Arabian Journal for Science and Engineering (2021) 46:8509–8523
https://doi.org/10.1007/s13369-021-05471-4

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

An improved faster-RCNN model for handwritten character recognition

Saleh Albahli[1] · Marriam Nawaz[2] · Ali Javed[2,3] · Aun Irtaza[2]

Received: 2 October 2020 / Accepted: 18 February 2021 / Published online: 30 March 2021

Abstract
Existing techniques for hand-written digit recognition (HDR) rely heavily on the hand-coded key points and requires prior knowledge. Training an efficient HDR network with these preconditions is a complicated task. Recently, work on HDR is mainly focused on deep learning (DL) approaches and has exhibited remarkable results. However, effective detection and classification of numerals is still a challenging task due to people's varying writing styles and the presence of blurring, distortion, light and size variations in the input sample. To cope with these limitations, we present an effective and efficient HDR system, introducing a customized faster regional convolutional neural network (Faster-RCNN). This approach comprises three main steps. Initially, we develop annotations to obtain the region of interest. Then, an improved Faster-RCNN is employed in which DenseNet-41 is introduced to compute the deep features. Finally, the regressor and classification layer is used to localize and classify the digits into ten classes. The performance of the proposed method is analyzed on the standard MNIST database, which is diverse in terms of changes in lighting conditions, chrominance, shape and size of digits, and the occurrence of blurring and noise effects, etc. Additionally, we have also evaluated our technique over a cross-dataset scenario to prove its efficacy. Experimental evaluations demonstrate that the approach is more competent and able to accurately detect and classify numerals than other recent methods.

Keywords DenseNet-41 · Deep learning · Faster-RCNN · Hand-written digits · MNIST

1 Introduction

Nowadays, recognizing hand-written data [1] play a vital role in the field of information processing, due to the presence of a vast collection of information. Moreover, digital data processing is more economical than handling traditional paper information. The goal of handwriting digit recognition (HDR) methods is to translate hand-written characters into machine-understandable formats. Recently, HDR has been given extensive consideration among researchers due to its various applications. These systems can understand what is written in hand-written documents and enable researchers to find meaningful data stored on historic pages and manuscripts, which seem unrecognizable through the naked eye [2]. Apart from historical revelations, hand-written recognition systems are significant for the digital transformation of any organization. Automatic handwriting recognition systems can have a range of uses: recognition of hand-written medical transcripts to help patients, staff and chemists; assisting psychologists who believe that the personality can be judged through handwriting; hand-written recognition in forensic analyses to interpret handwriting to trace a criminal and help to drastically reduce the crime rates in a city; automatic recognition of vehicle number plates and postal codes written on envelopes; or reading bank cheques [2], etc.

All these applications have huge databases and so require recognition systems with high recognition power, minimum

✉ Ali Javed
ali.javed@uettaxila.edu.pk

Saleh Albahli
salbahli@qu.edu.sa

Marriam Nawaz
marriam.nawaz@uettaxila.edu.pk

Aun Irtaza
aun.irtaza@uettaxila.edu.pk

[1] Department of Information Technology, College of Computer, Qassim University, Buraidah, Saudi Arabia

[2] Department of Computer Science, University of Engineering and Technology, Taxila, Pakistan

[3] Department of Software Engineering, University of Engineering and Technology, Taxila, Pakistan

computational time and reliable accuracy. Recently, several HDR methods [3, 4] have been introduced to memorize writing patterns [5–7]. The task of the hand-written pattern recognition system is to group digits that are involved in writing a similar language; however, variations in writing styles and patterns, languages, the resemblance between the shapes of characters and overlapped digits have increased the complexity of recognition systems. Moreover, writing pattern recognition frameworks require high recognition accuracy and consistency. To cope with these challenges associated with automated hand-written digit recognition systems, researchers have proposed various solutions using systems based on hand-coded features [8], artificial neural networks (ANN) [9–11] and deep learning (DL) methods [12, 13], etc.

There exist several handcrafted key point-oriented HDR systems [14–17]. Hand-coded feature extraction approaches are easier to implement and do not require large training datasets; however, these approaches are slow and require the expertise of trained human experts. Moreover, for hand-coded-based feature extraction techniques, there is always a trade-off between efficacy and recognition accuracy as the processing of large feature-sets increases computational complexity, while employing small key points degrade the performance of the recognition system. Therefore, these approaches are not very effective for automated digit recognition systems [18].

Recently, we observed the utility of DL-based methods (Convolutional neural networks (CNN) [19], Recurrent neural networks (RNNs) [20], deep belief networks [21] and deep Boltzmann machines [22]) in various research domains, including the HDR systems. DL-based methods like CNN are capable of automatically learning the representative features of images without any human interventions. CNN architectures are an extended form of multi-layer perceptron (MLP) framework. The functionality of the CNN framework mimics the processing of the human brain. Humans detect and recognize objects by their naked eyes through visualizing thousands of object images. CNN follows the same patterns for perceiving and recognizing objects. Some distinguished CNN instances are GoogleNet [23], AlexNet [24], VGG [25] and ResNet [26]. CNN networks combine the key point detection and classification steps with small preprocessing and computational effort. Additionally, CNN-based techniques provide robust performance for object recognition, even with a small amount of training data. The main benefit of employing CNN architecture is that it exploits the topological information from the input sample and is invariant to post-processing transformations like scale changes, translation, etc., whereas their predecessors, like MLP models, never considered detailed topology information of input and were unable to perform well over higher resolution images because of fully interconnected nodes. Therefore, CNN frameworks are more effective than MLP [27] for various applications, including hand-written digit recognition systems.

In the past, CNN frameworks have been widely explored for HDR systems over the benchmark MNIST database [28]. Some works have reported an accuracy of 98% to 99% for hand-written digit recognition [29]. In [30], an ensemble technique comprising of various CNN models was proposed for hand-written digit recognition and reported high accuracy of 99.73% on the MNIST dataset. In [9], the CNN model was used in combination with a support vector machine (SVM) classifier and achieved impressive recognition accuracy of 98.1% over the MNIST database. Similarly, DL-based ensemble approaches were proposed in [31, 32] to improve the classification performance, though at the expense of increased computational cost. Although existing works have achieved impressive recognition precision, there is still room for improvement of handwriting digit recognition performance in terms of time and accuracy. Therefore, there exists a need to thoroughly investigate existing conventional ML- and DL-based approaches that are able to effectively recognize hand-written digits with maximum efficiency. The ability of machine learning algorithms to solve complex real-world problems is amazingly superior to human intelligence. The main challenges of HDR techniques are their low efficiency and high computational time. ML-based HDR solutions result in lengthy implementation codes which increase the computational time. To overcome the problem of lengthy codes, deep neural networks (NN) have emerged as DL techniques have decreased the coding length, but at the expense of increased code complexity.

Efficient and effective automated identification and classification of numerals is still a challenging task because of varying writing styles and the presence of post-processing operations like rotation and scaling, etc. In this paper, we have tried to overcome these challenges by employing a customized faster regional convolutional neural network (Faster-RCNN) with DenseNet-41 at the feature extraction level to compute the deep features of input images and to localize and classify the hand-written digits. The proposed method is robust to variations in scale, angles, chrominance, intensity, contrast, illumination conditions, blurring and high-density noisy images. The major contributions of the proposed work are as follows:

1. Introduced an improved Faster-RCNN framework with DenseNet-41 for computation of features, which increased the performance in locating small objects while decreasing both training and testing time and complexity
2. Accurate localization of numerals due to the precise region proposal network of Faster-RCNN.

3. Effective classification of hand-written digits because of the capability of the Faster-RCNN framework to deal with over-fitted training data.
4. Rigorous experimentation has been performed against several of the latest numeral recognition methods on a standard MNIST database containing different distortions, i.e., blurring, high-density noisy images, variations in chrominance, intensity, rotation and scale, etc., to show the efficacy of the introduced framework.

The rest of the manuscript has the following structure: Sect. 2 presents related work, while the proposed framework is discussed in detail in Sect. 3. A performance evaluation of our framework is presented in Sect. 4, and finally, Section 5 concludes the proposed work.

