learning. Both methods involve some signs; as with the conventional machine learning approach, where the model can be trained with an available dataset, this approach worked based on the extraction of various features, such as statistical, curvature, and multiple different features. The machine-learning approach must extract features manually and pass them to the model for training. The deep-learning approach has some other aspects compared with this approach. In deep-learning models, there is no need to extract features manually because the models can extract the required or available features automatically. These features are passed for the model training and classification. The CNN approach extracts features and gives them to the model for classification. The proposed system used both to check where the model can deliver the best results for the problem.

The deep-learning approach requires more time for training due to the large volumes of datasets, whereas the conventional machine-learning approach takes less time with small

\[ V \]

**Figure 4.** The architecture of the hybrid models.
datasets. Deep-learning models are more complex than conventional learning approaches and need devices with high computational power to execute them.

4.1. Datasets

The first dataset used for this research was an Arabic MNIST dataset obtained from the MNIST database developed by LeCun et al. [49], which comprises 60,000 images for training and 10,000 images for testing, with 6000 images per class. A part of this dataset can be seen in Figure 5. In the same fashion, the Arabic MNIST dataset comprises 60,000 images from digit 0 to digit 9, with each class having 6000 handwritten images.

The second dataset [3] comprises 16,800 characters written by 60 participants. The images were scanned at a resolution of 300 dpi. Each image was segmented automatically using MATLAB 2016a, which automatically coordinated every image. The database was then partitioned into 13,400 images for training and 480 images for testing, with 120 images per class, which sums up to 3360 images in total.

The images were processed by converting them to grayscale using OpenCV so that the images had a single filter instead of three, then row-wise values were extracted side by side, with 784 columns. The labels for the images were used as the target values, thus generating a csv file. The Kaggle dataset already had a csv file with 1024 columns. These columns contained 0 for a black value and 1 for a white value for an image. Then, the dataset was separated into two csv files for training and testing.

4.2. Experimental Setup

The choice of datasets was the same for all the models trained (both standalone and hybrid), the datasets used are the Arabic MNIST dataset and the Arabic character dataset. All experiments were conducted using Google Colab (short for Colaboratory), which is a product from Google research that allows users to write and execute Python code with either a CPU or a GPU. As for libraries, the Keras library was employed to create the deep neural networks, the Python programming language (version 3.6.3) being used for all of them.

The proposed system was evaluated using the open-source MNIST Arabic dataset. This system calculated evaluation measures, such as accuracy, precision, recall, and F1-measure, to check the proposed system’s consistency and performance. Accuracy is the ratio of the number of correct predictions to the total number of predictions. Equation (1) shows the accuracy measure. These evaluation measures are described in terms of TPs (true positives), TNs (true negatives), FNs (false negatives), and FPs (false positives).

\[
Accuracy = \frac{TN + TP}{TP + TN + FP + Fn}
\]
Precision is the ratio of true positives to true and false positives. The equation of precision is shown in Equation (2).

\[
\text{Precision} = \frac{TP}{TP + FP}
\]  

(2)

The recall is the ratio of correctly identified positive examples to the total number of positive models. Equation (3) shows the recall evaluation measure.

\[
\text{Recall} = \frac{TP}{TP + FN}
\]  

(3)

The F1-measure is a valuable metric in machine learning, which sums up the predictive performances to the combination of two other metrics. Equation (4) shows the F1-measure evaluation measure.

\[
F1 - \text{Measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]  

(4)

4.3. Results and Analysis

The task of designing a system that can automatically detect Arabic handwritten characters is very necessary, so we set out to use every means possible to achieve the best performance. To do this, we split the work into two parts: the standalone and the hybrid models. The results obtained from these experiments are reported below to show the performance evaluation for the models trained on both the Arabic MNIST and Arabic character datasets.

4.3.1. Standalone Machine-Learning Models

The standalone models are deep neural networks that carry out the entire classification task, from feature extraction to the final detection of patterns and the ultimate classification of images. The first model trained was the XGBoost model, then the random forest model was trained. Next, the CatBoost model was trained for 100 epochs, with a learning rate of 0.1. Then, the logistic regression model was trained with a cost function of the log. Finally, a Support Vector Machine (SVM) was trained, but since the SVM is a non-linear model, two different kernels were used: the RBF and the linear kernel. The results obtained by the standalone machine-learning models broke down into two tables. Table 5 shows the results for the models trained on the Arabic MNIST digit dataset, while Table 6 show the results obtained by the models trained on the Arabic character dataset.

