ISSN NO: 2249-3034

Arabic Printed Numerals Classification Using Convolutional Neural Networks Model

Dr.CHANDRA MOHAN

[a] School of Computer Sciences, Kavayitri Bahinabai Chaudhari North Maharashtra University, Jalgaon, India

Correspondence should be addressed to Mohammed Rashed Almaamari; mohammedrashed555@gmail.com

Abstract

Optical character recognition (OCR) systems heavily depend on precise numeral recognition, which poses considerable difficulties in linguistically diverse languages like Arabic. This research introduces a convolutional neural network (CNN) developed for the classification of written Arabic numerals. Our proposed model attained an exceptional accuracy of 99.92%, exceeding the 96% accuracy of a previously modified model. By empirically adjusting hyperparameters such as epoch counts and batch sizes, we enhanced the model's performance, successfully reducing overfitting concerns. We also tested how simple it was to understand the CNN by using a confusion matrix to look at performance in each class. This gave us a better idea of how the model made its decisions. This study shows that the suggested CNN method works well at accurately finding all 10 types of numbers and demonstrates that it could be a good choice for OCR systems that work with Arabic and similar scripts. Our findings advance the domain of Arabic number identification and facilitate future improvements in deep learning models, targeting enhanced interpretability and applicability across other languages.

Keywords Arabic, OCR, CNN, DL, Recognizing, CV

1 Introduction

The fifth most widely used language worldwide is Arabic. Additionally, since 1973, it has served as one of the UN's official languages. It is spoken by 422 million people in 22 countries, in addition to its 290 million native speakers [1]. Artificial intelligence's fundamental goal is to have computer systems feel, remember, understand, and recognise everything like humans do. Due to a new multidisciplinary discipline that strives to make computer systems smarter, machine learning and deep learning have recently received a lot of attention. In a variety of OCR fields, including printed Arabic digits (0–9), printed Urdu text, and printed Arabic character recognition. OCR is gaining popularity and is used for a variety of tasks, such as reading license plates on cars, processing checks in banks, identifying zip codes at post offices, etc. Due to the distinctive design of each digit, the use of varied materials, and differences in orientation, automatic recognition of printed digits is currently remarkable and difficult [2]. Indian mathematicians came up with the Hindu-Arabic number system between the first and fourth centuries. It is the ancestor of the modern decimal number system. The approach was widely employed in Arabic mathematics by the ninth century, and by the High Middle Ages (about the 12th century), it had spread to Europe. In a text he wrote in Arabic in the year 825, the Indian numeral system was introduced to the Arabian Peninsula by Muhammad Ibn Musa al-Khwarizmi. He was a Persian astronomer and mathematician. The fundamental Indian numeric system was implemented differently in the Arabic world's Eastern and Western regions, and the modern-day European digits are descended from Western Arabic glyphs. Moreover, the Indian numerals, often known as Eastern Arabic numerals, are widely used in numerous nations to the east of the Arab world [3]. Pattern recognition and computer vision, which are regarded as crucial subfields of optical character recognition, are particularly dependent on printed digit recognition (OCR). Numerous practical applications have increased interest in both handwritten and printed numeral recognition [4]. The importance of handwritten and printed recognition systems has increased with the development of machine learning algorithms. Although digit recognition methods for online or offline handwritten and printed digits have been developed, research on an improved method that is writer-independent is still ongoing. Our motives and goals are to design and construct a digital translator that can recognise text and numbers in real-time, translate them into a voice for visually impaired students to hear, and translate speech into text for hearing-impaired students to read [5].

Here are the contributions this work makes:

(i) Based on this study's comparison of DL models and previous studies, the convolution neural network model is 99.92% accurate at classifying Arabic printed numerals.

(ii) It has helped the convolutional neural network model do a better job of classifying Arabic numbers.

(iii) A new CNN-based model for recognising Arabic numbers in low-resource settings that can also be used in real-time systems has been made.

The remainder of the essay is divided into the following sections: Section 2 includes a literature review on Arabic printed numerals. A review of the available research is presented in Section 2. Section 3 is about the dataset, Section

4 is about the materials and methods, Section 5 is about the results and analysis of the experiment, and Section 6 is about the conclusion. As shown in Table 1, it explains the different numerals, like European, Arabic-Hindi, and East Asian Arabic-Hindi numerals, and the originals for each of them and where they were used.

Table 1: Arabic-Hindi numeral system.

European (Origin from the west Arabic)	Arabic-Hindi	Easter Arabic-Hind (Urdu and Persian)
0	.	.
1	1	1
2	2	2
3	3	3
4	4	*
5	5	۵
6	6	۶
7	7	7
8	8	8
9	9	9

As shown in Table 1, this is the Arabic-Hindi numeral system, and we explain the European number and its origin in West Arabic. The Arabic-Hindi numeral is derived from Hindi, and the eastern Arabic-Hindi is used in Urdu and Persian numerals.

