Arabic Handwritten Characters Recognition using Convolutional Neural Network

Ahmed El-Sawy, Mohamed Loey
Benha University
Faculty of Computer & Informatics
Computer Science Department
Egypt
{ahmed.el_sawy, mohamed.loey}@fci.bu.edu.eg

Hazem EL-Bakry
Mansoura University
Faculty of Computer & Information Sciences
Information System Department
Egypt
helbakry5@yahoo.com

Abstract: Handwritten Arabic character recognition systems face several challenges, including the unlimited variation in human handwriting and large public databases. In this work, we model a deep learning architecture that can be effectively apply to recognizing Arabic handwritten characters. A Convolutional Neural Network (CNN) is a special type of feed-forward multilayer trained in supervised mode. The CNN trained and tested our database that contain 16800 of handwritten Arabic characters. In this paper, the optimization methods implemented to increase the performance of CNN. Common machine learning methods usually apply a combination of feature extractor and trainable classifier. The use of CNN leads to significant improvements across different machine-learning classification algorithms. Our proposed CNN is giving an average 5.1% misclassification error on testing data.

Key–Words: Arabic Character Recognition, Deep Learning, Convolutional Neural Network

1 Introduction

Recognition is an area that covers various fields such as, face recognition, finger print recognition , image recognition, character recognition, numerals recognition, etc. Handwritten Character Recognition System (HDR) [1] is an intelligent system able to classify handwritten Characters as human see. Character classification [2, 3] is an important part in many computer vision and image possessing problems like Optical character recognition, license Plate recognition, etc. The classification of handwritten characters is a more difficult task due to the different handwriting styles of the writers. Handwritten Arabic character recognition (HACR) has attracted considerable attention in recent decades. Researchers have made significant breakthroughs in this field with the rapid development of deep learning algorithms. Arabic is a kind of the Semitic language [4] used in countries of the Middle East as a mother language of millions people. Generally, the Arabic alphabet characters consist of twenty-eight alphabet characters that illustrated in Fig. 1.

خ	ح	ج	ث	ت	ب	أ
khah	hah	jeem	theh	teh	beh	alef
ص	ش	س	ز	ر	ذ	د
sad	sheen	seen	zain	reh	thal	dal
ق	ف	غ	ع	ظ	ط	ض
qaf	feh	ghain	ain	zah	tah	dad
ي	و	ه	ن	م	ل	ك
yeh	waw	heh	noon	meem	lam	kaf

Figure 1: Arabic alphabet characters

Deep Neural networks consist of input layer and multiple nonlinear hidden layers and output layer, so the number of connections and trainable parameters are very large. The deep neural network need very large set of examples to prevent over fitting. One class type of Deep Neural Network with comparatively smaller set of parameters and easier to train is Convolutional Neural Network (CNN) [5]. CNN is a multi-layer feed-forward neural network that extract features and properties from the input data (images or sounds). CNN trained with neural network back-propagation algorithm. CNN have the ability to learn from high-dimensional complex inputs, nonlinear mappings from very large number of data (images or sounds)[6]. The advantage of CNN is that it automatically extracts the salient features which are invariant and a certain degree to shift and shape distortions of the input characters [7]. Another major advantage of CNN is the use of shared weight in convolutional layers, which means that the same filter is used for each input in the layer. The share weight reduces parameter number and improves performance [8]. In related work, they applied the algorithms on a small database of handwritten images, and its a problem in testing all the variation in handwriting. This problem lead us to collect a big dataset of Arabic handwritten characters to make our result more reliable and effective.

The rest of the paper is organized as follows: Section 2 gives a review on some of the related work done in the area. Section 3 describes the proposed approach, Section 4 gives an overview of the experiment and results, and we list our conclusions in Section 5.

2 Related Work

Various methods have been proposed and high recognition rates are reported for the handwritten recognition of English [8, 9], Chinese [10, 11, 12, 13, 14], Hangul [15], Malayalam [16], Devanagari [17, 18], Telugu [19] using CNN. In recent years many researchers addressed the recognition of letters including Arabic. Hussien et al. [20] proposed an Optical Character Recognition (OCR) of Arabic Handwritten Characters using Hopfield neural network. they designed an OCR system and trained their system using eight Arabic letters. They used a small database for eight Arabic letters with a successful recognition rate of 77.25%.