2 Related Work

In the literature, a substantial amount of research has been performed in the field of HDR systems [33–42]. In 1995, SVM was employed for the first time for recognizing hand-written characters [43]. Later, the SVM classifier became the first choice for several classification problems, i.e., character recognition [40–42], face recognition [44, 45] and object detection [46–49], due to its ability to efficiently handle the curse of dimensionality. Moreover, SVM decreases the chances of empirical error, while maintaining the complexity level of the mapping function which allows it to better generalize its prediction behavior and perform well for unknown data samples. Boukharouba et al. [50] proposed an approach for automated HDR. Initially, features were computed using the transition information of image pixels in the vertical and horizontal directions, along with the Freeman chain code histogram approach [51]. The computed key points were then used to train the SVM classifier for HDR. This approach [50] is robust to hand-written numeral recognition, but it requires training on a large dataset.

HDR systems exhibit remarkable performance using shallow frameworks [52–55]. Recently, DL-based methods have proved their robustness in many fields [56–58]; therefore, many researchers have used DL-based techniques for numerals, characters and word classification. Three-layered deep belief networks (DBN), together with the 'greedy' algorithm, were analyzed for the MNIST database and attained an accuracy of 98.75% [59]. Pham et al. [60] employed a regularization technique of dropout to increase the robustness of RNNs for HDR. This method [60] improves accuracy of using RNN, with a substantial decrease in the character and word error rate. Shamim et al. [4] introduced a technique to facilitate off-line HDR by employing various machine learning approaches, like MLP, SVM, Random Forest, Naïve Bayes, J48, Bayes Net and Random Tree through WEKA. It was concluded in [4] that MLP exhibits better recognition performance than other classifiers.

Wang et al. [61] presented the quantum k-neighbor algorithm for hand-written digit recognition. This approach [61] reduced the computational complexity compared to the simple k-nearest neighbor technique, but still needs to improve recognition accuracy. Arbain et al. [62] employed the multi-zoning method [63] for feature computation and used these features to train the SVM and MLP classifiers for numeral recognition. The approach in [62] performs well for digit recognition; however, it is unable to perform well when numerals form a triangle shape. Assegie et al. [64] presented a pixel-based dense approach with Decision Trees for numeral classification. This method [64] is simpler to implement, but at the expense of increased computational cost and comparatively lower recognition performance than other approaches.

Recently, CNN has achieved significant performance improvement in off-line hand-written character recognition of Tamil [65], Arabic [66], Telugu [67], Urdu [68] and Chinese [69] languages. Because of these promising results, CNNs have also been tested heavily for numeral recognition [70–72]. Initially, in 2003, Simard et al. [73] proposed a generic CNN framework for document examination and refined the complicated approaches of NN training [73]. Shi et al. [74] merged the benefits of both the deep CNN and RNN, and named the result a 'convolutional recurrent neural network' (CRNN). This approach [74] was also employed for scene text recognition and exhibited better performance over traditional approaches of numeral recognition. A hybrid model comprised of a BP neural network (NN) model and CNN was proposed in [75]. For the BP NN model, the Gabor feature extraction technique was employed to compute the numeral features. Then both Gabor and deep features were combined to train the CNN for HDR. This technique [75] improved numeral recognition accuracy; however, it is computationally complex.

Ali et al. [76] introduced a technique for HDR, where the Java-based DL4J framework was used for feature extraction. Later, the computed key points were used to train CNN for HDR classification. It was concluded in [76] that, for small datasets, CNN with a small number of layers performs better. Aly et al. [77] presented a new deep learning framework for HDR, named the deep convolutional self-organizing maps (DCSOM) network. This framework [77] used multiple cascades of convolutional SOM layers to compute the hierarchical key points from training samples. The output layer of the DCSOM framework calculated the local histograms of the computed features to show the classification results. This approach [77] is robust to numeral recognition in the presence of noise; however, the performance of this method degrades for rotational changes.

Hafiz et al. [78] presented an efficient hybrid classifier through a combination of deep learning with Q-Learning-based Reinforcement Learning technique [79]. The approach in [78] is robust to numeral classification under rotational variations, but at the expense of higher computational cost. Kulkarni et al. [80] presented a three-layered spiking neural network (SNN) for recognizing hand-written digits. It was concluded in [80] that SNN-based deep learning frameworks exhibit better numeral recognition accuracy than simple ANN architectures with back-propagation techniques. However, for large benchmark classification problems, SNN may not perform well.

Qiao et al. [81] proposed an adaptive deep Q-learning technique that merged the key-point computation competence of DL and the decision-making power of reinforcement learning to build an adaptive Q-learning deep belief network (Q-ADBN). Initially, Q-ADBN calculated the key points of the input sample by employing an adaptive deep auto-encoder (ADAE) approach. The computed features were considered as the current states of the Q-learning algorithm. In the next step, Q-ADBN accepted the Q-function (reward signal) through recognition of the present states. Finally, HDR was implemented by exploiting the Q-function though the Q-learning technique. This approach [81] exhibits better HDR performance in the presence of noise. However, for rotational variations in the written pattern, the recognition results of [81] are not reported.

An overview of existing methods used for hand-written digit classification is presented in Table 1.

Table 1 Overview of existing techniques for HDR

Method	Technique	Limitation
Boukharouba et al. [50]	The transition information of image pixels along with the vertical and horizontal orientations by employing the Freeman chain code histogram approach with SVM classifier	Needs training on a large database
Pham et al. [60]	DL-based framework RNN was employed for localizing and classification for HDR	Suffers from high computational cost
Shamim et al. [4]	Several ML-based techniques namely MLP, SVM, Random Forest, Naïve Bayes, J48, Bayes Net and Random Tree through WEKA were utilized to recognize and categorize the numerals	MLP exhibits better classification accuracy than other techniques; however, performance needs further improvements
Wang et al. [61]	The quantum k-neighbor algorithm was applied for hand-written numeral recognition	Classification accuracy needs to be improved
Arbain et al. [62]	The multi-zoning method [63] together with SVM and MLP classifiers were used for hand-written digit recognition	Unable to perform well for numerals forming a triangle shape
Assegie et al. [64]	Image pixels along with Decision Tree were utilized for numeral classification	Performance needs further enhancement
Shi et al. [74]	A CRNN model was applied for numeral classification	Computationally complex method
Hou et al. [75]	Both hand-coded key points based on the Gabor feature extraction algorithm together with deep features were computed. The calculated key points were employed to train the CNN classifier	Needs huge training time which in turn increases the computational burden
Ali et al. [76]	A Java-based DL4J framework was employed for feature calculation which was later used to train a CNN classifier for HDR	Not robust for the datasets of large size
Aly et al. [77]	A hierarchical key point computation-based DL approach, namely DCSOM network, was applied for digit recognition and classification	The performance of the method degrades on rotational changes in the input samples
Hafiz et al. [78]	A hybrid classifier by combining deep features with Q-Learning based Reinforcement Learning technique [79] was used for digit recognition	Suffers from high computational cost
Kulkarni et al. [80]	A 3-layered spiking neural network (SNN) for classifying the numeral is presented	May not perform well over large datasets
Qiao et al. [81]	An adaptive deep Q-learning method for numeral classification was presented. This method worked by combining the features computation of DL and the decision-making power of reinforcement learning to build an adaptive Q-learning deep belief network (Q-ADBN)	Not robust for samples with intense rotational variations

The accuracy of CNNs is highly dependent on the selection of hyper-parameters [82], which are typically decided with a trial-and-error approach. Hyper-parameters consist of activation function, epoch size, learning rate, size of the kernel, hidden units and layers, etc. The choice of these parameters is significant, as it controls the algorithm's functionality [83]. Hyper-parameters differ from model elements and must be selected before starting the training process. Ahlawat et al. [27] presented a method to modify the pure CNN architecture to achieve comparable performance to ensemble techniques for hand-written digit recognition systems. In [27], the effect of changing the layers, stride, kernel size, receptive field, padding and dilution was investigated. Moreover, the impact of performing fine-tuning of hyper-parameters was also discussed. Similarly, the impact of varying the architecture of the CNN framework was analyzed in [84]. The main motivation of this study [84] was to examine the changes of results when employing a different combination of hidden layers and epochs for CNN frameworks.

3 Proposed Method

The proposed method comprises two main parts. The first is 'dataset preparation' and the second is Custom 'Faster-RCNN builder' for classification. The first module develops annotations for digits to locate the exact region of interest. The second component of the introduced framework builds a new type of Faster-RCNN. Figure 1 shows the generic workflow of the proposed method.