2 Related work and motivations

In this paper, Arabic handwritten numerals are recognised using a convolutional neural network model. Dropout regularisation gives the suggested approach the ability to solve the data overfitting issue. There are 3000 distinct handwritten numeral images in the Arabic handwritten digit dataset. They use the CMATERDB Arabic handwritten digit dataset to train the suggested CNN model, and they test it using a sample of test data from the same dataset. The suggested approach outperforms all prior studies on the dataset with an accuracy of 99.4% [6]. In this paper, they suggest a deep hybrid transfer model that has been modified using two well-known convolutional neural network models that have already been trained. These are further enhanced by the addition of recurrent neural networks, notably long-short-term memory structures, to recognise Arabic (Indian) handwritten digits. The CNN model learns the key characteristics of Arabic (Indian) digits, while the LSTM layer's sequence learning method extracts features for LSTM dependency. The experimental findings, utilising well-known datasets, demonstrate the remarkable performance attained by the modified transfer models. They achieved an accuracy of 98.92% [7]. In this paper, they offer the Hijja dataset, a new collection of Arabic letters produced solely by kids between the ages of 7 and 12
written by 591 people, make up th
they provide a convolutional neural network-based methodology for automatically recognising handwriting. They use the Arabic Handwritten Character Dataset and Hijja to train their model. Results show that their model does better than other models in the literature, with a level of accuracy of 97% on the AHCD dataset and 88% on the Hijja dataset [8]. In this research, they immediately noticed and recognised the Arabic numerals from the very first set of Ottoman Empire population registers, taken in the middle of the nineteenth century. To find these numbers, they initially used a segmentation technique based on CNN. In the second section, they used sizable open datasets of Arabic handwritten digits to test the deep transfer learning method, and they annotated a local dataset of Arabic written digits with the spotted numerals by selecting single-digit ones. They had a 96.06% accuracy rate [9]. In this work, they have developed a 10-class, one-class-per-numerical-digit, fusion-free, script-independent numeral recognition system for handwritten digits in more than one language. The issues with multilingual numerical recognition systems have been addressed. Exhaustive tests are conducted on numerical datasets of eight Indian and non-Indian scripts. Together, they have achieved an accuracy of 96.23% across all eight scripts. Using CNN, this technique achieved great accuracy in handwritten numeral identification [10]. In the paper, a deep learning technique for identifying Arabic handwritten numerals was demonstrated. LeNet-5 is a convolutional neural network that was trained and tested using pictures of Arabic numbers written by hand from the MADBase collection. This collection has 10,000 test images and 60,000 training images. This work had an accuracy rate of 88% [11]. This work that is being presented here suggests a fuzzy conceptual approach to categorising handwritten Arabic numerals based on moment aspects that are invariant and the segmentation of the number images into different portions. The American University in Cairo developed a database with 7,000 examples of each number from 700 different writers. 500 samples of each numerical image were used to test the algorithm, and 161 features were assessed for each image. The method's performance rate is determined to be 95.14% [12]. As shown in Table 2, we explain the different studies and the different accuracy levels for each study of Arabic printed numerals. As shown in Table 2, the related works and motivation, and in Table 2, we explain the different previous works related to the Arabic numeral classification. By comparing the previous accuracy with my accuracy, I achieved the highest accuracy.

Table 2: Related Works and Motivation

Authors names	Approach	Methodology and adoptive	Accuracy
Ashiquzzaman Et Al. [6]	Applying Data Augmentation To Handwritten Arabic Numeral	Convolutional Neural Network Model	99.4%

	Recognition Using Deep Learning Neural Networks		
Alkhawaldeh Et Al. [7]	Arabic (Indian) Digit Handwritten Recognition Using Recurrent Transfer Deep Architecture	Deep Hybrid Transfer Model	98.92%
Altwaijry Et Al. [8]	Arabic Handwriting Recognition System Using Convolutional Neural Network	Convolutional Neural Network Model	97%
Can Et Al. [9]	Automatic CNN-Based Arabic Numeral Spotting And Handwritten Digit Recognition By Using Deep Transfer Learning In Ottoman Population Registers	Convolutional Neural Network Model	96.06%
Gupta Et Al. [10]	CNN-Based Multilingual Handwritten Numeral Recognition: A Fusion-Free Approach	Convolutional Neural Network Model	96.23%
El-Sawy Et Al. [11]	CNN For Handwritten Arabic Digits Recognition Based On Lenet-5	Convolutional Neural Network Model	88%
Ali Et Al. [12]	FUZZY BASED RECOGNITION OF HANDWRITTEN ARABIC NUMERALS	Fuzzy Conceptual Approach Model	95.14%

.

3 Dataset

In this study, all images chosen from the 90 million Egyptian residents in the National ID dataset were picked. Data taken from photographs of the NID must be stored in databases as extracted data, according to this written agreement. These NID numbers will be extracted from ID photos. The computer vision community will assist them in locating each number's location before they extract it. CNN will also assist us in predicting each number. After producing datasets for each number class, they built a CNN model, which they subsequently enhanced to include every potential dataset case. Every ten classes in Gary's image contains around 2480 images, and each image has a pixel size of (64, 64). Consequently, the dataset's size is (24800, 64, 64, 1). This dataset is separated into training (70%), testing (10%), and validation (20%) categories. The link to this dataset is https://github.com/Rawash/Google-Colaboratory-datasets . As shown in Figure 1, an image sample from the dataset is displayed.

