IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2020, accepted May 8, 2020, date of publication May 14, 2020, date of current version June 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2994290

Comparative Study on Deep Convolution Neural Networks DCNN-Based Offline Arabic Handwriting Recognition

TARAGGY M. GHANIM[1], MAHMOUD I. KHALIL[2], AND HAZEM M. ABBAS[2], (Senior Member, IEEE)

[1]Faculty of Computer Science, Misr International University, Cairo 44971, Egypt
[2]Faculty of Engineering, Ain Shams University, Cairo 11517, Egypt

Corresponding author: Taraggy M. Ghanim (taraggy.ghanim@miuegypt.edu.eg)

ABSTRACT Recently, deep learning techniques demonstrated efficiency in building better performing machine learning models which are required in the field of offline Arabic handwriting recognition. Our ancient civilizations presented valuable handwritten manuscripts that need to be documented digitally. If we compared between Latin and the isolated Arabic character recognition, the latter is much more challenging due to the similarity between characters, and the variability of the writing styles. This paper proposes a multi-stage cascading system to serve the field of offline Arabic handwriting recognition. The approach starts with applying the Hierarchical Agglomerative Clustering (HAC) technique to split the database into partially inter-related clusters. The inter-relations between the constructed clusters support representing the database as a big search tree model and help to attain a reduced complexity in matching each test image with a cluster. Cluster members are then ranked based on our new proposed ranking algorithm. This ranking algorithm starts with computing Pyramid Histogram of Oriented Gradients (PHoG), and is followed by measuring divergence by Kullback-Leibler method. Eventually, the classification process is applied only to the highly ranked matching classes. A comparative study is made to assess the effect of six different deep Convolution Neural Networks (DCNNs) on the final recognition rates of the proposed system. Experiments are done using the IFN/ENIT Arabic database. The proposed clustering and ranking stages lead to using only 11% of the whole database in classifying test images. Accordingly, more reduced computation complexity and more enhanced classification results are achieved compared to recent existing systems.

INDEX TERMS Agglomerative hierarchical clustering, deep convolutional neural network, Kullback-Leibler divergence, offline arabic handwritting recognition, pyramid histogram of gradients.

I. INTRODUCTION

Psychological science [1] highlighted the importance of using handwriting in our personal life for retaining more information and improving personal performance and comprehension. Handwriting helps us to re-frame topics in our own words. This re-framing process triggers parts of our brains that are not activated by typing verbatim on digital devices. One of the most recent challenging tracks of computer vision is automatic Arabic handwriting recognition. Although it has been a pervasive research field for a long time, a lot of effort is still needed for achieving better recognition accuracy and fast response time. This field is important to represent handwritten text in a digitized symbolic form.

In spite of all the technological progress, we can never deny that ancient civilizations made significant contributions and presented valuable handwritten manuscripts that need to be documented digitally. Automated handwriting recognition aims to reserve old manuscripts and to create electronic libraries of digitized handwritten documents. It is useful in many other areas like forgery detection and signature analysis. Much efforts and time are saved when a handwritten text is automatically recognized and digitized.

Formerly, authors categorized the handwriting recognition process into online and offline processes [2]. Online recognition process uses more information than the offline recognition process, thus achieving higher accuracy levels.

The associate editor coordinating the review of this manuscript and approving it for publication was Xin Luo.

Online recognition is based on the sequential order of writing and the instant temporal information, while offline recognition is based solely on images and pixel information, and according to literature,is more challenging. This paper is concerned only with offline recognition.

Arabic is one of the major worldwide languages used in documenting sources [3], and it poses many challenges due to different handwriting styles and changeable letter shapes corresponding to different spatial localities. Around 27 languages use the Arabic alphabet. Arabic is the native language of more than 420 million people around the world, making it the sixth most widely spoken language [4]. It is a cursive language and has 28 different characters. Characters are not represented in lower or upper cases but each character has four different shapes according to its position in the word: isolated, start, middle, and end. Arabic alphabets are written from right to left. Each word may consist of several subwords called Piece of Arabic Words (PAWs). A PAW can be a single letter or a set of connected letters. Among the peculiar features of the Arabic language is that some characters do not join to their left neighbors, as shown in Figure 1a. One stand-alone character does not join to left or right neighbors, as shown in Figure 1b. Some characters are very similar and can be only distinguished by their dots number and positions, as shown in Figure 1c. Some diacritical marks can be placed above or below particular characters and change the sound and/or the meaning of the character.

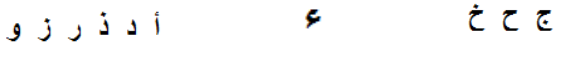


FIGURE 1. Different categories of arabic characters: (a) Arabic characters that do not join to their left neighbor, (b) Stand-alone character, (c) Different dots number and positions.

Automatic recognition of Arabic handwriting requires building a hybrid recognition system, computing different types of features, and applying various classifiers together to achieve improved performance. Different classifiers [5], [6], were applied in a parallel or cascading manner to achieve high automatic recognition rates. Data-mining raw databases [7] is also essential for better meaningful data representation and reduced testing time and consequently higher recognition rates.

Offline Arabic handwriting recognition approaches are mainly categorized into two main types. The first type represents the traditional approaches that are based on extracting features, while the second type represents the deep learning-based approaches that automatically extract features from raw images. The traditional approaches apply three main consequent steps: preprocessing, feature extraction, and finally classification.

Recently, deep learning techniques demonstrate better recognition performance than the other crafted feature detectors in the field of computer vision and building recognition systems. Deep convolutional neural networks (DCNN) is one of the most prominent deep learning methods. It is a sort of feed-forward network that can extract valuable topological features from raw images. Instead of extracting the traditional features, neural networks are capable of extracting features from large-scale raw data [8]. Recent surveys stated the effectiveness of deep learning techniques [9]–[11], in the field of character recognition. There are different proposed architectures in the field of deep Convolution Neural Networks DCNN [12]–[15]. Many research efforts were directed to design new effective architectures for feature extraction and classification issues.

We propose a state-of-the-art solution in Arabic handwriting words holistic recognition. This approach is composed of three consecutive stages: matching, ranking and classifying. The first main contribution is representing the IFN/ENIT as a big search tree-like model of inter-related clusters. Each cluster includes the database classes with similar regional and geometric features. This model aims to reduce the matching process complexity. The matching process relates each test image with one of the constructed clusters.

The second main contribution is proposing a new ranking approach that is applied before classification with CNN for reducing computational complexity. The ranking stage sorts the matching cluster's members from the nearest to the furthest relative to the test image. Accordingly, only a set of high-rank classes passes for final classification. Finally, we investigate the effect of the most popular DCNN architectures in the final classifying stage in terms of accuracy and different complexity measures. The proposed approach achieved the highest accuracy among all the previously proposed holistic approaches.

The efficiency of deep CNN requires a lot of training samples which increases the computational complexity. The proposed multi-stage approach aims to compensate this increased complexity by reducing the number of database classes participating in the final classification stage

Different DCNN architectures were recently introduced, each suggests a different innovative idea as a contribution to improve learning efficiency. Each architecture differs in number and design of the convolutional layers. Some of these different architectures were applied before for text recognition, while others weren't. It was challenging to apply the different architectures in our research problem and to investigate their effect on the final classification stage of our proposed approach. Section II summarizes previous related work. The section includes an analysis of offline Arabic handwriting recognition systems. Section III lists our contribution in points and introduces our proposed approach. Experimental results are presented in section IV. Complexity Analysis is presented in section V. Finally, conclusion and future work are presented in sections VI and VII, respectively.

II. LITERATURE REVIEW

A. CONCEPTS AND ANALYSIS

In the field of automatic handwriting recognition, applying a combination of hybrid cooperative classifiers together is

a thought-provoking issue to enhance the overall system performance [8]. There are two main categories of combination techniques: feature fusion and decision fusion. Feature fusion is based on combining different features into one feature vector followed by applying one classifier. Some approaches [16] emphasized the effectiveness of this early integration method. Decision fusion techniques integrate different classifiers' decisions; each classifier is trained on a different set of features. More recent approaches [8], [17], [18], stated that this high-level combination strategy improves the performance of handwriting recognition systems.

One of the important concepts related to building robust recognition systems is feature localization. Different automatic Arabic recognition systems [8],

References [19]–[21], applied sliding window-based feature extraction techniques to do localization. It was also stated [22] that applying multiple sliding windows achieved superior recognition rates. A special sort of localized features [23] called Pyramid histogram of oriented gradients (PHoG) was applied to achieve localization on the ordinary HoG features.

B. CATEGORIZATION BASED ON SEGMENTATION APPROACHES

Some approaches [24] are based on segmenting words into letters. Other recent systems [5], [25], segment words to different categories of segments: core shapes, sub-core shapes and diacritics. Still others [19], [26]–[28], recognize words in a holistic manner without any prior segmentation. This holistic way avoids recognition errors that could be caused due to wrong segmentation, but it requires robustness of features. The proposed holistic approaches [19], [26]–[28], stated that statistical and structural features achieved this required robustness. On the other hand, some proposed analytical approaches [19], [22], [26], are based on modeling each word by concatenating its character models.

C. DATABASES

The number of available Arabic databases is limited. Among the most well-known are HACDB, which is a character database, Ibn Sina dataset of Arabic manuscript, and IFN/ENIT database [29] of Arabic words (as described later in section IV).

D. FEATURES EXTRACTION

Different types of features were extracted for Arabic handwriting recognition. Some approaches [19], [26], [27], computed a combination of statistical and structural features, whereas others [23], [30], [31], computed skeleton-based features. Distribution of concavity features was obtained in different approaches [8], [24], [32], and has been proved effective in representing Arabic handwriting. Some recent approaches [33]–[35], applied Convolutional Neural Network (CNN) to automatically extract features from raw images.

Pyramidal features were also extracted and achieved high performance in representing handwriting scripts. Input strings were represented by pyramidal histograms of characters (PHOC) [36] to be encoded at different levels. Also, pyramid histogram of oriented gradients (PHoG) [37] outperformed the ordinary histograms of gradients (HoG) [7], [20], [38]. Based on literature [5], [39], derivatives were also computed in recently proposed systems.

E. INTEGRATION OF CLASSIFIERS

Different approaches [8], [34], applied a combination of multiple Bidirectional Long Short-Term Memory - Connectionist Temporal Classification (BLSTM-CTC) architectures. These applied deep neural networks were integrated together for building a more robust Arabic handwritten recognition system. The recurrent neural network (RNN) was also integrated with BLSTM [30] and showed effectiveness. A combination of multidimensional RNN (MDRNN) and Connectionist Temporal Classification (CTC) was proposed [40] and the error rate was 8.57% on IFN/ENIT. Another approach applied multidirectional long short term memory (MDLSTM) -based system [41] and achieved satisfactory results with a recognition rate of 89.9%. For overfitting avoidance, a combination of Deep Bidirectional LSTM (DBLSTM) and a special type of RNN was proposed [42].

Other approaches [35] integrated CNN and Support Vector Machine (SVM). SVM was applied with the RBF kernel for the final classification. CNN was also integrated with HMM in other approaches [43], [44]. CNN was applied for feature extraction and classification as well [45]. A combination of multiple residual networks (ResNet) [18] was applied for Arabic words recognition and achieved superior results that outperform the single residual network.