The second module comprises two sub-modules, of which the first is a CNN framework named DenseNet-41 and the second is the training component, which performs training of Faster-RCNN through employing the key points computed by the CNN model. Faster-RCNN accepts two types of input: the first is the image sample, while the second is the location of the digit in the input image. Figure 2 shows the functionality of the presented technique. First, an input sample is passed to the designated CNN framework, along with the annotation's bounding box (bbox). The bbox recognizes the region of interest (ROI) in CNN key points. From the bboxes, reserved key points from training samples are nominated. Based on the computed features, Faster-RCNN trains a classifier and generates a regressor estimator for the given areas. The Classifier module assigns a predicted class to the object and the regressor component learns to determine the coordinates of the potential bbox to pinpoint the location of the digit in each image.

3.1 Data Preparation

The location of digits against each sample is necessary to detect the affected region for the training process. In this study, we used the LabelImg [40] tool to annotate the images and manually created a bbox for each sample. Figure 3 exhibits samples of the generated ground truths. The annotations are saved in XML files which contain the class name of each object and their bbox values, i.e., xmin, ymin, xmax, ymax, width and height. For each image, we maintain an XML file that is compiled to generate the CSV file. Finally, the training file is generated from the CSV file, to be later employed in the training process. In our study, we have ten classes that consist of integer values from 0 to 9.

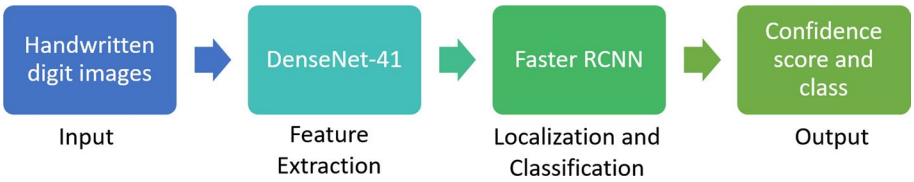

Fig. 1 General workflow of the proposed method

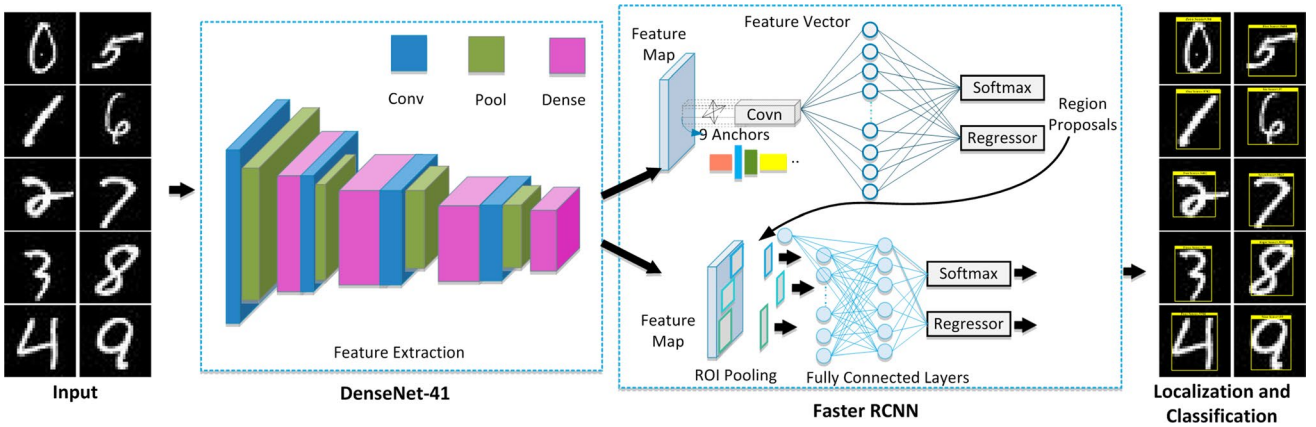

Fig. 2 Structural design of customized Faster-RCNN

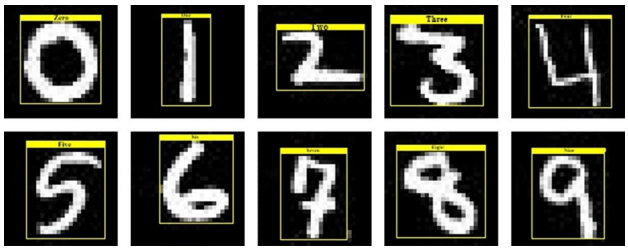

Fig. 3 Sample images with annotations

3.2 Faster-RCNN

The Faster-RCNN [57] algorithm is an extended form of existing approaches, i.e., R-CNN [58] and Fast-RCNN [59], which employed the Edge Boxes [61] technique to produce region proposals for possible object areas. However, the functionality of Faster-RCNN is changed from [58, 59] as it utilizes a region proposal network (RPN) as an alternative to the Edge Boxes algorithm to create the region proposals directly as part of the framework. This makes the computational complexity of Faster-RCNN for producing region proposals significantly less than in [61]. Put concisely, the selection of anchor boxes is finalized by RPN, which shows the most expected anchor boxes containing the regions of interest. So, in Faster-RCNN, region proposal generation is quick and better attuned to the input samples. Two types of outputs are generated by the Faster-RCNN: (1) classification that shows the class associated with each object, and (2) coordinates of the bounding box.

3.3 Custom Feature Faster-RCNN Builder

A CNN is a special type of NN that is essentially developed to perceive, recognize and detect visual attributes from 1D, 2D or ND matrices. In our study, image pixels are passed as input to the CNN framework. We employ DenseNet-41 [85] as a feature extractor in the Faster-RCNN approach. DenseNet [9] is the latest model of CNN, in which the current layer receives the inputs from all the preceding layers. DenseNet comprises a set of dense blocks that are sequentially interlinked with each other, with extra convolutional and pooling layers among successive dense blocks. DenseNet can represent complex transformations, which results in some degree of improvement in the issue of the absence of position information for the top-level key points of the target. DenseNet minimizes the number of parameters, which makes them cost-efficient. Moreover, DenseNet assists process of propagation of the key points and encourages their reuse, which makes them more suitable for digit classification. Thus, in the proposed work, we employ DenseNet-41 as a feature extractor for Faster-RCNN. The training parameters for customized Faster-RCNN are shown in Table 2.

Algorithm 1: Steps for numeral recognition by custom Faster-RCNN

START
INPUT: NI, annotation (position)
OUTPUT: Localized ROI, CFstDenseNet-41
NI: Total samples with digits.
annotation (position): bounding box coordinates of digit in sample
Localized ROI: digit position
CFstDenseNet-41-: Custom Faster-RCNN model based on DenseNe-41 key-points
imageSize ← [28 28]
// Bounding box approximation
α← AnchorsEstimation (NI, annotation)
// Customized FasterRCNN framework
CFstDenseNet-41← ConstructCustomDenseNet-41FasterRCNN **(*imageSize,* α)**
[T*r*, T*s*] ← partitioning of database into *train* and *test* set
// digit Identification Training Unit
For each sample ***i*** in →***Tr***
Compute ***DenseNet-41*** key-points →n***i***
End For
Training **CFstDenseNet-41** over n***i***, and measure **training time *t_dense***
η***_dense*** ← PredigitLoc(n***i***)
Ap_dense ← **Evaluate_AP(** ***DenseNet-41***, η***_dense)***
For each sample ***I*** in → ***Ts***
a) compute key-points through trained model **€→β*I***
b) **[*bounding_box*, *objectness_ score*, *class*]** ←Predict **(β*I*)**
c) show sample along with ***bounding_box*, *class***
d) η← **[**η ***bounding_box*****]**
End For
Ap_€← **Evaluate** model **€** using η
FINISH.