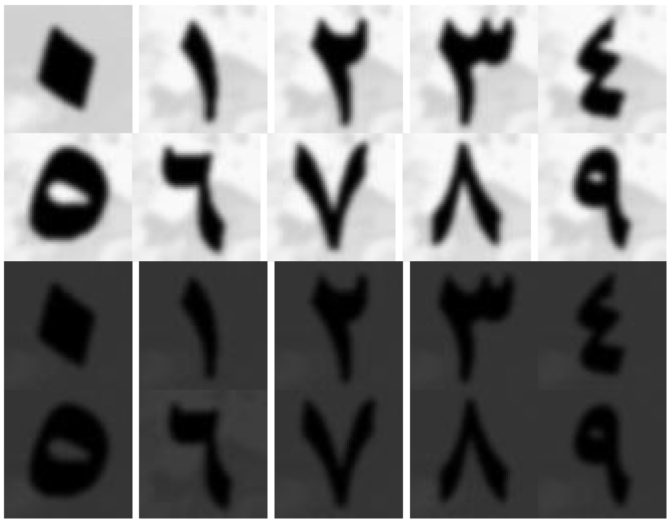


Figure 1: Dataset sample

As shown in Table 3, there are about 24800 samples in total. Each class has 1736 training images, 496 validation images, and 248 testing samples. Each class contains 2480 testing samples, 4960 validation images, and 17360 training images. There are 10 anticipated classes in all.

Classes	Total of samples
Total number of samples in the dataset.	24800
Total number of training sets	17360
Total number of testing set	2480
Total number of validation set	4960
Total number of predicted classes	10

Table 3: Subclasses of the Arabica numeral dataset.

4 Materials and approach

CNNs have recently been widely used for character and numeric classification methods. The performance of CNNs has shown to be the best model for a variety of pattern categorization issues. Because of recent improvements in deep learning methods, it is now much easier to recognize printed numbers in different scripts. The CNN-based system is better than the old ways of identifying printed numbers, which were based on how they were made. CNN is essentially a multilayered hierarchical model that is robust enough to handle a variety of pattern classification problems. The basic layers of the CNN model are explained in layman's terms below [13]. The proposed methodology and systems are divided into two phases. The first is a brief overview of the CNN phase, which includes the dense layer, activation layer, and a fully connected layer. The architecture of the applied CNN models is the subject of the second phase [14]. In Figure

ISSN NO: 2249-3034

2, we describe the materials and approach in the CNN model. We used the hidden layer and fully connected layer to make two significant changes to the materials and approach: in the input layer and hidden layer, we now activate each neuron using rectified linear units (ReLUs), and in the final classification layer, we now employ the Softmax function. During training, a certain number of neurons in each layer are randomly selected not to receive a gradient update in order to combat the overfitting issue. "Dropout" refers to this procedure [17].

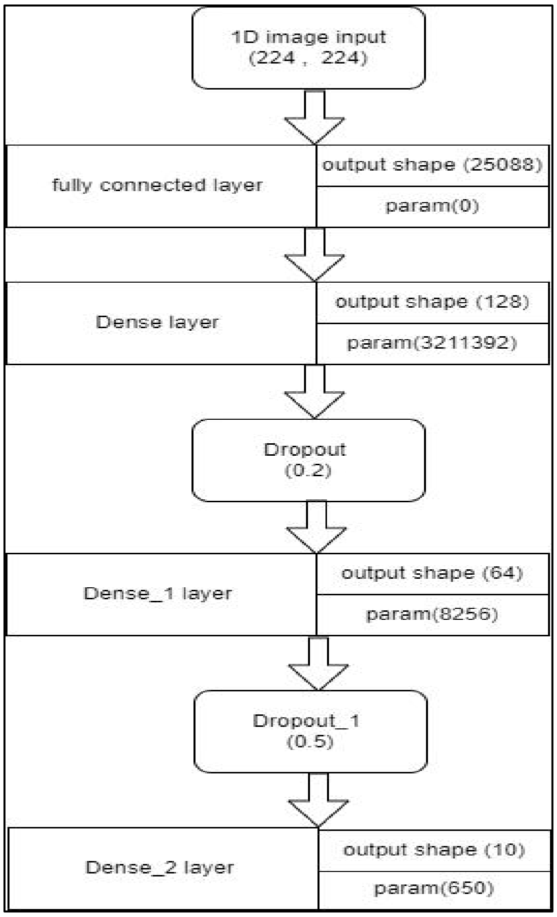


Figure 2: Materials and approach

4.1 A brief overview of CNN. Convolutional neural networks have produced ground-breaking breakthroughs in a variety of pattern recognition domains, from image processing to voice recognition, over the past ten years. The fact that CNNs reduce the number of parameters in ANNs is their most useful feature. The most crucial presumption concerning issues solved by CNNS is that they shouldn't contain features that depend on space [15]. The convolutional neural network (CNN) is effective for image classification due to its unique characteristics. The printed numeral images are first normalised, and then CNN is used to classify individual numerals. The CNN-based method outperforms some other existing methods in classification accuracy [16]. As shown in Figure 2 and Figure 3 , the brief overview of the CNN is explained by the entire architecture of the convolutional neural network.

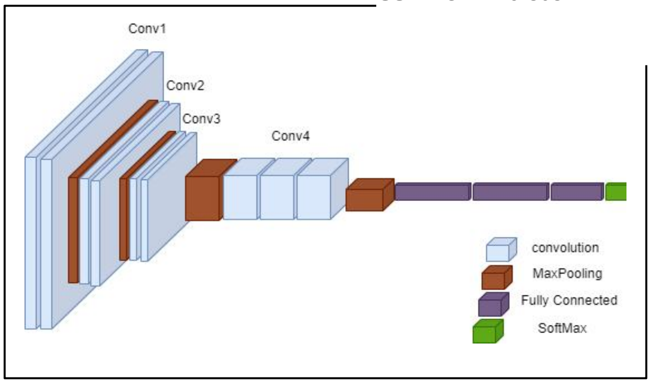


Figure 3: The CNN architecture provides examples of system components that it can explain, including the convolutional layer, the Max pooling layer, the fully connected layer, and the softmax layer.