Different sets of multi-stage Hidden Markovian Models (HMM) were applied [5] for the classification of different word segments. Multi-Stream HMM (MSHMM) architectures [26], [46], were proposed too and outperformed ordinary HMM-based approaches [17], [24], [28], [47], [39].

Different types of deep networks, like deep belief network (DBN) [48] and multi-column deep neural network (MCDNN) [49], were recently proposed and achieved satisfactory results but still need to be improved. In a comparison [50] between DBNs and Vertical-Horizontal HMM, the latter showed more feasibility when applied to the IFN/ENIT. Another approach [51] proved that the combination between a Convolutional DBN (CDBN) and Support Vector Machine (SVM) outperforms the DBN-based approaches.

F. POST-PROCESSING SUPPORT

It was observed that some proposed approaches achieved high recognition rates due to the support of the post-processing stage after classification. Some approaches [8], [34], applied the Connectionist Temporal Classification layer (CTC) for sequence labeling after classification. Others [5], [25], [30],

achieved high recognition rates due to using a dictionary for post-processing.

G. CONCLUSIONS AND RESULTS

Different systems applied a combination of hybrid cooperative classifiers together. Such a combination has been described as thought-provoking as it enhances the overall system performance. Accordingly, our approach includes different classifiers that use a combination of different extracted features in a cascading fashion.

Based on the literature, [35] pre-trained and fine-tuned CNNs were suggested to be explored. Applying Support Vector Machine on features extracted by CNN [35] achieved 92.95% recognition rate. This CNN-based features outperformed other SVM approaches when applied on other traditional features [7], [20], [38]. Accordingly, our approach studied the effect of SVM and CNN in the classification phase.

It was stated [19], [27], [28], that local densities are suitable for all cursive languages, and statistical features are computed faster than other types of features. When MSHMM was trained on a set of statistical and structural features, [46] achieved 91.1% recognition rate [26], and when trained on a combination of density-based features and contour-based features it achieved 79.8% recognition rate.

MSHMM [46] outperforms Semi-continuous HMM (SCHMM) [23], [27], [31], when applied on the same type of features. Although multi-stream frameworks prvides the possibility to combine feature streams, their implementation is higher in complexity.

The analytical approaches [26] such as HMMs outperformed holistic ones, but they have a large number of unstructured parameters and Viterbi algorithm is expensive in terms of time and memory. The HMM outperforms BLSTM [8], [24], [32], when applied on the distribution of concavity features due to character modeling, while the main advantage of the holistic approaches [19] is the absence of the pre-segmentation step. Relatively low recognition rates were achieved by holistic approaches.

K-means algorithm was applied for clustering [7] and achieved 85% recognition rate. The clustering process enhanced the final performance of SVM when applied on HoG features. On the other hand, PHoG [37] outperformed the ordinary HOG features. Based on the out-performance of the PHoG, it is employed in our approach and achieved satisfactory results.

Word error rate (WER) was reduced in different approaches by applying multi-staging techniques [17] and applying multiple sliding windows [22]. Feature selection methods, like Minimal-redundancy-maximal-relevance (mRMR), were effective in reducing WER [30].

The performance of traditional RNN was enhanced by applying different residual models. The temporal residual achieved higher accuracy than hybrid temporal and spatial residual models. On the other hand, the training process is accelerated by hybrid temporal and spatial residual models.

H. APPLICATIONS ON DEEP LEARNING AND WORD RECOGNITION

Some applications in the field of Arabic handwriting recognition were proposed recently. An approach [52] based on hybrid peer-to-peer (P2P), Grid computing, and agent technology was applied to distribute tasks among multiple peers and grids for tolerating several types of faults. The approach [52] used character segmentation and pattern matching, achieving fast execution time, and optimum speedup factor. Another approach [53] combined Histogram of Oriented Gradients (HOG) and Gray Level Run Length (GLRL) features, and applied decision fusion on scores for implementing a better writer identification system. Another application [54] was designed for the early detection of Parkinson's disease. Short time series were used to train long short-term memory (LSTM) neural networks. Other adaptive neural networks (NNs) [55] were designed for tracking and controlling non-linear systems. Another deep learning approach [56] diagnosed faults of electric motors.

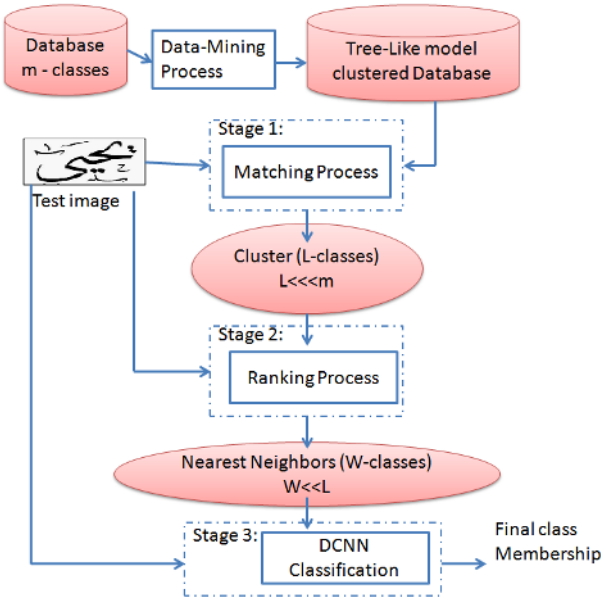


FIGURE 2. The proposed system overview (m: total number of database classes, L: number of classes per cluster, W: number of high-rank classes).

III. THE PROPOSED APPROACH

This approach is composed of three consecutive stages: matching, ranking, classifying, as shown in Figure 2. The output of each stage is used as an input to the next stage. The matching and ranking stages aim to pass a small set of training classes for the final classification stage. This aims to reduce the classification complexity and to increase the final recognition accuracy. The effect of our matching and ranking techniques on the performance of different popular DCNN architectures is studied in the final classification phase. In Figure 2, the labels: m, L, and W represent the total number of database classes, number of classes per cluster, and

number of high-rank classes that pass to the final classification stage, respectively. It is worth noting that L value is much less than m, and W is much less than L.

The contribution of the proposed multi-stage approach is summarized in the following points:

1) Applying Hierarchical Agglomerative Clustering (HAC) on the IFN/ENIT database to construct small inter-related clusters. Inter-relations between the constructed clusters support representing the database as a big search tree during the training phase. This constructed tree-like search model is used to match each test image with a cluster. Accordingly, the matching process complexity becomes $O(log(v))$ instead of $O(v)$, where $O(v)$ is the complexity of the ordinary search, and v is the number of clusters.
2) Observing that the largest cluster includes 423 different classes, which is 49.2% of the database size. Only this matching set of database classes is involved in the next phases instead of the whole database classes to reduce computation complexity.
3) Applying Random Forest Selection technique to measure the importance of Dileep features [57] to adapt Arabic, rather than English handwriting recognition.
4) Proposing details on our new ranking approach in section III-B. The effect of different parameters on the true positive rate (TPR) is recorded. The ranking is applied on the matching set per test image to vote for the best subset of the nearest database classes.
5) Observing that the correct class is always included in the first 100 ranked classes in our experiments. That's why we applied final classification on the highest 100 ranked classes in the matching set, which is 11% of the total database classes. This means more reduction in computation complexity.
6) Improving final classification accuracy through multi staging to be around 95.6% using DCNN. This reduction in error rate is due to applying classification on only 11% of the whole database. This enhanced the final performance achieved by DCNN architectures, which is due to the effect of clustering and ranking stages.
7) Comparing the effect of the most popular DCNN architectures in the field of Arabic handwriting recognition.

Before applying the matching stage, the data-mining technique is applied to the IFN/ENIT dataset to split the raw classes into clusters of similar classes, as detailed in section III-A. Based on the literature review, clustering was applied to enhance final recognition results, as reviewed in section II-G.

A. DATA-MINING PROCESS

The applied data-mining technique, shown in Figure 3, aims to represent the training section of IFN/ENIT as a big decision tree. Each node per tree-level includes a cluster of similar dataset classes. The similarity is determined based on a range of regional and geometric feature values [57]. Some constructed clusters intersect together by including some common classes. Some clusters are subsets of other bigger ones. That is why the dataset is represented as a tree-like model. This model relates smaller subsets to bigger supersets.

After building the training database model, each test image is matched against one of the constructed clusters. The Big O notation of this matching process is $O(log(v))$, where v is the number of clusters in the database. It is observed that the biggest cluster includes 423 different classes, i.e. the worst case is to pass only 49.2% of the database size to the second stage. Details of how the training tree-like model is built will be shown in the following three subsections.

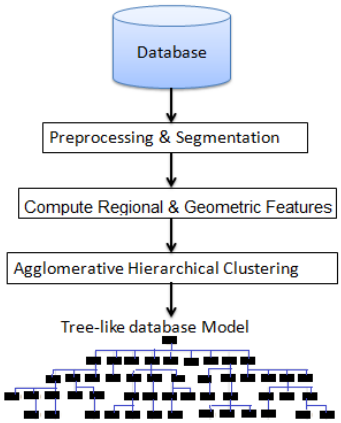


FIGURE 3. Database data-mining.

As shown in Figure 3, the database data-mining process starts by applying the preprocessing and the segmentation, as described in the next section III-A1.

1) PREPROCESSING AND SEGMENTATION

Passing testing and training database images through preprocessing and segmentation is essential for consistent post-analysis. This leads to better final recognition rates.

The database binary images are cropped and normalized to one pixel wide. This removes any variations due to using different writing tools or different handwriting styles. Image negatives are computed to concentrate computations on the word's pixels, as shown in Figure 4.

(a) Cropped Image (b) Image Negative (c) Image Thinning

FIGURE 4. A sample image during preprocessing.

After the preprocessing and segmentation process, a set of 13 regional and geometrical features is computed [57], as described in the next section III-A2.

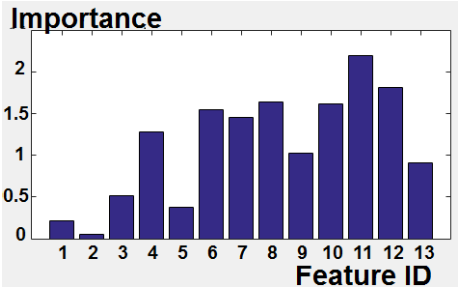
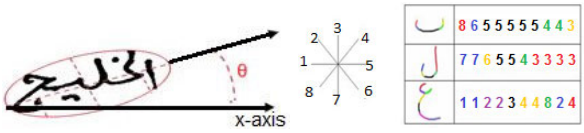


FIGURE 5. Feature importance.

2) REGIONAL AND GEOMETRIC (REG-GEO) FEATURES EXTRACTION

After computing the 13 regional and geometrical features [57], the Random Forest feature selection technique is then applied to select the best set of effective features from these 13 features. Feature selection is based on measuring feature importance. Importance measurement depends on how much the final recognition accuracy is affected by including the feature in the recognition process. Figure 5 shows the measured importance (y-axis) of the 13 features (x-axis). Accordingly, a set of 9 features are selected to be computed in this stage, numbered 4 and 6 to 13. The x-axis value 4 is for 'Extent' regional feature, while the values from 6 to 13 are a set of geometric features [57]. These nine features satisfy best-computed importance levels. The other features with lower importance are excluded from our approach. Those excluded features labeled 1, 2, 3 and 5, are the count of piece of Arabic words (PAWs), number of holes, eccentricity, and orientation with the x-axis, respectively.