| Model         | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|---------------|---------------------------|------------------------|--------------------------|----------------|
| Random Forest | 0.99                      | 0.99                   | 0.99                     | 0.98714        |
| CatBoost      | 0.95                      | 0.95                   | 0.95                     | 0.95476        |
| SVM (Linear)  | 1.0                       | 1.0                    | 1.0                      | 0.99628        |
| SVM (RBF)     | 1.0                       | 1.0                    | 1.0                      | 0.99628        |
| AdaBoost      |                           |                        |                          | 0.69600        |
| XGBoost       | 1.0                       | 1.0                    | 1.0                      | 1.00000        |
| Logistic Regression | 1.0                      | 1.0                    | 1.0                      | 0.99957        |

The proposed system was evaluated on the Arabic MNIST digit dataset. Evaluation measures were calculated against this dataset, with precision, recall, F1-measure, and accuracy calculated for the test dataset. Machine-learning algorithms, named Random Forest, CatBoost, SVM, AdaBoost, XGBoost, and Logistic Regression, were used for the model training and testing on the Arabic MNIST digit dataset. The XGBoost results were
the highest compared to the other algorithms, but Logistic Regression and Linear SVM also performed outstandingly. The results shown in Table 5 were remarkable for the dataset.

Table 6. Standalone machine-learning models trained on the Arabic character dataset.

| Model                             | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|-----------------------------------|---------------------------|------------------------|--------------------------|-----------------|
| Random Forest                     | 0.64                      | 0.64                   | 0.63                     | 0.6336          |
| CatBoost                          | 0.47                      | 0.47                   | 0.46                     | 0.4643          |
| SVM (Linear)                      | 0.41                      | 0.41                   | 0.41                     | 0.4233          |
| SVM (RBF)                         | 0.66                      | 0.65                   | 0.65                     | 0.6529          |
| Feed-Forward Neural Network (Adam)| 0.66                      | 0.61                   | 0.63                     | 0.6101          |
| XGBoost                           | 0.49                      | 0.49                   | 0.48                     | 0.4854          |
| Logistic Regression               | 0.66                      | 0.65                   | 0.65                     | 0.6529          |

As can be seen from Table 5, above, almost all the models performed extremely well. This goes to show that the models were fine-tuned in such a way that they conformed to the dataset used. The Arabic MNIST digit dataset is a standard dataset used for artificial intelligence projects. This means that it has been standardized to avoid overfitting, which indicates that our models are valid; however, as shown in Table 6, the results obtained were average, with the highest being those for Logistic Regression and the SVM (RBF kernel), both of which achieved an accuracy of 0.65. The accuracies were as low as 0.43 for the SVM (linear kernel).

Table 6 shows the evaluated results against the Arabic character dataset for the following machine-learning algorithms. The machine-learning algorithms were Random Forest, SVM, Neural Network, XGBoost, and Logistic Regression. Based on the feature extraction in the machine learning, the results were evaluated for the Arabic character dataset. SVM (RBF) had better results than the other algorithms, but the overall results were not the best.

4.3.2. Standalone Deep-Learning Models

The second phase of the research consisted of the experiments on the standalone deep neural networks which performed the tasks of feature extraction and classification. Three deep neural networks were trained and evaluated, including the feed-forward network, the convolutional neural network, and the neural network integrated with transfer learning. These models were tested with different optimizers to see how they performed, and a summary of the results obtained is presented in Table 7.

Table 7. Standalone deep-learning models trained on the Arabic MNIST digit dataset.

| Model                           | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|---------------------------------|---------------------------|------------------------|--------------------------|-----------------|
| Feed-Forward Neural Network (Adam) | 1.0                       | 1.0                    | 1.0                      | 0.98714         |
| CNN                             | 1.0                       | 1.0                    | 1.0                      | 0.99063         |
| Transfer Learning               | 1.0                       | 1.0                    | 1.0                      | 0.99673         |

Figure 6 shows the loss- and accuracy-curve graphs for the feed-forward network, for which the variation was very low after the 10th iteration. The purpose was to check the variation in the iterations, so that more iterations were required. The variational changes after the 10th iteration were only minor.

Figure 7 shows the curves for both the loss and accuracy of the training and validation; the model had an almost perfect training accuracy but a more realistic validation accuracy, which goes to show that the model performed well.
These models were tested with different optimizers to see how they performed, and a summary of the results obtained is presented in Table 7.

| Model                | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|----------------------|---------------------------|------------------------|--------------------------|----------------|
| Feed-Forward Neural Network (Adam) | 1.0                        | 1.0                    | 1.0                      | 0.98714         |
| CNN                  | 1.0                        | 1.0                    | 1.0                      | 0.99063         |
| Transfer Learning    | 1.0                        | 1.0                    | 1.0                      | 0.99673         |

Figure 6 shows the loss- and accuracy-curve graphs for the feed-forward network, for which the variation was very low after the 10th iteration. The purpose was to check the variation in the iterations, so that more iterations were required. The variational changes after the 10th iteration were only minor.

Figure 7 shows the curves for both the loss and accuracy of the training and validation; the model had an almost perfect training accuracy but a more realistic validation accuracy, which goes to show that the model performed well.