4.2 Dense layer. A simple layer of neurons is referred to as a "dense layer" if every neuron in that layer receives input from every neuron in the previous layer. Images are classified using a dense layer based on the output of convolutional layers. Such neurons, which function as a single neuron, are found in large numbers within a layer of the brain [18]. The regular layer of a neural network with many connections is a dense layer. It is the most widely used and well-liked layer. It is also referred to as a "fully connected layer," since each neuron receives input from the layer before. The convolutional neural network (CNN), one of the most widely used deep learning methods, has a topology that makes it ideal for processing images and videos. A CNN has a dense layer and a convolutional layer [19]. In DL-based networks with Arabic printed numerals, the network size is crucial. The number of filters in the dense layer must be limited in order to reduce the size of the network. A classifier with fewer dense layers is produced as a result of this adjustment, which enables the CNN to extract useful characteristics from the dataset [20]. Additionally, it informs us that a dense layer is how the formula output = activation (dot (input, kernel) + bias) is implemented. This means that we are using the dot product of our dense layer's weight kernel matrix and the input tensor. As we can see in Figure 4, the input layer, the hidden layer, and the output layer are all explained. The dense layer is another name for the hidden layer.

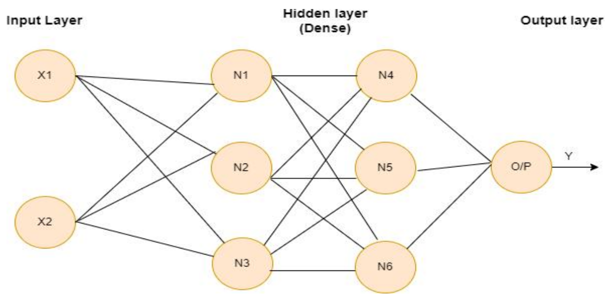


Figure 4: A dense layer consists of neurons that receive input from every neuron in the preceding layer.

4.3 **Activation layer.** Finding and designing activation functions that can enhance neural network performance is of great interest. Since new activation functions are essential to the expressiveness of artificial neural networks, they have helped to rekindle interest in neural networks among scientists. Since other expressions, like transfer functions or output functions, have also been employed, the term "activation function" has not always been used with the current meaning. The ReLU, leaky ReLU, and parametric ReLU are a few examples of activation functions [21]. In artificial neural networks, activation functions are used specifically to convert input signals into output signals, which are then sent as input to the subsequent layer in the stack. In an artificial neural network, the inputs are added together with the necessary weights, an activation function is applied to produce the layer's output, and the output is then used as the input for the layer that follows. The number of layers a neural network uses and, more importantly, the type of activation function it uses affect how accurate its predictions are [22]. In artificial neural networks, each neuron typically has a fixed, nonlinear activation function. However, in finite networks, the choice of nonlinearity affects both how the network learns and how well it can express itself. A sufficiently large neural network adopting any of these typical nonlinear functions can approximate arbitrarily complex functions [23]. Additionally, y = Activation Function (∑ (weights * input + bias)) is the output of the activation layer formula. As shown in Figure 5, it explains the activation layer mathematically.

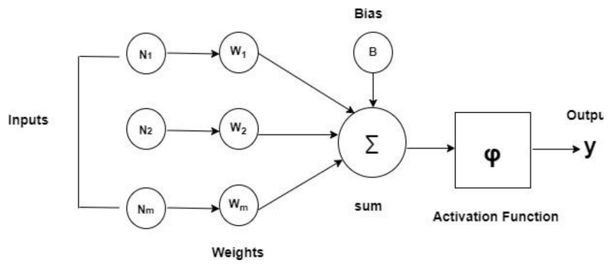


Figure 5: An activation function in neural networks is a mathematical function applied to a neuron's output.

4.4 **Fully connected layer.** The fully connected layers of the neural network are used for high-level thinking. All neurons from the preceding layer are accepted by and connected to every single neuron in a completely connected layer. After a fully connected layer, there can't be any more convolutional layers because they no longer know where they are in space [24]. Fully connected layers are a need for convolutional neural networks (CNNs), which have proven to be extremely effective in picture recognition and classification for computer vision. Each activation in the preceding layer is completely connected to every neuron in the layer that is fully connected. The matrix can be multiplied by a bias offset to determine their activation [25]. The output of convolutional or pooling layers is used by a fully connected layer to predict the r the image. After a fully conne softmax layer is frequently added, which applies a softmax function to the data it receives [26]. Each output unit's weight is decoupled from all other units' weights while calculating a fully connected layer. The number of weights is equal to x*y for the inputs of x and output of y. Furthermore, each output node has a bias, so you are at (x + 1) * y parameters. As we can see in Figure 6, it shows how the fully connected layer works in the CNN model.

