Advances in Science, Technology and Engineering Systems Journal Vol. 5, No. 5, 137-147 (2020)

www.astesj.com

Special Issue on Innovative Research in Applied Science, Engineering and Technology

ASTESJ

ISSN: 2415-6698

Towards Directing Convolutional Neural Networks Using Computational Geometry Algorithms: Application to Handwritten Arabic Character Recognition

Mohsine Elkhayati*, Youssfi Elkettani

Mathematics department, Faculty of Science, Ibn tofail university, Kenitra 14000, Morocco

ARTICLE INFO

Article history:
Received: 28 June, 2020
Accepted: 19 August, 2020
Online: 09 September, 2020

Keywords:
Convolutional neural network
Computational geometry
Gabriel's graph
Relative neighborhood graph
Appearance rate
Handwritten Arabic characters

ABSTRACT

Suppose we want to classify a query item Q with a classification model that consists of a large set of predefined classes L and suppose we have a knowledge that indicates to us that the target class of Q belongs to a small subset from L. Naturally, this filtering will improve the accuracy of any classifier, even random guessing. Based on this principle, this paper proposes a new classification approach using convolutional neural networks (CNN) and computational geometry (CG) algorithms. The approach is applied and tested on the recognition of isolated handwritten Arabic characters (IHAC). The main idea of the proposed approach is to direct CNN using a filtering layer, which reduces the set of possible classes for a query item. The rules of the relative neighborhood graph (RNG) and Gabriel's graph (GG) are combined for this purpose. The choice of RNG-GG was based on its great capacity to correctly reduce the list of possible classes. This capacity is measured by a new indicator that we call "the appearance rate". In recent years and due to strong data growth, CNNs have performed classification tasks very well. On the contrary, CG algorithms yield limited results in huge datasets and suffer from high computational time, but they generally reach high appearance rates and do not require any training phase. Consequently, the proposed approach uses an optimal architecture to exploit the advantages of the two techniques and overcome the computational time issue. Experiments carried out on the IFHCDB database have shown that the suggested approach outperforms a normal CNN and yield satisfactory results.

1. Introduction

Supervised classification aims to assign a new item to a class from a given set of classes according to its feature values and to a training set [1]. Supervised classification requires previously classified reference samples in order to train the classifier and subsequently classify unknown data [2]. Different supervised classification methods are available. The simplest methods do not require training but rather rely on the notions of proximity between pre-known samples and the unknown sample [2] such as Computational Geometry (CG) algorithms. Training-based algorithms such as decision trees and Neural Networks (NN) form a second set of methods. Many algorithms are widely used in the literature for classification tasks, including Support Vector Machine (SVM), artificial NNs, decision trees, and K-Nearest Neighbors (KNN). This paper proposes a new classification approach, which attempts to combine the robustness of a deep artificial NN, which is a Convolutional Neural Network (CNN), and the advantages of two CG algorithms, which are the relative neighborhood graph (RNG) and Gabriel's graph (GG). The approach is applied to the recognition of Isolated Handwritten Arabic Characters (IHAC).

Recently, deep learning algorithms have emerged and dominated many research areas. They have obtained excellent results and solved many difficult issues. Due to strong data growth, they have outperformed conventional classification algorithms and in some tasks, they have surpassed human capabilities. Deep learning is a sub-domain of machine learning that uses high-level hierarchy architectures to learn high-level abstractions in data [3]. Deep architecture models have been successfully implemented to solve many visual recognition problems such as image recognition [4], text recognition [5], and character recognition [6]. Deep learning generally relies on Deep Neural Networks (DNN). The main contrast between a NN and a

*Mohsine Elkhayati, Ibn tofail university, faculty of science, Kenitra, Morocco, +212-677343723 & MohsinElkhayati@gmail.com

DNN is the depth of the network, in other words, the number of hidden layers used in the system. DNNs can be classified into five different categories: Feedforward Neural Networks (FFNNs), Recurrent Neural Networks (RNNs), Radial Basis Function Neural Networks (RBFNNs), Kohonen Self Organizing Neural Networks (KSONNs), and Modular Neural Networks (MNNs) [7]. In FFNNs, the information flows in one direction, there is no looping or cycle. Unlike FFNNs, the units of RNNs form a cycle; the output of a unit becomes the input to itself. The hidden layer of an RBFNN includes a radial basis function and each unit represents a cluster center. KSONN organizes itself the network model in the input data using unsupervised learning. The MNN partition a huge network into smaller independent neural network modules [7]. CNN is an FFNN, and one of the most commonly used NNs in the literature, particularly in the area of pattern recognition. It is one of the most remarkable learning techniques where the network layers are trained with great robustness [8]. In general, CNNs are based on three main types of layers: convolutional layers (CLs), pooling layers (PLs), and fully connected layers (FCLs); each layer plays an important role in the CNN architecture.

Computational geometry consists of studying the algorithms that can be stated in terms of geometry, it is worth mentioning that there is a bunch of purely geometric problems that cannot belong to the computational geometry [9]. Several CG algorithms and techniques have been proposed to solve classification problems, such as GG, RNG, K-Nearest Neighbors (KNN), the Delaunay triangulation, the Voronoi diagram, the convex hull, and polygon triangulation. The most commonly used is KNN because of its simplicity and the no need for knowledge about the distribution of training data [10]. Unlike deep learning algorithms, CG algorithms have been effective in small databases, but with the strong growth in database sizes, the capacity and the computational complexity of these algorithms are problematic. Except for KNN, these algorithms have not been widely used in the literature, and to our knowledge; they have never been used for the classification of IHAC. However, based on certain experiences, it has been noticed that they obtain excellent results in terms of a new indicator that we call "Appearance Rate". A CG algorithm applies its proximity rules to connect between instances that can belong to different classes. The appearance rate is the probability that the query item is connected to at least one instance of its correct target class. This indicator assesses the ability of a CG algorithm to act as a class filter. This is an important indicator and is the main motivation behind the approach and behind the use of the hybridization of RNG and GG (RNG-GG). To our knowledge, the appearance rate indicator has never been considered before, and this paper is the first attempt to introduce it.

Suppose we have to classify a new item with a classification model that consists of a large set of classes L and suppose that we have information that confines the new item's correct target class to a small subset of L. Of course, this filtering will improve the accuracy of any classifier, even random guessing. Based on this principle, the idea of the approach is to use a filtering layer to direct CNN by reducing the set of possible classes for a new item. The filtering layer applies the rules of RNG-GG on the features extracted by the CLs of CNN in order to extract a list of possible classes. This list is then provided to the output layer of CNN. At this level, other classes are excluded from the classification and the class in the list with the maximum softmax value is considered the target class.