The loss and accuracy curves for the convolutional neural network can be seen in Figure 7. The model encountered a few problems at the beginning of the experiment, which caused it to move rapidly; however, with more epochs, the line smoothed out, which signifies the model’s effectiveness. Figure 8 shows the confusion matrix of the CNN classifier. Variations in the graphs show significant changes in the loss and accuracy for the training and validation. This impact was seen due to the significant change in the dataset. During training, the changes in loss and validation for the loss and accuracy were clearly seen.

The results obtained for the experiments conducted on our two datasets were impressive. The MNIST dataset was used to train the feed-forward network, the CNN, and the transfer-learning models, which achieved 0.9871, 0.9906, and 0.9967 accuracies, respectively. These are very effective accuracies, and it can be confidently said that the models can be used. On the other hand, the Arabic character dataset was used to train only the CNN with an adagrad optimizer, and it achieved an accuracy of 0.8992, see Table 8, which is not as high as that of MNIST but still remarkably high.
The loss and accuracy curves for the convolutional neural network can be seen in Figure 7. The model encountered a few problems at the beginning of the experiment, which caused it to move rapidly; however, with more epochs, the line smoothed out, which signifies the model’s effectiveness. Figure 8 shows the confusion matrix of the CNN classifier. Variations in the graphs show significant changes in the loss and accuracy for the training and validation. This impact was seen due to the significant change in the dataset. During training, the changes in loss and validation for the loss and accuracy were clearly seen.

Table 8. Standalone deep-learning models trained on the MNIST Arabic digit dataset.

| Model                  | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|------------------------|---------------------------|------------------------|--------------------------|----------------|
| CNN (Adagrad)          | 0.90                      | 0.90                   | 0.90                     | 0.8992         |

4.3.3. Hybrid Models

The final phase of this research was the experiment on the hybrid models, which were combinations of more than one model. These hybrid networks were a combination of a convolutional neural network (CNN), which performed the task of feature extraction because of such networks’ eminent suitability for the task. Then, the extracted features were passed on to various machine-learning models, which carried out the classification. The machine-learning models used were the SVM, Random Forest, AdaBoost, XGBBoost, CatBoost, and Logistic Regression. A summary of all the results obtained is shown in Tables 9 and 10, below.

Table 9. Hybrid machine-learning models trained on the Arabic MNIST digit dataset.

| Model                      | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|----------------------------|----------------------------|------------------------|--------------------------|----------------|
| CNN + SVM                  | 0.97                       | 0.97                   | 0.97                     | 0.9710          |
| CNN + Random Forest        | 0.94                       | 0.94                   | 0.94                     | 0.9390          |
| CNN + AdaBoost             | 0.57                       | 0.55                   | 0.54                     | 0.5470          |
| CNN + XGBoost              | 0.93                       | 0.93                   | 0.93                     | 0.9320          |
| CNN + Logistic Regression  | 0.94                       | 0.94                   | 0.94                     | 0.9388          |

The hybrid models were purely experimental. Since we combined models that naturally are standalone, it can be seen from the MNIST dataset experiment that all the models did well, with results exceeding 0.9, with the exception of CNN + AdaBoost, which achieved a classification accuracy of 0.55. The Arabic character dataset experiments did...
not produce such good results, with the highest performance attained by the CNN + SVM hybrid model, which had a classification accuracy of 0.872, and the lowest by the CNN + AdaBoost, which had a very poor accuracy score of 0.1690. This result goes to show that the CNN + AdaBoost hybrid model is not suitable for this classification task because it had the lowest classification accuracy for both datasets. In the hybrid approach, the combinations were set to check the model performance, the CNN + SVM set performing better than the other varieties. CNN + AdaBoost performed very poorly as compared with the other combinations. According to this system analysis, these types of machine-learning algorithms were not best-suited to this problem. Ultimately, which combination of algorithms is best depends on the nature of the problem to be solved.

### Table 10. Hybrid machine-learning models trained on the Arabic character dataset.

| Model                  | Precision (Macro Average) | Recall (Macro Average) | F1-Score (Macro Average) | Accuracy Score |
|------------------------|--------------------------|------------------------|--------------------------|----------------|
| CNN + SVM              | 0.87                     | 0.87                   | 0.87                     | 0.872          |
| CNN + Random Forest    | 0.78                     | 0.78                   | 0.87                     | 0.804          |
| CNN + AdaBoost         | 0.14                     | 0.17                   | 0.13                     | 0.1690         |
| CNN + XGBoost          | 0.78                     | 0.77                   | 0.77                     | 0.7990         |
| CNN + Logistic Regression | 0.83                    | 0.83                   | 0.83                     | 0.8560         |

### 5. Discussion and Comparisons

The primary purpose of this study was to develop a hybrid model that recognizes Arabic handwritten characters accurately. In this section, the performances of the hybrid CNN-based architectures will be discussed from three perspectives: the basic CNN architectures and the ML classifiers utilized in the hybrid models.