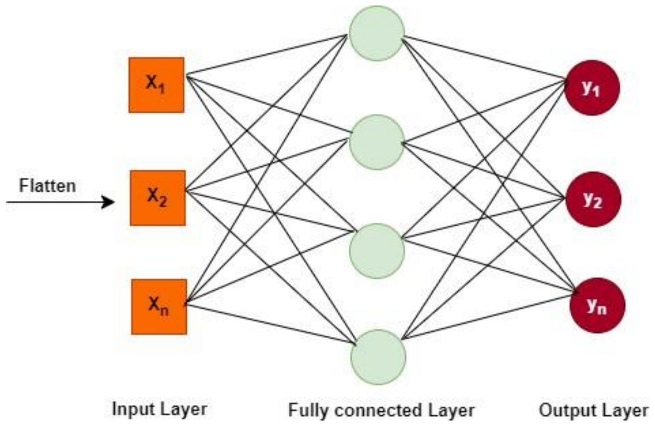


Figure 6: A fully connected layer denotes a neural network where each neuron performs a linear modification on the input vector via a weights matrix.

4.5 **Dropout layer.** Dropout, a way to stop overfitting, was made in 2012. It was later used in the winning entry for the Large Scale Visual Recognition Challenge in 2012, which brought deep neural network research back into the spotlight. The initial approach excluded every neuron in the neural network with a chance of 0.5 during each iteration of training, while including every neuron during testing. It has been demonstrated that using this method greatly enhances performance on a number of tasks. Since then, a variety of stochastic methods for use with deep learning models have been presented, many of which were inspired by the original dropout method. In general, the dropout technique involves simulating this procedure while randomly changing the activations or parameters of neural networks during training. The uses of dropout algorithms have grown since they were first employed to prevent overfitting [27]. However, a significant problem with these networks is overfitting. Large networks make it difficult to prevent overfitting by combining the predictions of several large neural networks at the test time, in addition to being sluggish to exploit. Dropout is one approach to addressing this problem [28]. During training, Dropout randomly sets the concealed unit activity to 0, with a probability of 0:5. Thus, it is possible to think of every training example as giving gradients for a unique, randomly selected design, resulting in a neural network that effectively reflects a vast ensemble of neural networks and has the potential to generalise well [29]. The relationship between the number of units in the network and the number of potential thin networks is exponential. An estimated model averaging approach is used at the test time to integrate all potential thin networks. This approximate model combination, followed by dropout training, considerably decreases overfitting and provides significant advantages over other regularisation techniques [30]. As shown in Figure 7, it explains how the dropout layer is applied in the CNN model.

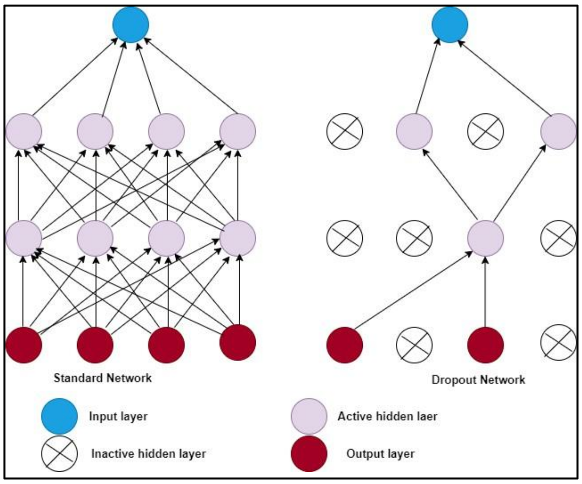


Figure 7: Dropout layers function as a mask that eliminates the contribution of some neurons to the subsequent layer. This establishes a novel network design that inhibits units from rectifying one another's errors, perhaps resulting in overfitting..

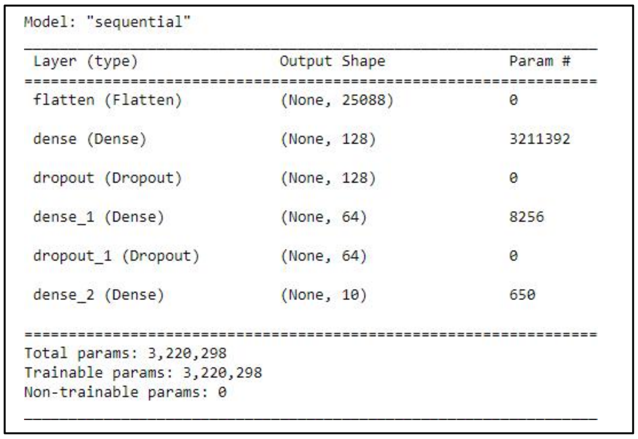
5 Experimental results and analysis

In the first part of this section, we discuss the software and hardware requirements for conducting experiments as well as how the CNN model is configured during the training phase. The proposed enlarged dataset is then used for a number of experiments, and a thorough analysis of the results is offered. Finally, performance comparisons between current research and cutting-edge techniques are given. The research exhibits multiple limitations. Firstly, its dependence on a particular dataset may limit generalizability, as it may not encompass all variants in Arabic numerical representations, including diverse fonts and handwriting styles. Moreover, elevated accuracy prompts apprehensions regarding overfitting, particularly when the dataset is deficient in diversity. The model's intricacy hinders interpretability, rendering its decision-making process harder to comprehend. Computational requirements may impede real-time performance in settings with constrained processing capabilities. Furthermore, the model may exhibit suboptimal performance with numerals from other languages, and environmental variables such as lighting and image quality can influence accuracy. Ultimately, depending exclusively on accuracy measurements may not yield a comprehensive assessment, especially in the presence of an imbalanced dataset. Mitigating these constraints is crucial for next research to improve the model's resilience and relevance.