Extent is a measure of the normalized word skeleton area. It depends on the word's area and its ratio to area of its bounding ellipse, as shown in Figure 6a. Image contours are represented by the set of geometric features, numbered 6 to 13. Geometric features are computed in different six image partitions [57]. Partitioning images into six different zones aims to consider line segments' position as a feature. Eight different concavity-based features are computed to represent the normalized length and number of occurrence $\hat{N}$ per line type in each partition. An example is shown in Figure 6b.



(a) Word Bounding Ellipse. (b) Concavity Features

FIGURE 6. Sample of geometric and regional features.

Line types are either horizontal or vertical or left/right diagonal. Normalized length $\hat{L}$ and occurrence $\hat{N}$ of each line type is defined by equations (1) and (2), respectively, where A_z is the zone area.

$$\hat{L} = \frac{L}{A_z} \tag{1}$$

$$\hat{N} = 1 - \frac{2N}{10} \tag{2}$$

The impact of feature values on model final accuracy is shown in Figure 7. The x-axis represents the number of constructed decision trees and the y-axis represents the overall percentage of matching error (ER). The Figure summarizes the effect of the 13 features on ER, using Random Forest feature selection technique. It is clear in Figure 7a that features labeled: 1 (number of holes), 2 (PAWs), 3 (eccentricity), and 5 (word orientation) cause 21% misclassification error, while Figure 7b shows that feature 4 (Extent) reduces misclassification error exponentially from 21% to 9%. Including the geometric features labeled 6-13 to feature 4(Extent) again reduces misclassification error to 6%, as shown in Figure 7c. That is why the chosen set of features, numbered (4,6 till 13), is only concerned in this stage.

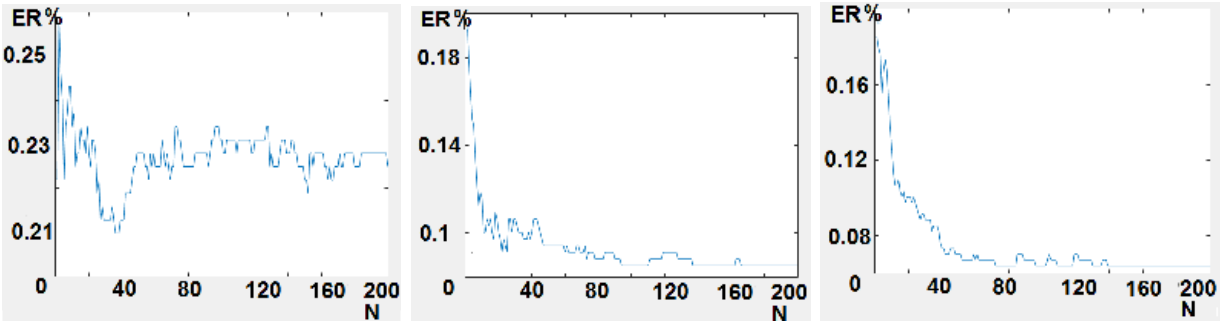
The database classes are then clustered based on the selected set of features, as described in the next section III-A3.

3) HIERARCHICAL AGGLOMERATIVE CLUSTERING HAC

The clustering technique [58] is used to structure database classes as a tree-like model. There are two different categories of hierarchical clustering algorithms: top-down and bottom-up. In bottom-up, each data point is initially a single cluster. Single clusters are then merged iteratively. Agglomerate is an alternative name to the merge operation. Merge operations are consecutively done, based on a distance measure between each pair of clusters, until all clusters are included in one whole cluster in a hierarchical manner. That is why this category of clustering is known as Hierarchical Agglomerative Clustering (HAC). The root of the tree is the entire database classes included in one unique cluster.

The first advantage of this clustering technique is that the number of clusters is not necessarily become known before building the tree-like model; it is determined while building the tree. The second advantage is that the hierarchical clustering criteria fit the hierarchical structure of the database classes. The main disadvantage is the quadratic time complexity $O(n^3)$ relative to other linear clustering algorithms, for example, K-Means and GMM. Although linearity would seem an attractive property in K-Means and GMM, the number of clusters should be pre-determined and there should not be any intersections between the constructed clusters. That is why HAC is preferred and more applicable to our working conditions.

Testing images are matched with one of the clusters after applying preprocessing, segmentation and feature computation as mentioned in section III-A1 and III-A2. The matching process is performed based on satisfying the defined range of feature values.



(a) Features 1,2,3,5 (b) Feature 4 (c) Feature 4,6-13

FIGURE 7. Effect of The selected features on classification error, (N = number of decision trees), (ER% = error rate), (1 = count of piece of Arabic words (PAWs), (2 = number of holes), (3 = eccentricity), (4 = extent), (5 = orientation with x-axis), (6-13 = eight geometric features).

Algorithm 1 Matching Algorithm

```
input: testimage t, Database of clusters DB[c], Max Cluster Size S;
initialize:list Parent[c];
//sort database clusters in ascending order
sort(DB[c]);
//build database tree model
for cluster size s = 1 to S do
    found = 0;;
    ∀ Cluster C of size s & ∀ SuperCluster SC of size >s
    if C ⊂ SC&found ≠ 1 then
        Parent(C) = SC;
        found = 1;
    end
end
//Compute Regional and Geometric Features Reg-Geo(t);
//Search the tree for the matching cluster of test image t SearchTree(t, Parent);
Result: C_t = MatchingCluster(t)
```

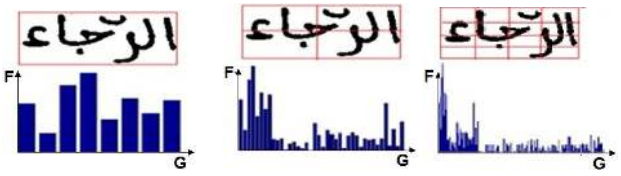
Each matching cluster includes classes with the same range of the selected regional and geometric features. Different clusters/sets may include common classes. Some smaller sets may become a subset of other bigger sets that may allow the matching process to be implemented as a binary search tree, reducing complexity to $O(log(v))$ where v is the number of clusters in the database. The test image is now ready to be matched with a set of similar database classes included in one of the clusters. The test image and its matching cluster are both ready for the ranking stage. Algorithm 1 represents the matching process and how the clusters are related to each other in the form of a tree. The inputs to this algorithm are the test image, the database clusters, and the max cluster size. The algorithm runs iteratively on all the clusters in an ascending order to relate each cluster C with its super-cluster SC. By default, the database of all the clusters is a parent node to all the smaller clusters. Finally, the output of the algorithm is the cluster matching the input test image based on the computed nine Reg-Geo features.

The next stage III-B describes how the classes of the matching cluster are ranked relative to the input test image.

B. STAGE 2: RANKING STAGE (PHoG AND KL-DIVERGENCE)

Our ranking approach starts by computing the pyramid histogram of orientation gradients (PHoG) features [59]. These features are computed for each input image and its matching cluster's members. They represent a statistical-descriptor which was stated to achieve robustness, as reviewed in section II-B and II-D.

This statistical-descriptor is based on dividing images into segments at different resolutions called levels L, then extracting histogram of quantized oriented gradients (HOG) [60]. This form of calculation analyzes orientation information at different resolutions and levels L, as shown in Figure 8, which extracts fine details and effectively discriminate between the information of different words skeletons [37].



(a) Level=1 (b) Level=2 (c) Level=3

FIGURE 8. Pyramid histogram of gradients (F: frequency, G: oriented gradient).

Representing oriented gradients by a single histogram, as a quantized feature space, provides visual vocabulary known as bag-of-words [61]. This way of feature representation is an effective way of measuring similarity. PHoG effectiveness is achieved by representing local shapes of the Arabic handwritten words, in addition to extracting the spatial layout of word shape information. The ordinary HOG (quantized into K bins) is responsible for the extraction of local shapes. Each bin represents the frequency of edges within a certain range of angular orientations.

The main contribution in the PHoG compared to the ordinary HOG is satisfying the concept of spatial layout representation by segmenting the image into sub-regions. At each level L, segmented regions/grids are doubled relative to the number of grids at level $L-1$, as shown in Figure 8. The x-axis represents the quantized oriented gradients and the y-axis represents their corresponding normalized histograms. The spatial distribution of edges is captured in a 1-dimensional feature vector. This vector concatenates HOG of all segmented sub-region at each level L. The feature vector length is proportional to the number of bins K and the number of levels L, as defined in (3).

$$K * \sum_{l \in L} 4^l \tag{3}$$

Although according to literature, the most preferred gradient mask for computing the oriented gradients was the centered 1-D mask $[-1, 0, 1]$, the Prewitt 2-D edge detector mask [62] works better in computing edge information in our case. Orientation values $[0, 360°]$ are quantized to eight orientations. The effect of bins numbers and orientation range on the final true positive rate (TPR) of the system is shown in Figure 9. The figure shows an increase of the TPR when more ranks are considered. The x-axis represents the top N ranks, at $N = 1,20,40,60$.

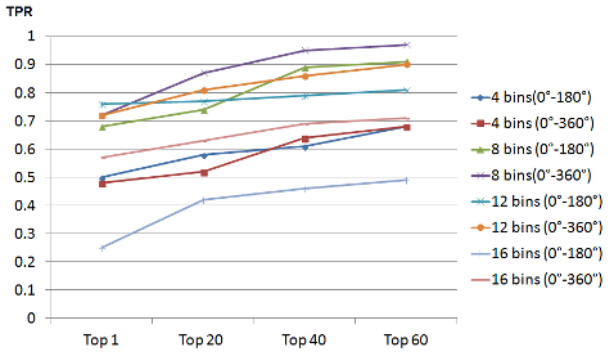


FIGURE 9. Effect of number of orientation bins on true positive rate (TPR).

Range of orientation bins is tested when evenly spaced over $[0°, 180°]$ and another time when spaced over $[0°, 360°]$. The $[0°, 180°]$ range represents the unsigned gradients, while the $[0°, 360°]$ range represents signed gradient. It is clear in Figure 9 that fine coding and increasing the number of orientation bins show better performance for a number of bins less not exceeding eight, then no improvement is achieved beyond that. Increasing the number of bins to more than eight causes overfitting in our situation. It is shown that ignoring signed gradients degrades performance, even after doubling the number of bins. As a conclusion, including sign information is essential for preserving information representing Arabic handwritten words.

The eight angular histogram bins are normalized to form the feature vector using L1-sqrt. Considering 8 bins $[0°, 360°]$, the effect of different block normalization methods [60] on the true positive rate (TPR) is shown in Table 1. It is clear in the table that L2-hys, L2-norm, and L1-sqrt achieve almost the same performance, L2-norm outperforms L1-sqrt in early top ranks (first three left columns), while the similarity between them appears in high top ranks (forth column). L1-sqrt is preferred for lower computation complexity due to passing the top 100 ranks from this stage to the next final stage. L1-norm achieves relatively lower performance, while total neglection of normalization causes great degradation as shown in Table 1. The definition of L2-norm, L1-sqrt, and L1-norm block normalization methods are defined by equations (4), (5), and (6) respectively. L2-Hys normalization is simply the original L2-norm clipped to 0.2 and re-normalized [63].