The Arabic MNIST and Arabic character datasets were used to assess the hybrid models developed in the current study. Ten standalone machine-learning models using the Arabic MNIST dataset, eight standalone machine-learning models trained on the Arabic character dataset, and five hybrid models for both the Arabic MNIST and Arabic character datasets were developed. According to the results, the performances of several CNN-based models for the Arabic character dataset were considerably inferior to the performances for the Arabic MNIST [49] dataset. Therefore, the Arabic character dataset can be considered a more complicated and challenging dataset. In this study, the best performance for the Arabic MNIST dataset was 97% (Figures 9 and 10), achieved with the hybrid model that combined a CNN and an SVM.

![Figure 9](attachment:image.png)

**Figure 9.** The accuracy of the hybrid models on the MNIST dataset.
5.2. Arabic Character Dataset

Figure 9. The accuracy of the hybrid models trained on the Arabic character dataset.

For the Arabic MNIST dataset, the Logistic Regression model and the XGBOOST model outperformed all the models in the research field in this area; the previous record of 99.71% was outperformed by our machine-learning approach. The hybrid models, such as CNN + SVM, reached near accuracy. However, hybrid models are more robust than standalone CNN models. A higher recall rate was observed for the hybrid models. The models which had the worst performances were the AdaBoost and CNN + AdaBoost models.

For the Arabic MNIST digit dataset, the machine-learning methods Random Forest, CatBoost, SVM, AdaBoost, XGBoost, and Logistic Regression were employed for model training and testing. In comparison to the other algorithms, XGBoost yielded the best results, while Logistic Regression and Linear SVM also performed quite well. Each of these models performed well. This demonstrates that the models were fine-tuned in accordance with the datasets used. The Arabic MNIST digit dataset is a typical dataset for artificial intelligence research. This implies that it has been standardized to prevent overfitting, indicating that our models are legitimate. However, on the Arabic character dataset, the results obtained were average, with Logistic Regression and the SVM (RBF kernel) achieving the best accuracy (0.65), followed by the SVM (linear kernel). The SVM accuracy dropped as low as 0.43 (with the linear kernel). The machine-learning methods examined were Random Forest, SVM, Neural Network, XGBoost, and Logistic Regression. On the basis of feature extraction in machine learning, the findings for the Arabic character dataset were reviewed, and the SVM (RBF) yielded superior results compared to the other algorithms, although the results were not the greatest overall.

In contrast, three deep neural networks were trained and assessed, including a feed-forward network, a convolutional neural network, and a neural network along with transfer learning. These models were evaluated using several optimizers to determine their performances. Experiments performed on our two datasets yielded amazing outcomes. The MNIST dataset was used to train the feed-forward network, the CNN, and the transfer-learning models, which produced corresponding accuracy levels of 0.9871, 0.9906, and 0.9967. These are very reliable accuracies; thus, it is fair to assume that the models can be used. In parallel, the Arabic character dataset was used to train the CNN solely with an adagrad optimizer, and it achieved an accuracy of 0.8992, which was not as high as that achieved for the MNIST dataset but still remarkable.

The last batch of models consisted of hybrid models. These hybrid networks incorporated a convolutional neural network (CNN), which performed the feature extraction, since such networks are superior for the task. The extracted features were then sent to several machine-learning models, which performed classification. SVM, Random Forest, AdaBoost, XGBoost, CatBoost, and Logistic Regression were the machine-learning models used. Regarding the hybrid model approach, the model combination performances were
evaluated to assess the approach. The CNN + SVM set was proven to outperform the others. However, the CNN + AdaBoost fared very badly compared to the other combinations.

5.1. Arabic MNIST Dataset

For the standalone models trained using the Arabic MNIST dataset, most of the models performed excellently, with accuracies above 95%. The best performances were observed for XGBoost (with values for precision, recall, F1-score, and accuracy of 1), Logistic Regression (with precision, recall, and F1-score values of 1.0 and an accuracy of 0.99957), the SVMs (with precision, recall, and F1-score values of 1 and an accuracy of 0.99628), followed by the transfer-learning model (with an accuracy of 0.99743) and the CNN (with an accuracy of 0.99063, and precision, recall, and F1-score values of 1). AdaBoost, on the other hand, obtained the lowest performance, with an accuracy of 0.696.

Considering the hybrid models, see Figure 9, the best classification performance was observed for the combination of CNN and SVM (with an accuracy of 0.9710, and precision, recall, and F1-score values of 0.97), followed by the CNN and Random Forest (with an accuracy of 0.9390), CNN and Logistic Regression (accuracy of 0.9388), and CNN and XGBoost (accuracy of 0.9320). The CNN and AdaBoost hybrid model obtained the weakest performance, with an accuracy of 0.5470, precision of 0.57, recall of 0.55, and an F1-score of 0.54.

5.2. Arabic Character Dataset

The standalone Convolutional Neural Network outperformed all the models trained using the Arabic character dataset. The experiments performed using this dataset recorded relatively lower accuracies than those trained using the Arabic MNIST dataset due to the large class size and low interclass difference coupled with a higher variance. Regarding the Arabic character dataset, the applied algorithms obtained weak-to-moderate performances, with accuracies ranging from 0.4233 (linear SVM) to 0.8992 (CNN), suggesting that the CNN might be the best choice for classification problems on this dataset. Hybrid models obtained similar performances, with the highest accuracy score of 0.872 for the CNN and SVM model, see Figure 10, followed by the CNN and Logistic Regression model (accuracy of 0.8560), the CNN and Random Forest model (accuracy of 0.804), and CNN and XGBoost (accuracy of 0.7990). The lowest performance was obtained by the combination of the CNN and AdaBoost algorithm, with an accuracy of 0.1690, a precision of 0.14, a recall of 0.17, and an F1-score of 0.13.

5.3. Comparison

As already discussed in Section 2, the literature contains reports of various attempts to improve Arabic handwriting recognition approaches. In the current work, the hybrid model of machine-learning and deep-learning algorithms (CNN + SVM) achieved an improved result. Tables 1 and 2 present summaries of the CNN models (standalone and hybrid) and the datasets used for the Arabic handwriting recognition experiments.

Various types of datasets, AHCD, HMBD, AIA9k, OIHAC, HACDB, and Hijja, have been used to train normal CNN models. Most of the studies have used the dataset AHCD [3], for which the highest accuracy of 99.98% was achieved [9] using AlexNet.

For the hybrid models, HACDB and AHCD were used. The accuracy reached for AHCD upon its initial publication was 94.90%, but the accuracy achieved by the hybrid model [50] was 95.07%.

In the current work, the performances of the proposed models applied on the MNIST dataset were generally better than those applied on the Arabic character dataset. Thus, one can infer that the latter dataset is more complicated than the former one. Researchers are encouraged to further investigate how performance can be improved for the Arabic character dataset. Table 11 presents a comparison of the proposed work and the other common architectures trained on the common datasets.
Table 11. Comparison with other architectures.

| Authors                        | Model                  | Datasets with Accuracy Rates (%) |
|--------------------------------|------------------------|----------------------------------|
|                                |                        | AHCD  | Hijja | AIA9k | MNIST |
| El-Sawy et al. [3]             | CNN                    | 94.9  |       |       |       |
| Altwaijry et al. [1]           | CNN                    | 97    | 88    |       |       |
| Alrobah and Albahli [29]       | Hybrid CNN models: CNN + FCL, CNN + SVM, CNN + XGBoost | 94.5  | 96.3  |       |       |
| Balaha et al. [30]             | CNN                    | 97.3  | -     | 98.4  |       |
| Younis et al. [31]             | CNN                    | 97.6  | -     | 94.8  |       |
| Al-Taani et al. [38]           | ResNet                 | 99.5  | -     | -     |       |
| Current work                   | CNN + SVM              |       |       |       | 97.1  |

6. Challenges and Open Problems

The task at hand is a very challenging one, since this is a field in which not much research has been carried out, especially in the area of artificial intelligence. Most of the challenges and problems to be discussed in this section could be mitigated or completely eradicated in the near future when more work has been completed. The challenges and open problems are discussed below:

- Inadequate dataset: For this experiment, two databases were utilized to acquire dataset images for both digits and characters; even so, the dataset was not sufficient, because one of the requirements for training both machine-learning and deep-learning models is an enormous supply of data. Lack of sufficient data leads to model overfitting.

- Cursive nature of the Arabic language: The Arabic language is cursive, meaning written characters are joined together, even in digital forms. If what is to be predicted was digitally written, it would have been much easier, but the project focused on handwriting recognition. Since every writer has a unique way of writing, it is difficult for the designed system to effectively detect handwritten characters and digits.

- Imbalanced dataset: The datasets used for the experiments were subjected to normalization by the database managers. However, the real-world problems that will be encountered might not be as clean as the ones in the dataset. For example, the images to be analyzed might be zoomed-in, unclear, or blurry, such that the system cannot be used to achieve effective results.

- Presence of special characters: The Arabic language is one that has a lot of special characters, such as dots and diacritics, making the classification problem a tedious one, in that misplacing just a single dot may completely change the meaning of the conveyed message and the presence or absence of a single stroke may change what is said completely. Hence, clear images are required for the system to make correct classifications.

The challenges and open problems are summarized in Figure 11.

The major contribution of the proposed system was to apply deep learning and conventional approaches to the crucial problem of Arabic-language character recognition. The proposed system addressed Arabic handwritten character recognition and various methods were explored to find the best solution to the problem. A deep-learning CNN approach, a conventional standalone approach, and a hybrid approach were introduced to address the crucial problem of handwritten character recognition. No approach has been applied to solve this issue prior to the system proposed here.
which had a 0.9967 accuracy. On the other hand, the results of the hybrid models achieved good records using the MNIST dataset, with accuracies exceeding 0.9 for all the hybrid models. It should be noted that the results for the hybrid models using the Arabic character dataset were very poor, which might mean that the problem lies with the dataset itself.

7. Conclusions and Future Studies

In this study, combining machine-learning and deep-learning algorithms to build the recognition models was worthwhile. The study involved standalone deep-learning experiments and focused attention on the idea of exploiting machine learning in classification and the advantages of deep learning in feature extraction. Several deep-learning and hybrid models were created. The best result for the standalone deep-learning models trained on the two datasets was achieved with the transfer-learning model on the MNIST dataset, which had a 0.9967 accuracy. On the other hand, the results of the hybrid models achieved good records using the MNIST dataset, with accuracies exceeding 0.9 for all the hybrid models. It should be noted that the results for the hybrid models using the Arabic character dataset were very poor, which might mean that the problem lies with the dataset itself.

To conclude, more studies are required to improve Arabic handwriting recognition and overcome the challenges it presents. Arabic handwriting might have some unique characteristics that need to be considered by researchers. What is appropriate for one language might not be appropriate for others. As mentioned in the Introduction to this study, English handwriting recognition might have received a large amount of interest in the literature. Although there is always room for improvement, the Arabic language is still in need of more attention from researchers, to improve handwriting-recognition methods and the use of appropriate datasets in experiments. It is hoped that the challenges discussed in this work will grab the attention of researchers and stimulate the proposal of more solutions and better contributions towards improvements in the field of Arabic handwriting recognition.

Author Contributions: Funding acquisition, W.A.; Methodology, W.A. and S.A.; Project administration, S.A.; Supervision, S.A.; Writing—original draft, W.A. and S.A.; Writing—review & editing, W.A. and S.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The experiment used a public dataset shared by LeCun et al. [49], and they have already published the download address for the dataset in their paper.

Acknowledgments: The researchers would like to thank the Deanship of Scientific Research, Qassim University, for funding the publication of this project.

Conflicts of Interest: The author declares that they have no conflict of interest.
References

1. Altwaijry, N.; Al-Turaiki, I. Arabic handwriting recognition system using convolutional neural network. *Neural Comput. Appl.* 2020, 33, 2249–2261. [CrossRef]
2. Sarkhel, R.; Das, N.; Saha, A.K.; Nasipuri, M. A multi-objective approach towards cost effective isolated handwritten Bangla character and digit recognition. *Pattern Recognit.* 2016, 58, 172–189. [CrossRef]
3. El-Sawy, A.; Loey, M.; El-Bakry, H. Arabic handwritten characters recognition using convolutional neural network. *WSEAS Trans. Comput.* Res. 2017, 5, 11–19.
4. Weldegebriel, H.T.; Liu, H.; Haq, A.U.; Bugingo, E.; Zhang, D. A new hybrid convolutional neural network and eXtreme gradient boosting classifier for recognizing handwritten Ethiopinan characters. *IEEE Access* 2020, 8, 17804–17818. [CrossRef]
5. Babbel Magazine. Available online: https://www.babbel.com/en/magazine/ (accessed on 18 March 2022).
6. Ghadhban, H.Q.; Othman, M.; Samsudin, N.A.; Ismail, M.N.B.; Hammoodi, M.R. Survey of offline Arabic handwriting word recognition. In *International Conference on Soft Computing and Data Mining*; Springer: Cham, Switzerland, 2020; pp. 358–372.
7. Boufenar, C.; Kerboua, A.; Batouche, M. Investigation on deep learning for off-line handwritten arabic character recognition. *Cognit. Syst. Res.* 2018, 50, 180–195. [CrossRef]
8. Palatnik de Sousa, I. Convolutional ensembles for arabic handwritten character and digit recognition. *PeerJ Comput. Sci.* 2018, 4, e167. [CrossRef]
9. Elleuch, M.; Lahiani, H.; Kherallah, M. Recognizing arabic handwritten script using support vector machine classifier. In Proceedings of the 15th International Conference on Intelligent Systems Design and Applications (ISDA), Brussels, Belgium, 22–27 June 2015; pp. 551–556. [CrossRef]
10. Mudhsh, M.; Almodfer, R. Arabic handwritten alphanumeric character recognition using very deep neural network. *Information 2017*, 8, 105. [CrossRef]
11. Niu, X.-X.; Suen, C.Y. A novel hybrid CNN-SVM classifier for recognizing handwritten digits. *Pattern Recognit.* 2012, 45, 1318–1325. [CrossRef]
12. Alwaqqi, Y.M.; Mohamad, M.; Al-Taani, A.T. Generative Adversarial Network for an Improved Arabic Handwritten Characters Recognition. *Int. J. Adv. Soft. Comput. Appl.* 2022, 14. [CrossRef]
13. Alrobah, N.; Albalhi, S. A hybrid deep model for recognizing arabic handwritten characters. *IEEE Access* 2021, 9, 87058–87069. [CrossRef]
14. Balaha, H.M.; Ali, H.A.; Badawy, M. Automatic recognition of handwritten Arabic characters: A comprehensive review. *Neural Comput. Appl.* 2021, 33, 3011–3034. [CrossRef]
15. Manwatkar, P.M.; Singh, K.R. A technical review on text recognition from images. In Proceedings of the 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, India, 9–10 January 2016; pp. 1–5.
16. Alsanousi, W.A.; Adam, I.S.; Rashwan, M.; Abdou, S. Review about off-line handwriting Arabic text recognition. *Int. J. Comput. Sci. Mob. Comput.* 2017, 6, 4–14.
17. Ding, H.; Chen, K.; Yuan, Y.; Cai, M.; Sun, L.; Liang, S.; Huo, Q. A compact CNN-DBLSTM based character model for offline handwriting recognition with Tucker decomposition. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9–15 November 2017; Volume 1, pp. 507–512.
18. Ji, S.; Zeng, Y. Handwritten character recognition based on CNN. In Proceedings of the 2021 3rd International Conference on Artificial Intelligence and Advanced Manufacturing, Manchester, UK, 23–25 October 2021; pp. 2874–2880.
19. Alwzwazy, H.A.; Albehadili, H.M.; Alwan, Y.S.; Islam, N.E. Handwritten digit recognition using convolutional neural networks. *Int. J. Innov. Res. Comput. Commun. Eng.* 2016, 4, 1101–1106.
20. Ahlawat, S.; Choudhary, A.; Nayyar, A.; Singh, S.; Yoon, B. Improved handwritten digit recognition using convolutional neural networks (CNN). *Sensors 2020*, 20, 3344. [CrossRef] [PubMed]
21. Dutta, K.; Krishnan, P.; Mathew, M.; Jawahar, C.V. Improving CNN-RNN hybrid networks for handwriting recognition. In Proceedings of the 2018 16th international conference on frontiers in handwriting recognition (ICFHR), Niagara Falls, NY, USA, 5–8 August 2018; pp. 80–85.
22. Ahmed, R.; Gogate, M.; Tahir, A.; Dashtipour, K.; Al-Tamimi, B.; Hawalah, A.; Hussain, A. Novel deep convolutional neural network-based contextual recognition of Arabic handwritten scripts. *Entropy 2021*, 23, 340. [CrossRef]
23. Gan, J.; Wang, W.; Lu, K. Compressing the CNN architecture for in-air handwritten Chinese character recognition. *Pattern Recognit. Lett.* 2020, 129, 190–197. [CrossRef]
24. He, M.; Zhang, S.; Mao, H.; Jin, L. Recognition confidence analysis of handwritten Chinese character with CNN. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 61–65.
25. Zhong, Z.; Jin, L.; Xie, Z. High performance offline handwritten Chinese character recognition using GoogLeNet and directional feature maps. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 846–850.
26. Wu, C.; Fan, W.; He, Y.; Sun, J.; Naoi, S. Handwritten Character Recognition by Alternately Trained Relaxation Convolutional Neural Network. In Proceedings of the 2014 14th International Conference on Frontiers in Handwriting Recognition (ICFHR), Crete, Greece, 1–4 September 2014; pp. 291–296.
27. Zhong, Z.; Jin, L.; Feng, Z. Multi-font printed Chinese character recognition using multipooling convolutional neural network. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 96–100.

28. Yang, W.; Jin, L.; Xie, Z.; Feng, Z. Improved deep convolutional neural network for online handwritten Chinese character recognition using domain-specific knowledge. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 551–555.

29. Alrobah, N.; Saleh, A. Arabic handwritten recognition using deep learning: A survey. Arab. J. Sci. Eng. 2022, 47, 9943–9963. [CrossRef]

30. Balaha, H.M.; Ali, H.A.; Saraya, M.; Badawy, M. A new Arabic handwritten character recognition deep learning system (AHCR-DLS). Neural Comput. Appl. 2021, 33, 6325–6367. [CrossRef]

31. Younis, K.S. Arabic handwritten character recognition based on deep convolutional neural net-works. Jordanian J. Comput. Inf. Technol. (JJCIT) 2017, 3, 186–200.

32. Najdat, H.M.; Alshboul, A.A.; Alabeled, A.F. Arabic Handwritten Characters Recognition using Convolutional Neural Network. In Proceedings of the 2019 10th International Conference on Information and Communication Systems, ICICS 2019, Irbid, Jordan, 11–13 June 2019; pp. 147–151.

33. Alyahya, H.; Ismail, M.M.B.; Al-Salman, A. Deep ensemble neural networks for recognizing isolated Arabic handwritten characters. Accent. Trans. Image Process. Comput. Vis. 2020, 6, 68–79. [CrossRef]

34. Almansari, O.A.; Hashim NN, W.N. Recognition of isolated handwritten Arabic characters. In Proceedings of the 2019 7th International Conference on Mechatronics Engineering (ICOMET), Putrajaya, Malaysia, 30–31 October 2019; pp. 1–5.

35. Daš, N.; Molla, A.F.; Saha, S.; Haque, S.S. Handwritten Arabic Numeral Recognition using a Multi Layer Perceptron. arXiv 2010, arXiv:1003.1891.

36. Elleuch, M.; Kherallah, M. Boosting of Deep Convolutional Architectures for Arabic Handwriting Recognition. Int. J. Multimed. Data Eng. Manag. 2019, 10, 26–45. [CrossRef]

37. Noubigh, Z.; Mezghani, A. Contribution on Arabic Handwriting Recognition Using Deep Neural Network. In International Conference on Hybrid Intelligent Systems; Springer: Cham, Switzerland, 2021. [CrossRef]

38. Al-Taani, A.T.; Ahmad, S.T. Recognition Of Arabic Handwritten Characters Using Residual Neural Networks. Jordanian J. Comput. Inf. Technol. (JJCIT) 2021, 7, 192–205. [CrossRef]

39. Aljarrah, M.N.; Zyout, M.M.; Duwairi, R. Arabic Handwritten Characters Recognition Using Convolutional Neural Network. In Proceedings of the 12th International Conference on Information and Communication Systems (ICICS), Valencia, Spain, 24–26 May 2021; pp. 182–188. [CrossRef]

40. Elkhayati, M.; Elkettani, Y.; Mourchid, M. Segmentation of handwritten Arabic graphemes using a directed convolutional neural network and mathematical morphology operations. Pattern Recogn. 2022, 122, 108288. [CrossRef]

41. Elleuch, M.; Tagougui, N.; Kherallah, M. Arabic handwritten characters recognition using Deep Belief Neural Networks. In Proceedings of the IEEE 12th International Multi-Conference on Systems, Signals & Devices (SSD15), Sfax, Tunisia, 16–19 March 2015; pp. 1–5. [CrossRef]

42. Kef, M.; Chergui, L.; Chikhi, S. A novel fuzzy approach for handwritten Arabic character recognition. Pattern Anal. Appl. 2016, 19, 1041–1056. [CrossRef]

43. Korichi, A.; Slatnia, S.; Aiadi, O.; Khalidi, B. A generic feature-independent pyramid multilevel model for Arabic handwriting recognition. Multimed. Tools Appl. 2022, 81, 1–21. [CrossRef]

44. Korichi, A.; Slatnia, S.; Aiadi, O.; Tagougui, N.; Kherallah, M. Arabic handwriting recognition: Between handcrafted methods and deep learning techniques. In Proceedings of the 2020 21st International Arab Conference on Information Technology (ACIT), Muscat, Oman, 21–23 December 2020; pp. 1–6.

45. Korichi, A.; Slatnia, S.; Tagougui, N.; Zouari, R.; Kherallah, M.; Aiadi, O. Recognizing Arabic Handwritten Literal Amount Using Convolutional Neural Networks. In Proceedings of the International Conference on Artificial Intelligence and its Applications, Vienna, Austria, 29–30 October 2022; pp. 153–165.

46. Albahli, S.; Nawaz, M.; Javed, A.; Irtaza, A. An improved faster-RCNN model for handwritten character recognition. Arab. J. Sci. Eng. 2021, 46, 8509–8523. [CrossRef]

47. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A.V.; Gulin, A. CatBoost: Unbiased boosting with categorical features. Adv. Neural Inf. Process. Syst. 2018, 31.

48. Yang, J.; Ma, J. Feed-forward neural network training using sparse representation. Expert Syst. Appl. 2019, 116, 255–264. [CrossRef]

49. LeCun, Y.; Cortes, C.; Burges, C.J. MNIST Handwritten Digit Database. 2010, 18. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 15 August 2022).

50. Shams, M.; Elsonbaty, A.; ElSawy, W. Arabic handwritten character recognition based on convolution neural networks and support vector machine. Int. J. Adv. Comput. Sci. Appl. 2020, 11, 144–149. [CrossRef]