Table 2 Training parameters of the presented methodology

Network parameters	Value
Epochs	30
Learning rate	0.001
IOU threshold	0.90
Matched threshold	0.5
Unmatched threshold	0.5

Table 3 Architecture of DenseNet-41

Layer	Densenet-41	
	Size	Stride
Convolutional_layer_1	7×7 *conv*	2
Pooling_layer_1	3×3 *Max_pooling*	2
Dense_block_1	$\begin{bmatrix} 1\times1 conv \\ 3\times3 conv \end{bmatrix}\times3$	1
Transition_layer		
Convolutional_layer_2	1×1 *conv*	
Pooling_layer_2	2×2 *Avg_pooling*	
Dense_block_2	$\begin{bmatrix} 1\times1 conv \\ 3\times3 conv \end{bmatrix}\times6$	1
Transition_layer		
Convolutional_layer_3	1×1 *conv*	
Pooling_layer_3	2×2 *Avg_pooling*	
Dense_block_3	$\begin{bmatrix} 1\times1 conv \\ 3\times3 conv \end{bmatrix}\times6$	1
Transition_layer		
Convolutional_layer_4	1×1 *conv*	
Pooling_layer_4	2×2 *Avg_pooling*	
Dense_block_4	$\begin{bmatrix} 1\times1 conv \\ 3\times3 conv \end{bmatrix}\times3$	1
Classification_layer	7×7 *Avg_pooling*	
	Fully connected layer	
	SoftMax	

The main process of digit classification through Faster-RCNN can be divided into four steps. First, the input sample, along with the annotation, is fed to DenseNet-41 to compute the feature map. Second, the computed key points are used as input to the RPN component to obtain the feature information of the region proposals. In the third step, the ROI pooling layer produces the proposed feature maps by using the calculated feature map and proposals from the convolutional layers and RPN unit, respectively. In the last step, the classifier unit shows the class associated with each digit, while the bbox generated by the bounding box regression is used to show the final location of the identified digit. The detailed process of this component is presented in Algorithm 1.

3.4 DenseNet-41 Architecture

DenseNet-41 has two potential differences from the traditional DenseNet: (1) Densenet-41 has fewer parameters than the base DenseNet model; i.e., DenseNet-41 contains 24 channels on the first convolution layer, instead of 64, and the size of the kernel is 3×3 instead of 7×7; and (2) the number of layers within each dense block is attuned to deal with the computational complexity required. Table 3 shows the architecture of the presented DenseNet-41 model and names the layers through which the key points are taken for advance processing by Faster-RCNN.

The dense block is the fundamental part of DenseNet-41 as shown in Fig. 4, in which $n\times n\times m_0$ shows the feature maps (FPs) of the *K*-1 layer. The size of the FPs is *n* and the number of channels is denoted by m_0. A nonlinear transformation function *H*(.) containing different operations (i.e., batch normalization layer (BN), rectified linear unit (Relu) activation function, a 1×1 convolution layer (ConvL)) is used to reduce the number of channels and 3×3 ConvL is employed for feature restructuring. The dense connection is represented by the long-dashed arrow which joins the *K*-1 layer to the *K* layer and creates concatenation with the results

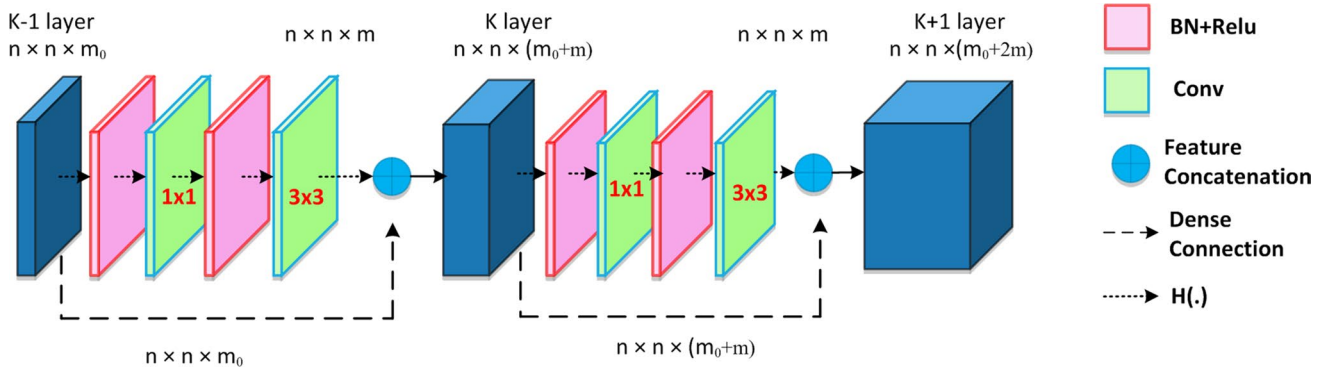

Fig. 4 Structure of dense block with two dense connections

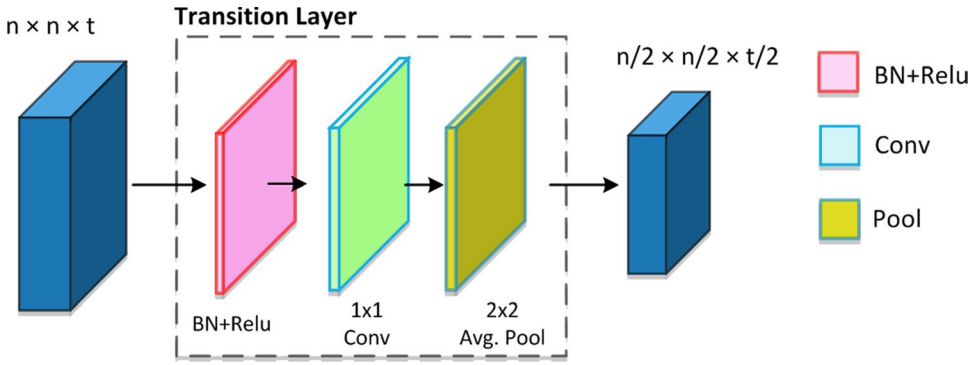

Fig. 5 Structure of transition layer

of $H(.)$. Finally, $n \times n \times (m_0 + 2\ m)$ is the output of the $K+1$ layer.

After multiple dense connections, the number of FPs will rise significantly, so the transition layer (TL) is added to minimize the feature dimension from the preceding dense block. The structure of TL is shown in Fig. 5: it comprises BN and a 1×1 ConvL to decrease the number of channels to half, followed by a 2×2 average pooling layer that decreases the size of FPs, where t and pool represent the number of channels and average pooling, respectively.

3.5 Detection Process

Faster-RCNN is a deep learning-based technique that is not dependent on methods like selective search for its proposal generation. Therefore, the input sample with annotation is given as input to the network, on which Faster-RCNN directly computes the bbox to show the digit's location and associated class.

4 Experiment and Results

This section provides a detailed analysis of the results obtained after conducting different experiments to assess the performance of the proposed technique. Details of the dataset are also given in this section.

4.1 Dataset

The evaluation of the presented technique is performed on the standard Modified National Institute of Standards and Technology (MNIST) database [28]. MNIST is a large-scale standard dataset of hand-written numerals that has been employed in training several image processing systems. The MNIST database comprises a total of 60,000 training and 10,000 testing images. The images in the MNIST dataset are diverse in terms of variations in rotation, scale and illumination, blurring and distortions, etc., which makes it a challenging dataset for digit classification. Figure 6 shows some sample images from the MNIST database.

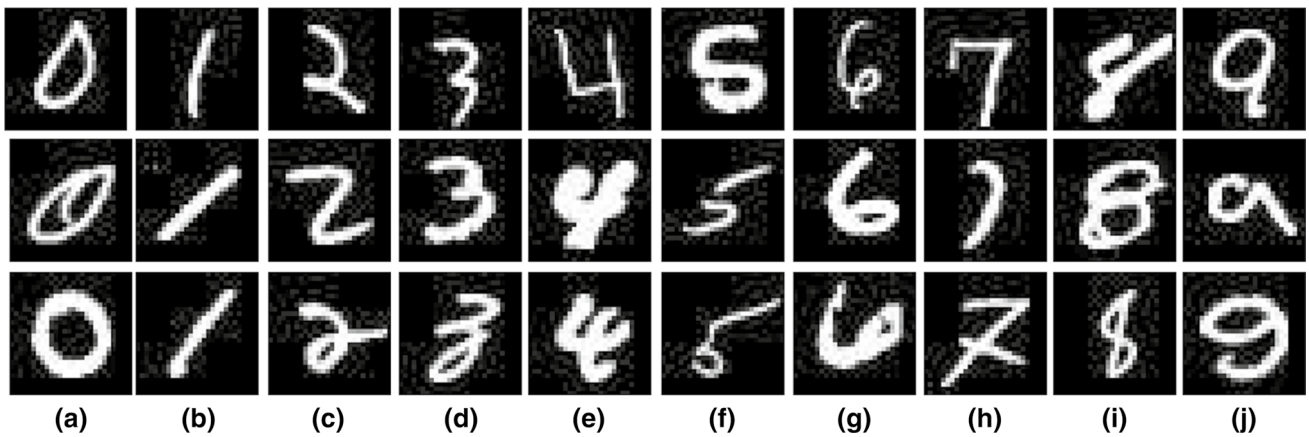

Fig. 6 Sample Images from MNIST dataset; **a** Zero, **b** One, **c** Two, **d** Three, **e** Four, **f** Five, **g** Six, **h** Seven, **i** Eight, **j** Nine

4.2 Evaluation metrics

We employed the Precision (P), Recall (R), accuracy (acc), mean average precision (mAP) and intersection over union (IOU) metrics to analyze the results of the presented approach. We computed the precision, recall, F1-score, accuracy, mAP and IOU metrics as follows:

$$P = \frac{tp}{tp + fp} \tag{1}$$

$$R = \frac{tp}{tp + fn} \tag{2}$$

$$F1\text{-}score = \frac{2 \times P \times R}{P + R} \tag{3}$$

$$acc = \frac{tp + tn}{tp + fp + tn + fn} \tag{4}$$

$$mAP = mean \frac{tp}{tp + fp} \tag{5}$$

$$IOU = \frac{tp}{tp + fn + fp} \times 2, \tag{6}$$

where tp, tn, fp and fn represent the true positive, true negative, false positive, and false negative, respectively.