ElAdel et al. [21] presented a Neural Network (NN) architecture based on Fast Wavelet Transform (FWT) based on Multi-Resolution Analysis (MRA) and Adaboost algorithm. The learning and testing of the Arabic handwriting character classification system were conducted using IESK-arDB dataset which include 6000 segmented characters. The classification rate for the different groups of characters are 93.92%. Elleuch [22] introduced an Arabic handwritten characters recognition using Deep Belief Neural Networks. The approach was tested on the HACDB database. The HACDB database contains 6.600 shapes of handwritten characters written by 50 persons. The dataset is divided into a training set of 5.280 images and a test set of 1.320 images. The result was promising with an error classification rate of 2.1%.

Shatnawi and Abdallah [23] model a real distortions in Arabic handwriting using real handwritten character examples to recognize characters. They used these distortion models to synthesize handwritten examples that are more realistic. The dataset used in these experiments contains 48 examples of each Arabic handwritten character class (28 letters and 10 digits) obtained from 48 different writers. They achieve a recognition rate of 73.4% with feature set 3 for Arabic letters. Kef et al. [24] introduced a Fuzzy ARTMAP neural network for handwritten Arabic character recognition. Fuzzy ARTMAP neural network is an incremental supervised learning classifier. they used five Fuzzy ARTMAP neural networks to classify characters. The IFN/ENIT database contains 3840 handwritten character images. The training data is 2304 and the testing data is 1536. The average result of recognition rate is 93.8%.

Alabodi and Li [25] proposed a new recognition system based on geometrical features of Arabic characters. The IFN/ENIT database was used in their experimental results. Sixty percent of the words were chosen randomly as a training data set, and the remaining words were used as the testing data set. The average result of recognition rate is 93.3% for 596 words. Lawgali et al. [26] presented a new framework for the recognition of handwritten Arabic words based on segmentation. An ANN was used to identify the shape of character by using its features obtained by applying DCT. There work has been tested with IFN/ENIT database which contain 6033 characters. The average result of recognition rate is 90.73%.

3 Proposed Approach

3.1 Motivation

The motivation of this study is to use cross knowledge learned from multiple works to enhancement the performance of Arabic handwritten character recognition. In recent years, Arabic handwritten characters recognition with different handwriting styles as well, making it important to find and work on a new and advanced solution for handwriting recognition. A deep learning systems needs a huge number of data (images) to be able to make a good decisions. In [20-26] they applied the algorithms on a small database of Arabic handwritten characters images, and it's a problem to test the variation of several handwriting Arabic characters.

3.2 Architecture

In a convolutional neural network data and functions have additional structure. The input data $x_1, x_2, ..., x_n$ are images or sounds. Formally, the input to a convolutional layer is $M \times M \times C$ image where M is the height and width of the image, $M \times M$ is number of pixels in image and C is number of channels per pixel. For gray scale image have one channel $C = 1$ but RGB image have three channels $C = 3$. A CNN consists of a number of layers (convolutional layers, pooling layers, fully connected layers). The convolutional layer will have K filters (kernels) of $N\ x\ N \times R$ where N is height and width of filter (kernels) and R is the same number of image channels C or less and may vary for each filter (kernel). The filter (kernel) convolved with the image to produce k feature maps of size $M-N+1$ shown in Fig. 2. Each feature map is then pooled typically with mean or max pooling over $q \times q$ where q is range between 2 to 5 for large inputs that illustrated in Fig. 3. After the convolutional layers and pooling layers there may be any number of fully connected layers as in a standard multi-layer neural network.

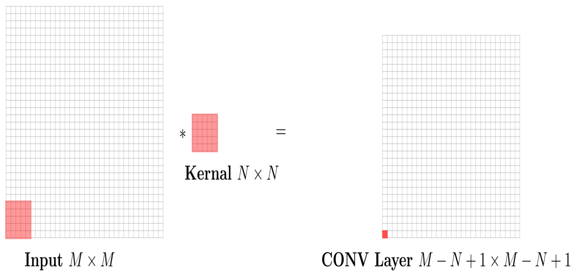


Figure 2: Illustration of convolution process on input image of size $M \times M$ with kernal of size $N \times N$. If the input is 32×32 and the kernal is 5×5 then the output of convolution is 28×28.