TABLE 1. Effect of normalization techniques on true positive rate (TPR).

Normalization (Technique)	Top 1	Top 20	Top 40	Top 60
L2-hys	0.71	0.82	0.95	0.96
L2-norm	**0.72**	**0.87**	**0.95**	0.97
L1-Sqrt	0.70	0.81	0.94	**0.97**
L1-norm	0.62	0.69	0.78	0.82
without normalization	0.58	0.63	0.75	0.79

$$\hat{v} = \sqrt{\frac{v}{\| v \|_2^2 + \epsilon^2}} \tag{4}$$

$$\hat{v} = \sqrt{\frac{v}{\| v \|_1 + \epsilon}} \tag{5}$$

$$\hat{v} = \frac{v}{\| v \|_1 + \epsilon} \tag{6}$$

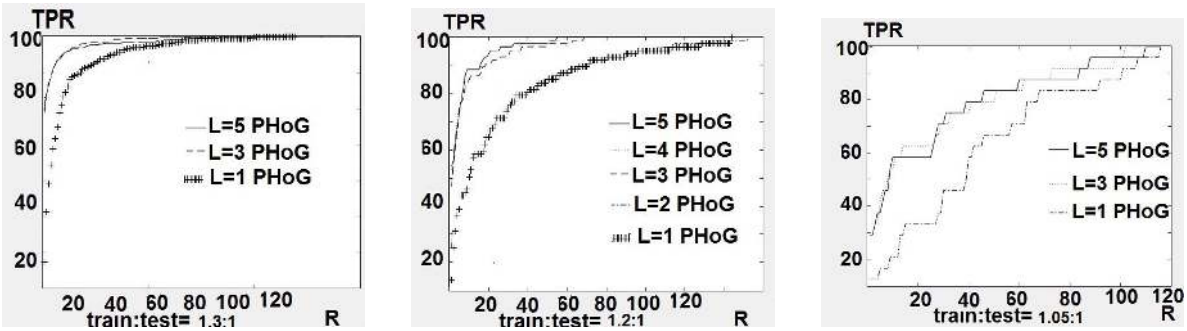
where v is the feature vector before normalization, $\hat{v}$ is the normalized feature vector, and ϵ is a small corrective constant value.

At this point, handwritten words are characterized as probability distributions of local gradients. This characterization enables ranking of matching classes relative to the test image by measuring divergence between their probability distributions [64]. Accordingly, the next step in our ranking algorithm is measuring Kullback-Leibler (KL) divergence [65]. It is a generalized measure of Shannon entropy, as defined by (7). The significance of entropy in measuring information was stated [66], and complemented by KL-Divergence [67].

$$H(p(x)) = -\sum_{x \in X} p(x) \ln p(x) \tag{7}$$

KL-Divergence is measured between PHoG feature vector of the test image and PHoG feature vectors of cluster's members. Based on the measured divergence, the classes of the matching cluster are ranked in ascending order relative to the test image. This process aims to pass only highly ranked cluster's classes to the final classification stage as shown in our system overview (Figure 2).

KL-divergence $D_{KL}(p(x) \parallel q(x))$, defined by equation (8), is a non-symmetric measure of divergence between probability distributions. It is a non-negative measure, measured value



(a) $Label = 30$ (b) $Label = 20$ (c) $Label = 5$

FIGURE 10. Expected true positive recognition rates TPR for database classes labeled 30, 20, and 5 versus correct class rank *R*.

from one distribution $q(x)$ to another distribution $p(x)$ differs from $q(x)$ to $p(x)$. $p(x) = q(x)$ is the only case that causes divergence to become zero.

$$D_{KL}(p(x) \parallel q(x)) = \sum_{x \in X} p(x) \ln \left(\frac{p(x)}{q(x)}\right) \tag{8}$$

KL-divergence $D_{KL}(p(x) \parallel q(x))$ (8) can be expressed as a relation in Shannon entropy [68] as shown in equation set (9)

$$\begin{aligned} D_{KL}(p(x) \parallel q(x)) &= \sum_{x \in X} p(x) \ln \left(\frac{p(x)}{q(x)}\right) \\ &= \sum_{x \in X} p(x) \ln p(x) - \sum_{x \in X} p(x) \ln q(x) \\ &= -H(p(x)) - H(q(x)) \\ &\quad - 2 \sum_{x \in X} \frac{p(x) + q(x)}{2} \ln q(x) \end{aligned} \tag{9}$$

KL-divergence $D_{KL}(p(x) \parallel q(x))$ is quite effective in measuring divergence between probability distributions due to the following properties,

1) Non-negative measure, where $D_{KL}(p(x) \parallel q(x)) \geq 0$, and it equals 0 if and only if P = Q.
2) Convex measure in respect to both P and Q.
3) Satisfy Independence, that is when two independent factors X and Y are considered, then (10) is satisfied

$$\begin{aligned} D_{KL}(p(x, y) \parallel q(x, y)) &= D_{KL}(p(x) \parallel q(x)) \\ &\quad + D_{KL}(p(y) \parallel q(y)) \end{aligned} \tag{10}$$

According to the measured divergence, matching classes are ranked before classifying stage. Figure 10 shows the relation between the expected TPR (y-axis) and the different ranks (x-axis). It was found that the correct class rank is always included in the top 100 ranks as shown in Figure 10a, 10b, and 10c. That is why only 100 classes, which are only 11% of the total database classes, are passed to the final stage for classification. The steps of the ranking algorithm are shown in Algorithm 2.

It is shown in Figures 10a, 10b, and 10c that faster saturation is always achieved by higher PHoG levels. The proposed approach categorized the database into three main parts:

Algorithm 2 Ranking Algorithm

```
input: testimage t, cluster c[x], level L;
initialize:list d[x] = 0;
for i = 1 to x do
    // measure divergence between PHoG feature vectors
    d[i] = D_KL(PHoG(t, L), PHoG(c[i], L))
end
// sort classes based on d
sd = sort(d, cluster);
Result: sd[1 : 100]
```

labeled 30, 20, and 5, according to the average increase in training samples over testing samples.

The fastest saturation achieved in Figure 10a of the database part labeled 30 is due to the availability of enough training samples relative to testing ones. This is a crucial property that is required in our DCNN comparative study. DCNN is a classifier that needs enough training samples for efficient training. Saturation in Figures 10b and 10c is slower than the one achieved in Figure 10a i.e., high recognition rates are not easily attainable. The final classification stage is detailed in the next section III-C.

C. STAGE 3: CLASSIFICATION USING DEEP CONVOLUTIONAL NEURAL NETWORK

Convolutional neural networks (CNNs) were introduced 20 years ago [69] for visual object detection and recognition. The usage of CNNs achieved robust feature extraction and classification, as reviewed in section II-E. Many contributions in the CNN structure were recently reported to construct deep-learned DCNNs. In deep learning, propagating information or gradient of input images through many layers may cause excluding many important parts of this information. That is why many recent publications proposed different architectures as solutions to create a short path of layers while achieving the deep learning concept. The most popular CNN architectures are Residual Networks (ResNets) [15], AlexNet architecture [70], VGG-16 architecture [13], GoogleNet architecture [14], ResNeXT [71], and DenseNet [72].

These different CNN architectures vary in number and type of layers. These variations are based on the nature of application, data size, and complexity. Different types of layers are input layer, convolution layer, batch normalization layer, pooling layer, dropout layer, and output layer [73]. Convolution process constructs feature maps using filter weights, and bias.

Designing deep architectures is a challenging task with a big set of hyper-parameters (width, filter, sizes, strides…etc.). Research efforts concentrated on two main architecture-based categories of CNN as shown in Figure 11, i.e. classic network architectures and modern network architectures.

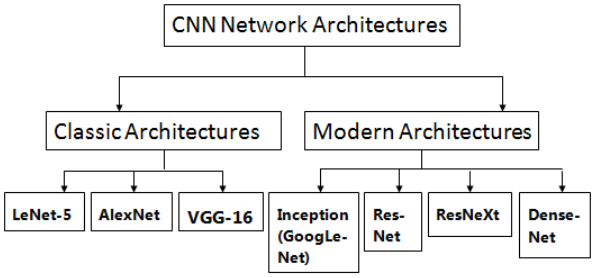


FIGURE 11. CNN architectures.

Classic network architectures represent the traditional CNNs, resembling the LeNet-5 architecture [74], where successive stacked convolutional layers are applied with few new modifications. LeNet-5 was the first effective CNN model that was designed to recognize handwritten digits in postal service. It outperformed its former architectures in the field of document recognition and reading bank checks. The modern architectures [15], [71], [75], [76], added more impressive contributions to the traditional CNNs. New innovative ways for constructing convolutional layers are applied to achieve more efficient learning. These architectures were applied previously in different applications and especially in text recognition, as described in the following subsections.

1) AlexNet

AlexNet architecture [70], as shown in Figure 12, would seem similar to LeNet-5 architecture, but relatively deeper.

The architecture includes five convolutional layers in addition to some max-pooling layers and three fully-connected ones. The softmax function is applied to the output of the final layer to differentiate between the 859 classes in IFN/ENIT database. Rectified Linear Unit (ReLU) used at the convolution and the fully-connected layers. ReLU breaks up the linearity that may be caused by the convolution process. The number and size of convolutional filters are shown in Figure 12.

2) VGG-16

A new very deep architecture [13] called VGG network was introduced for offline handwriting recognition [45], [77].

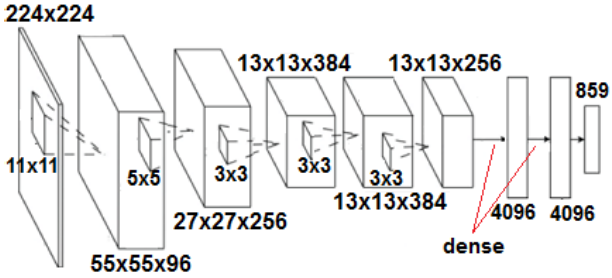


FIGURE 12. AlexNet architectures [70].

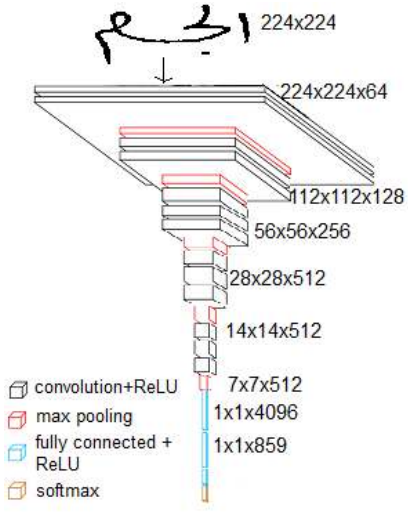


FIGURE 13. VGG-16 CNN architecture [13].

Although it is deeper as shown in Figure 13, yet it is simpler than the previous architecture [70].

Simplicity was achieved by proposing a new strategy to design very deep networks based on equally sized building blocks. The increased depth is compensated by applying only small convolution filters of size 3×3.

Input images of size 224×224 are passed through a sequence of convolutional layers. Layers' details are shown in Figure 13. ReLU non-linear function is employed. Response normalization did not contribute and consumed more memory and time.

3) INCEPTION (GoogLeNet)

The concept of Inception network was first introduced by researchers at Google [14]. This type of network was applied for classification and detection purposes. The model adopts the concept of the ''Inception cell''. In the inception cell, shown in Figure 14, multiple convolution operations are applied at different scales. Input channel depth was reduced by applying 1×1 convolutions. For each cell, a set of 1×1, 3×3, and 5×5 convolution filters were used. These multiple convolutions aim to extract and merge features at different scales. Max pooling and same-padding were applied to unify dimensions of features to enable their concatenation.

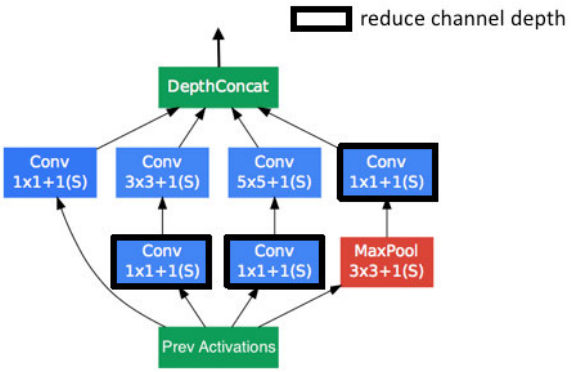
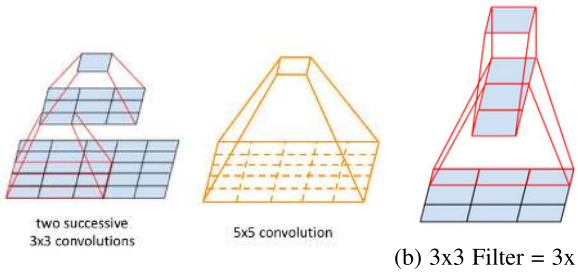


FIGURE 14. Inception cell.



(a) 5x5 Filter = Two 3x3 Filters (b) 3x3 Filter = 3x1 and 1x3 Filters

FIGURE 15. Replacing big spatial filters by successive smaller ones.

The pooling layer is responsible for non-linear down-sampling and reduces feature map spatial size.

Large convolution filters, for example, 5×5 or 7×7, are capable to extract features at large scale; however, they suffer from high computation complexity. An alternative solution was proposed to replace such large filters, as shown in Figure 15a. It was found that a 5×5 convolution filter can be replaced by two stacked 3×3 successive filters. In that way, instead of using a number of $25c$ parameters per $5 \times 5 \times c$ filter, this alternative filter design requires only $18c$ parameters when applying two $3 \times 3 \times c$ filter with linear activation between the two filters, where c is the number of channels. For better computation complexity, 3×3 convolution filter is alternatively replaced by another two successive convolution filters of sizes, 3×1, and 1×3, as shown in Figure 15b.

4) ResNet

Deep residual networks (ResNet) are deeper networks with hundreds of layers instead of tens of layers. The target of building very deep networks is learning more complex data with high accuracy. Nonetheless, adding a large number of layers may cause some negative effects, and the occurrence of the degradation problem. Residual Networks (ResNet) is one of the proposed solutions to overcome degradation. This type of network is composed of residual blocks as shown in Figure 16. The identity shortcut, x, is added to the output of the applied residual mapping function to preserve the computations of the stack layers without adding extra

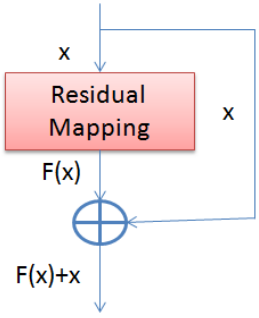


FIGURE 16. Residual block.

parameters or increasing computation complexity. Different residual blocks [15] were introduced to ease training and achieve high accuracy.

5) ResNeXt

Another contribution to the deep residual network is ResNeXt architecture [71] by applying the ''split-transform-merge'' strategy instead of the standard residual block. Instead of calculating a series of convolutions over the full feature map, convolutional filters are applied in a parallel manner and merged to form feature maps.

It is a multi-branch architecture with a small set of hyper-parameters. Group convolutions are applied in a parallel to save computation and processing time. The main contribution is extracting features of various characteristics by the split process.

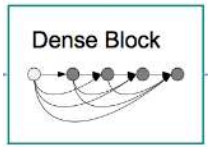
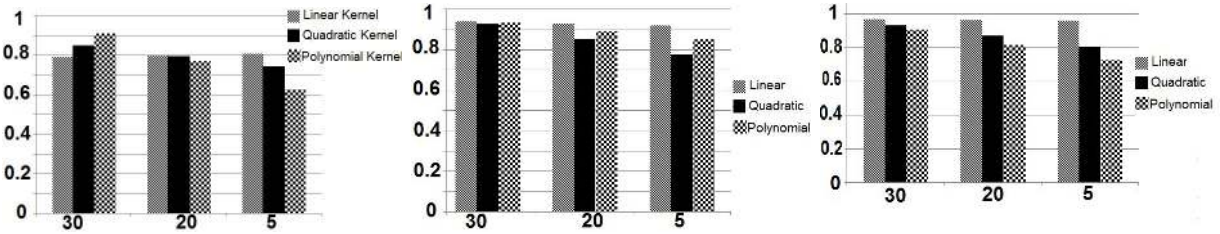


FIGURE 17. DenseNet block.

6) DenseNet

The traditional CNNs are composed of L layers with L connections; one connection per two successive layers. Dense CNNs are composed of $L(L+1)/2$ direct connections. Each layer makes use of all feature-maps computed in all preceding layers. This enables such networks to outperform the previously described ones. It is considered a very powerful solution to the vanishing-gradient problem [78], by propagating feature maps, and reducing the number of parameters. Less computation is resulted while higher performance is achieved. Concatenation of all referenced feature maps from earlier successive layers constructs final feature maps. The number of filters used per layer is called growth rate k, i.e. each layer adds k more channels than its preceding layer. According to literature, DenseNet achieved better performance when compared to the ResNet models. Finally, DenseNet architecture is constructed by replacing the main unit in ResNet model architecture by the dense block, as shown in Figure 17.



(a) Level 1 PHoG (b) Level 3 PHoG (c) Level 5 PHoG

FIGURE 18. System recognition rate using SVM, x-axis represents the three labeled database parts, y-axis represents the final achieved recognition accuracy.

IV. EXPERIMENTAL RESULTS

The IFN/ENIT database [29] is one of the challenging Arabic handwriting datasets. It provides training and testing images to support constructing handwriting recognition systems. It is composed of about 2200 (300 dpi) binary images of handwriting sample forms from different 411 writers. 26,000 binary word images represent town/village names that have been extracted from the forms and saved individually in five different sets. These five sets are labeled *a*, *b*, *c*, *d* and *e*. Formerly, sets *a*, *b*, and *c* were used for training, while set *d* was used for testing. Experiments were also achieved using sets *a*, *b*, *c*, and *d* for training, while set *e* for testing. The number of different classes per set is shown in Table 2. The number of samples varies from one class to another. The database provides researchers with a ground truth file for each word. We categorized the database into 3 parts. The first part includes the classes where the number of training samples exceeds the number of testing samples by 5%. The second part includes the classes where the number of training samples exceeds the number of testing samples by 20%. The third part includes the classes where the number of training samples exceeds the number of testing samples by 30%. Table 2 summarizes the statistics of the database part used in our experiments.

TABLE 2. Detailed description of the IFN/ENIT [29].

Set	No. of classes label 5	No. of classes label 20	No. of classes label 30	Total No. of Classes	Total No. of words	Total No. of Paws	Total No. of characters
a	207	443	174	824	6,537	28,298	51,984
b	242	482	160	884	6,710	29,220	53,862
c	188	486	156	830	6,477	28,391	52,155
d	208	483	168	859	6,735	29,511	54,166
e	250	421	158	829	6,033	22,640	45,169

Before testing our approach, ground truth files have been compiled and images of the same city name are grouped in a folder. By doing that, the whole dataset is organized in folders instead of having raw data images. Each folder represents a class that has a city name and contains all the images of that city written by different writers.

A. MATCHING PROCESS SENSITIVITY

There are a different number of classes per cluster. Different clusters' sizes have different frequencies. Sensitivity [79] is

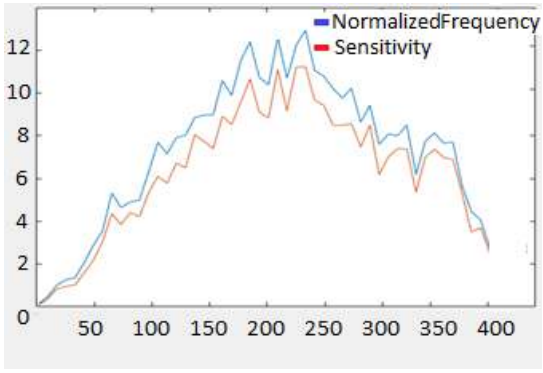


FIGURE 19. Matching process sensitivity (x-axis: cluster size).

measured by the degree of correct class inclusion in the matching set. Based on the Area Under Curve (AUC) calculation in Figure 19, the difference between the normalized frequency of clusters sizes, and its corresponding measured sensitivity represents 13% of the whole area under the curve. The weighted average matching set size computed from equation (11) is approximately equal to 172.6 different classes. This average is about 18% of the whole database number of classes. This causes passing only an average of 18% of the database to the next stage for ranking.

$$\bar{x} = \frac{\sum_{i=1}^{n}(x_i * w_i)}{\sum_{i=1}^{n} w_i} \tag{11}$$

$\bar{x}$ is the weighted average, x is the set size and w is the normalized weights of sets. The largest set includes 423 different classes (worst case) which is 44.7% of the total database 946 training classes, as shown in Figure 19.

TABLE 3. Effect of clustering algorithms on Alexnet-CNN and IFN/ENIT.

Proposed Approach	Training-testing	Recognition rate	Biggest Cluster Size	Training time (min)
Without Clustering Without Ranking	abc-d	76%	946 classes (whole dataset)	72.1
With Clustering & Without Ranking	abc-d	84.1%	423 classes	36.5
With Clustering & Ranking	abc-d	**95.6%**	100 classes	18.15

Table 3 shows the effect of the clustering algorithm on the performance of the Alexnet-CNN classifier when applied to

the IFN/ENIT dataset. This experiment used sets *a*, *b*, and *c* for training, while set *d* was used for testing. When AlexNet was applied to the whole 946 database classes, without any clustering or ranking, it was trained in 72.1 minutes and achieved 76% recognition rate. On the other hand, when clustering was applied prior to using Alexnet and without any ranking, the recognition rate was increased to 84.1%. Finally, the outperforming recognition rate was achieved when applying clustering and ranking consecutively, before classification. It is shown in Table 3 that the training time was reduced exponentially in the final third case. The best recognition rate was achieved with the support of clustering and ranking stages.

B. THE RANKING PROCESS

For each test image, the ranking process causes the correct class to advance and occupies an early position among the cluster members. This causes faster higher recognition rates as shown in Figure 10a. Fast saturation, approximating 100% recognition rates, proves the effectiveness of the ranking process, and its positive effect on the final classification stage.

Ranking cluster's members and leading of correct classes' ranks in advanced positions enable passing only a subset of high-rank classes to the classification stage. After clustering, the maximum number of classes used for training the classifier is reduced from 845 to 423. While ranking the cluster members caused only 100 classes to train the classifier, reducing the number of participant classes in final classification improved final recognition rates from 84.1% to 95.6%, as shown in Table 4 and thus reduced computation and time complexity.