4.3 Evaluation of DenseNet-41 Model

We designed an experiment to investigate the effectiveness of the DenseNet-41 model for HDR in comparison with existing deep learning models. For this purpose, the detection power of the presented Faster-RCNN with DenseNet-41 is compared with other base models like GoogleNet, Alexnet, ResNet and DenseNet-121. All models are implemented using Python with TensorFlow and run on Nvidia GTX1070 GPU based system. Moreover, classifiers are trained using various base networks and applied to locate digits from the MNIST dataset with 30 epochs and 0.001 learning rate.

The comparison of our technique with base models, both in terms of evaluation parameters and performance results, is reported in Table 4. From the reported results, it can be observed that the custom Faster-RCNN with DenseNet-41 outperforms GoogleNet, Alexnet, ResNet and DenseNet-121. From Table 4, we can observe that AlexNet is computationally most expensive and took 2235 s for execution, whereas our custom DenseNet-41 model is computationally most efficient and took 1002 s for processing. The base approaches suffer from high computational cost and may not exhibit better detection accuracy in the presence of noise, blurring, rotational and scale variations. Thus, our work addresses the limitations of existing works by presenting an effective framework for key-point computation and presents complex transformations accurately, leading to improved performance in post-processing operations. From this experiment, we can conclude that our custom DenseNet-41-based Faster-RCNN framework outperforms the base models evaluated in terms of both accuracy and efficiency.

4.4 Hand-written Digits Localization Results

The accurate localization of digits is crucial to develop effective HDR methods. Thus, we performed an experiment to investigate the accuracy of our digit localization approach. For this experiment, we used all samples from the MNIST testing dataset and reported the qualitative results of 100 images, as shown in Fig. 7. We can see from the resultant images that the presented framework can precisely localize the numerals even with the occurrence of blurring, distortion, and variations in illumination. Moreover, our approach can accurately diagnose digits of varying sizes and orientations.

The localization power of the Faster-RCNN method enables it to accurately detect and differentiate hand-written digits. The regression layer of Faster-RCNN localizes numerals with improved mAP and IOU. The mAP and IoU metrics are employed to determine how well each class of digit is recognized and localized by our framework. More specifically, we achieved the mAP and mean IOU of 0.993 and 0.991, respectively. From these qualitative and quantitative evaluations, we can conclude that the proposed method can reliably be used to localize numerals.

Table 4 Comparison of the presented technique with base models

Parameters	ResNet	AlexNet	GoogleNet	DenseNet-121	DenseNet-41
Total Parameters	23,595,908	62,378,344	7,844,327	7,037,508	6,031,422
Trainable parameters	23,542,788	58,178,211	6,432,287	6,955,908	5,921,356
Accuracy	0.940	0.777	0.888	0.963	0.997
Precision	0.920	0.861	0.880	0.940	0.983
Recall	0.892	0.790	0.782	0.910	0.971
Execution time (sec)	1338	2235	1320	1217	1002

Fig. 7 Test results of custom Faster-RCNN for numeral localization

4.5 Class-wise Performance

The correct recognition and classification of various numerals are essential to measure the robustness of a model. Therefore, the competence of the introduced approach in identifying the class of digits was also analyzed through performing an experiment. For this purpose, we employed the trained Faster-RCNN classifier over all the samples from the MNIST dataset. The class-wise hand-written digit detection performance of the presented approach in terms of precision, recall and F1-score are shown in Table 5. It can be clearly visualized that the introduced method has obtained remarkable precision, recall and F1-score values. The main reason for the better numeral recognition accuracy is the robustness of the introduced feature computation technique, which represents each class in a better manner. Although a little association is found between class-one and class-seven, both are recognizable. Therefore, because of the accurate key-point computation, our approach provides accurate class-wise

Table 5 Class-wise performance of the presented approach

Classes	Precision	Recall	F1-Score
Zero	1	0.96	0.979
One	0.992	0.94	0.965
Two	0.997	1	0.979
Three	0.97	0.99	0.979
Four	0.974	0.983	0.978
Five	0.99	0.997	0.993
Six	1	0.992	0.995
Seven	0.99	0.99	0.99
Eight	0.98	1	0.989
Nine	1	0.98	0.989

numeral recognition performance that shows the efficacy of the introduced method.

To further evaluate the class-wise performance of the proposed method, we have plotted the accuracies of ten classes in a boxplot, as this shows the summaries of results more clearly by showing the maximum, minimum and median of the accuracies obtained for all the classes (Fig. 8). Our technique has attained average accuracy values of 0.997, 0.996, 0.995, 0.998, 0.998, 0.997, 0.996, 0.995, 0.999 and 0.998 for classes from zero to nine, respectively. The proposed method achieved an average accuracy of 0.997, which signifies the effectiveness of the presented framework.

We also designed a confusion matrix analysis to clearly summarize the classification performance of the proposed method in terms of actual and predicted class. Figure 9 presents the confusion matrix of the proposed system. From this confusion matrix, we can observe that our system achieves the best results for classes *zero* and *two* with a true-positive rate (TPR) of 99%. However, the presented framework achieves the lowest results on class *one* with a TPR of 90% due to its visual similarity with other classes (i.e., four, seven, nine). Our method also achieves better classification performance for the remaining classes.

4.6 Comparative Analysis

To evaluate the significance of our method's superiority over existing approaches, we designed an experiment to provide a comparative analysis of the proposed and existing state-of-the-art HDR methods. For this purpose, we compared our approach with the latest HDR methods [29, 30, 59, 62, 86–90] and reported the results in Table 6. The proposed framework performed best and achieved an accuracy of 98.6%. Ciresan et al. [30] achieved the second-best results with the accuracy of 99.65%, whereas, Ge et al. [90] performed the worst with an accuracy of 95.7%. The results of this experiment clearly show that our method provides superior detection performance over the techniques compared. It is important to mention that the techniques compared

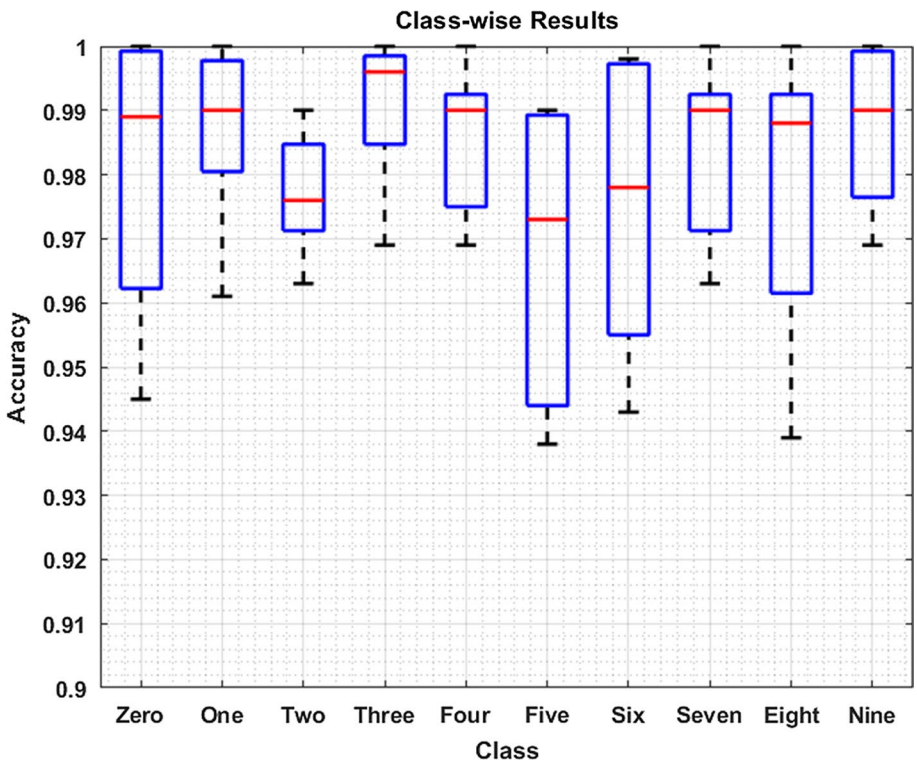

Fig. 8 Class-wise accuracies of the proposed methodology

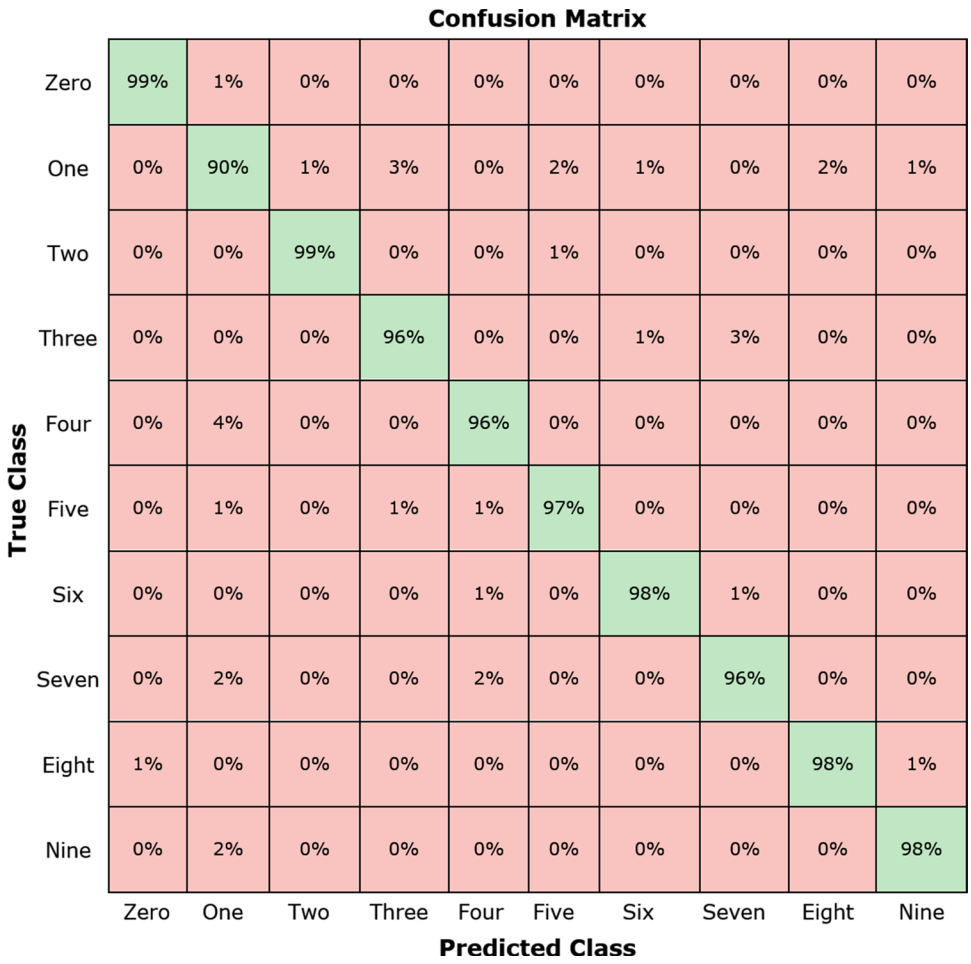

Fig. 9 Confusion matrix of the proposed framework

Table 6 Comparison with state-of-the-art approaches

Methods	Accuracy (%)
Zhao et al. [86]	97.5
Zhao et al. [87]	98.1
Enriquez et al. [88]	98
Maghari et al. [89]	98.08
Ge et al. [90]	95.7
Arbain1 et al. [62]	96.51
Jarrett et al. [29]	99.47
Ciresan et al. [30]	99.65
Hinton et al. [59]	98.75
Proposed	99.78

employed very deep networks that can easily result in overfitting. As the model used in our method has fewer layers, we can conclude that our approach is more efficient for classification of hand-written digits.

4.7 Cross-Dataset Validation

We designed an experiment to analyze the detection accuracy of our presented approach over a cross-dataset scenario. The main objective of performing cross-dataset validation is to evaluate the generalization power of our technique. For this purpose, we trained our method on the MNIST dataset and tested it over the USPS [91] database. USPS comprises a total of 7291 training and 2007 test images with 10 classes from 0 to 9. We used a box plot to show the evaluation results of our technique for cross-dataset evaluation in Fig. 10, where the accuracy of the training and test sets is distributed over the number line into quartiles, median and outliers. According to the reported results in Fig. 10, we obtained an average accuracy of 0.99 for training and 0.985 for testing, which demonstrates that our proposed framework is also capable of better recognizing digits from unseen samples. Therefore, it can be concluded that the proposed method is robust to numeral recognition and classification.

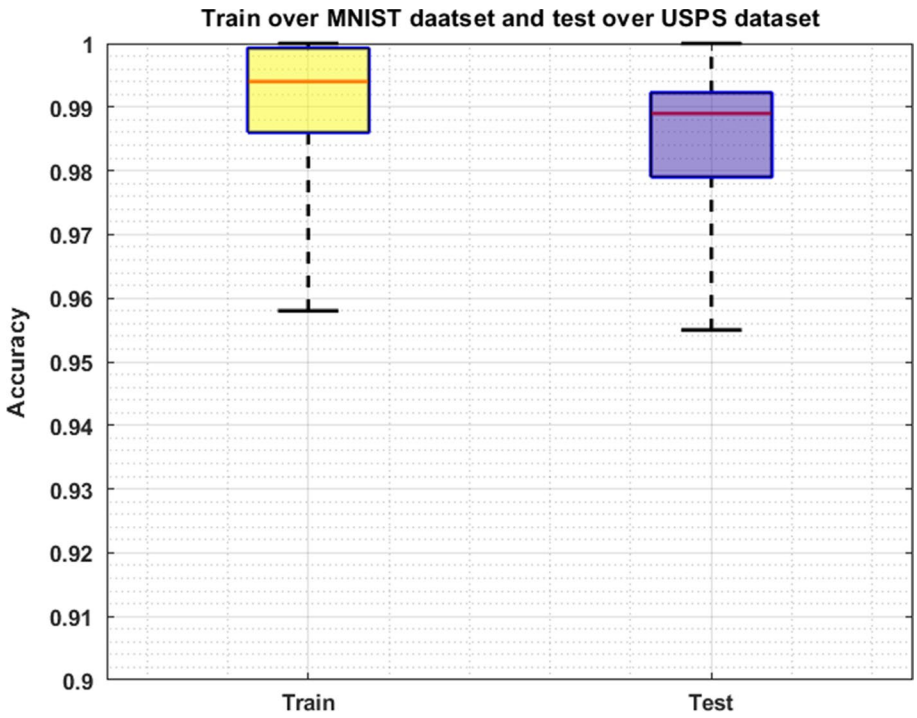

Fig. 10 Cross-dataset validation

5 Conclusion

This work introduces a novel method for the automated recognition and classification of hand-written digits by employing the Faster-RCNN deep learning technique along with the DenseNet-41 framework at the feature extraction level. In the presented technique, we also introduced the application of Faster-RCNN for numeral classification. More specifically, we employed the DenseNet-41 for deep feature computation and digit detection. Finally, we used the deep features to train Faster-RCNN's classifier for hand-written digit classification. The proposed method effectively localizes the digits from the input image and classifies them into 10 classes, representing integer values from 0 to 9. Our approach is robust to various artifacts, i.e., noise, blurring, chrominance changes, variations in light, digit size, rotational and scale variations and the presence of distortions. Experimental results on single- and cross-dataset scenarios confirmed that the presented framework outperforms the existing state-of-the-art techniques. In the future, we plan to test the presented framework over real-world scenarios and extend it to classification of other languages. Moreover, we will test our technique over more challenging datasets and will also consider other latest DL models.

Acknowledgements The authors would like to thank the Deanship of Scientific Research, Qassim University for covering publication of this project.

Declarations

Conflict of interest The authors declare that they have no conflicts of interest.

References

1. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
2. Al-wajih, E.; Ghazali, R.; Hassim, Y.M.M.: Residual neural network vs local binary convolutional neural networks for bilingual handwritten digit recognition. In: International Conference on Soft Computing and Data Mining, pp. 25–34. Springer (2020)
3. Abdulrazzaq, M.B.; Saeed, J.N.: A comparison of three classification algorithms for handwritten digit recognition. In: 2019 International Conference on Advanced Science and Engineering (ICOASE), pp. 58–63. IEEE (2019)
4. Shamim, S.; Miah, M.B.A.; Angona Sarker, M.R.; Al Jobair, A.: Handwritten digit recognition using machine learning algorithms. Global J. Comput. Sci. Technol. **18**(1), 1–8 (2018)
5. Abualigah, L.M.Q.: Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering. Springer, Berlin (2019)
6. Abualigah, L.M.; Khader, A.T.; Hanandeh, E.S.: Hybrid clustering analysis using improved krill herd algorithm. Appl. Intell. **48**(11), 4047–4071 (2018)
7. Abualigah, L.M.; Khader, A.T.; Hanandeh, E.S.: A new feature selection method to improve the document clustering using particle swarm optimization algorithm. J. Comput. Sci. **25**, 456–466 (2018)
8. Lauer, F.; Suen, C.Y.; Bloch, G.: A trainable feature extractor for handwritten digit recognition. Pattern Recogn. **40**(6), 1816–1824 (2007)

9. Niu, X.-X.; Suen, C.Y.: A novel hybrid CNN–SVM classifier for recognizing handwritten digits. Pattern Recogn. **45**(4), 1318–1325 (2012)
10. Goltsev, A.; Gritsenko, V.: Investigation of efficient features for image recognition by neural networks. Neural Netw. **28**, 15–23 (2012)
11. Kang, M.; Palmer-Brown, D.: A modal learning adaptive function neural network applied to handwritten digit recognition. Inf. Sci. **178**(20), 3802–3812 (2008)
12. Larochelle, H.; Bengio, Y.; Louradour, J.; Lamblin, P.: Exploring strategies for training deep neural networks. J. Mach. Learn. Res. **10**(1), 1–40 (2009)
13. Wang, Y.; Wang, X.; Liu, W.: Unsupervised local deep feature for image recognition. Inf. Sci. **351**, 67–75 (2016)
14. Verma, R.; Kaur, R.: An efficient technique for character recognition using neural network & surf feature extraction. Int. J. Comput. Sci. Inf. Technol. **5**(2), 1995–1997 (2014)
15. Verma, R.; Kaur, R.: Enhanced character recognition using surf feature and neural network technique. Int. J. Comput. Sci. Inf. Technol. **5**, 5565–5570 (2014)
16. Mapari, S.; Dani, A.: Recognition of handwritten benzene structure with support vector machine and logistic regression a comparative study. In: The International Symposium on Intelligent Systems Technologies and Applications, pp. 147–159. Springer (2016)
17. Hua, L.; Xu, W.; Wang, T.; Ma, R.; Xu, B.: Vehicle recognition using improved SIFT and multi-view model. J. Xi'an Jiaotong Univ. **4**(47), 92–99 (2013)
18. Ahlawat, S.; Choudhary, A.: Hybrid CNN-SVM classifier for handwritten digit recognition. Proc. Comput. Sci. **167**, 2554–2560 (2020)
19. Fukushima, K.: Biological cybernetics neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. **36**, 193–202 (1980)
20. Schuster, M.; Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**(11), 2673–2681 (1997)
21. Hinton, G.E.: Deep belief networks. Scholarpedia **4**(5), 5947 (2009)
22. Salakhutdinov, R.; Hinton, G.: Deep boltzmann machines. In: *Artificial intelligence and statistics*, pp. 448–455 (2009)
23. Christian Szegedy, W.L.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A.: Googlenet: going deeper with convolutions. Comput. Vis. Pattern Recognit. **1**(1), 1–9
24. Krizhevsky, A.; Sutskever, I.; Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
25. Simonyan, K.; Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv* (2014)
26. Targ, S.; Almeida, D.; Lyman, K.: Resnet in resnet: generalizing residual architectures. arXiv preprint arXiv:.08029 (2016)
27. Ahlawat, S.; Choudhary, A.; Nayyar, A.; Singh, S.; Yoon, B.: Improved handwritten digit recognition using convolutional neural networks (CNN). Sensors **20**(12), 3344 (2020)
28. Deng, L.: The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Process. Mag. **29**(6), 141–142 (2012)
29. Jarrett, K.; Kavukcuoglu, K.; Ranzato, M.A.; LeCun, Y.: What is the best multi-stage architecture for object recognition? In: 2009 IEEE 12th international conference on computer vision, pp. 2146–2153: IEEE (2009)
30. Cireşan, D.C.; Meier, U.; Masci, J.; Gambardella, L.M.; Schmidhuber, J.: High-performance neural networks for visual object classification. arXiv preprint arXiv (2011)
31. Ciregan, D.; Meier, U.; Schmidhuber, J.: Multi-column deep neural networks for image classification. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3642–3649. IEEE (2012)
32. Qu, X.; Wang, W.; Lu, K.; Zhou, J.: Data augmentation and directional feature maps extraction for in-air handwritten Chinese character recognition based on convolutional neural network. Pattern Recogn. Lett. **111**, 9–15 (2018)
33. Graves, A.; Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: Advances in Neural Information Processing Systems, pp. 545–552 (2009)
34. Sayre, K.M.: Machine recognition of handwritten words: a project report. Pattern Recogn. **5**(3), 213–228 (1973)
35. Plamondon, R.; Srihari, S.N.: Online and off-line handwriting recognition: a comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell. **22**(1), 63–84 (2000)
36. Yuan, A.; Bai, G.; Jiao, L.; Liu, Y.: Offline handwritten English character recognition based on convolutional neural network. In: 2012 10th IAPR International Workshop on Document Analysis Systems, pp. 125–129. IEEE (2012)
37. Manisha, C.N.; Reddy, E.S.; Krishna, Y.: Role of offline handwritten character recognition system in various applications. Int. J. Comput. Appl. **135**(2), 30–33 (2016)
38. Sánchez, J.A.; Bosch, V.; Romero, V.; Depuydt, K.; De Does, J.: Handwritten text recognition for historical documents in the transcriptorium project. In: Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage, pp. 111–117 (2014)
39. Plötz, T.; Fink, G.A.: Markov models for offline handwriting recognition: a survey. Int. J. Doc. Anal. Recogn. **12**(4), 269 (2009)
40. Choudhary, A.; Ahlawat, S.; Rishi, R.: A binarization feature extraction approach to OCR: MLP vs. RBF. In: International Conference on Distributed Computing and Internet Technology, pp. 341–346: Springer (2014)
41. Choudhary, A.; Rishi, R.; Ahlawat, S.: Off-line handwritten character recognition using features extracted from binarization technique. Aasri Proc. **4**, 306–312 (2013)
42. Choudhary, A.; Rishi, R.: A fused feature extraction approach to OCR: MLP vs. RBF. In: ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India, Vol I, pp. 159–166. Springer (2014)
43. Cortes, C.; Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
44. Pontil, M.; Verri, A.: Support vector machines for 3D object recognition. IEEE Trans. Pattern Anal. Mach. Intell. **20**(6), 637–646 (1998)
45. Osuna, E.; Freund, R.; Girosit, F.: Training support vector machines: an application to face detection. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 130–136. IEEE (1997)
46. Burges, C.J.: A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. **2**(2), 121–167 (1998)
47. Guo, G.-D.; Jain, A.K.; Ma, W.-Y.; Zhang, H.-J.: Learning similarity measure for natural image retrieval with relevance feedback. IEEE Trans. Neural Netw **13**(4), 811–820 (2002)
48. Weston, J.A.E.: Extensions to the support vector method. Ph.D. Thesis, Citeseer (2000)
49. Muller, K.-R.; Mika, S.; Ratsch, G.; Tsuda, K.; Scholkopf, B.: An introduction to kernel-based learning algorithms. IEEE Trans. Neural Netw. **12**(2), 181–201 (2001)
50. Boukharouba, A.; Bennia, A.: Novel feature extraction technique for the recognition of handwritten digits. Appl. Comput. Inf. **13**(1), 19–26 (2017)
51. Iivarinen, J.; Visa, A.J.: Shape recognition of irregular objects. In: Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, and Materials Handling, vol. 2904, pp. 25–32. International Society for Optics and Photonics (1996)
52. Choudhary, A.; Ahlawat, S.; Rishi, R.: A neural approach to cursive handwritten character recognition using features extracted from binarization technique. In: Complex System Modelling and Control Through Intelligent Soft Computations. Springer, pp. 745–771 (2015)
53. Choudhary, A.; Rishi, R.; Ahlawat, S.: Handwritten numeral recognition using modified BP ANN structure. In: International