RNG and GG are computationally expensive when applied to huge datasets and high dimensional spaces, while CNNs work well when trained on huge datasets. This imposed itself as a real challenge. In this context, we adopted an optimal CNN architecture to reduce computational time without losing CNN's high performance. The architecture uses a reduced number of filters and many PLs to provide the smallest possible feature vectors to the filtering layer. In addition, in the filtering layer, we only apply the rules of RNG-GG against constructing full graphs. These two strategies avoided unrealistic execution time.

The approach has been tested on the recognition of IHAC using the IFHCDB database [11]. The results are satisfactory and outperform the normal CNN's performance by approximately 3%. These results give a new breath to CG algorithms, open another vision angle on the usefulness of these algorithms, and make us wonder about the benefits of directing robust classifiers using a class filtering. On the other hand, even the CG algorithms do not require a training phase and even if we adopted strategies that considerably reduce the computational time, the execution time of the proposed approach is even a little higher compared to that of a normal CNN in the classification phase. Generally, with powerful computers, this small difference in computing time becomes negligible.

The rest of the paper is organized as follows: Section 2 presents some recent works that have applied CNNs to IHAC recognition. Section 3 describes the proposed approach and the used algorithms. Section 4 Shows performed experimentations and obtained results. Finally, Section 5 concludes the paper.

2. Related works

Recently, supervised learning using CNNs has been very successful in image classification tasks, and this success is due mainly to the large scale labeled datasets [12]. An acceptable number of works have used CNNs for IHAC recognition. On the contrary and except KNN, CG algorithms have been rarely used in recent years and no attempt to use them for IHAC recognition in the literature to our knowledge. In this section, we will focus on the most recent works using CNNs for the recognition of IHAC and languages using the Arabic alphabet such as Urdu.

The IHAC recognition is a complicated task for several reasons: firstly, the Arabic alphabet contains characters that change their shapes depending on their position in the word: beginning, middle, final, or isolated [13] (see Figure 1). Secondly, there is a set of characters that have the same shape, the only difference between them is the number or the position of diacritical points [14], e.g. (ش, س) (د, ذ) (ح, خ, ج) (ب, ت, ث). Thirdly, 60% of the characters contain diacritical points; these diacritics are isolated and are written above or below the character. Sometimes, these points may be related to each other

or to the main body of the character due to writing errors or writing styles. The presence of these isolated or linked diacritics disturbs the features extraction process because they can modify the character shape, can be treated as noise or as an isolated character. Moreover, a set of characters have the same number of diacritics, which reduces the inter-character variability. Figure 1 shows the shape of the handwritten Arabic characters according to their position.

Name	Isolated	Start	Middle	End	Name	Isolated	Start	Middle	End
Alif	أ	أ	ـا	ـا	Daad	ض	ضـ	ـضـ	ـض
Baa	ب	بـ	ـبـ	ـب	TTaa	ط	طـ	ـطـ	ـط
Taa	ت	تـ	ـتـ	ـت	Dhaa	ظ	ظـ	ـظـ	ـظ
Thaa	ث	ثـ	ـثـ	ـث	Ayn	ع	عـ	ـعـ	ـع
Jim	ج	جـ	ـجـ	ـج	Ghayn	غ	غـ	ـغـ	ـغ
Haa	ح	حـ	ـحـ	ـح	Faa	ف	فـ	ـفـ	ـف
Khaa	خ	خـ	ـخـ	ـخ	Qaaf	ق	قـ	ـقـ	ـق
Dall	د	د	ـد	ـد	Kaaf	ك	كـ	ـكـ	ـك
Dhall	ذ	ذ	ـذ	ـذ	Laam	ل	لـ	ـلـ	ـل
Raa	ر	ر	ـر	ـر	Meem	م	مـ	ـمـ	ـم
Zayn	ز	ز	ـز	ـز	Noon	ن	نـ	ـنـ	ـن
Seen	س	سـ	ـسـ	ـس	Haa	ه	هـ	ـهـ	ـه
Sheen	ش	شـ	ـشـ	ـش	Waw	و	و	ـو	ـو
Saad	ص	صـ	ـصـ	ـص	Yaa	ي	يـ	ـيـ	ـي

Figure 1: Handwritten Arabic characters and their shape according to their position

In [15], the authors proposed a new approach to recognize offline Urdu handwritten characters and numerals using CNNs. The authors extract geometrical features of the characters and numerals first in order to embed them with pixel-based data that will be extracted by CNN. The proposed CNN model consists of 3 CLs, 2 PLs, and 2 FCLs. The authors suggested a local database on which they experimented with their approach. Regarding characters classification, they got 98.3% accuracy. It is worth mentioning that they reduced the number of classes into 10 by grouping similar characters into groups.

In [16], the authors proposed a hybridization between CNN and the multidimensional long short-term memory neural network (MDLSTM) to recognize Urdu characters written in Nastaliq font. CNN is used to extract the low level translational invariant features in characters. These features are fed to MDLSTM, which in turn extracts high order features and make the classification. The approach achieved 98.12% as recognition accuracy, which outperformed the state-of-the-art on the UPTI dataset [17].

In [18], the authors proposed an approach for IHAC recognition using a CNN architecture composed of 2 CLs, 2 PLs, and 2 FCLs. The Rectified Linear Unit (RELU) activation function was used after the CLs and the first FCL. The PLs had no overlapping regions, which down-samples the feature maps by 2 in each direction. The proposed model achieved 94.9% accuracy on a local database of 16,800 images.

Inspired by the success of the very deep model VGGNet, [19] proposed an alphanumeric VGGnet to recognize handwritten Arabic alphanumeric characters. They tried to reduce the overall complexity of VGGNet while maintaining high performance. The proposed model consists of 13 CLs, 2 PLs, and 3 FCLs. Dropout and augmentation techniques are adopted to prevent overfitting. The approach was tested on two databases, ADBase database for digits and HACDB for characters. An accuracy of 99.57% was obtained on ADBase and 97.32% on HACDB.

The authors in [20] suggested a hybridization of CNN and SVM to recognize IHAC. CNN is used to extract features from images and SVM for recognition. CNN architecture includes 2 CLs, 2 PLs, and 2 FCLs. Since the CLs do not contain a large number of parameters, dropout was applied only on FCLs to avoid overfitting. The outputs of the last FCL are then taken by SVM as a feature vector to continue the training process. The classification phase is done only with SVM. The approach has been tested with and without the dropout technique. The best accuracy (94.17%) was achieved using dropout.

The authors in [21] proposed an IHAC recognition approach using CNN and transfer learning strategies. The network architecture is built around AlexNet CNN, it consists of eight layers (5 CLs and 3 FCLs). Hidden layers are equipped with RELU and max-pooling is applied after the first, the second, and the fourth CL. The architecture was tested using three learning strategies; learning from scratch and two transfer learning strategies (CNN as feature extractor and fine-tune CNN). Experimentations have shown that learning from scratch brings better results compared to both transfer learning strategies and can reach a 100% accuracy in some conditions.

In [22], an automatic handwriting recognition model based on CNN was proposed. The model consists of 3 CLs, 3 PLs, and 4 FCLs. RELU activation function is used after each CL and 80% dropout is used in FCLs to combat overfitting. Besides this, the authors introduced a new database called Hijja, which included 47,434 Arabic characters. Therefore, the proposed model was evaluated on the new database and the AHCD [18] database. It achieved 88% accuracy on Hijja and 97% on AHCD. A comparison made in the paper showed that the proposed model outperformed the CNN-for-AHCD proposed by [18].

The othors of [23] trained a CNN holistically to recognize Arabic names. They proposed a CNN architecture composed of 4 CLs and 4 FCLs. Each CL is equipped with the RELU activation function. Max-pooling is used after the second and the fourth CL. The first FCL is equipped with RELU and the second with the softmax activation function. Batch normalization layers are used to normalize inputs and hence speed up the learning process. The dropout technique is used after the first FCL with a keep probability parameter of 0.2. The proposed CNN model obtained a 99.14% accuracy on the SUST-ARG names database [24]. The authors did not find many works in the literature that experimented with the same dataset. Therefore, they made a comparison with two works and their approach performances were much better compared to the two works.

A CNN model for handwritten Arabic characters recognition was designed in [25]. The architecture of the model consists of 3 CLs, an FCL of 200 neurons, and an output layer of 28 neurons (as the number of classes). Batch normalization, ReLU activation

function, and dropout of 0.5 follow each CL. Categorical Cross-Entropy [26] was used as a cost function. Experimental results showed that the proposed model was able to achieve an accuracy of 94.8% and 94.7% on the AIA9K [27] and AHCD datasets respectively.

3. Method

3.1. Material

3.1.1. Convolutional neural network

CNN is a feedforward neural network that uses CLs to extract high-level characteristics and properties included in the input data. It is a type of NNs, which has been used in many fields and has solved important problems. CNN is inspired by the biological mechanism of the animal cortex where the CLs play the role of receptive fields. CNN is generally trained with a back-propagation algorithm and can learn from high dimensional inputs, non-linear mappings of huge amounts of data [28]. CNN has many advantages; it automatically detects and extracts the invariant salient characteristics [29] and uses shared weights in CLs, which can reduce the number of parameters and improve performance [30].

Generally, CNN is based on three principal kinds of layers, which are CLs, PLs, and FCLs, each kind of layer plays an important role in the CNN architecture. In new deep learning libraries, another layer called Rectified Linear Unit (ReLU) [31] is considered to introduce the non-linearity in the network. Besides, dropout layers [32] are used in order to reduce overfitting. Figure 2 illustrates the general pipeline of CNN architecture.

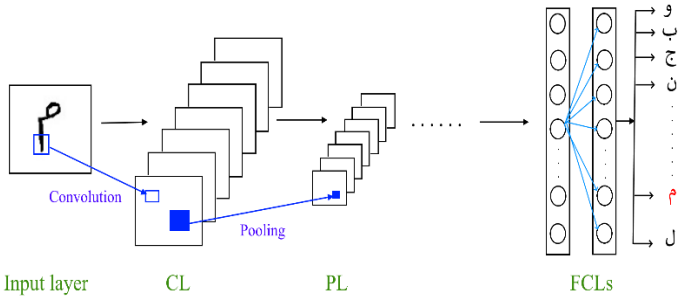

Figure 2: The pipeline of the general CNN architecture

CLs are feature extractors that use mathematical convolutions. Using some matrix filters, the CL tries to find features throughout the entire image. It generates various feature maps according to the number of filters; each filter is specified for a feature. The output of this layer is a set of filtered images [33].

The subsampling or PLs reduce the dimensionality of the CL's output images. They may be placed after CLs and they are translation-invariant because they take into account neighboring pixels as CLs do. The most widely used subsampling techniques are max pooling and average pooling [3]. The output of the PL has the same number of images with fewer pixels, but it retains important features.

FCLs are performed like a traditional NN, in which each neuron in a layer is connected to all neurons in the next layer. The input layer uses a 1D feature vector instead of the 2D matrix used by CLs and PLs. In the output layer, in most cases, a softmax function is used to assign a value between 0 and 1 to each neuron knowing that each neuron represents a class. The class with the maximum value is considered the predicted class. The FCLs represent the high-level reasoning of the network [18]. They contain around 90% of the parameters and aim to classify the input data in an appropriate class [3].

The ReLU layers are generally used to introduce non-linearity on the network. A ReLU layer converts the negative pixels of each feature map to 0 and maintains positive pixel values. This increases the non-linearity on the network without affecting the receptive fields of the CLs [31].

Dropout is a popular technique used in CNNs. The dropout layers aim to reduce the risk of overfitting and to speed up the training process [32]. The technique was proposed by [34] and explained in depth by [35]. The dropout layer randomly nullifies the weights and outputs of a number of units to disable their influence on the forward pass and backpropagation.

3.1.2. Relative neighborhood graph

In computational geometry, the relative neighborhood graph (RNG) is an undirected graph defined on a set of points of the Euclidean plane by the connection of two points P_i and P_j by an edge whenever there is no third point P_k closer to both points P_i and P_j, $1 \leq i, j, k \leq n$. This graphic was proposed by Godfried Toussaint [36] in 1980 as a means of defining a structure from a set of points [37].

Let $\Omega = \{P_1, P_2, \ldots\ldots, P_n\}$ be a set of n points in the m-dimensional space, and $d(P_i, P_j)$ denotes the distance between P_i and P_j. The RNG (Ω) connects all pairs of points (P_i, P_j), $(i \neq j)$ for which there is no other point P_k with the property that the greater of the two distances $d(P_i, P_j)$ and $d(P_j, P_k)$ is less than the distance $d(P_i, P_j)$ [38]. Which is expressed by the following formula:

$$d(P_i,P_j) \leq Max(d(P_i,P_k),d(P_j,P_k)), \forall P_k \in \Omega \setminus P_k \neq P_i,P_j \quad (1)$$

In other words, the two points P_i and P_j are neighbors if the lune in Figure 3 (a) obtained by the intersection of the hyperspheres of centers P_i and P_j, and of radius the length of the edge (P_i, P_j) does not contain any other point of Ω.

3.1.3. Gabriel's graph

Gabriel's graph (GG) [39] is a connected graph in which, if two points P_i and P_j are connected by an edge, then the hypersphere of diameter $\delta(P_i, P_j)$ does not contain any points of Ω.

If we call μ the center of the edge (P_i, P_j), the vertices P_i and P_j will be neighbors in the sense of Gabriel if and only if they verify the following property:

$$\delta(P_k,\mu) \geq \delta(P_i,\mu) = \delta(P_j,\mu) = \frac{\delta(P_i,P_j)}{2}, \forall P_k \in \Omega \setminus P_k \neq P_i,P_j \quad (2)$$

Since the hatched circle in Figure 3 (b) which has the diameter $\delta (P_i, P_j)$ is empty, the two points P_i and P_j are connected by an edge.

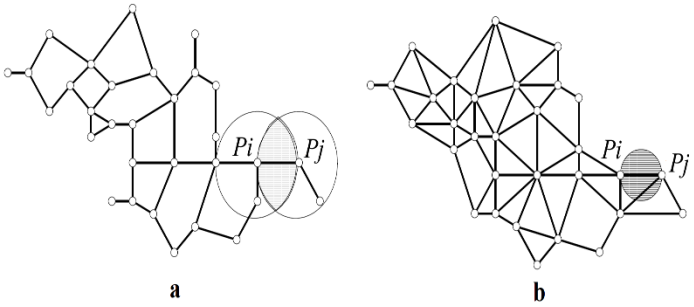

Figure 3: Diagram explaining RNG (a) and GG (b)

3.2. Architecture

Suppose we have prior knowledge that the correct target class of the query item *Q* belongs to a small subset of classes instead of the whole database's classes set. This will limit choices for a classifier and then increase the probability of having a correct decision. Based on this principle, the main idea behind the approach is to direct CNN by using a filtering layer that attempts to reduce the list of possible classes for *Q*. The filtering layer hybrids the rules of two CG algorithms (RNG and GG) to provide a reduced list of possible classes to CNN. CNN ignores other classes and attempts to classify *Q* based on the provided list. Figure 4 illustrates the proposed approach's architecture.

The performed CNN architecture consists of 3 CLs, 3 PLs, 1 Dropout layer, and 3 FCLs including an input layer, a hidden layer, and an output layer. The input layer of the network is a binary image of size (28×28×1). The first CL consists of 16 filters of 5×5×1 size. In the second CL, 24 filters of 3×3×1 are used, and in the third CL, 32 filters of 3×3×1 are used. In order to reduce the resolution of the images by the half, we use the max-pooling strategy with a 2×2 window and stride 2 in all PLs. The first FCL consists of the feature vector resulted from the last PL (512 values). The second FCL consists of 128 neurons and the output layer consists of *n* neurons as the number of classes (In our case it is 28 classes). The softmax activation function is used in the output layer. The ReLU activation function is used by all CLs. The dropout technique is used before the first FCL with a keep probability parameter of 0.5.

The filtering layer does not intervene in the training phase, but it just captures the features extracted by CNN in order to use them at the classification phase. The output of the last PL is 32 filtered images of 4×4×1 size, which will be transformed into a 1D feature vector (512 values). This vector will be fed to FCLs, and at the same time to a 512D space. Once the training phase is finished, we will get a trained CNN and a 512D space which includes the feature vectors of all training samples.

Since RNG/GG does not require a training phase, the filtering layer only intervenes during the classification phase. At this phase,the query item *Q* follows the same architecture and once its feature vector is introduced into the *512D* space, the rules of RNG-GG are applied to *Q*, which connects it to a certain number of instances. The list of classes of these instances is provided to the CNN output layer (In our example the list contains *C2*, *C3*, and *C5*). From this list and according to the assigned softmax values, the class with the maximum value is considered the target class of *Q* (In our example the target class is *C5* since it has the maximum softmax value from the list), while classes outside the list are definitively excluded from the classification.

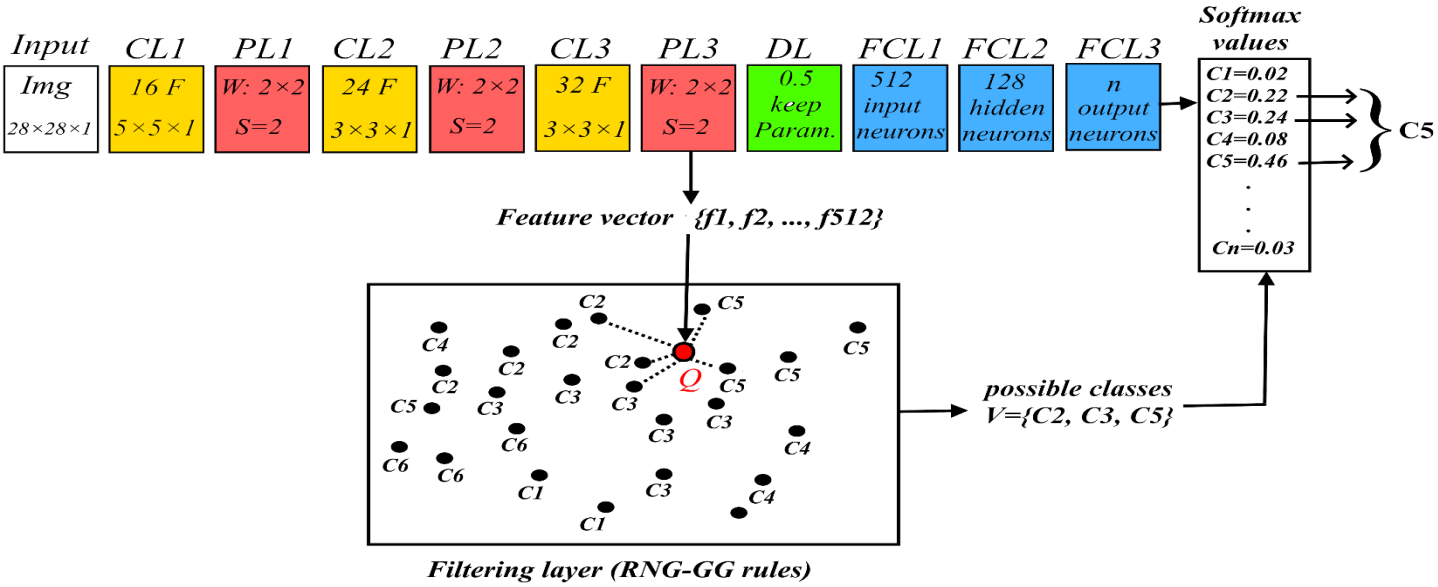

Figure 4: The proposed approach architecture

3.3. Filtering layer

As mentioned in the previous section, the filtering layer applies the rules of RNG-GG in order to provide a reduced list of possible classes to CNN. The motivation behind using this layer and the reasons for using RNG-GG are explained and discussed in this section. Our points of view are reinforced in the experiments and results section. CG algorithms such as RNG/GG generally give acceptable classification accuracies when applied to small datasets and low dimensional spaces. However, in the case of large datasets and high dimensional spaces, they are computationally expensive and do not give satisfactory results. In this section, we will discuss a new advantage of these algorithms, which we call appearance rate. We will introduce it, explain its importance, and discuss how to exploit it for a classification problem. After that, we will address the computational complexity issue. Finally, we will illustrate the hybridization technique between RNG and GG.

3.3.1. Appearance rate

CG algorithms are based on a simple technique for classifying new data. Once a query item *Q* is processed, it will be connected to a set of instances based on the proximity rules of the used algorithm. The classes of these instances will then pass to a voting system, in which the class with the maximum number of instances will be selected as the target class *Cq*. Based on some experiments, CG algorithms yield unsatisfactory classification accuracies, but it was noticed that *Cq* passes to the voting system with a high probability. This means that the problem of CG lies in the voting system. In other words, the prediction based on the number of connected instances does not bring good results. The probability that *Cq* appears in the voting system is what we call the appearance rate, i.e. the appearance rate is the probability that *Q* will be connected to at least one instance of *Cq*. More clarification, the appearance rate allows us to know the ability of an algorithm to detect the target class of *Q* (before the voting system), while the classification rate shows its ability to perform a final classification (after the voting system). As far as we know, the appearance rate indicator has not been considered before; therefore, this paper represents the first attempt to introduce it. This indicator is the key point of the approach and the main motivation for combining RNG-GG and CNNs. More directly, RNG-GG has approached 100% appearance rate (see **section 4.2**). This inspired us to use it to reduce the list of possible classes before making a final classification by CNN.

Overall, to classify *Q* using RNG/GG, we construct the graph based on their proximity rules and extract the classes' labels of the instances that are connected to *Q*. These labels are saved into a vector, *V= {C1, C2,…, Cn }*. Finally, a voting system is applied in which the most voted label of *V* is considered the target class of *Q*. We are not interested in the voting phase. However, to know the ability of an algorithm to connect *Q* with at least one instance of *Cq*, we introduced the appearance rate indicator (*AR*) which is calculated using the following formula.

$$AR = \frac{\sum_{q=1}^{TN} f(C_q)}{TN} * 100 \quad (3)$$

$$\text{with} \quad f(C_q) = \begin{cases} 1 & if \ C_q \subset V \\ 0 & Otherwise \end{cases}$$

Denotes *f(Cq)* is the appearance function of *Cq* in *V*; *TN* is the total number of test items.

The appearance rate is an important indicator because it gives us the possibility of confining *Cq* in a small defined list instead of the whole list of classes. If this indicator is high, it will help any classifier to make the right prediction because it reduces choices. Otherwise, if this indicator is weak, it will decrease the classification accuracy, consequently, the class filtering will not be useful.

3.3.2. Computational complexity

Unlike small datasets and low dimensional spaces, constructing RNG/GG graphs in the case of large datasets and large dimensional spaces is time-consuming and memory-intensive. The two parameters that complicate the RNG/GG calculations are the number of instances and space dimensionality. Constructing a graph on a space requires calculating all the distances between each instance and all the other instances of the space and checking the proximity rules each time. Consequently, the higher the number of instances and dimensions, the more the calculations increase. Generally, to perform the classification, RNG/GG is applied directly to the instances of the space and does not require a training phase. This is an advantage because it does not need time for training, but it is also a disadvantage because it takes a long time to construct a graph each time to classify a new item. We can think for example to introduce the new element into an already existing graph. This idea appears ideal to avoid reconstructing the graph each time, but actually, the introduction of a new item will change a significant number of edges (create new ones and delete existing ones), which will create a change in the entire structure of the graph. Once again, we find ourselves facing an obligation to reconstruct the graph from scratch.

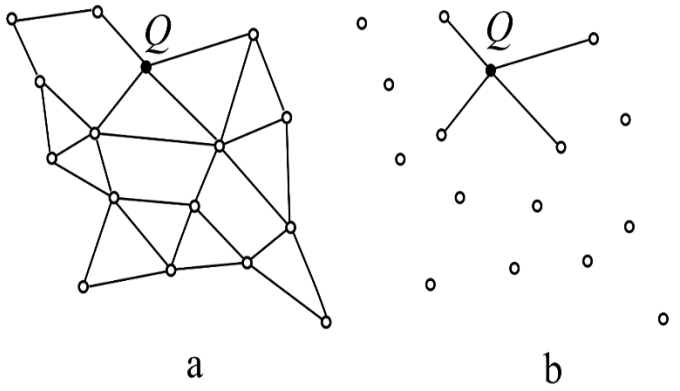

Figure 5: Illustration of the GG construction on a set of points; (a) full graph; (b) mini-graph

The good news is that our model only needs edges of the query item *Q*, while other edges in the graph are not necessary. On this basis, it is not necessary to construct the full graph because it does not add any benefit to the process. The idea is then to construct a mini-graph that consists only of the edges of *Q* instead of the full graph. Figure 5 shows the difference between the construction of a GG full graph (a) and the construction of a mini-graph (b). This strategy significantly reduced the computational time by ignoring unprofitable calculations (see **section 4.3**).

3.3.3. Hybridization

Our approach is based on the use of a robust CNN's architecture and a filtering layer including the rules of RNG and GG together. The hybridization between RNG and GG is applied as follows: Once a query item is introduced into the *512D* space, GG and RNG rules are applied to it at the same time, which produces a mini-graph composed of GG and RNG edges around the query item. The classes' labels of all the instances connected to *Q* are then stored in a vector *V*. Figure 6 shows an example of a constructed mini-graph based on the proposed hybridization.

Based on the example figured in Figure 6, the vector will contain the following labels, *V = {C2, C3}*. *V* will be provided then to the CNN output layer to perform classification. Other classes are ignored and the class with the maximum softmax value from *V* will be considered the target class *Cq*.

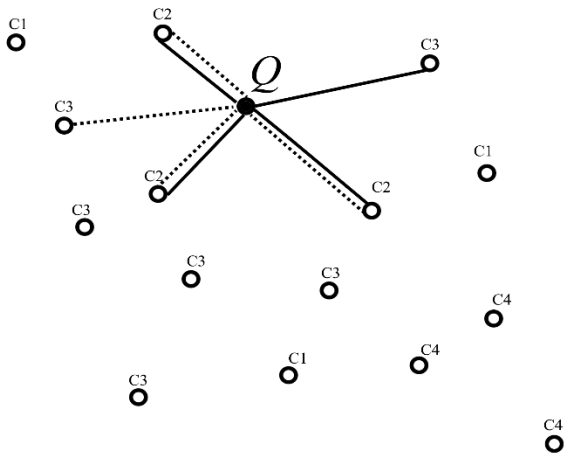

Figure 6: The constructed mini-graph from the hybridization of GG (solid edges) and RNG (dotted edges).

4. Experimentations and results

This section presents and discusses the results obtained from three experiments. Section 4.1 describes the database used in all the experiments. Section 4.2 Evaluates and compares the appearance rate of certain CG algorithms. This experiment aims to justify the choice of RNG-GG and to show the difference between the classification rate and the appearance rate of CG algorithms. In section 4.3, we compare the computational complexity of a mini-graph construction with a full graph construction. Finally, in section 4.4, we present and discuss the obtained results.

4.1. Dataset

To evaluate the performances of the proposed approach, we used the Isolated Farsi Handwritten Character Database (IFHCDB) [11]. The database contains 52,380 single characters and 17,740 digits. The distribution of characters is not uniform, which means that the number of samples is not the same for all characters. We are only interested in the 28 Arabic characters, which represent 97% of the characters in the database. Therefore, the experiments are carried out using a dataset from IFHCDB containing 35,989 characters for training and 15,041 for tests. All the presented experiments were carried out using this dataset.

4.2. Appearance rate results

In this section, we compare the appearance rate, the reduction rate, and the classification rate of four algorithms that are 9-NN, RNG, GG, and RNG-GG. The choice of KNN was due to its popularity and the choice of nine neighbors (9-NN) was due to the experiments carried out. The reduction rate concerns the percentage of ignored classes. Table 1 shows the results of the experiment.

Table 1: Comparison between the appearance rate, the classification rate, and the reduction rate of the studied algorithms

Classifier	9-NN	RNG	GG	RNG-GG
Appearance rate	69.2 %	74.5 %	91.3 %	99.1 %
Classification rate	65.6 %	59.4 %	58.5 %	64.2 %
Reduction rate	66.3 %	72.3 %	45.9 %	43.6 %

The results presented in Table 1 reinforce what was mentioned previously on the difference between the classification rate and the appearance rate of the studied algorithms. 9-NN did not show any significant change in the appearance rate compared to the classification rate; however, GG, RNG, and RNG-GG have shown great changes. This proves the idea that the problem of classification using RNG/GG lies in the voting system. RNG-GG yields an excellent appearance rate (99.1%). This result can be read as follows: in 99.1% of cases, the correct target class of the query item is detected by the algorithm and then passed to the next stage; on the other hand, the algorithm can reduce the number of classes that reach the final classification stage by a rate of 43%.

It is worth reminding that the idea of the proposed approach was born from the results of this experiment and the related conclusions. The high appearance rate obtained by RNG-GG encouraged us to consider using it as a filtering layer for a robust classifier such as CNN.

4.3. Computational complexity results

In this section, we compare the computational time of RNG and GG full graphs construction with mini-graphs construction. Feature vectors (512 values) of the training set (35,989 images) were extracted using CNN. Therefore, the graphs are constructed on a *512D* space of 35,989 training points.

The experiment was performed on a computer with an Intel Core i3-3110M processor with 2.40 GHz frequency, 4GB RAM, and Windows 10 64-bit operating system. Table 2 shows the results of the experiment.

Table 2: The computational time of full graphs and mini-graphs construction in a *512D* space of 35,989 points.

Algorithm	**GG**		**RNG**	
	Full graph	***Mini-graph***	***Full graph***	***Mini-graph***
Computational time	18.4 minutes	13.20 seconds	19.6 minutes	13.65 seconds

As shown in Table 2, the computational time of an RNG/GG mini-graph construction is too short than the full graph construction. Constructing a full graph every time we need to classify a new item has no benefit, is time-consuming and unrealistic. If for example, we want to classify words, paragraphs, or documents, it can take hours, even days and this is not realistic. However, the mini graph consumed about 80 times much less time than the full graph in this experiment and seems to be more realistic to some extent. Even both algorithms do not require a training phase, but they are computationally expensive and this remains the big drawback of these algorithms, especially when applied to huge databases. Indeed, the used strategy significantly reduced the computational time, but generally, the execution time of the approach is a little higher of a normal CNN since it consumes more time to make filtering, but this is can be negligible with the presence of powerful computers.

4.4. Accuracy results and discussion

4.4.1. Normal CNN against directed CNN

In this section, we compare the results obtained by a normal CNN with those obtained by the proposed directed CNN. The first experiment was performed using the proposed CNN architecture without the intervention of the filtering layer. The second experiment was performed with the intervention of the filtering layer. Both experiments were carried out using the same hyper-parameters. Table 3 compares the accuracies obtained by the two experiments.

Table 3: Comparison between the results obtained by normal CNN and directed CNN

Approach	**Normal CNN**	**Directed CNN**
Classification rate	94.2 %	97.4 %

From Table 3, we note that the use of RNG-GG as a filtering layer has improved the classification accuracy of CNN. Indeed, this strategy would not have succeeded if we did not have a strong filtering layer that provides reliable results; otherwise, the correct class could be rejected in the filtering layer, which would decrease the classification accuracy. Therefore, the obtained results are mainly thanks to the high appearance rate of RNG-GG.

Table 4: The confusion matrix of the proposed approach

	ا	ب	ت	ث	ج	ح	خ	د	ذ	ر	ز	س	ش	ص	ض	ط	ظ	ع	غ	ف	ق	ك	ل	م	ن	ه	و	ى	Accuracy
ا	297																						29						99.03 %
ب		291	8	6																3	1				3			3	92.38 %
ت		4	407	7																					4			4	95.53 %
ث			2	10																									83.33 %
ج					161	2	3											1	1										95.83 %
ح					4	354	7											4											95.93 %
خ						1	106												1										98.14 %
د								776	10	9	3																		97.24 %
ذ								2	10																				83.33 %
ر								4		128	17												10						97.65 %
ز									3	12	320																		95.52 %
س												693	12								2	4			3				97.05 %
ش												6	160																96.38 %
ص														138	2									1					97.87 %
ض														1	107														99.07 %
ط																60													100 %
ظ																	15												100 %
ع						4												248	9										95.01 %
غ																		1	41										97.61 %
ف																				265					10				96.36 %
ق																				10	131								92.90 %
ك																						192							100 %
ل	14									6													877	3					97.44 %
م														14				11						145		11			97.58 %
ن			8	5							9														112			6	97.56 %
ه																		6						10		738	5		97.23 %
و										7	4													8			389		95.34 %
ى		9	5	5																4	3				5			132	97.71 %
Total number of test images = 15041																													
Total number of correct classification = 14650 = 97.4%																													
Total number of miss-classification = 391 = 2.6%																													

4.4.2. Confusion matrix

Against printed / Latin languages, Arabic handwriting has many challenges and difficulties, which make the task of recognition very difficult for any classifier. The similarities between the characters and the writing style are the main factors responsible for the confusion between the characters and thus lead to miss-classifications. Table 4 shows the confusion matrix of the proposed approach.

From the confusion matrix, we can conclude that the majority of miss-classifications are due to the great similarity between some characters such as (س، ش), (ل ،ا), (ب، ت، ث، ن), etc. It is also worth mentioning that the distribution of the database is not uniform, some characters are represented with a large number of samples and some others are represented with a small number. This presents a great challenge for the classifier and negatively affects the classification accuracy. From Table 4, we also notice that the average of accuracies is not necessarily equal to the overall accuracy since the distribution is not uniform. We expect that normalizing the distribution by increasing the number of samples for small classes will generally improve the performance of the approach.

4.4.3. Comparison with recent approaches

This section compares the results of the proposed approach with other recent works in the literature. Table 5 shows the comparison results.

Table 5: A comparison between the obtained accuracy with other recent works

Reference	Method	Database	Accuracy
[20]	CNN based-SVM + Dropout	HACDB	94.17 %
[19]	Very deep NN	HACDB	97.32 %
[20]	CNN based-SVM + Dropout	IFN/ENIT	92.95 %
[18]	CNN	AHCD	94.90%
[27]	Windows-based descriptors	AIA9k	94.28 %
[25]	CNN	AIA9K	94.8 %
[40]	specificity and the singularity	IFHCDB	96.31 %
[41]	Modified Bitmap Sampling+ local binary pattern	IFHCDB	97.18 %
[42]	Derivative projection profile + Hamming Neural Network	IFHCDB	96.91 %
Proposed method	Directed CNN	IFHCDB	97.4 %

From Table 5, it appears that the proposed approach outperforms other approaches. Usually, this comparison is for internalization as it cannot be completely reliable for some reasons:

- The databases used are not the same;
- the splitting schemes used are not the same (Percentage of training, validation, and test data);
- Some works used untrained data for testing and others did not;
- Some works used pre-processing techniques to make the classifier's job easier, while others have not.

It must be reminded that this paper aims to show the importance of directing a robust classifier such as CNN by using a filtering layer. For this purpose, experiments were carried out on the IHAC recognition. Indeed, this work does not deal with specific characteristics or particular cases and does not use any particular treatment adopted to Arabic handwriting. Despite this blind application, the results were satisfactory and outperformed other approaches. Therefore, the authors of this article expect that an in-depth focus on the characteristics of the application domain will improve the performance of the approach.

Generally, the obtained results prove the effectiveness of the proposed approach and its ability to improve CNN's performance. This can open up a new avenue of research based on the direction of robust classifiers using class filtering. On the other hand, these results breathe new life into CG algorithms by thinking of using them as classifier helpers. This breath comes from the appearance rate indicator, which in turn is an important indicator that assesses the filtering capacity of algorithms.

5. Conclusion

In this paper, we presented a new classification approach based on CNN and RNG-GG. The approach was tested on the IHAC recognition. The main idea of the approach was to direct CNN by using a filtering layer, which limits the number of possible classes for a query item. This filtering layer relies on the rules of RNG-GG. The choice of RNG-GG was based on some experiments and conclusions that showed its high appearance rate. The appearance rate is a new indicator that we introduced in this paper. It evaluates the filtering capabilities of a CG algorithm. This indicator is very important and was the main motivation for using a filtering layer to direct CNN.

Generally, CNNs yield good classification accuracies on huge datasets. Conversely, CG algorithms face great challenges in computational time and classification accuracy when applied to huge databases, but as said before, they yield an excellent appearance rate, in particular, RNG-GG. On this basis, we have tried to propose an optimal architecture, which exploits the advantages of the used algorithms and overcomes their limits. Consequently, the proposed CNN architecture uses an optimal number of filters and three PLs in order to minimize the size of the feature vector, which will be fed to RNG-GG. This makes it possible to reduce the dimensionality of the space and then the computational time of RNG-GG. Besides this, we applied only the rules of RNG-GG on the query item against constructing full graphs, which significantly reduced the computational time.

Some experiments were carried out on the IFHCDB database to assess the performance of the proposed approach. The results were satisfactory and the approach outperformed a normal CNN and other recent approaches in the literature. Therefore, these results could open up new perspectives and impose new questions such as how directing a classifier through class filtering can be beneficial in terms of performance, and how to exploit the high appearance rate given by RNG-GG in classification tasks. Furthermore, the obtained results gave new life to CG algorithms as they have not been widely used for classification tasks in the

literature, especially in recent years due to their limitations on huge datasets. On the other hand, the proposed approach requires a little more execution time than a normal CNN. This time is consumed during class filtering. This issue becomes negligible with the presence of powerful computers, and despite that, we are trying to reduce it further.

In future work, we intend to find a more efficient solution for time complexity in order to have the freedom to improve the architecture of the network by adding more layers and more filters. Besides, we intend to try more combinations between other CG algorithms and other kinds of deep neural networks.

References

[1] G. M. Di Nunzio, A. Sordoni, "Picturing bayesian classifiers," in: Data Mining Applications with R, Academic Press, 35–61, 2014, doi:10.1016/B978-0-12-411511-8.00002-5.

[2] X. Ceamanos, S. Valero, "Processing hyperspectral images," in: Optical Remote Sensing of Land Surface, Elsevier, 163–195, 2016, doi:10.1016/B978-1-78548-102-4.50004-1.

[3] A. C. YanmingGuo, A. YuLiu, C. ArdOerlemans, C. SongyangLao, A. SongWu, S. Michael, A. C. Lew, "Deep learning for visual understanding: a review," Neurocomputing, **187**, 27–48, 2016, doi:/10.1016/j.neucom.2015.09.116.

[4] A. Krizhevsky, I. Sutskever, G. E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," in NIPS'12: Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, 1097–1105 , 2012, doi:10.1145/3065386.

[5] T. Wang, D. J. Wu, A. Coates, A. Y. Ng, "End-to-End Text Recognition with Convolutional Neural Networks," in the 21st International Conference on Pattern Recognition (ICPR2012), Tsukuba, IEEE, 3304–3308, 2012.

[6] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, A. Y. Ng, "Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning," in International Conference on Document Analysis and Recognition, Beijing, IEEE, 440–445, 2011, doi:10.1109/ICDAR.2011.95.

[7] B. Koyuncu, H. Koyuncu, "Handwritten character recognition by using convolutional deep neural network: a review," International Journal of Engineering and Technologies-IJET, **5**(1), 1–6, 2019, doi:10.19072/ijet.528775.

[8] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, **86**(11), 2278–2324, 1998, doi:10.1109/5.726791.

[9] M. Bundzel, T. Kasanický, "Using algorithms of computational geometry for pattern recognition and comparison to support vector machine," 2005.

[10] G. T. Toussaint, C. Berzan, "Proximity-graph instance-based learning, support vector machines, and high dimensionality: an empirical comparison," in: Machine Learning and Data Mining in Pattern Recognition, Springer, Heidelberg, 222–236 2012, doi:10.1007/978-3-642-31537-4_18.

[11] S. Mozaffari, K. Faez, F. Faradji, M. Ziaratban, S. M. Golzan, "A Comprehensive Isolated Farsi/Arabic Character Database for Handwritten OCR Research," in Tenth International Workshop on Frontiers in Handwriting Recognition, La Baule, France, Suvisoft, 385–389, 2006.

[12] Z. Chen, R. Ding, T. W. Chin, D. Marculescu, "Understanding the impact of label granularity on CNN-based image classification," CoRR, abs/1901.07012, 2019.

[13] J. Chen, Information Preserving Processing of Noisy Handwritten Document Images, Ph. D. Thesis, Lehigh University, 2015.

[14] N. Amara, F. Bouslama, "Classification of Arabic script using multiple sources of information: State of the art and perspectives," International Journal on Document Analysis and Recognition, **5**(4), 195–212, 2003, doi:10.1007/s10032-002-0092-6.

[15] M. Husnain, M. M. Saad, S. Mumtaz, M. Z. Jhanidr, M. Coustaty, M. Muzzamil Luqman, J. M. Ogier, G. S. Choi, "Recognition of Urdu handwritten characters using convolutional neural network," Appl. Sci., **9**(13), 2758–2778, 2019, doi:10.3390/app9132758.

[16] S. Naz, A. I. Umar, R. Ahmad, I. Siddiqi, S. B. Ahmed, M. I. Razzak, F. Shafait, "Urdu Nastaliq recognition using convolutional recursive deep learning," Neurocomputing, **243**, 80–87, 2017, doi:10.1016/j.neucom.2017.02.081.

[17] N. Sabbour, F. Shafait, "A segmentation-free Approach to Arabic and Urdu OCR," in: Document Recognition and Retrieval XX, SPIE, 8658, 215–226, 2013, doi:10.1117/12.2003731.

[18] A. El-Sawy, M. Loey, H. EL-Bakry, "Arabic handwritten characters recognition using convolutional neural network," WSEAS Trans. Comput. Res., **5**, 11–19, 2017.

[19] M. Mudhsh, R. Almodfer, "Arabic handwritten alphanumeric character recognition using very deep neural network," Information, **8**(3), 105–118, 2017, doi:10.3390/info8030105.

[20] M. Elleuch, R. Maalej, M. A. Kherallah, "New design based-SVM of the CNN classifier architecture with dropout for offline Arabic handwritten recognition," Procedia Comput. Sci., **80**, 1712–1723, 2016, doi:10.1016/j.procs.2016.05.512.

[21] C. Boufenar, A. Kerboua, M. Batouche, "Investigation on deep learning for off-line handwritten Arabic character recognition," Cognitive Systems Research, **50**, 180–195, 2018, doi:10.1016/j.cogsys.2017.11.002.

[22] N. Altwaijry, I. Al-Turaiki, "Arabic handwriting recognition system using convolutional neural network," Neural Comput & Applic, 2020, doi:10.1007/s00521-020-05070-8.

[23] M. E. Mustafa1, M. K. Elbashir, "A Deep learning approach for handwritten Arabic names recognition," International Journal of Advanced Computer Science and Applications (IJACSA), **11**(1), 678–682, 2020, doi:10.14569/IJACSA.2020.0110183.

[24] M.E. Musa, "Towards building standard datasets for Arabic recognition," International Journal of Engineering and Advanced Research Technology (IJEART), **2**(2), 16–19, 2016.

[25] K. Younis, "Arabic handwritten characters recognition based on deep convolutional neural networks," Jordan J Comput Inform Technol (JJCIT), **3**(3), 186–200, 2018.

[26] P. Golik, P. Doetsch, H. Ney, "Cross-Entropy vs. Squared Error Training: A Theoretical and Experimental Comparison," in 14th Annual Conference of the International Speech Communication Association, Lyon, France, 1756–1760, 2013.

[27] M. Torki, M. E. Hussein, A. Elsallamy, M. Fayyaz, S. Yaser, "Window-based descriptors for Arabic handwritten alphabet recognition: A comparative study on a novel dataset," arXiv preprint arXiv: 1411.3519, 2014.

[28] D. S. Maitra, U. Bhattacharya, S. K. Parui, "CNN Based Common Approach to Handwritten Character Recognition of Multiple Scripts," in 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, IEEE, 1021–1025, 2015, doi:10.1109/ICDAR.2015.7333916.

[29] H. C. Shin, H. R. Roth, M. Gao, M. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, R. M. Summers, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," IEEE Transactions on Medical Imaging, **35**(5), 1285–1298, 2016, doi:10.1109/TMI.2016.2528162.

[30] J. Bai, Z. Chen, B. Feng, B. Xu, "Image Character Recognition Using Deep Convolutional Neural Network Learned from Different Languages," in 2014 IEEE International Conference on Image Processing (ICIP), Paris, 2560–2564, 2014, doi:10.1109/ICIP.2014.7025518.

[31] V. Nair, G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in the 27th International Conference on Machine Learning, 807–814, 2018.

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," J. Mach. Learn. Res., **15**(1), 1929–1958, 2014.

[33] Y. LeCun, K. Kavukcuoglu, C. Farabet, "Convolutional Networks and Applications in Vision," in 2010 IEEE International Symposium on Circuits and Systems, Paris, 253–256, 2010, doi:10.1109/ISCAS.2010.5537907.

[34] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," arXiv e-prints, abs/1207.0580, 2012.

[35] P. Baldi, P. J. Sadowski, "Understanding Dropout," in Proceedings of the NIPS, Advances in Neural Information Processing Systems 26, Curran Associates, Inc., 2814–2822, 2013.

[36] G. T. Toussaint, "The relative neighborhood graph of a finite planar set," Pattern Recognition **12**(4), 261–268, 1980, doi:10.1016/0031-3203(80)90066-7.

[37] F. Muhalenbach, Evaluation de la Qualité de Représentation de Fouille de Données, Ph. D. Thesis, University of Lumière Lyon II, 2002.

[38] J. Katajainen, O. Nevalainen, "An almost naive algorithm for finding relative neighborhood graphs in Lp metric," Theoretical Informatics and Applications, **2**(21), 199–215, 1987.
[39] K. R. Gabriel, R. R. Sokal, "A new statistical approach to geographic variation analysis," Systematic Biology **18**(3), 259–278, 1969, doi:10.2307/2412323.
[40] Y. Boulid, A. Souhar, M. Y. Elkettani, "Handwritten character recognition based on the specificity and the singularity of the Arabic language," International Journal of Interactive Multimedia and Artificial Inteligence, **4**(4), 45–53, 2017, doi:10.9781/ijimai.2017.446.
[41] Y. Boulid, A. Souhar, M. M. Ouagague, "Spatial and textural aspects for Arabic handwritten characters recognition," International Journal of Interactive Multimedia and Artificial Intelligence, **5**(1), 86–91, 2018, doi:10.9781/ijimai.2017.12.002.
[42] M. Askari, M. Asadi, A. Asilian Bidgoli, H. Ebrahimpour, "Isolated Persian/Arabic handwriting characters: Derivative projection profile features, implemented on GPUs," Journal of AI and Data Mining, **4**(1), 9–17, 2016, doi:10.5829/idosi.JAIDM.2016.04.01.02.