TABLE 4. Effect of clustering and ranking algorithms on Alexnet-CNN and IFN/ENIT.

Proposed Approach	Training-Testing	Recognition rate using ALEXnet	No. of classes in Classification
Without Ranking	abc-d	84.1%	423 classes
With Ranking	abc-d	**95.6%**	100 classes

Figure 20 shows a sample of the database clusters after applying the ranking algorithm. Each test image is attached with its matching cluster. Only a small part of the cluster members are shown in the figure. It is clear that the rank of the correct matching class varies per test image, but proceeds in early positions.

C. CLASSIFICATION STAGE

Classification is applied only to a subset of high-rank classes. Two different types of classifiers were applied; SVM and DCNNs. The performance of SVM is shown in section IV-C1. Detailed experiments using DCNNs are shown in section IV-C2.

1) CLASSIFICATION USING SUPPORT VECTOR MACHINES (SVM)

The SVM is applied with different kernel types at different levels of PHoGs, as shown in Figure 18. The figure shows

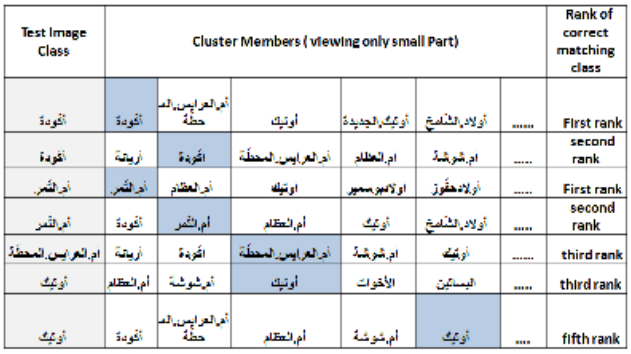
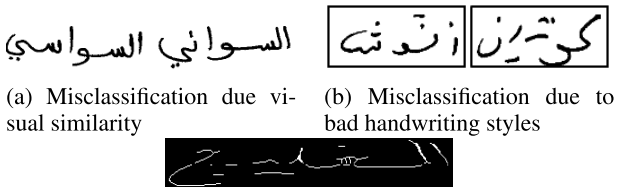


FIGURE 20. Sample of clusters and early ranks of correct matching class.

the final recognition accuracy (*y*-axis) that was reported on the three labeled database parts (*x*-axis). The linear SVM with the fifth PHoG level achieved the highest recognition rate. The linear kernel gave the superior classification results, while the polynomial kernel was better than the quadratic. The classification error rate was due to different reasons: visual similarity as some samples have common PAWs, as shown in Figure 21a, bad handwriting styles that may mislead the system, as shown in Figure 21b, inaccurate thinning results that may remove important details, as shown in Figure 21c and cause misclassification, and finally, lack of training samples in some database classes.



(a) Misclassification due visual similarity

(b) Misclassification due to bad handwriting styles

(c) Misclassification due to inaccurate thinning results

FIGURE 21. Samples of misclassified images.

Table 5 represents similar systems' experiments and their results compared to our proposed approach using SVM. Some classification systems applied word segmentation to character level, while others did not, as shown in the table. Segmentation helps in predicting the correct sequence of word characters based on a previously defined dictionary, but it increases computation complexity, and segmentation error affects final recognition rates. The proposed approach is applied without segmentation and without use of the previously defined dictionary. Similar systems are evaluated using the word error rate (WER) metric. The approach is applied one time by using sets *a*, *b*, and *c* as training sets, while set *d* is used as testing. The approach is applied another time using sets *a*, *b*, *c*, and *d* as training sets, while set *e* is used as testing. It is shown in Table 5 that some approaches applied HMM classifiers. The HMM-based approaches [39] depend on character modeling, as shown in Figure 22. Each model has a defined number of hidden states and transitions.

TABLE 5. Comparative results of systems applied on IFN/ENIT.

Features-Classifiers [Approach reference]	Training-testing	Segmentation/ Character Modelling	WER
Derivatives-HMM[17]	abc-d abcd-e	Character Modelling	**5.59** % **14.55**%
Distribution & Concavity-HMM[39]	abc-d	Character Modelling	12.8%
Distribution, & Baseline-HMM[24]	abc-d	Segmentation and Character modelling	9.04%
Structure & baseline -HMM[47]	abc-d	Character modelling	10.76%
Statistical & Structural-HMM[28]	abc-d	Character modelling	11.88%
Statistical, Skeleton & Structural-SCHMM[27]	abc-d	Character modelling	16.21%
PCA sliding window -HMM[21]	abc-d	character modelling	7.14%
Sliding & Skeleton-SCHMM[23]	abc-d	Character modelling	10.9%
Intensity-SCHMM[31]	abc-d	Character modelling	10.26%
Distribution & -Concavity -HMM[32]	abcd-e	Character modelling	15.85%
MSHMM[26]	abc-d abcd-e	Character modelling	16.9% 27.6%
Denstity-HMM[19]	abcd-e	Character modelling	21.05%
Multiple Sliding Windows-HMM[22]	abc-d abcd-e	Character modelling	**2.3** % **6.56**%
Surf-SVM[7]	18 classes	without	15%
Derivatives-Multi [5] stage HMM	abc-d abcd-e	Sub-Core-Shape models	**2.29%** **5.24%**
Proposed approach using SVM	abc-d abcd-e	Without Segmentation Without Character Modeling	**3.6**% **14.6%**

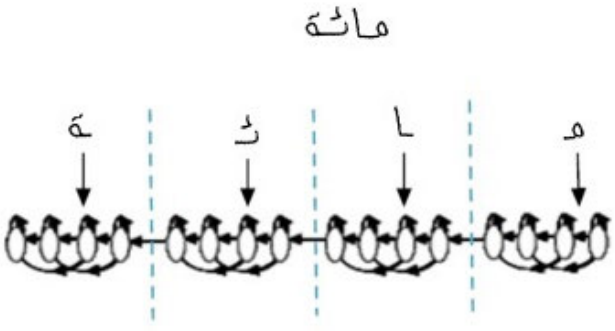


FIGURE 22. Character modeling in HMM.

Modeling defines transition and observation probabilities. Recognition is based on the sequence of the characters that achieve maximum likelihood. Features are extracted using a sliding window.

It is shown in Table 5 that our proposed solution is among the four highest achieved accuracy. However, the proposed approach did not model any characters, and words were recognized in a holistic manner without any character-level segmentation. Some approaches proposed a combination of different HMMs for final decision [22], [24], to improve final performance.

Other approaches [47] applied re-ranking to improve HMM performance. Still others [27] stated outperforming of gamma distributions than the Gaussian distribution in modeling states. Semi-continuous HMM (SCHMM) [23], [27], [31], achieved improved performance than the HMM. In our approach, different techniques were applied in a consecutive fashion.

2) CLASSIFICATION USING DEEP CONVOLUTION NEURAL NETWORKS (DCNN)

Applying DCNN classifier requires the existence of enough training samples. That is why it is applied in our study on IFN/ENIT dataset classes labeled 30 only. These classes were tested after training the system with their cluster members. The number of training classes per cluster is 100 based on the analysis done during the ranking stage.

Figure 23 shows the achieved experiments using the different CNN architectures. The summarized experiments in the figure are based on using sets *a*, *b*, and *c* for training and using set *d* for testing. AlexNet achieved the best testing accuracy at the learning rate of 0.01. Almost all the testing samples were recognized correctly, as shown in Figure 23a. AlexNet outperformed the other architectures in time and accuracy, as shown in Figure 23a. Accuracy is measured at three different learning rates. Increasing the learning rate causes a decrease in total training time. The second best architecture is VGG-16, the highest accuracy is also achieved at 0.01 learning rate. VGG-16 shows more time and memory complexity than AlexNet. Based on Figure 23, it can be deduced that the more advanced and complex the architecture is, the lower accuracy is achieved on IFN/ENIT dataset. The clarification of this behavior is that the complexity of the CNN architecture can be compatible with data complexity. IFN/ENIT dataset is composed of binary images, and all images are thinned to one pixel wide after our early preprocessing stage. Clearly, this is a simple form of data that requires CNN of moderate complexity.

It is shown in Figure 23 that some architectures failed to classify test images at specific learning rate values. The learning rate [80] is one of the most important hyper-parameters that deserves an effort to be tuned and to reach the optimal value. Small learning rates may cause high training time and high training errors. On the other hand, large learning rates may cause a high increase in the CNN weights and this may result in overfitting. Therefore, learning rates should neither be too large nor too small. The typical values of learning rates should range from 1 to 0.000001. A comprehensive research is essentially needed to find the optimal design of a CNN for achieving the best performance [55]. Another justification for the achieved low results in Figure 23 is the deepness of some architectures that do not suit small-sized dataset. It was stated that the deeper the architecture is, the lower the identification rate is achieved [81]. Deep architectures are also sensitive to noise and image degradation [82] that may be attributed to the process of thinning images to one pixel wide.

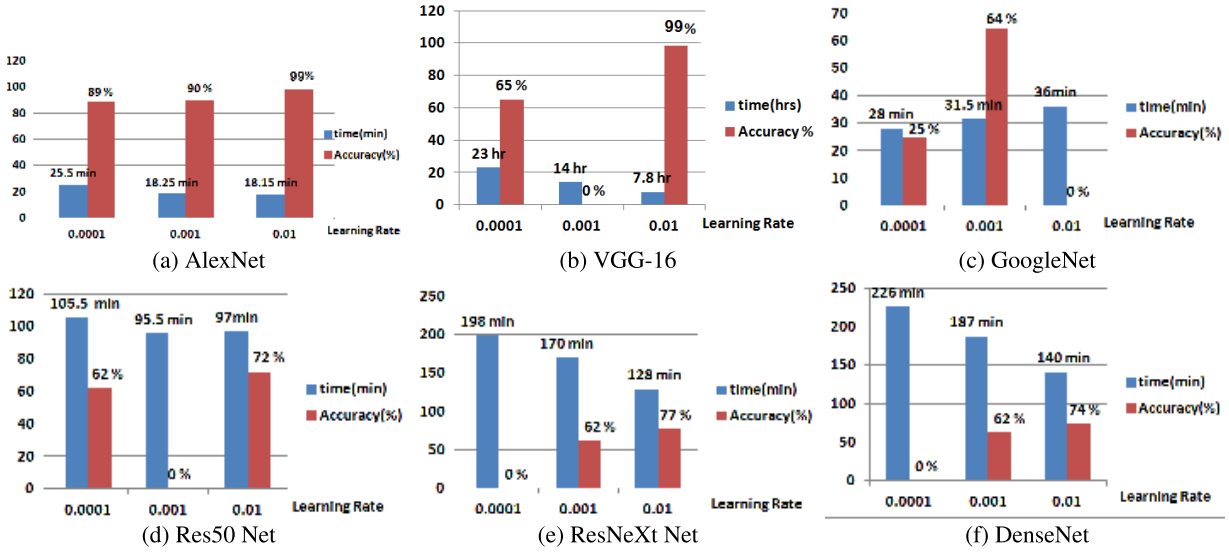


FIGURE 23. Achieved accuracy and testing time versus learning rate using DCNN (abc-d).