Conference on Computer Science and Information Technology, pp. 56–65. Springer (2011)
54. Cai, Z.-W.; Huang, L.-H.: Finite-time synchronization by switching state-feedback control for discontinuous Cohen–Grossberg neural networks with mixed delays. Int. J Mach. Learn. Cybern. **9**(10), 1683–1695 (2018)
55. Zeng, D.; Dai, Y.; Li, F.; Sherratt, R.S.; Wang, J.: Adversarial learning for distant supervised relation extraction. Comput. Mater. Continua **55**(1), 121–136 (2018)
56. O'Shea, T.; Hoydis, J.: An introduction to deep learning for the physical layer. IEEE Trans. Cognit. Commun. Netw. **3**(4), 563–575 (2017)
57. Aceto, G.; Ciuonzo, D.; Montieri, A.; Pescapè, A.: MIMETIC: Mobile encrypted traffic classification using multimodal deep learning. Comput. Netw. **165**, 106944 (2019)
58. Aceto, G.; Ciuonzo, D.; Montieri, A.; Pescapé, A.: Toward effective mobile encrypted traffic classification through deep learning. Neurocomputing **409**, 306–315 (2020)
59. Hinton, G.E.; Osindero, S.; Teh, Y.-W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
60. Pham, V.; Bluche, T.; Kermorvant, C.; Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: 2014 14th International Conference on Frontiers in Handwriting Recognition, pp. 285–290. IEEE (2014)
61. Wang, Y.; Wang, R.; Li, D.; Adu-Gyamfi, D.; Tian, K.; Zhu, Y.: Improved handwritten digit recognition using quantum K-nearest neighbor algorithm. Int. J. Theor. Phys. **58**(7), 2331–2340 (2019)
62. Arbain, N.A.; Azmi, M.S.; Muda, A.K.; Muda, N.A.; Radzid, A.R.: Offline handwritten digit recognition using triangle geometry properties. Int. J. Comput. Inf. Syst. Ind. Manag. Appl. **10**, 87–97 (2018)
63. Azmi, M.S.; Omar, K.; Nasrudin, M.F.; Idrus, B.; Wan Mohd Ghazali, K.: Digit recognition for Arabic/Jawi and Roman using features from triangle geometry. In: AIP Conference Proceedings, vol. 1522(1), pp. 526–537. American Institute of Physics (2013)
64. Assegie, T.A.; Nair, P.S.: Handwritten digits recognition with decision tree classification: a machine learning approach. Int. J. Electr. Comput. Eng. **9**(5), 4446 (2019)
65. Kavitha, B.; Srimathi, C.: Benchmarking on offline handwritten tamil character recognition using convolutional neural networks. J. King Saud Univ. Comput. Inf. Sci. **1**(1), 1–8 (2019)
66. Boufenar, C.; Kerboua, A.; Batouche, M.: Investigation on deep learning for off-line handwritten Arabic character recognition. Cogn. Syst. Res. **50**, 180–195 (2018)
67. Dewan, S.; Chakravarthy, S.: A system for offline character recognition using auto-encoder networks. In: International Conference on Neural Information Processing, pp. 91–99. Springer (2012)
68. Ahmed, S.B.; Naz, S.; Swati, S.; Razzak, M.I.: Handwritten Urdu character recognition using one-dimensional BLSTM classifier. Neural Computing Applications **31**(4), 1143–1151 (2019)
69. Wu, Y.-C.; Yin, F.; Liu, C.-L.: Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models. Pattern Recogn. **65**, 251–264 (2017)
70. Tabik, S.; Alvear-Sandoval, R.F.; Ruiz, M.M.; Sancho-Gómez, J.-L.; Figueiras-Vidal, A. R.; Herrera, F.: MNIST-NET10: A heterogeneous deep networks fusion based on the degree of certainty to reach 0.1% error rate. Ensembles overview and proposal. Inf. Fus. **62**(1), 73–80 (2020)
71. Lang, G.; Li, Q.; Cai, M.; Yang, T.; Xiao, Q.: Incremental approaches to knowledge reduction based on characteristic matrices. Int. J. Mach. Learn. Cybern. **8**(1), 203–222 (2017)
72. Badrinarayanan, V.; Kendall, A.; Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(12), 2481–2495 (2017)
73. P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In: Icdar 2003(3) (2003)
74. Shi, B.; Bai, X.; Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2016)
75. Hou, Y.; Zhao, H.: Handwritten digit recognition based on depth neural network. In: 2017 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), pp. 35–38. IEEE (2017)
76. Ali, S.; Shaukat, Z.; Azeem, M.; Sakhawat, Z.; Mahmood, T.; ur Rehman, K.: An efficient and improved scheme for handwritten digit recognition based on convolutional neural network. SN Appl. Sci. **1**(9), 1125 (2019)
77. Aly, S.; Almotairi, S.: Deep convolutional self-organizing map network for robust handwritten digit recognition. IEEE Access (2020)
78. Hafiz, A.M.; Bhat, G.M.: Reinforcement learning based handwritten digit recognition with two-state Q-learning. arXiv preprint arXiv:.01193 (2020)
79. Watkins, C.J.; Dayan, P.: \cal Q-learning. Mach. Learn. **8**(3–4), 279–292 (1992)
80. Kulkarni, S.R.; Rajendran, B.: Spiking neural networks for handwritten digit recognition—supervised learning and network optimization. Neural Netw. **103**, 118–127 (2018)
81. Qiao, J.; Wang, G.; Li, W.; Chen, M.: An adaptive deep Q-learning strategy for handwritten digit recognition. Neural Netw. **107**, 61–71 (2018)
82. Cui, H.; Bai, J.: A new hyperparameters optimization method for convolutional neural networks. Pattern Recogn. Lett. **125**, 828–834 (2019)
83. Tso, W.W.; Burnak, B.; Pistikopoulos, E.N.: HY-POP: Hyperparameter optimization of machine learning models through parametric programming. Comput. Chem. Eng. **139**, 106902 (2020)
84. Siddique, F.; Sakib, S.; Siddique, M.A.B.: Recognition of handwritten digit using convolutional neural network in python with tensorflow and comparison of performance for various hidden layers. In: 2019 5th International Conference on Advances in Electrical Engineering (ICAEE), pp. 541–546. IEEE (2019)
85. Wang, Y.; Li, H.; Jia, P.; Zhang, G.; Wang, T.; Hao, X.: Multiscale DenseNets-based aircraft detection from remote sensing images. Sensors **19**(23), 5270 (2019)
86. Zhao, H.; Liu, H.: Algebraic fusion of multiple classifiers for handwritten digits recognition. In: 2018 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR), pp. 250–255: IEEE (2018)
87. Zhao, H.-H.; Liu, H.: Multiple classifiers fusion and CNN feature extraction for handwritten digits recognition. Granul. Comput. **5**(3), 411–418 (2020)
88. Enriquez, E.A.; Gordillo, N.; Bergasa, L.M.; Romera, E.; Huélamo, C.G.: Convolutional neural network vs traditional methods for offline recognition of handwritten digits. In: Workshop of Physical Agents, pp. 87–99. Springer (2018)
89. Ghosh, M.M.A.; Maghari, A.Y.: A comparative study on handwriting digit recognition using neural networks. In: 2017 international conference on promising electronic technologies (ICPET), pp. 77–81. IEEE (2017)
90. Ge, D.-y.; Yao, X.-f.; Xiang, W.-j.; Wen, X.-j.; Liu, E.-c.: Design of high accuracy detector for MNIST handwritten digit recognition based on convolutional neural network. In: 2019 12th International Conference on Intelligent Computation Technology and Automation (ICICTA), pp. 658–662. IEEE (2019)
91. Maji, S.; Malik, J.: Fast and accurate digit classification. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS--159 (2009)