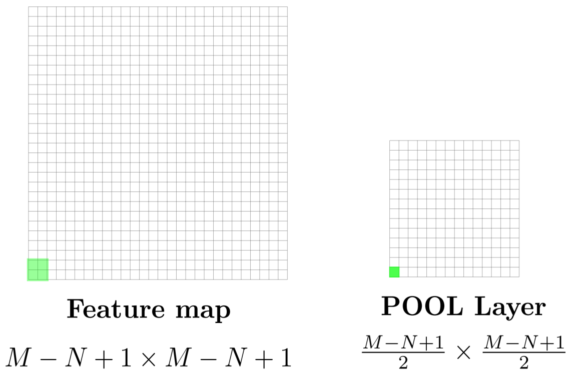

Feature map
$M - N + 1 \times M - N + 1$
POOL Layer
$\frac{M-N+1}{2} \times \frac{M-N+1}{2}$

Figure 3: Illustration of pooling process on the feature map that produce a new feature map of size $\frac{M-N+1}{2}$. If the input ia 28×28 then the output of pooling layer is 14×14

3.3 Layers

3.3.1 Convolutional Layers

Let layer l be a convolutional layer. Suppose that we have some $M \times M$ square neuron nodes which is followed by our convolutional layer. If we use an $N \times N$ filter W then convolutional layer output will be of size $(M-N+1) \times (M-N+1)$ which produce k-feature maps that illustrated in Fig. 4. Convolutional layer acts as a feature extractor that extracts salient features of the inputs such as corners, edges, endpoints. In order to compute the pre-nonlinearity input to some unit. Then, the input of layer $l-1$ comprises is computed as:

$$Y_i^l = B_i^{(l)} + \sum_{a=1}^{M} \sum_{b=1}^{M} W_i \ X_{(i+a)(j+b)}^{l-1} \qquad (1)$$

where $B_i^{(l)}$ is a bias matrix and $W_i^{(l)}$ is the filter of size $N \times N$. Then, the convolutional layer applies its activation function as:

$$Z_i^l = \sigma(Y_i^l) \qquad (2)$$

The activation function that is used in our proposed approach is Rectified Linear Units (ReLU) that applies the non-saturating $\sigma(Y_i^l) = max(0, Y_i^l)$. We apply the ReLU non-linearity as an activation fuction to the output of every convolutional layer and fuly connected layer. The ReLU [27] increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer.

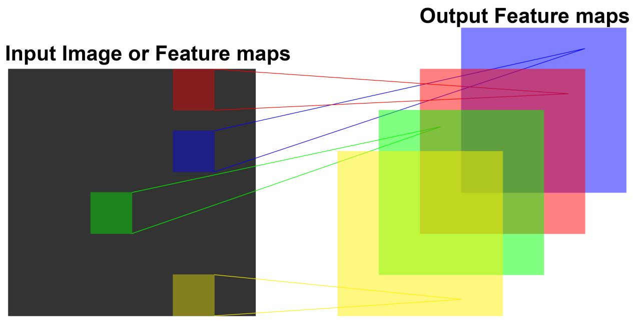


Figure 4: Illustration of a single convolutional layer that produce k-feature maps

3.3.2 Pooling Layers

After each convolutional layer, there may be a pooling layer. Fig. 5 shown the pooling layer takes a

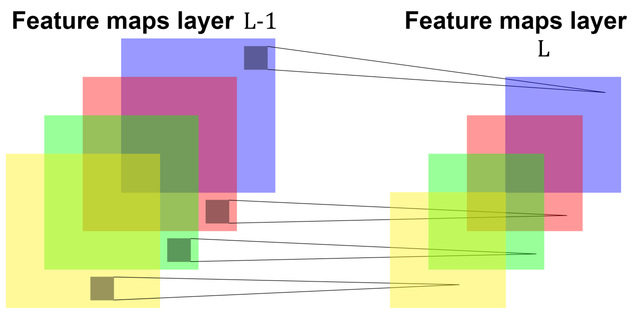


Figure 5: Illustration of pooling layer that produce the same k-feature maps

small rectangular blocks from the convolutional layer and subsamples it to produce a single output from that block. The pooling layer reduces the resolution of the image that reduce the precision of the translation (shift and distortion) effect. There are several ways to do this pooling, such as taking the average or the maximum, or a learned linear combination of the neurons in the block. Our pooling layers will always be max-pooling layers; that take the maximum of the block that they are pooling. All the max-pooling is carried out over a 2×2 pixel window.

3.3.3 Fully-Connected Layers

Finally, after several convolutional and pooling layers, the high-level reasoning in the neural network is done via fully connected layers. A fully connected layer takes all neurons in the previous layer (be it fully connected, pooling, or convolutional) and connects it to every single neuron it has. Fully connected layers are not spatially located anymore (you can visualize them as one-dimensional), so there can be no convolutional layers after a fully connected layer. Fig. 6 shown the proposed CNN approach for Arabic handwritten character recognition that describe as follow : INPUT $\rightarrow$ CONV $\rightarrow$ RELU $\rightarrow$ POOL $\rightarrow$ CONV $\rightarrow$ RELU $\rightarrow$ POOL $\rightarrow$ FC $\rightarrow$ RELU $\rightarrow$ FC. For one direction in a channel (feature map) of the convolutional layer C1, the output is ((32 5) +1) = 28. The max pooling layer S2 has non overlapping regions, so it down-samples by 2 in each direction,the output is 28/2 = 14. For one direction in a channel (feature map) of the convolutional layer C3, the output is ((14 5) +1 = 10. The max pooling layer S4 has non overlapping regions, so it down-samples by 2 in each direction, the output is 10/2 = 5 on fourth layer. The output image size of max pooling layer is 5 * 5 = 25. There are 64 channels in the convolutional layer and the same in pooling layer, so the total output of the max pooling layer is 25 * 128 = 1600. This is the size of the input to the fully connected layer FC5. Layer FC6 (Output), contain 28 units and is fully connected to FC5. Finally, the FC6 is composed of softmax classifier to produce 28 output class.

3.4 CNN Optimization

Learning Rate. Learning rate α is used during the weight update of such architecture. This parameter is crucial in determining the successful of convergence and generalization of such neural network. A too small learning rate leads to slow convergence and oppositely leads to divergence.
Activation Function. The activation function that is used in our proposed approach is Rectified Linear Units (ReLU) [28] that applies the non-saturating $\sigma(Y_i^l) = max(0, Y_i^l)$. We apply the ReLU nonlinearity as an activation fuction to the output of every convolutional layer and fuly connected layer. ReLU is more biologically plausible it can be engaged as better activation functions than the widely used logistic sigmoid and hyperbolic tangent functions. The ReLU increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer. Th use of ReLUs map more plausibly to biological neurons, makes the training of deep neural network significantly faster and improves its generalization ability.
Stochastic Gradient Descent. The gradient descent algorithm updates the parameters (weights and biases) so as to minimize the error function by taking small steps in the direction of the negative gradient of the loss function:

$$P_{i+1} = P_i - \alpha \nabla E(P_i) \tag{3}$$

where i stands for the iteration number, $\alpha > 0$ is the learning rate, P is the parameter vector, and $E(P_i)$ is the loss function. The gradient of the loss function, $\nabla E(P_i)$, is evaluated using the entire training set, and the standard gradient descent algorithm uses the entire data set at once.
Mini-batch. The stochastic gradient descent algorithm evaluates the gradient, hence updates the parameters, using a subset of the training set. This subset is called a mini-batch. Mini-batch optimization [29] is to divide the dataset into small batches of examples, compute the gradient using a single batch, make an update, then move to the next batch. Each evaluation of the gradient using the mini-batch is an iteration. At each iteration, the algorithm takes one step towards minimizing the loss function. The full pass of the training algorithm over the entire training set using mini-batches is an epoch. We specify the mini-batch

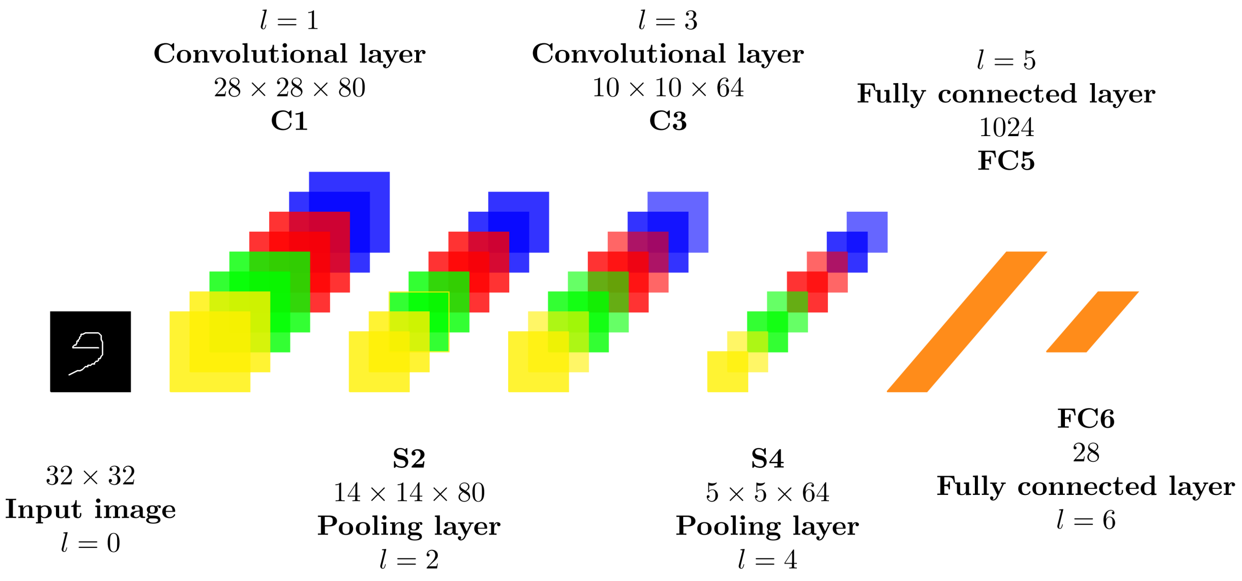


Figure 6: The proposed CNN for Arabic handwritten character recognition

size as 256 and the maximum number of epochs is 30.
Momentum. The gradient descent algorithm might oscillate along the steepest descent path to the optimum. Adding a momentum term [30] to the parameter update is one way to prevent this oscillation. The SGD update with momentum is

$$P_{i+1} = P_i - \alpha \nabla E(P_i) + \gamma(P_i - P_{i-1}) \tag{4}$$

where γ determines the contribution of the previous gradient step to the current iteration. By default, before training the input data shuffles.
L2 Regularization. One problem with machine learning estimation is that it can result in overfitting. Adding a regularization term for the weights to the loss function $E(P_i)$ is a way to reduce overfitting. The loss function with the regularization term takes the form:

$$E_R(P_i) = E(P_i) + \lambda\Omega(w) \tag{5}$$

where w is the weight vector, λ is the regularization factor (coefficient), and the regularization function, $\Omega(w)$ is:

$$\Omega(w) = \frac{1}{2} w^t w \tag{6}$$

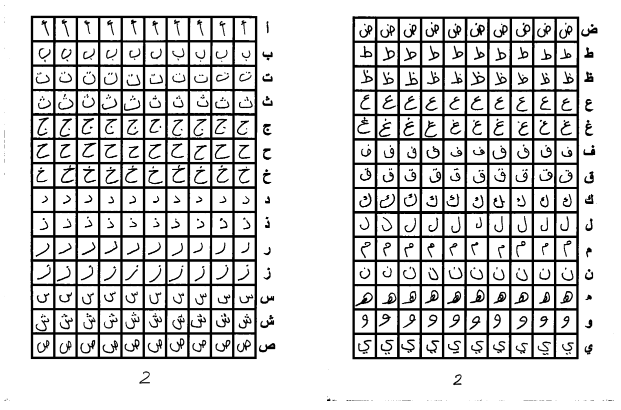
Softmax classifier. The softmax classifier is used in various probabilistic multi-class classification methods. To obtain a probability value for each class, the softmax function is applied to the final output units of the network:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{N} e^{x_k}} \tag{7}$$

4 Experiment

4.1 DataSet

Convolutional neural network need a big training data of handwritten characters images to get a good result, so we collect and made a dataset of Arabic handwritten characters. The dataset is composed of 16,800 characters written by 60 participants, the age range is between 19 to 40 years, and 90% of participants are right-hand. Each participant wrote each character (from 'alef' to 'yeh') ten times on two forms as shown in Fig. 7(a) & 7(b). The forms were scanned at the resolution of 300 dpi. Each block is segmented automatically using Matlab 2016a to determining the coordinates for each block. The database is partitioned into two sets: a training set (13,440 characters to 480 images per class) and a test set (3,360 characters to 120 images per class). Writers of training set and test set are exclusive. Ordering of including writers to test set are randomized to make sure that writers of test set are not from a single institution (to ensure variability of the test set). Creating the proposed database presents more challenges because it deals with many issues such as style of writing, thickness, dots number and position. Some characters have different shapes while written in the same position. For example the teh character has different shapes in isolated position. Some pre-processing methods were applied to reduce the effect of noise and to increase the readability of the input image. The pre-processing stage is quite essential for any recognition system. Generally, pre-processing stage contains convert RGB im-



(a) Form A (b) Form B

Figure 7: Data collection for Arabic characters

age to grayscale image, filtering and smoothing. The database is available for free at our google drive [1] for researchers.

4.2 Results

In this section, the performance of CNN was investigated for training and recognizing Arabic characters. The experiments are conducted in MATLAB 2016a programming environment with CUDA SDK v.7.5. The experiment was performed on a 2.6 GHz Core i7 PC with 8G memory and GPU NVIDIA Geforce 840M running on windows. At first for evaluating the performance of CNN on Arabic charters, incremental training approach was used on the proposed approach with the mini-batch mode. As shown in figure 8, the miss-classification rate for training data has reach 0% on epochs 25 to 30. Here our approach is trained for 30 epochs, but from epoch 25, the CNN shows a low miss-classification error.

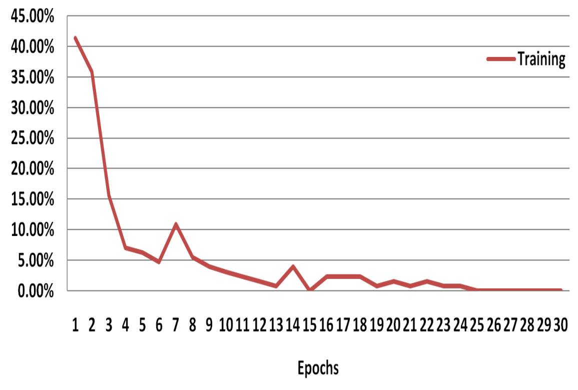


Figure 8: Miss-Classification rate for training data

Table 1: Miss-Classification & correct-Classification rate and number of wrong and correct recognition on testing data

Class	Character	Miss Classification	# of Missed	Correct Classification	# of Correct
1	alef	0.0%	0	100%	120
2	beh	3.3%	4	96.7	116
3	teh	8.3%	10	91.7%	110
4	theh	8.3%	10	91.7%	110
5	jeem	4.2%	5	95.8%	115
6	hah	2.5%	3	97.5%	117
7	khah	6.7%	8	93.3%	112
8	dal	5.0%	6	95.0%	114
9	thal	8.3%	10	91.7%	110
10	reh	0.0%	0	100%	120
11	zain	12.5%	15	87.5%	105
12	seen	2.5%	3	97.5%	117
13	sheen	4.2%	5	95.8%	115
14	sad	1.7%	2	98.3%	118
15	dad	9.2%	11	90.8%	109
16	tah	3.3%	4	96.7%	116
17	zah	8.3%	10	91.7%	110
18	ain	5.8%	7	94.2%	113
19	ghain	6.7%	8	93.3%	112
20	feh	5.0%	6	95.0%	114
21	qaf	7.5%	9	92.5%	111
22	kaf	5.0%	6	95.0%	114
23	lam	0.8%	1	99.2%	119
24	meem	0.8%	1	99.2%	119
25	noon	11.7%	14	88.3	106
26	heh	5.0%	6	95.0%	114
27	waw	4.2%	5	95.8%	115
28	yeh	3.3%	4	96.7%	116
Miss-Classification Rate= 5.1%					
Total Number of Miss-Classification = 173					
Total Number of Correct Classification = 3187					

Table 2: Arabic characters stroke similarity

Similar characters	Master stroke
ب،ت،ث	ٮ
ج،ح،خ	ح
د،ذ	د
ر،ز	ر
س،ش	س
ط،ظ	ط
ص،ض	ص
ع،غ	ع
ف،ق	ٯ
ل،ك	ل

[1] https://goo.gl/Y8KDNw

Table 3: Comparison between Proposed Approach and Other Approach

Authors	**Database**	**Training Data & Testing Data**	**Misclassification Error**
Hussien et al. [20]	Private dataset	small database of eight Arabic letters	77.25%
ElAdel et al. [21]	IESK-arDB	6000 images	93.92%
Elleuch [22]	HACDB	6600 images 5.280 Training images 1.320 Testing images	97.9%
Shatnawi and Abdallah [23]	Private dataset	1824	73.4%
Kef et al. [24]	IFN/ENIT	3840 images 2304 training images 1536 testing images	93.8%
Alabodi and Li [25]	IFN/ENIT	60% Training images 40% Testing images	93.3%
Lawgali et al. [26]	IFN/ENIT	6033 images	90.73%
Our Approach	**Our dataset**	**16800 images** **13440 Training images** **3360 Testing images**	**94.9%**

The miss-classification rate and the number of missed classification on the testing dataset is given in Table 1. It can be seen that we achieve a very exciting average miss-classification rates of 0.0%, 0.0%, 0.8%, 0.8% for class 1,10,23,24. The total of wrong classification is 173. This shows the great potential and recognition ability of Our CNN to classify Arabic handwritten characters. On the contrast, for the characters with high miss-classification rate, we found they have some characteristics in common, which are summarized as:

- Some characters are very confusable with similar character stroke that shown in Table 2. The tiny distinction of stroke structure bring challenges for some similar character pairs, such as sad and dad. The difference of sad and dad is the dot that above character dad.
- Some characters are very confusable with the structure of other characters. The Arabic character zain have a highest number of wrong classification 15. The character zain easily confused by four Arabic characters dal,thal and reh.
- Some character strokes are missing with the structure of character. Other characters may have extra stroke or stroke touching that shown in Fig. 9.

The above three phenomena may be the main reason why characters of such kinds of Arabic characters are very hard to be recognized. Besides, we also note that the average miss-classification of these classes is commonly higher.

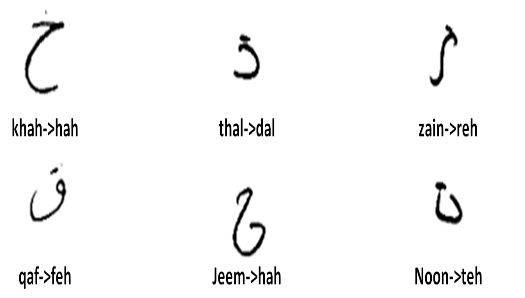


Figure 9: Misclassified character samples.The lower label is ground-truthprediction. The misclassified samples are due to extra stroke, missing strokes and stroke touching. Most of these samples are difficult for a proposed approach to make a correct prediction.

Finally, in Table 3 shown the obtained results with CNN on the database that we built and collected. CNN need a huge training data to get a good result, so we collect a big dataset of Arabic handwritten characters. We need more of training data to make rise up our CNN result. When CNN use a small dataset like IFN/ENIT that contain between 6033-3840, it would lead to average results. It can be seen from Table 3 that the proposed approach have the large database of Arabic image characters and have a good classification rate. The results are better than the results reported in related work [20,21,23-26], but the related work [22] have trained and tested his method on a small database. However, it is sometimes hard to compare, because previous work has not experimented with large database. The proposed method obtained 5.1% miss-classification error on testing data.

5 Conclusion

Handwritten Character Recognition for Arabic characters is an active research area which always needs an improvement in accuracy. This work is based on recognition of Arabic characters using convolutional neural networks (CNN). Compare to other deep learning architectures, CNN has better performance in both images and big data. The purpose to use deep learning was to take advantages of the power of CNN that are able to manage large dimensions input and share their weights. In an experimental section we showed that the results were promising with 94.9% classification accuracy rate on testing images. In future work, we plan to work on improving the performance of handwritten Arabic character recognition. The other plan is to work on Arabic handwritten word recognition using deep learning architectures.

References:

[1] R. Sarkhel, N. Das, A.K. Saha, and M. Nasipuri, A multi-objective approach towards cost effective isolated handwritten Bangla character and digit recognition, Pattern Recognition, 58, 2016, pp. 172-189.

[2] M. Tounsi, I. Moalla, A.M. Alimi, and F. Lebouregois, Arabic characters recognition in natural scenes using sparse coding for feature representations, Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 1036-1040.

[3] D.K. Sahu and C.V. Jawahar, Unsupervised feature learning for optical character recognition, Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 1041-1045.

[4] M.N. Ibrahim, S.N. Bilmas, A. Babiker, and M. Idroas, A Framework of an Online Self-Based Learning for Teaching Arabic as Second Language (TASL), 2013 Fifth International Conference on Computational Intelligence, Modelling and Simulation, 2013, pp. 255-260.

[5] Y. Liang, J. Wang, S. Zhou, Y. Gong, and N. Zheng, Incorporating image priors with deep convolutional neural networks for image super-resolution, Neurocomputing,194, 2016, pp. 340-347.

[6] D.S. Maitra, U. Bhattacharya, and S.K. Parui, CNN based common approach to handwritten character recognition of multiple scripts, Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 1021-1025.

[7] H.C. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R.M. Summers, Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning, IEEE Transactions on Medical Imaging, 35, 2016, pp. 1285-1298.

[8] J. Bai, Z. Chen, B. Feng, and B. Xu, Image character recognition using deep convolutional neural network learned from different languages, 2014 IEEE International Conference on Image Processing (ICIP), 2014, pp. 2560-2564.

[9] A. Yuan, G. Bai, L. Jiao, and Y. Liu, Offline handwritten English character recognition based on convolutional neural network, Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on, 2012, pp. 125-129.

[10] C. Wu, W. Fan, Y. He, J. Sun, and S. Naoi, Handwritten Character Recognition by Alternately Trained Relaxation Convolutional Neural Network, Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on, 2014, pp. 291-296.

[11] Z. Zhong, L. Jin, and Z. Feng, Multi-font printed Chinese character recognition using multi-pooling convolutional neural network, Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 96-100.

[12] W. Yang, L. Jin, Z. Xie, and Z. Feng, Improved deep convolutional neural network for online handwritten Chinese character recognition using domain-specific knowledge, Proc. Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 551-555.

[13] M. He, S. Zhang, H. Mao, and L. Jin, Recognition confidence analysis of handwritten Chinese character with CNN, Proc. Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 61-65.

[14] Z. Zhong, L. Jin, and Z. Xie, High performance offline handwritten Chinese character recognition using GoogLeNet and directional feature maps, Proc. Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 846-850.

[15] I.-J. Kim, and X. Xie, Handwritten Hangul recognition using deep convolutional neural networks, International Journal on Document Analysis and Recognition (IJDAR), 18, 2015, pp. 1-3.

[16] R. Anil, K. Manjusha, S.S. Kumar, and K.P. Soman, Convolutional Neural Networks for the Recognition of Malayalam Characters, Proc. Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, 2015, pp. 1041-1045.

[17] S. Acharya, A.K. Pant, and P.K. Gyawali, Deep learning based large scale handwritten Devanagari character recognition, 2015 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), 2015, pp. 1-6.

[18] P. Singh, A. Verma, and N.S. Chaudhari, Deep Convolutional Neural Network Classifier for Handwritten Devanagari Character Recognition, Information Systems Design and Intelligent Applications: Proceedings of Third International Conference INDIA ,2 , 2016, pp. 551-561.

[19] S.T. Soman, A. Nandigam, and V.S. Chakravarthy, An efficient multiclassifier system based on convolutional neural network for offline handwritten Telugu character recognition,Proc. Communications (NCC), 2013 National Conference on , 2013, pp. 1-5.

[20] R.S. Hussien, A.A. Elkhidir, and M.G. Elnourani, Optical Character Recognition of Arabic handwritten characters using Neural Network, Proc. Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), 2015 International Conference on , 2015, pp. 456-461.

[21] A. ElAdel, R. Ejbali, M. Zaied, and C.B. Amar, Dyadic Multi-resolution Analysis-Based Deep Learning for Arabic Handwritten Character Classification, Proc. Tools with Artificial Intelligence (ICTAI), 2015 IEEE 27th International Conference on , 2015, pp. 807-812.

[22] M. Elleuch, N. Tagougui, and M. Kherallah, Arabic handwritten characters recognition using Deep Belief Neural Networks, Proc. Systems, Signals & Devices (SSD), 2015 12th International Multi-Conference on , 2015, pp. 1-5.

[23] M. Shatnawi, and S. Abdallah, Improving Handwritten Arabic Character Recognition by Modeling Human Handwriting Distortions, ACM Trans. Asian Low-Resour. Lang. Inf. Process. , 15, 2015, pp. 1-12.

[24] M. Kef, L. Chergui, and S. Chikhi, A novel fuzzy approach for handwritten Arabic character recognition, Pattern Analysis and Applications, 2015, pp. 1-16.

[25] J. Al Abodi, and X. Li, An effective approach to offline Arabic handwriting recognition, Pattern Analysis and Applications, 40, 2014, pp. 1883-1901.

[26] A. Lawgali, M. Angelova and A. Bouridane, A Framework for Arabic Handwritten Recognition Based on Segmentation, International Journal of Hybrid Information Technology, 7, 2014, pp. 413-428.

[27] V. Nair, and G.E. Hinton, Rectified Linear Units Improve Restricted Boltzmann Machines, Machine Learning -International Workshop Then Conference, 2010, pp. 807-814.

[28] F. Hussain and J. Je ong, Efficient deep neural network for digital image compression employing rectified linear neurons, Journal of Sensors, 2016, pp. 1-7.

[29] M. Li, T. Zhang, Y. Chen, and A. J. Smola, Efficient mini-batch training for stochastic optimization, in Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 14, (New York, NY, USA), 2014, pp. 661-670.

[30] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, On the importance of initialization and momentum in deep learning, in Proceedings of the 30th International Conference on Machine Learning (ICML-13), vol. 28, 2013, pp. 1139-1147.