TABLE 6. Comparative results of NN-based systems applied on IFN/ENIT (abcd-e).

Reference	Features Extraction	Classifier	Recognition rate
[30]	BLSTM	RNN	75.72%
[25]	DCT	NN	90.73 %
[34]	CNN	BLSTM-CTC	92.21%
[35]	CNN	SVM	92.95%
[45]	CNN	CNN	**97.07**%
[48]	RNN	RNN	87.94%
[36]	PCOH	CNN	92.14%
[83]	MCDNN	MCDNN	91.5%
[43]	CNN	HMM	88.95%
[44]	CNN	HMM	89.23%
[49]	DBN	DBN	**94.99**%
[41]	MDLSTM	Maxout	89.9%
[18]	ResNet	multiple ResNet models	93.37%
[40]	RNN	CTC	91.43%
[51]	Convolutional DBN (CDBN)	SVM	83.7%
[33]	CNN	Dynamic Bayesian Network	**95.2**%
Proposed approach	CNN	AlexNet	**95.6**%

Table 6 compares our achieved results and others' results, using different types of neural networks (NN). This comparison is based on using sets *a*, *b*, *c*, and *d* for training, and set *e* for testing. The CNN when applied as feature extractor and Bidirectional Long Short-Term Memory (BLSTM) followed by a Connectionist Temporal Classification layer (CTC) as a classifier on the IFN/ENIT database achieved 92.21% [34], and when CNN was applied as feature extractor and SVM as a classifier on only 56 classes [35], the recognition rate was 92.95%. When CNN was applied as a classifier [36] on Pyramidal Histogram of Characters (PHOC) features, the achieved recognition rate was 92.14%. A recent approach [49] applied a deep belief network (DBN) for feature extraction and classification, and the achieved recognition rate was 94.99%. Their enhanced results were based on character-level segmentation using a morphological algorithm, but there was an additional 6.5% segmentation error. Long Short-Term Memory (LSTM) network [48] achieved a 12.06% character error rate when applied to IFN/ENIT. A multi-column deep neural network (MCDNN) [83] achieved better recognition results in comparison to Recurrent Neural Network (RNN), when applied on IFN/ENIT.

It was always demonstrated that the achieved results using CNN features [43] are more efficient than those obtained by other hand-crafted features. Another recent approach [33] achieved superior recognition rates by applying CNN as a feature extractor followed by a dynamic Bayesian network for classification. The present multi-stage approach when applied to the database, achieved 95.6% using AlexNet CNN architecture without any character-level segmentation. Observation states the effectiveness of the proposed multi-staging approach with AlexNet CNN as a classifier. Based on the literature, the superior recognition results [45] shown in Table 6 were due to building a set of Arabic characters and training the CNN on the Arabic uni-grams rather than the whole words. Classification in their outperforming approach [45] was supported by a matching dictionary.

V. COMPLEXITY ANALYSIS

The complexity of the hierarchical clustering technique in terms of database total number of classes is $O(n^2)$ for space and $O(n^3)$ for time. The similarity matrix is 2-D, and needs square on n to be stored in RAM, while n iterations are required for updating the similarity matrix and restoring it. After storing the database as a tree-like model of clusters along with their computed range of features, test images are matched with one of these clusters. Although the complexity

of the clustering technique is very high, building the tree-like model reduces matching process complexity to $O(log(v))$ where v is the number of clusters. The biggest cluster includes 423 different classes; this is only 49.2% of the whole database size.

If the number of classes in a cluster is L, then KL divergence performs L operations in the second stage. This results in $O(L)$ space and time complexity. After ranking clusters members, it was observed that the correct class maximum rank is 100, as shown in Figure 10a with the 5^{th} level PHoG features. This observation enables passing only a subset of 100 (maximum) classes to the classification stage. This maximum number of classes is only about 11% of the total 859 database classes. This reduction in the number of classes involved in classification causes a reduction in recognition complexity, and thus higher recognition accuracy can be expected.

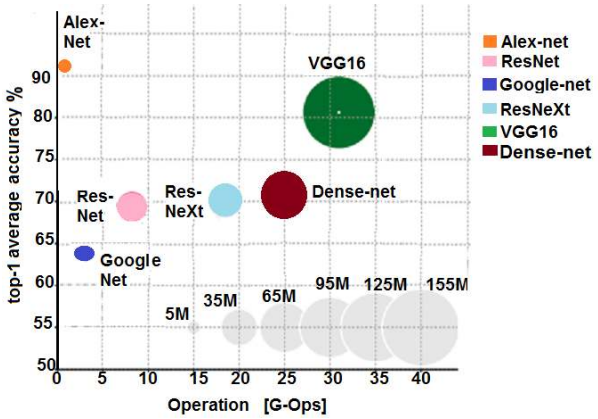


FIGURE 24. Complexity evaluation on IFN/ENIT dataset.

A comparison between different architectures on IFN/ENIT dataset is shown in Figure 24. The Figure relates the top 1 average accuracy to memory computational cost. Experiments is done on Intel(R) Core(TM) i7-2670QM processor. VGG is the most expensive architecture due to the recorded number of operations and memory complexity, as shown in Figure 24. In some other architectures reduced complexity affects accuracy negatively. The x-axis represents the required amount of operations per architecture. The required memory is represented by circles of different sizes. The memory ranges from 5MB to 155MB.

VI. CONCLUSION

The paper presents a multi-stage approach that aims to data-mine big dataset before classification. The ranking stage is an intermediate stage that passes a set of nearest classes per test image for less complex final classification. For classification, SVM and CNN effect were reported independently. SVM classification is based on a feature vector, which is extracted from an input image, while CNN classification is based on the raw pixel values of an input image.

VII. FUTURE WORK

Although KL-divergence is one of the most recommended methods of measuring divergence between probability distributions based on literature [84], we plan for applying other different methods and propose a comparison between their performance. These methods include Pearson (PE) Divergence, Relative Pearson (rPE) Divergence and L^2-Distance for divergence measure [85]. It is worth noting that these methods based on quadratic computation, while our used method is linear. Also, the relation between the different learning rates and the achieved results should be deeply studied. It is essential to implement an algorithm that computes the optimum learning rate.

REFERENCES

[1] P. A. Mueller and D. M. Oppenheimer, "The pen is mightier than the keyboard: Advantages of longhand over laptop note taking," *Psychol. Sci.*, vol. 25, no. 6, pp. 1159–1168, 2014.
[2] T. Saba, A. S. Almazyad, and A. Rehman, "Online versus offline Arabic script classification," *Neural Comput. Appl.*, vol. 27, no. 7, pp. 1797–1804, Oct. 2016.
[3] A. Hasasneh, N. Salman, and D. Eleyan, "Towards offline Arabic handwritten character recognition based on unsupervised machine learning methods: A perspective study," *Int. J. Comput. Academic Res. (IJCAR)*, vol. 8, no. 1, pp. 1–8, 2019.
[4] L. Campbell and V. Grondona, "Ethnologue: Languages of the world," *Language*, vol. 84, no. 3, pp. 636–641, 2008.
[5] I. Ahmad and G. A. Fink, "Handwritten Arabic text recognition using multi-stage sub-core-shape HMMs," *Int. J. Document Anal. Recognit.*, vol. 22, pp. 329–349, Aug. 2019.
[6] I. A. Doush, F. Alkhateeb, and A. H. Gharaibeh, "A novel Arabic OCR post-processing using rule-based and word context techniques," *Int. J. Document Anal. Recognit.*, vol. 21, nos. 1–2, pp. 77–89, Jun. 2018.
[7] A. Al-Dmour and M. Abuhelaleh, "Arabic handwritten word category classification using bag of features," *J. Theor. Appl. Inf. Technol.*, vol. 89, no. 2, p. 320, 2016.
[8] S. K. Jemni, Y. Kessentini, S. Kanoun, and J.-M. Ogier, "Offline Arabic handwriting recognition using BLSTMs combination," in *Proc. 13th IAPR Int. Workshop Document Anal. Syst. (DAS)*, Apr. 2018, pp. 31–36.
[9] X. Liu, G. Meng, and C. Pan, "Scene text detection and recognition with advances in deep learning: A survey," *Int. J. Document Anal. Recognit.*, vol. 22, no. 2, pp. 143–162, Jun. 2019.
[10] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.
[11] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 1107–1116.
[12] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2014, pp. 818–833.
[13] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *Proc. 3rd IAPR Asian Conf. Pattern Recognit. (ACPR)*, Nov. 2015, pp. 730–734.
[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[16] R. Bertolami and H. Bunke, "Early feature stream integration versus decision level combination in a multiple classifier system for text line recognition," in *Proc. 18th Int. Conf. Pattern Recognit. (ICPR)*, vol. 2, Aug. 2006, pp. 845–848.
[17] P. Dreuw, G. Heigold, and H. Ney, "Confidence-based discriminative training for model adaptation in offline Arabic handwriting recognition," in *Proc. 10th Int. Conf. Document Anal. Recognit.*, Jul. 2009, pp. 596–600.

[18] M. Awni, M. I. Khalil, and H. M. Abbas, "Deep-learning ensemble for offline Arabic handwritten words recognition," in *Proc. 14th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2019, pp. 40–45.
[19] E. M. Hicham, H. Akram, and S. Khalid, "Using features of local densities, statistics and HMM toolkit (HTK) for offline Arabic handwriting text recognition," *J. Electr. Syst. Inf. Technol.*, vol. 4, no. 3, pp. 387–396, Dec. 2017.
[20] Y. Elfakir, G. Khaissidi, M. Mrabti, and D. Chenouni, "Handwritten Arabic documents indexation using HOG feature," *Int. J. Comput. Appl.*, vol. 126, no. 9, pp. 14–18, Sep. 2015.
[21] P. Dreuw, S. Jonas, and H. Ney, "White-space models for offline Arabic handwriting recognition," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.
[22] S. A. Azeem and H. Ahmed, "Effective technique for the recognition of offline Arabic handwritten words using hidden Markov models," *Int. J. Document Anal. Recognit.*, vol. 16, no. 4, pp. 399–412, Dec. 2013.
[23] H. El Abed and V. Margner, "Comparison of different preprocessing and feature extraction methods for offline recognition of handwritten Arabic words," in *Proc. 9th Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 2, Sep. 2007, pp. 974–978.
[24] R. Al-Hajj Mohamad, L. Likforman-Sulem, and C. Mokbel, "Combining slanted-frame classifiers for improved HMM-based Arabic handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1165–1177, Jul. 2009.
[25] A. Lawgali, M. Angelova, and A. Bouridane, "A framework for Arabic handwritten recognition based on segmentation," *Int. J. Hybrid Inf. Technol.*, vol. 7, no. 5, pp. 413–428, Sep. 2014.
[26] K. Jayech, M. A. Mahjoub, and N. E. B. Amara, "Arabic handwriting recognition based on synchronous multi-stream HMM without explicit segmentation," in *Proc. Int. Conf. Hybrid Artif. Intell. Syst.*, vol. 9121. Springer, 2015, pp. 136–145.
[27] A. Benouareth, A. Ennaji, and M. Sellami, "Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition," *Pattern Recognit. Lett.*, vol. 29, no. 12, pp. 1742–1752, Sep. 2008.
[28] A. Benouareth, A. Ennaji, and M. Sellami, "HMMs with explicit state duration applied to handwritten Arabic word recognition," in *Proc. 18th Int. Conf. Pattern Recognit. (ICPR)*, vol. 2, Aug. 2006, pp. 897–900.
[29] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri, "IFN/ENIT-database of handwritten Arabic words," in *Proc. CIFED*, vol. 2, 2002, pp. 127–136.
[30] G. A. Abandah, F. T. Jamour, and E. A. Qaralleh, "Recognizing handwritten Arabic words using grapheme segmentation and recurrent neural networks," *Int. J. Document Anal. Recognit.*, vol. 17, no. 3, pp. 275–291, Sep. 2014.
[31] M. Pechwitz and V. Maergner, "HMM based approach for handwritten Arabic word recognition using the IFN/ENIT–database," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, 2003, pp. 890–894.
[32] D. Xiang, H. Yan, X. Chen, and Y. Cheng, "Offline Arabic handwriting recognition system based on HMM," in *Proc. 3rd Int. Conf. Comput. Sci. Inf. Technol.*, vol. 1, Jul. 2010, pp. 526–529.
[33] A. Khémiri, A. K. Echi, and M. Elloumi, "Bayesian versus convolutional networks for Arabic handwriting recognition," *Arabian J. Sci. Eng.*, vol. 44, pp. 9301–9319, May 2019.
[34] R. Maalej and M. Kherallah, "Convolutional neural network and BLSTM for offline Arabic handwriting recognition," in *Proc. Int. Arab Conf. Inf. Technol. (ACIT)*, Nov. 2018, pp. 1–6.
[35] M. Elleuch, R. Maalej, and M. Kherallah, "A new design based-SVM of the CNN classifier architecture with dropout for offline Arabic handwritten recognition," *Procedia Comput. Sci.*, vol. 80, pp. 1712–1723, 2016.
[36] S. Sudholt and G. A. Fink, "PHOCNet: A deep convolutional neural network for word spotting in handwritten documents," in *Proc. 15th Int. Conf. Frontiers Handwriting Recognit. (ICFHR)*, Oct. 2016, pp. 277–282.
[37] A. Saidani and A. K. Echi, "Pyramid histogram of oriented gradient for machine-printed/handwritten and Arabic/Latin word discrimination," in *Proc. 6th Int. Conf. Soft Comput. Pattern Recognit. (SoCPaR)*, Aug. 2014, pp. 267–272.
[38] G. Khaissidi, Y. Elfakir, M. Mrabti, M. E. Yacoubi, D. Chenouni, and Z. Lakhliai, "Segmentation-free word spotting for handwritten Arabic documents," *Int. J. Interact. Multimedia Artif. Intell.*, vol. 4, no. 1, p. 6, 2016.
[39] R. El-Hajj, L. Likforman-Sulem, and C. Mokbel, "Arabic handwriting recognition using baseline dependant features and hidden Markov modeling," in *Proc. 8th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug./Sep. 2005, pp. 893–898.
[40] A. Graves, "Offline Arabic handwriting recognition with multidimensional recurrent neural networks," in *Guide to OCR For Arabic Scripts*. Springer, 2012, pp. 297–313.
[41] R. Maalej and M. Kherallah, "Maxout into MDLSTM for offline Arabic handwriting recognition," in *Proc. Int. Conf. Neural Inf. Process.* Springer, 2019, pp. 534–545.
[42] R. Maalej, N. Tagougui, and M. Kherallah, "Online Arabic handwriting recognition with dropout applied in deep recurrent neural networks," in *Proc. 12th IAPR Workshop Document Anal. Syst. (DAS)*, Apr. 2016, pp. 417–421.
[43] M. Amrouch and M. Rabi, "Deep neural networks features for Arabic handwriting recognition," in *Proc. Int. Conf. Adv. Inf. Technol., Services Syst.* Springer, 2017, pp. 138–149.
[44] M. Amrouch, M. Rabi, and Y. Es-Saady, "Convolutional feature learning and CNN based HMM for Arabic handwriting recognition," in *Proc. Int. Conf. Image Signal Process.* Springer, 2018, pp. 265–274.
[45] A. Poznanski and L. Wolf, "CNN-N-gram for handwritingword recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2305–2314.
[46] Y. Kessentini, T. Paquet, and A. Ben Hamadou, "Off-line handwritten word recognition using multi-stream hidden Markov models," *Pattern Recognit. Lett.*, vol. 31, no. 1, pp. 60–70, Jan. 2010.
[47] J. H. AlKhateeb, J. Ren, J. Jiang, and H. Al-Muhtaseb, "Offline handwritten Arabic cursive text recognition using hidden Markov models and re-ranking," *Pattern Recognit. Lett.*, vol. 32, no. 8, pp. 1081–1088, Jun. 2011.
[48] R. Yan, L. Peng, G. Bin, S. Wang, and Y. Cheng, "Residual recurrent neural network with sparse training for offline Arabic handwriting recognition," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 1031–1037.
[49] N. Essa, E. El-Daydamony, and A. A. Mohamed, "Enhanced technique for Arabic handwriting recognition using deep belief network and a morphological algorithm for solving ligature segmentation," *ETRI J.*, vol. 40, no. 6, pp. 774–787, Dec. 2018.
[50] A. Khemiri, A. K. Echi, A. Belaid, and M. Elloumi, "Arabic handwritten words off-line recognition based on HMMs and DBNs," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 51–55.
[51] M. Elleuch, N. Tagougui, and M. Kherallah, "Deep learning for feature extraction of Arabic handwritten script," in *Proc. Int. Conf. Comput. Anal. Images Patterns*. Springer, 2015, pp. 371–382.
[52] H. Hassen, T. Zied, and K. Maher, "The p2p–grid–agent distributed platform: A distributed and dynamic platform for developing and executing large-scale application based on deep learning techniques," in *Intelligent Decision Technologies 2019*. Springer, 2019, pp. 25–35.
[53] Y. Hannad, I. Siddiqi, C. Djeddi, and M. E.-Y. El-Kettani, "Improving Arabic writer identification using score-level fusion of textural descriptors," *IET Biometrics*, vol. 8, no. 3, pp. 221–229, May 2019.
[54] T. D. Pham, K. Wardell, A. Eklund, and G. Salerud, "Classification of short time series in early Parkinson's disease with deep learning of fuzzy recurrence plots," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1306–1317, Nov. 2019.
[55] P. M. Kebria, A. Khosravi, S. M. Salaken, and S. Nahavandi, "Deep imitation learning for autonomous vehicles based on convolutional neural networks," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 1, pp. 82–95, Jan. 2020.
[56] E. Principi, D. Rossetti, S. Squartini, and F. Piazza, "Unsupervised electric motor fault detection by using deep autoencoders," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 2, pp. 441–451, Mar. 2019.
[57] D. Dileep, "A feature extraction technique based on character geometry for character recognition," Dept. Electron. Commun. Eng., Amrita School Eng., Kollam, India, Tech. Rep., 2012.
[58] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2785–2797, 2015.
[59] A. Bosch, A. Zisserman, and X. Munoz, "Representing shape with a spatial pyramid kernel," in *Proc. 6th ACM Int. Conf. Image Video Retr. (CIVR)*, 2007, pp. 401–408.
[60] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.
[61] K. Grauman and T. Darrell, "The pyramid match kernel: Efficient learning with sets of features," *J. Mach. Learn. Res.*, vol. 8, pp. 725–760, Apr. 2007.
[62] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. London, U.K.: Pearson, 2007.
[63] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[64] L. Pardo, *Statistical Inference Based on Divergence Measures*. London, U.K.: Chapman & Hall/CRC, 2018.
[65] T. M. Ghanim, M. I. Khalil, and H. M. Abbas, "PHoG features and Kullback-Leibler divergence based ranking method for handwriting recognition," in *Proc. IAPR Workshop Artif. Neural Netw. Pattern Recognit.* Springer, 2018, pp. 293–305.
[66] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.
[67] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.
[68] J. C. Principe, "Information theory, machine learning, and reproducing kernel Hilbert spaces," in *Information Theoretic Learning*. Springer, 2010, pp. 1–45.
[69] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.
[70] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[71] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1492–1500.
[72] Y. Zhu and S. Newsam, "DenseNet for dense flow," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 790–794.
[73] K. Seetala, W. Birdsong, and Y. B. Reddy, "Image classification using tensorflow," in *Proc. 16th Int. Conf. Inf. Technol.-New Generat. (ITNG)*. Springer, 2019, pp. 485–488.
[74] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[75] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
[76] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[77] H. Ding, K. Chen, Y. Yuan, M. Cai, L. Sun, S. Liang, and Q. Huo, "A compact CNN-DBLSTM based character model for offline handwriting recognition with tucker decomposition," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 507–512.
[78] A. Menshawy, *Deep Learning By Example: A Hands-on Guide to Implementing Advanced Machine Learning Algorithms and Neural Networks*. Birmingham, U.K.: Packt, 2018.
[79] N. A. Macmillan and C. D. Creelman, *Detection Theory: A User's Guide*. London, U.K.: Psychology Press, 2004.
[80] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[81] M. Peng, C. Wang, T. Chen, and G. Liu, "NIRFaceNet: A convolutional neural network for near-infrared face identification," *Information*, vol. 7, no. 4, p. 61, Oct. 2016.
[82] K. Grm, V. Štruc, A. Artiges, M. Caron, and H. K. Ekenel, "Strengths and weaknesses of deep learning models for face recognition against image degradations," *IET Biometrics*, vol. 7, no. 1, pp. 81–89, Jan. 2018.
[83] R. Almodfer, S. Xiong, M. Mudhsh, and P. Duan, "Multi-column deep neural network for offline Arabic handwriting recognition," in *Proc. Int. Conf. Artif. Neural Netw.* Springer, 2017, pp. 260–267.
[84] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective*. New York, NY, USA: Academic, 2015.
[85] M. Sugiyama, S. Liu, M. C. D. Plessis, M. Yamanaka, M. Yamada, T. Suzuki, and T. Kanamori, "Direct divergence approximation between probability distributions and its applications in machine learning," *J. Comput. Sci. Eng.*, vol. 7, no. 2, pp. 99–111, Jun. 2013.

TARAGGY M. GHANIM received the B.Sc. and M.Sc. degrees in electrical engineering from Ain Shams University, Cairo, Egypt, in 2006 and 2012, respectively. She has worked as an Assistant Lecturer with Misr International University. Her current research interests include pattern recognition, computer vision, and neural networks.

MAHMOUD I. KHALIL received the B.Sc. and M.Sc. degrees in electrical engineering from Ain Shams University, Cairo, Egypt, in 1992 and 1996, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON, Canada. His current research interests include pattern recognition, computer vision, and neural networks.

HAZEM M. ABBAS (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical and computer engineering from Ain Shams University, Cairo, in 1983 and 1988, respectively, and the Ph.D. degree from Queen's University, Canada, in 1993. He is currently a Professor of computer and systems engineering. He has served as the Chairman of the Department of Computer and Systems Engineering, Ain Shams University, and the Dean of the Faculty of Media Engineering and Technology, German University, Cairo. He worked for the Royal Military College, Kingston, Canada, the IBM Toronto Laboratory, and Mentor Graphics. He chaired the IEEE Signal Processing Chapter in Cairo. His research interests include machine learning and computational intelligence.