5.1 **Model configuration and experimental setup.** Using the Arabic-Number-OCR-Keras dataset, we train, validate, and test both the modified and our suggested models. With a batch size of
model is trained across 50 epocl
of the various hyper-parameters employed in the proposed CNN model. The total number of trainable parameters for the modified model and the proposed model is 3,220,298 and 3,220,298, respectively. After the training phase, the proposed CNN model was generated as a layer-wise model summary with trainable parameters, which is displayed in Figure 8. On a laptop with an Intel(R) Core(TM) i5-4210U CPU running at 1.70 GHz and 4 GB of RAM, the trials are conducted. The "Python framework" (version 3.9.7) and "Keras library" are employed in terms of software needs. The software was written using the Jupyter Notebook interface.

Table 4: A list of the various hyper-parameters employed in the proposed CNN model

Optimizer	RMSprop
Learning rate	lr=1e-4
Loss-function	categorical_crossentropy
No. of epoch	50
Batch Size	32



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 25088)             0

 dense (Dense)               (None, 128)               3211392

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 64)                8256

 dropout_1 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 10)                650

=================================================================
Total params: 3,220,298
Trainable params: 3,220,298
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: Summary of the CNN model with trainable parameters.

5.2 **Results and analysis on the developed dataset.** the updated model had a 96% accuracy rate. On the same dataset, the proposed CNN model achieved 99.92% accuracy. The models' entire set of parameters was selected empirically. By changing the number of epochs and batch sizes, experiments were run. When we first started, we used 70 epochs with 128 and 256 batch sizes, which caused overfitting and increased model error rates. As a result, it was discovered that batch sizes of 32 and 50 epochs were the best options. We have just concentrated on a few facets of the proposed CNN model in the section that follows. We delve further since classification accuracy alone might be deceptive. We delve further since classification accuracy alone might be deceptive. However, deep learning is frequently linked to the phrase "black box." Understanding how CNN

ISSN NO: 2249-3034

perceives and comprehends a training model is crucial. It aids in our model's evaluation and comprehension. Therefore, we created a performance matrix for the outcomes. Over the whole number of predictions, the fundamental classification accuracy yields a number of accurate predictions. However, it might be challenging to determine whether or not all classes are equally anticipated when there are more than two. An easy way to express this view is with a confusion matrix, as seen in Figure 9. Have TN and FN stand for True Negative and False Negative numbers, respectively, and TP and FP for True Positive and False Positive numbers, respectively. Once this is done, the precision, or positive predictive value is calculated.

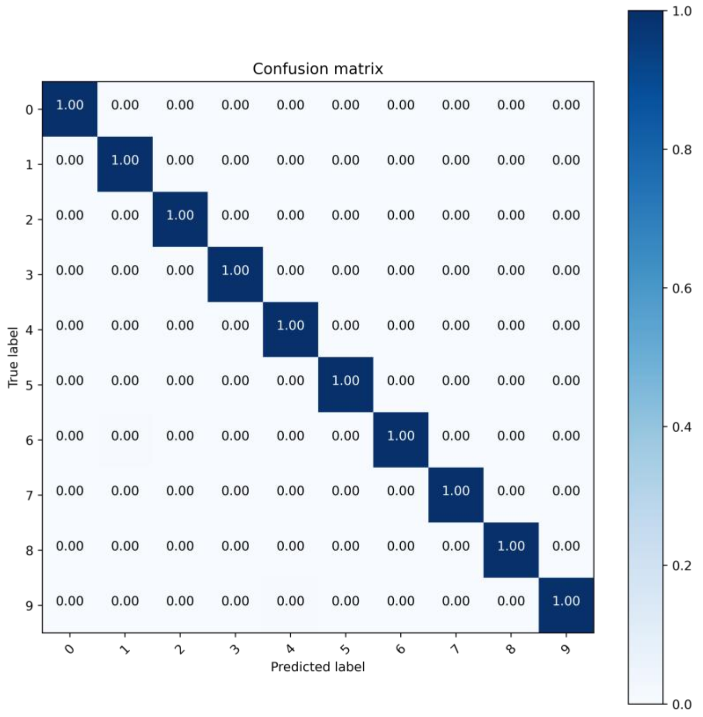
$$\frac{TP}{TP+FP} \qquad (1)$$

The sensitivity, recall, and rate of true positives are computed as

$$\frac{TP}{TP+FN} \qquad (2)$$

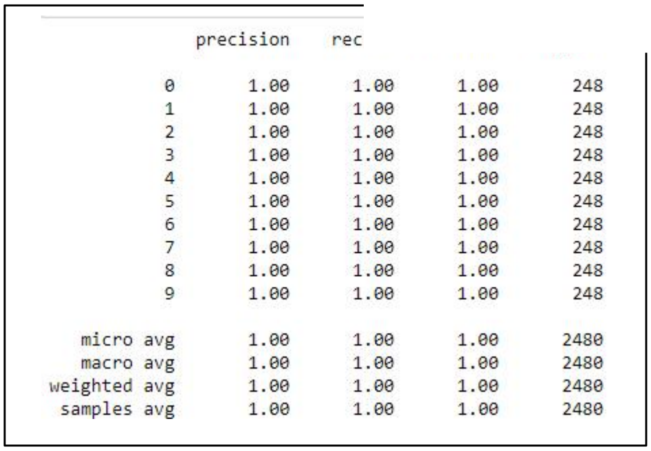
The formula for the F1-score is

$$\frac{2xPxR}{P+R} \qquad (3)$$



	0	1	2	3	4	5	6	7	8	9
0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

Figure 9: The proposed CNN model's confusion matrix for recognising printed Arabic numerals.



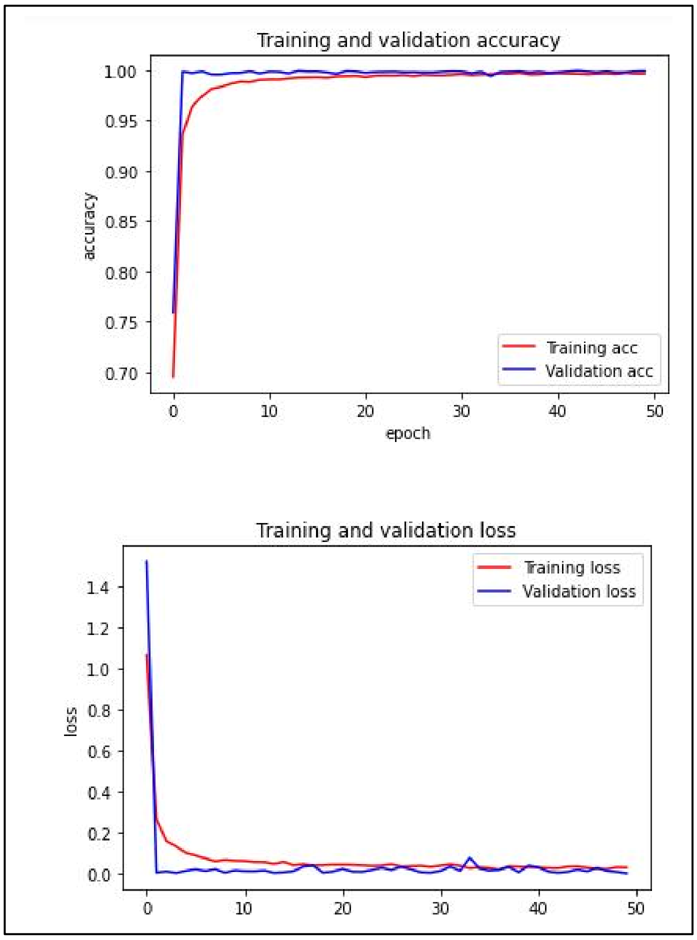
```
              precision    rec

           0       1.00      1.00      1.00       248
           1       1.00      1.00      1.00       248
           2       1.00      1.00      1.00       248
           3       1.00      1.00      1.00       248
           4       1.00      1.00      1.00       248
           5       1.00      1.00      1.00       248
           6       1.00      1.00      1.00       248
           7       1.00      1.00      1.00       248
           8       1.00      1.00      1.00       248
           9       1.00      1.00      1.00       248

   micro avg       1.00      1.00      1.00      2480
   macro avg       1.00      1.00      1.00      2480
weighted avg       1.00      1.00      1.00      2480
 samples avg       1.00      1.00      1.00      2480
```

Figure 10: Evaluation of the proposed CNN model's performance on a class-by-class basis for printed Arabic numerals.

Figure 10 details how the proposed CNN model measured the class's performance in detail and demonstrates that all classes had 100% F1-scores, which led to excellent recognition accuracy. The study's outcomes indicate exceptional metrics, including 100% precision, recall, and F1-score, alongside a total accuracy of 99.92%. Moreover, all ten classes achieved a flawless score of 100%.



7

ISSN NO: 2249-3034

Figure 11. Performance evaluation during the training and validation phases is represented graphically in 1. Accuracy curve for training and validation, 2. Loss curve for training and validation.

Tuning hyper-parameters is a crucial step in achieving the greatest performance of any architecture. We have experimented to find the optimum optimizer, ideal batch size, and number of epochs in order to achieve that. Additionally, it aids in the monitoring of model overfitting. With different epochs, we have compared the recognition accuracy. Through experimentation, it has also been found that the RMSprop optimizer is the best method for optimising and that 32 is the right batch size for the proposed CNN model. Any neural network model may be properly analysed using graphics, which also aids in identifying key errors that occur in the model. With the tuning and optimization of hyperparameters, it also aids in monitoring the model performance, which is shown in Figure 11. Recognition accuracy and loss curves collected throughout the training and validation phases of the proposed CNN model can be used to evaluate its performance. No overfitting is noticed in the suggested CNN model until the very last epoch. On the created dataset, our suggested CNN model produces 99.92% recognition accuracy. Additionally, this model outperforms cutting-edge techniques for recognising printed Arabic numerals.

6 Conclusion

This study successfully employed a convolutional neural network (CNN) to classify written Arabic numerals, achieving an impressive accuracy of 99.92%. Our findings underscore the efficacy of the proposed method in identifying all ten numerical classes and suggest its potential as a robust solution for optical character recognition systems in Arabic and similar scripts. While initial model configurations led to overfitting, careful adjustments to epoch counts and batch sizes significantly enhanced the model's reliability and accuracy. This research emphasizes the critical importance of hyperparameter tuning in deep learning models to optimize performance. Additionally, our analysis of the model's interpretability highlights the necessity of understanding CNN decision-making processes. By utilizing a confusion matrix, we were able to evaluate performance by class, providing insights that go beyond simple accuracy metrics. Overall, our results advance the field of Arabic number identification and set the stage for future research aimed at improving both interpretability and efficiency in deep learning models. The exceptional accuracy and reliability of our CNN model validate its practical applicability, paving the way for advancements in optical character recognition technologies across multiple languages.

this study has several limitations, including reliance on a specific dataset that may restrict generalizability, concerns about overfitting, and challenges with model interpretability. Additionally, computational demands may hinder real-time performance, and external factors like lighting can affect accuracy. Futur on addressing these limitations tc robustness. Efforts should prioritize expanding the dataset and exploring additional features to enhance the model's effectiveness and versatility.

Author Contributions M.R.A. created and planned the research, took part in gathering, analysing, and interpreting data, wrote and edited the article, and gave the authorization for its submission in its final form. as well as took part in the data interpretation and analysis, wrote and edited the paper, and gave authorization for its submission in its final form. supervised the research: made revisions to the preliminary report. R.J.R. carried out the experimental testing and made revisions to the paper draft.

Funding No funding was received for this article

Data availability statements We used one of the many huge datasets available on github for arabic printed Number, which is th https://github.com/Rawash/Google-Colaboratory-datasets .

7 Declarations

Conflict of interest The author(s) did not disclose any potential conflicts of interest.

Submission Declaration and verification: We affirm that the content of the article is original to the authors. We affirm that the text has not been submitted for publication elsewhere, in entirety or in part, nor has it been previously evaluated for publication. The corresponding author shall assume full responsibility for the submission on behalf of all co-authors. Every author has substantially contributed to the text, reviewed the work, and verified the accuracy of the data and its interpretation.

Declaration of generative AI in scientific writing This work was completed without the aid of generative artificial intelligence (AI). The author explicitly prohibits any organisation from using this publication, including but not limited to those capable of producing works in the same genre or style, for the purpose of training artificial intelligence (AI) technologies to generate text.

8 References

1. Ashiquzzaman, A., Tushar, A. K., Rahman, A., & Mohsin, F. (2019). An efficient recognition method for handwritten arabic numerals using CNN with data augmentation and dropout. In Data management, analytics and innovation (pp. 299-309). Springer, Singapore.
2. Ali, S., Sahiba, S., Azeem, M., Shaukat, Z., Mahmood, T., Sakhawat, Z., & Aslam, M. S. (2022). A recognition model for handwritten Persian/Arabic numbers based on optimized deep

convolutional neural network. Multimedia Tools and Applications, 1-24.
3. Tushar, A. K., Ashiquzzaman, A., Afrin, A., & Islam, M. (2018). A novel transfer learning approach upon hindi, arabic, and bangla numerals using convolutional neural networks. In Computational Vision and Bio Inspired Computing (pp. 972-981). Springer, Cham.
4. Abdulhussain, S. H., Mahmmod, B. M., Naser, M. A., Alsabah, M. Q., Ali, R., & Al-Haddad, S. A. R. (2021). A robust handwritten numeral recognition using hybrid orthogonal polynomials and moments. Sensors, 21(6), 1999.
5. Alghazo, J. M., Latif, G., Elhassan, A., Alzubaidi, L., Al-Hmouz, A., & Al-Hmouz, R. (2017). An online numeral recognition system using improved structural features–a unified method for handwritten Arabic and Persian numerals. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 9(2-10), 33-40.
6. Ashiquzzaman, A., Tushar, A. K., & Rahman, A. (2017). Applying data augmentation to handwritten arabic numeral recognition using deep learning neural networks. arXiv preprint arXiv:1708.05969.
7. Alkhawaldeh, R. S. (2021). Arabic (Indian) digit handwritten recognition using recurrent transfer deep architecture. Soft Computing, 25(4), 3131-3141.
8. Altwaijry, N., & Al-Turaiki, I. (2021). Arabic handwriting recognition system using convolutional neural network. Neural Computing and Applications, 33(7), 2249-2261.
9. Can, Y. S., & Kabadayı, M. E. (2020). Automatic cnn-based Arabic numeral spotting and handwritten digit recognition by using deep transfer learning in Ottoman population registers. Applied Sciences, 10(16), 5430.
10. Gupta, D., & Bag, S. (2021). CNN-based multilingual handwritten numeral recognition: a fusion-free approach. Expert Systems with Applications, 165, 113784.
11. El-Sawy, A., El-Bakry, H., & Loey, M. (2016, October). CNN for handwritten arabic digits recognition based on LeNet-5. In International conference on advanced intelligent systems and informatics (pp. 566-575). Springer, Cham.
12. Ali, Abdulbari & Ramteke, Rakesh. (2011). FUZZY BASED RECOGNITION OF HANDWRITTEN ARABIC NUMERALS. International Journal of Machine Intelligence. 3. 116-120. 10.9735/0975-2927.3.3.116-120.
13. Ahamed, P., Kundu, S., Khan, T., Bhateja, V., Sarkar, R., & Mollah, A. F. (2020). Handwritten Arabic numerals recognition using convolutional neural network. