information

MDPI

Article

Arabic Handwritten Alphanumeric Character Recognition Using Very Deep Neural Network

MohammedAli Mudhsh and Rolla Almodfer *

School of Computer Science, Wuhan University of Technology, Luo Shi Road, Wuhan 430070, China; mudhish@whut.edu.cn

* Correspondence: rollajamil@whut.edu.cn

Received: 5 August 2017; Accepted: 25 August 2017; Published: 31 August 2017

Abstract: The traditional algorithms for recognizing handwritten alphanumeric characters are dependent on hand-designed features. In recent days, deep learning techniques have brought about new breakthrough technology for pattern recognition applications, especially for handwritten recognition. However, deeper networks are needed to deliver state-of-the-art results in this area. In this paper, inspired by the success of the very deep state-of-the-art VGGNet, we propose *Alphanumeric VGG* net for Arabic handwritten alphanumeric character recognition. *Alphanumeric VGG* net is constructed by thirteen convolutional layers, two max-pooling layers, and three fully-connected layers. The proposed model is fast and reliable, which improves the classification performance. Besides, this model has also reduced the overall complexity of VGGNet. We evaluated our approach on two benchmarking databases. We have achieved very promising results, with a validation accuracy of 99.66% for the ADBase database and 97.32% for the HACDB database.

Keywords: alphanumeric recognition; Arabic handwritten; deep learning; VGGNet; dropout; augmentation

1. Introduction

In the last few decades, Arabic handwritten alphanumeric character recognition has become one of the challenging areas of research in the field of document image processing. While the recognition of handwritten Latin has been extensively investigated using various techniques, little work has been done on handwritten Arabic recognition, and none of the existing techniques are accurate enough for practical application. Recognizing an Arabic character or text is a complicated task due to the unlimited variation in human handwriting, the large variety of Arabic character shapes, the presence of ligature between characters, and overlapping of the components. The main distinction between Arabic and other Roman-based languages is that Arabic words and characters within words are written from right to left, as opposed to English words which are written from left to right. Nevertheless, the digits of an Arabic number are written from left to right [1].

The existing approaches to recognizing handwriting can be divided into two groups: the handcrafted approach and the unsupervised/supervised learning approach. For the first group, the most commonly used methods are scale-invariant feature transforms (SIFT) [2–4] and Gabor features [5,6]. The second group is known as the group of deep learning approaches. This approach has acquired a reputation for solving many computer vision problems, and its application to the field of handwriting recognition has been shown to provide significantly better results than traditional methods [7]. Reviewing the literature, several methods have been proposed for the recognition of Arabic handwritten digits, characters, and words. For example, Zaiz et al. [8] proposed a technique that uses a Support Vector Machine (SVM) classifier to recognize Arabic handwritten words in the IFN/ENIT database [9] based on two passes, horizontal and vertical. Then, a post-processor based on a Puzzle algorithm applied to improve the recognition rate, especially for ambiguous characters. Amrouch et al. [10] present a work that aims to compare learning features with Convolutional Neural Networks (CNN) and handcrafted features. Experiments have been performed on the benchmark IFN/ENIT database. The authors conclude that the obtained results with the CNN features surpass those achieved using the handcrafted features. Maalej et al. [11] proposed a system for Online Arabic Handwriting Recognition using Deep Bidirectional Long Short-Term Memory (DBLSTM). The system was tested on the ADAB database [12]. The result achieved by the proposed system exceeds 99.98%.

Considering the recognition of Arabic handwritten digits, Al-Omari and Al-Jarrah [13] presented a recognition system for the online handwritten Arabic digits one to nine. The system skeletonizes the digits, and the geometrical features of the digits are extracted. Probabilistic neural networks (PNNs) are used for recognition. The developed system is translation, rotation, and scaling invariant. Abdelazeem [14] studied the performance of a different set of classifiers for Arabic digit recognition. Different features were used, and different combinations of features and classifiers were analyzed. Gradient features with an SVM gave the best results of 99.48% for the ADBase database. Parvez and Mahmoud [15] utilized a polygonal approximation of the contour of a character and a classifier based on turning functions for Arabic alphanumeric character recognition. The authors obtained 97.17% accuracy for the ADBase database of Arabic digits.

Many researchers have tried to use a deep learning approach for handwritten Arabic characters. In this vein, a Deep Belief Network (DBN) was used to extract image features for raw data inputs, and proceed with a greedy layer-wise unsupervised learning algorithm for recognizing offline Arabic characters in [7]. The experimental results achieved a recognition accuracy equal to 96.36% when applied to the Handwritten Arabic Characters Database (HACDB) [16] with 66 class labels. Elleuch et al. [17] suggested an offline Arabic handwritten character recognition system using a Convolutional Neural Network (CNN) for extracting features information from the raw images and a Support Vector Machine (SVM) functions as the recognizer. This system was evaluated using the HACDB database, and the result was an accuracy of 94.17%. In [18], the authors presented two models for the recognition of handwritten Arabic characters. The first is based on deep learning. The second model focuses on handcrafted feature extraction. These two models were tested on the HACDB database; the result for the first was 91.36% and the result for the second was 87.60%. Coefficients of Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT) techniques were used with Neural Networks (NN) and the Hidden Markov Model (HMM) for the classification of the shapes of characters by Lawgali in [19]. The experiments were applied on the HACDB database. The results have demonstrated that the DCT yields the higher recognition rate (equal to 75.31%). In spite of the previous works, using Deep architectures is relatively scarce on Arabic handwriting character recognition comparative to other languages. Deep Neural Networks (DNN) have brought about new breakthrough technology for Latin and Handwritten Chinese Character Recognition (HCCR) with great success. For example, the HCCR-GoogLeNet is designed very deeply yet slim, with a total of 19 layers. Experiments on the ICDAR 2013 offline HCCR competition database show that a single HCCR-GoogLeNet is superior regarding both accuracy and storage performance [20].

In 2014, Simonyan and Zissermaanother presented VGGNet [21]. VGGNet is a very deep architecture that has achieved a high classification accuracy of the massive Imagenet database [22]. Nevertheless, this network has a high number of parameters compared to GoogLeNet, which makes it computationally more expensive to evaluate and requires a significant amount of memory for optimizing the learning parameters.

In this paper, inspired by the success of VGGNet, we propose *Alphanumeric VGG* net for Arabic handwritten alphanumeric recognition. *Alphanumeric VGG* net is straightforward to implement and shows effectiveness in improving classification performance. Moreover, it reduces the overall complexity of *VGGNet* while keeping the same good performance of the net. We experiment using the Mean Square Error (MSE) function and cross-entropy error (CEE) function. To prevent *Alphanumeric VGG* net from overfitting, two regularization methods are adopted; namely, *dropout* and *augmentation*. We provide an in-depth performance evaluation of these two critical factors. Because of the lack of one alphanumeric benchmark database, we conducted our experiments on two different databases: the ADBase database (a database of Arabic handwritten digits from 0 to 9) and the HACDB database (a database of handwritten Arabic characters). After several experiments and parameters adjustments, we achieved two state-of-the-art recognition accuracies equal to 99.57% for the ADBase database and 97.32% for the HACDB database.

The remainder of the paper consists of the following. Section 2 introduces Arabic handwriting characteristics and challenges. Section 3 presents the standard VGGNet and our proposed *Alphanumeric VGG* net architectures. Regularization methods are also addressed in this section. We demonstrate the results and analyze the architecture's performance in Section 4. Finally, conclusions are drawn in Section 5.

2. Arabic Handwriting Characteristics and Challenges

Arabic script is cursive by nature, which makes its recognition very challenging. Arabic is composed of 28 main characters and is written from right to left in a cursive manner. Each character has two or four shapes depending on its position in the word, resulting in a total of 81 shapes. The shape of some of the characters is similar, but the number and position of dots may change. There are 52 basic character shapes without dots. Some challenging structural characteristics of the characters in the database are described below:

1. The writing style is different for each writer. The same character can be written in various shapes as in Figure 1a.
2. Certain characters are very similar in shape and are referred to as preplexing characters; Figure 1b shows a representative set of pairs of preplexing characters.
3. Different characters have very similar shapes with a slight modification, as in Figure 1c.

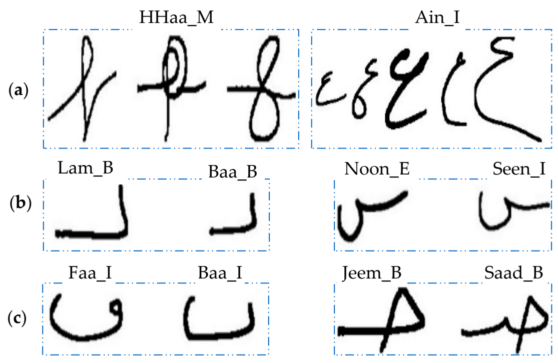


Figure 1. Illustration of some challenges of Arabic characters. (**a**) An isolated Ain has a different size and shape depending on the writer's style. Additionally, the Middle HHAA is written in three different shapes; (**b**) End Noon and isolated Seen have very similar shapes; (**c**) Beginning Saad has a small stroke differentiating it from Beginning Jeem; isolated Faa with a small number of points as a loop distinguishes it from isolated Baa.

3. Method

In this section, we briefly summarize the standard VGGNet. We then describe our proposed *Alphanumeric VGG* net for offline handwriting. An overview of the deep network overfitting problem is also introduced.

3.1. VGGNet

The VGGNet [21] created by Simonyan and Zisserman presents a very deep, very simple, and homogeneous architecture as shown in Figure 2. The aim of building VGGNet is to investigate how the Network depth affects the accuracy in a large-scale image recognition setting. In VGGNet, a given image is passed through a stack of convolutional layers; the filters are generally with size: 3×3. The stride and the spatial padding are both fixed to one pixel. The width of the convolutional layers starts from 64 to 512. The max-pooling layers are followed by some of the convolutional layers. Max pooling is performed over a 2×2 pixel window, with stride 2. Finally, three Fully-Connected (FC) layers follow the stack of convolutional layers: the first two have 4096 channels each, and the third contains 1000 channels. The final layer is the softmax layer. All convolutional layers are equipped with the rectified functions (Relu), which is formulated by:

$$f(x) = \max(0, x) \tag{1}$$

where x denotes a feature value produced over the former layer. Since our proposed model is inspired by VGG_16, we will compare the configuration of VGG_16 with our proposed model in Table 1.

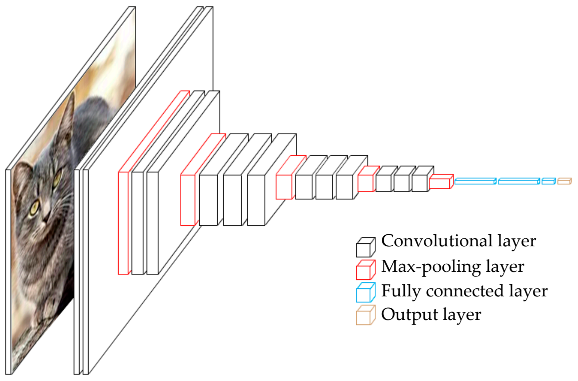


Figure 2. The Architecture of VGGNet.

Table 1. Comparison between VGG_16 and *Alphanumeric VGG* net Configurations.

VGG_16	*Alphanumeric VGG_66*	*Alphanumeric VGG_10*
Input 224 × 224 RGB image	Input 28 × 28 binary image	
Convolutional 3_64 Convolutional 3_64	Convolutional 3_8 Convolutional 3_8	Convolutional 3_8 Convolutional 3_8
Maxpooling		
Convolutional 3_128 Convolutional 3_128	Convolutional 3_16 Convolutional 3_16	Convolutional 3_16 Convolutional 3_16
Maxpooling		
Convolutional 3_256 Convolutional 3_256 Convolutional 3_256	Convolutional 3_32 Convolutional 3_32 Convolutional 3_32	Convolutional 3_32 Convolutional 3_32 Convolutional 3_32
Maxpooling		
Convolutional 3_512 Convolutional 3_512 Convolutional 3_512	Convolutional 3_64 Convolutional 3_64 Convolutional 3_64	Convolutional 3_64 Convolutional 3_64 Convolutional 3_64
	Maxpooling	
Convolutional 3_512 Convolutional 3_512 Convolutional 3_512	Convolutional 3_64 Convolutional 3_64 Convolutional 3_64	Convolutional 3_64 Convolutional 3_64 Convolutional 3_64
	Maxpooling	
FC-4096 FC-4096 FC-1000	FC-512 FC-512 FC-66	FC-512 FC-512 FC-10

RGB: Red, Green and Blue color model; FC: fully connected.

3.2. Alphanumeric VGG

In this section, we introduce our proposed architecture that improves the recognition performance of the given databases. The architecture of *Alphanumeric VGG* net is shown in Figure 3. *Alphanumeric VGG* net follows the standard model of VGGNet, which contains 16 layers with some differences. The filters numbers are divided by factor of 8 throughout the network. The filters numbers in a convolutional layer are set to 8, 16, 32, and 64, and the first two fully connected layers are set to 512. Since the original VGGNet was trained on 1000 classes, its last fully connected layer produces 1000 outputs. We replace this layer with a new fully connected layer that has as many outputs as the number of classes (10 for the ADBase database and 66 for HACDB databases). The final layer is the soft-max layer. The network's configurations, evaluated in this paper, are outlined in Table 1, one per column.

The important issue in *Alphanumeric VGG* net design is the selection of input image size. When the image size is set to 28 × 28, the complexity of the network is very low. However, there are several drawbacks, which cannot be ignored using this setting. In one hand, the max -pooling layers have significance to the performance of a deep network. Max pooling partitions the input image into a set of non-overlapping rectangles, and, for each such sub-region, outputs the maximum value. It leads to a faster convergence rate by selecting superior invariant features, which improve generalization performance. When the image size is 28 × 28, the number of max -pooling layers is hard to determine. However, after several experiments, we kept the fourth and the fifth max -pooling layers and discarded the first three. On the other hand, if we want to remove all the max -pooling layers, the deep model size will be very large since the number of weights in the fully connected layers increases significantly. The number of parameters of *Alphanumeric VGG* net is very low compared to the number of parameters of standard VGG_16 when applied to binary images. In Table 2, we report the number of parameters for each network configuration. In spite of a large depth, the number of parameters in our network is around 2 million while in VGG_16 it is higher than 138 million.

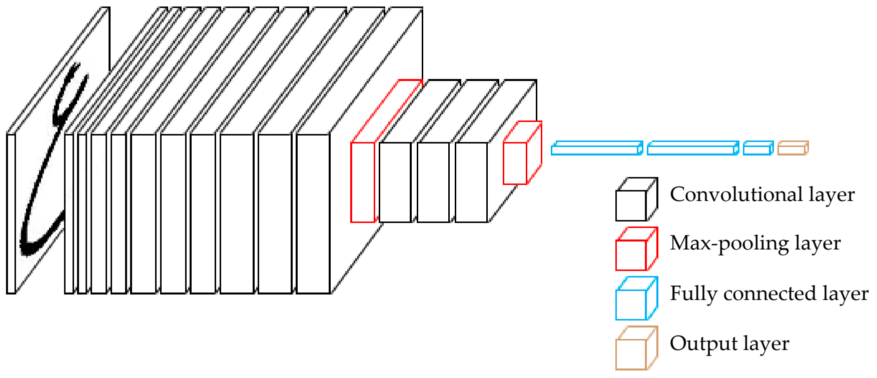


Figure 3. The Architecture of *alphanumeric VGG*.

Table 2. Number of parameters.

Network	VGG_16	Alphanumeric VGG-66	Alphanumeric VGG-10
Number of parameters	138,357,544	2,133,082	2,104,354

3.3. Overfitting in Deep Network

Training a deep neural network is mainly dependent on the availability of large quantities of training data, which are important to have the model learn a nonlinear function from input to output that generalizes well and yields a high classification accuracy on unseen data. Not having enough quality data will generate overfitting; that is, the network is highly biased to the data it has seen during training, and, therefore, the network cannot generalize the learned model to any other sample data [21]. The elegant solutions to this problem are *dropout data* and *augmentation*.

Dropout refers to "dropping out units" (representing both hidden and visible units in the deep network). Dropped-out neurons do not contribute to the forward pass and backpropagation. Dropping a unit out means temporarily removing it from the network with all its incoming and outgoing connections. Dropout regularization reduces the complex co-adaptations of neurons, as a neuron cannot rely on the existence of other neurons. Therefore, the neurons are forced to learn the robust features that are effective in conjunction with the other neurons [23]. Dropout has been reported to have achieved success on several benchmark databases [24]. At test time, all the neurons are used, but their outputs are multiplied by the dropout value, which is a reasonable approximation to taking the geometric mean of the predictive distributions produced by the exponentially many dropout networks [25].

Data augmentation: The other widely used strategy to avoid overfitting and improve generalization performance is called augmentation. Data augmentation is the process of expanding a database with same data created from the information in that database [26]. Data augmentation often includes the application of blurring, rotation, and translation to existing images that allow a network to generalize better. However, not all data augmentation methods can improve performance, and should not be used "blindly" [27]. For example, on MNIST (a Latin handwritten number database), if rotation is used, the network will be unable to distinguish accurately between handwritten "6" and "9" digits [21]. Likewise, on ADBase (the Arabic handwritten number database), if one adds rotation, the network will be unable to distinguish between handwritten "٢" and "٨" digits, and between "٦" and "٧" as shown in Figure 4. If flipping is used, the network would be unable to distinguish between handwritten "٧" and "٨". On HACDB (the Arabic handwritten characters database), the network may become incapable of discerning many characters if the training set was augmented by the indiscriminate flipping or rotating of images. Figure 4 shows some characters that can be badly affected if augmentation is applied blindly. Figure 4a shows, for example, the character Dal_Isolated (د) can be the same as the character Noon_Isolated (ں) when rotation is applied. From Figure 4b, if flipping used, the network would be unable to distinguish properly between handwritten Lam_beginning (لـ) and Alef_End (ـا) and between handwritten Hamza (ء) and Jeem_Isolated (ح) [18]. However, the main data augmentation employed in this work is for the HACDB database, which contains 6600 shapes of handwritten characters.

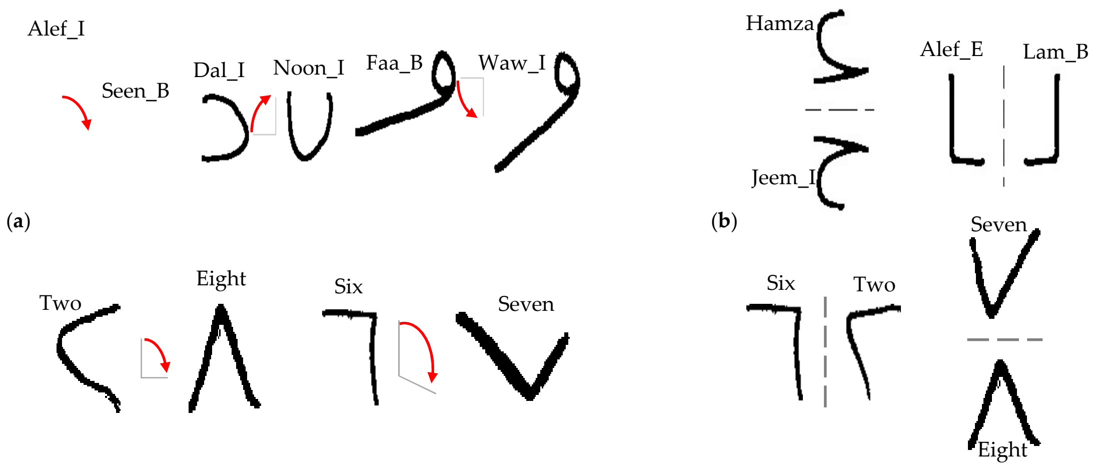


Figure 4. Arabic characters and digits that can be badly affected by rotation and flipping (**a**) Confusion caused by rotation effect; (**b**) Confusion caused by flipping.

3.4. Error Criteria

Each error criteria leads to one of the objective functions that guides the training process. Among them, the Mean Square Error (MSE) is a popular preference. If $D = (d_1, d_2, \ldots, d_c)$ is the target value for the training sample, MSE is defined in Equation (2), where X_c^N is the actual network value for the Cth class, and C is the number of classes [28].

$$E_{MSE} = \frac{1}{2}\frac{\sum_c \left(X_c^N - d_c\right)^2}{C} \tag{2}$$

The MSE error function enables the network to have good performance but slow convergence to the final outcome. To accelerate the backpropagation algorithm, a Cross-Entropy Error (CEE) function is an alternative. When CEE is used in conjunction with the softmax activation function, the CEE has the form shown in Equation (3).

$$E_{CE} = -\sum_c d_c log(X_c^N) \tag{3}$$

CEE maximises the relative size of the true class output regarding the outputs of other class nodes, while MSE minimizes the absolute error at each output node. CEE works well with the softmax activation function. Given the softmax function, the desired output D contains a one for the true class and zeroes for the other classes [29]. The gradient of CEE with regard to the output is computed in Equation (4).

$$\frac{\partial E_{CE}}{\partial X_c^N} = -\sum_c d_c \frac{1}{X_c^N} \tag{4}$$

The error signal propagating back from each output unit becomes directly proportional to the difference between the target value and actual value. This can lead to better network performance with a shorter stagnation period [30].

4. Experimental Results and Performance Analysis

We utilize two databases for our experimentations: a database of Arabic handwritten numerals, and a database of Arabic handwritten characters. With this set of experiments, we experiment on the effects of error criteria, dropout, and data augmentation on the overall classifier performance.

4.1. Training Method

In this study, for each database, we perform an experiment with two main parts: the first part uses the CEE function, and the second part uses the MSE function. For CEE, we use the Adam optimizer to minimize the categorical cross entropy. Adam is a first-order gradient-based algorithm developed for the optimization of stochastic objective functions with adaptive weight updates based on lower-order moments. The Adam optimizer has four parameters: the learning rate, the exponential decay rates (beta_1) for the moving averages of the gradient, the squared gradient (beta_2), and the smoothing term (epsilon). After related experiments, we left the parameters to their default values, with the learning rate equal to 0.001, the decay rates equal to 0.9, the squared gradient equal to 0.999, and the smoothing term equal to 1×10^{-8}. For MSE, we use the RMSprop optimizer to minimize the Minimum Square Error, as it was found through experiment that the Adam optimizer was hard to train with the MSE function. RMSprop is a gradient-based optimization technique that normally has three parameters: the learning rate, the decay rate (alpha), and the smoothing term (epsilon). We left the parameters to their default values, with the learning rate equal to 0.001, the decay rates equal to 0.9, and the smoothing term equal to 1×10^{-8}.

The dropout regularization for the first two fully connected layers was set to a dropout ratio of 0.5. The type of nonlinearity used was Rectified Linear Unit (ReLU). *Alphanumeric VGG* net was trained for 100 epochs. The training procedure took 2 hours on a desktop personal computer (PC) with an Intel i7 3770 processor, an NVidia GTX780 graphics card, and 16 gigabytes of onboard random access memory (RAM). VGGNet was trained on four NVidia Titan Black graphical processing units (GPUs) for two to three weeks.

4.2. Overfitting

Dropout: The first two fully connected layers in standard VGGNet are regularized by a 0.5 dropout ratio. Dropout means setting to zero the output of each hidden neuron with probability of 0.5. If the neurons in the CNN are dropped out, they do not contribute to the forward pass and do not participate in backpropagation. During testing, we use all the neurons but multiply their outputs by 0.5. However, *Alphanumeric VGG* net suffered from overfitting even when it dropped out values for the first two fully connected layers. To prevent *Alphanumeric VGG* net from overfitting, the training was regularized by dropout regularization for the first two fully connected layers and the two max-pooling layers. The dropout ratio was set to 0.5. *The Alphanumeric VGG* net with dropout regularization is depicted in Figure 5.

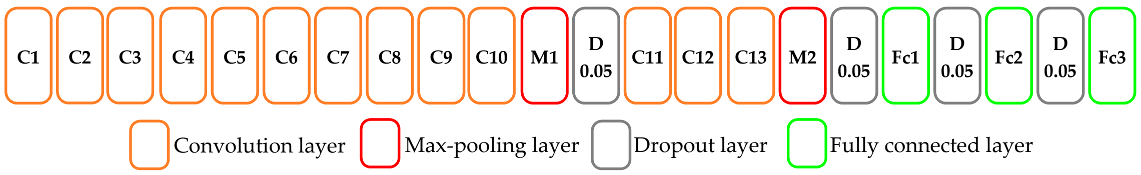


Figure 5. *Alphanumeric VGG* net with dropout.

Data augmentation: For the HACDB database that has 6600 images for 66 classes, if we do not adopt data augmentation, our network will suffer from overfitting. By doing augmentation, for each original character image in the database, we can generate ten samples for each image; therefore, the number of possible training samples would be 66,000. The augmentation improves the results significantly.

4.3. ADBase Database

The ADBase contains 70,000 digits written by 700 writers. Each writer wrote each digit (from "0" to "9") ten times. The database is partitioned into two sets: a training set (which contains 60,000 digits: 6000 images for each class) and a test set (which contains 10,000 digits: 1000 images for each class). Sample images of digits (0–9) from the ADBase database are shown in Table 3. The ADBase database is available on the website [31]. We want to mention here that no data augmentation applied to this database, as it includes enough samples to achieve a high result.

Table 3. Sample images of digits (0–9) from the ADBase database.

Arabic digit	١	٢	٣	٤	٥	٦	٧	٨	٩	٠
English digit	1	2	3	4	5	6	7	8	9	0
Image										

4.3.1. The Impact of CEE Function

In this subsection, we will compare the results obtained without adopting dropout and after adopting dropout regularization. The experiments were conducted using the CEE function. The experimental results are presented in Table 4. Without dropout, we achieved a classification accuracy equal to 98.75% on the validation set that does not hold on the test set. After adopting dropout, we achieved a classification accuracy equal to 99.57%. Figure 6a shows the improvement in the performance of *Alphanumeric VGG* net when using the dropout regularization method (the green and the orange lines).

4.3.2. The Impact of MSE Function

From Table 4, without dropout, we achieved a classification accuracy equal to 98.83% on the validation set that does not hold on the test set. With dropout, we achieved classification accuracy equal to 99.66%. The experiments were conducted using the MSE function. The CEE function was slightly better than MSE when we trained the model without dropout. However, using dropout, the MSE function showed better results. Figure 6b shows the improvement in the performance of *Alphanumeric VGG* net when using dropout regularization method (the green and the orange lines).

Table 4. Experimental results on the ADBase database. CCE: cross-entropy error; MSE: mean square error.

Learning Method	No Dropout	Dropout
CCE/adam	98.83%	99.57%
MSE/RMSprop	98.75%	99.66%

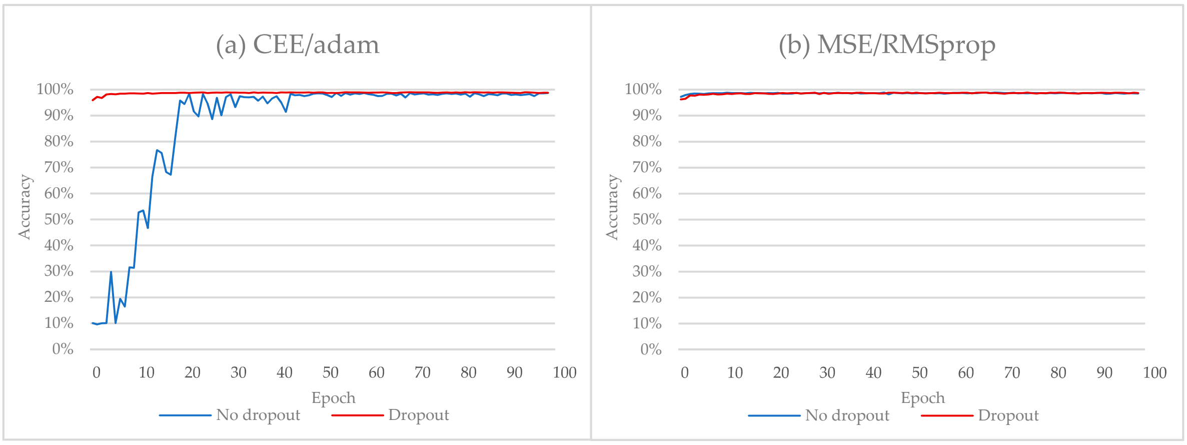


Figure 6. The performance of *Alphanumeric VGG* net over the ADBase database using different error criteria. (**a**) CEE/adam; (**b**) MSE/RMSprop.

4.3.3. Comparison with the State-of-the-Art

Unlike Latin, the task of Arabic handwritten digits recognition suffers from the lack of a benchmarking database. To the best of our knowledge, this is the first work to incorporate a deep learning approach for recognizing ADBase database digits. Table 5 compares our best results with two previous works on the ADBase database. Particularly, our best performance is noticeably higher than the result achieved using a Fuzzy Turning Function [8] and the result achieved using an SVM [7]. We achieved state-of-the-art results for the ADBase database.

Table 5. Comparison with state-of-the-art methods on the ADBase database.

Author(s)	Method	Accuracy
Proposed	*Alphanumeric VGG/MSE*	99.66%
Proposed	*Alphanumeric VGG/CEE*	99.57%
Abdelazeem et al. [14]	SVM with RBF kernel	99.48%
Parvez et al. [15]	Fuzzy Turning Function	97.17%

SVM: support vector machine; RBF: radial basis function kernel.

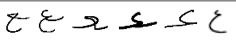
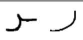
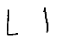
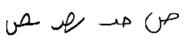
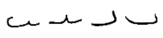
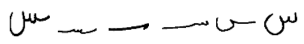
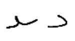
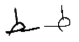
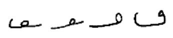
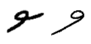
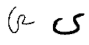
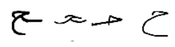
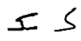
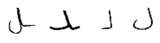
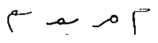
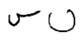
4.4. HACDB Database

The HACDB database [16] (shown in Table 6) contains the 52 basic shapes of characters, 6 different styles for only certain characters, and 8 shapes of overlapping characters for a total of 66 shapes of Arabic characters written by 50 people. Each person wrote each character twice. The number of character shapes collected totaled 6600 shapes of unconstrained handwritten Arabic characters.

4.4.1. The Impact of MSE Function

The experiment results of using MSE function are presented in Table 7. When no dropout and no data augmentation is adopted, *Alphanumeric VGG* net had a classification accuracy of 90.61% on the validation set that does not hold on the test set. After adopting dropout, the classification accuracy improved slightly to 92.42%. We repeated the previous experiment, this time augmenting the original data 10-fold with each image. We see that data augmentation increases the validation set accuracy dramatically, i.e., from 90.61% to 95.95%. Figure 7a shows the improvement in the performance (the blue line) of *Alphanumeric VGG* net. However, we obtained the first state-of-the-art result, which is equal to 96.58%, when adopting the dropout and data augmentation methods together. Figure 7a shows the improvement in the performance (the red line) for *Alphanumeric VGG* using the MSE function.

Table 6. HACDB database: 66 classes of individual shapes of Arabic handwritten characters, 16 classes of basic characters, and eight classes of overlapping characters.

Character	Shape	Character	Shape
Ain (ع)		Raa (ر)	
Alef (ا)		Saad (ص)	
Baa (ب)		Seen (س)	
Dal (د)		TTaa (ط)	
Faa (ف)		Waw (و)	
HHaa (ه)		Yaa (ي)	
Hamza (ء)		Alef_Lam_Jemm (الجـ)	
Jeem (ج)		Lam_Alef (لا)	
Kaf (ك)		Lam_Jeem (لجـ)	
Lam (ل)		Lam_Meem (لمـ)	
Meem (م)		Meem_Jeem (مجـ)	
Noon (ن)		Lam_Meem_Jeem (لمجـ)	

4.4.2. The Impact of CEE Function

In this set of experiments, the CEE function outperformed the MSE function in all cases. The results are tableted in Table 7, and the details are as follows:

Table 7. Experimental results on the HACDB database.

Learning Method	No Dropout and No Augmentation	Dropout	Augmentation	Dropout and Augmentation
CCE/adam	91.97%	93.48%	96.42%	97.32%
MSE/RMSprop	90.61%	92.42%	95.95%	96.58%

Without dropout and data augmentation, *Alphanumeric VGG* net had a classification accuracy of 91.97% on the validation set that does not hold on the test set. After applying dropout, the classification accuracy improved slightly to 93.48%. Figure 7 shows the improvement in the performance of *Alphanumeric VGG* net when using the dropout regularization method (the green and the orange lines).

We repeated the previous experiment, this time with augmented data. We observe an improvement in classification accuracy from 91.97% to 96.42%. However, we got the second state-of-the-art result, which is equal to 97.32%, when we adopted data augmentation and dropout together. Figure 7b shows the overall improvement in the performance (the red line) of *Alphanumeric VGG* net when using the dropout and data augmentation methods together.

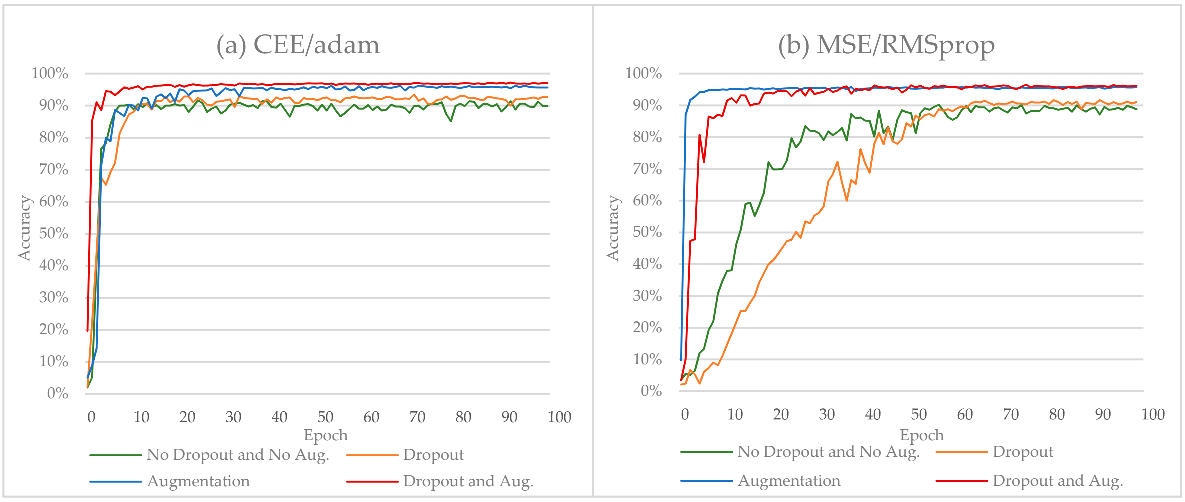


Figure 7. The performance of *Alphanumeric VGG* net over HACDB database using different error criteria. (**a**) CEE/adam; (**b**) MSE/RMSprop.

4.4.3. Comparison with the State-of-the-Art

The existing Arabic handwritten character recognition methods can be categorized into two groups: handcrafted-based methods and deep learning-based methods. In this subsection, to evaluate the effect of our proposed deep model, we compared the performance of the model with those of the deep learning-based methods. Table 8 compares our two state-of-the-art results with the previous best works on the HACDB database. According to these results, our *Alphanumeric VGG* net outperforms other deep learning methods, such as Deep Believe Network (DBN), and obtains the best-published results on the HACDB database to date. Very recently, Elleuch et al. [7] obtained a good result on the HACDB database. However, this result is obtained by using ensemble methods, such as Convolutional Neural Network (CNN) and Support Vector Machine (SVM). In contrast, our model is straightforward and generic to apply, so it may also work well with the handwritten characters of other languages, such as Latin and Chinese.

Table 8. Comparison with state-of-the-art methods on the HACDB database.

Authors	Method	Accuracy
Proposed	*Alphanumeric VGG/CEE*	97.32%
Proposed	*Alphanumeric VGG/MSE*	96.58%
Elleuch et al. [7]	DBN	96.36%
Elleuch et al. [17]	CNN + SVM	94.17%
Elleuch et al. [18]	Deep SVM	91.36%
Lawgali [19]	DCT + ANN	75.31%

DBN: Deep Believe Network; DCT: Discrete Cosine Transform; ANN: Artificial Neural Network.

5. Conclusions

In this paper, we proposed *Alphanumeric VGG* net for the Arabic handwritten alphanumeric (character/digit) recognition task. *Alphanumeric VGG* net is an optimized version of the very popular VGGNet. We show incremental improvements of the alphanumeric recognition comparable to approaches that use Deep Belief Network (DBN) or Recurrent Neural Network (RNN). *Alphanumeric VGG* network improved the classification accuracy and reduced the overall complexity of VGGNet by a factor of 8. With different network parameters, dropout, and augmentation we improved the overall performance as follows:

1. Regarding the ADBase database: without dropout, we achieved classification accuracies equal to 98.83% using the MSE function and 98.75% using the CEE function on the validation set that does not hold on the test set. With dropout, we achieved classification accuracies equal to 99.57% and 99.66 using CEE and MSE, respectively. The MSE function achieved better results over the CEE function.
2. Regarding the HACDB database: without dropout and augmentation, we achieved 90.61% and 91.97% classification accuracy using MSE and CEE, respectively, on the validation set that does not hold on the test set. With dropout, the classification accuracy improved slightly to 92.42% and 93.48% using MSE and CEE, respectively. We repeated the previous experiment, this time augmenting the original data 10-fold with each image. We see that data augmentation increases the validation set accuracy dramatically, i.e., from 90.61% to 95.95% using the MSE function and from 91.97% to 96.42% using CEE. However, we got the two state-of-the-art results when we applied data augmentation and dropout together. The two state-of-the-art results are 96.58% using the MSE function and 97.32% using the CEE function.

Our best results might not be statistically significant compared to the previous state-of-the-art; however, our model is quite simple and generic to apply, so it may also work well with the isolated handwritten characters of other languages. As for future work, we plan to discover the performance of other deep networks like AlexNet, GoogLeNet, and ResNet on the two databases. Moreover, we will experiment with the applicability of VGGNet on the IFN/ENIT Arabic handwritten words database.

Acknowledgments: Wuhan University of Technology, School of Computer Science and Technology for supporting this research.

Author Contributions: MohammedAli Mudhsh proposed the model, design the experiment and write the first draft; Rolla Almodfer write the first draft and gave the technichal support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ashiquzzaman, A.; Tushar, A.K. Handwritten Arabic numeral recognition using deep learning neural networks. In Proceedings of the 2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Dhaka, Bangladesh, 13–14 February 2017; pp. 1–4.
2. Chergui, L.; Kef, M. SIFT descriptors for Arabic handwriting recognition. *Int. J. Comput. Vis. Robot.* **2015**, *5*, 441–461. [CrossRef]
3. Assayony, M.O.; Mahmoud, S.A. An Enhanced Bag-of-Features Framework for Arabic Handwritten Sub-words and Digits Recognition. *J. Pattern Recognit. Intell. Syst.* **2016**, *4*, 27–38.
4. Tharwat, A.; Gaber, T.; Hassanien, A.E.; Shahin, M.; Refaat, B. Sift-based arabic sign language recognition system. In *Afro-european Conference for Industrial Advancement*; Springer: Berlin, Germany, 2015; pp. 359–370.
5. Chen, J.; Cao, H.; Prasad, R.; Bhardwaj, A.; Natarajan, P. Gabor features for offline Arabic handwriting recognition. In Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, Cambridge, MA, USA, 9–11 June 2010; pp. 53–58.
6. Elzobi, M.; Al-Hamadi, A.; Saeed, A.; Dings, L. Arabic handwriting recognition using gabor wavelet transform and SVM. In Proceedings of the 2012 IEEE 11th International Conference on Signal Processing (ICSP), Beijing, China, 21–25 October 2012; pp. 2154–2158.

7. Elleuch, M.; Tagougui, N.; Kherallah, M. Towards Unsupervised Learning for Arabic Handwritten Recognition Using Deep Architectures. In Proceedings of the International Conference on Neural Infomation Processing, Istanbul, Turkey, 9–12 November 2015; pp. 363–372.
8. Zaiz, F.; Babahenini, M.C.; Djeffal, A. Puzzle based system for improving Arabic handwriting recognition. *Eng. Appl. Artif. Intell.* **2016**, *56*, 222–229. [CrossRef]
9. Pechwitz, M.; Maddouri, S.S.; Märgner, V.; Ellouze, N.; Amiri, H. IFN/ENIT-database of handwritten Arabic words. In Proceedings of the 7th Colloque International Francophone sur l'Ecrit et le Document (CIFED), Hammamet, Tunis, 21–23 October 2002; pp. 127–136.
10. Amrouch, M.; Rabi, M. The Relevant CNNs Features Based HMM for Arabic Handwriting Recognition. *Int. J. Imaging Robot.* **2017**, *17*, 98–109.
11. Maalej, R.; Tagougui, N.; Kherallah, M. Online Arabic handwriting recognition with dropout applied in deep recurrent neural networks. In Proceedings of the 2016 IEEE 12th IAPR Workshop on Document Analysis Systems (DAS), Santorini, Greece, 11–14 April 2016; pp. 417–421.
12. Boubaker, H.; Elbaati, A.; Tagougui, N.; El Abed, H.; Kherallah, M.; Alimi, A.M. Online Arabic databases and applications. In *Guide to OCR for Arabic Scripts*; Springer: Berlin, Germany, 2012; pp. 541–557.
13. Al-Omari, F.A.; Al-Jarrah, O. Handwritten Indian numerals recognition system using probabilistic neural networks. *Adv. Eng. Inform.* **2004**, *18*, 9–16. [CrossRef]
14. Abdleazeem, S.; El-Sherif, E. Arabic handwritten digit recognition. *Int. J. Doc. Anal. Recognit.* **2008**, *11*, 127–141. [CrossRef]
15. Parvez, M.T.; Mahmoud, S.A. Arabic handwritten alphanumeric character recognition using fuzzy attributed turning functions. In Proceedings of the Workshop in Frontiers in Arabic Handwriting Recognition, 20th International Conference in Pattern Recognition (ICPR), Istanbul, Turkey, 22–26 August 2010.
16. Lawgali, A.; Angelova, M.; Bouridane, A. HACDB: Handwritten Arabic characters database for automatic character recognition. In Proceedings of the 4th European Workshop on Visual Information Processing, Paris, France, 10–12 June 2013; pp. 255–259.
17. Elleuch, M.; Maalej, R.; Kherallah, M. A New Design Based-SVM of the CNN Classifier Architecture with Dropout for Offline Arabic Handwritten Recognition. *Procedia Comput. Sci.* **2016**, *80*, 1712–1723. [CrossRef]
18. Elleuch, M.; Kherallah, M. An Improved Arabic Handwritten Recognition System using Deep Support Vector Machines. *Int. J. Multimed. Data Eng. Manag.* **2016**, *7*, 1–20. [CrossRef]
19. Lawgali, A. An Evaluation of Methods for Arabic Character Recognition. *Int. J. Signal Process. Image Process. Pattern Recognit.* **2014**, *7*, 211–220. [CrossRef]
20. Zhong, Z.; Jin, L.; Xie, Z. High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In Proceedings of the 2015 IEEE 13th International Conference on Document Analysis and Recognition (ICDAR), Nancy, France, 23–26 August 2015; pp. 846–850.
21. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014.
22. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database, Computer Vision and Pattern Recognition. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Miami, FL, USA, 20–25 June 2009; pp. 248–255.
23. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in neural information processing systems, Lake Tahoe, California, USA, 3–8 December 2012; pp. 1097–1105.
24. Fraser-Thomas, J.; Côté, J.; Deakin, J. Understanding dropout and prolonged engagement in adolescent competitive sport. *Psychol. Sport Exerc.* **2008**, *9*, 645–662. [CrossRef]
25. Wu, H.; Gu, X. Towards dropout training for convolutional neural networks. *Neural Netw.* **2015**, *71*, 1–10. [CrossRef] [PubMed]
26. Van Dyk, D.A.; Meng, X.-L. The art of data augmentation. *J. Comput. Gr. Stat.* **2001**, *10*, 1–50. [CrossRef]
27. Lemley, J.; Bazrafkan, S.; Corcoran, P. Smart Augmentation-Learning an Optimal Data Augmentation Strategy. *IEEE Access* **2017**, *5*, 5858–5869. [CrossRef]
28. Polson, N.G.; Willard, B.T.; Heidari, M. A statistical theory of deep learning via proximal splitting. *arXiv*, 2015.
29. Kim, I.-J.; Xie, X. Handwritten Hangul recognition using deep convolutional neural networks. *Int.J. Doc. Anal. Recognit.* **2015**, *18*, 1–13. [CrossRef]

30. Kline, D.M.; Berardi, V.L. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Comput. Appl.* **2005**, *14*, 310–318. [CrossRef]
31. Abdelazeem, S.; El-Sherif, E. The Arabic Handwritten Digits Databases ADBase & MADBase. Available online: http://datacenter.aucegypt.edu/shazeem/ (accessed on 24 August 2017).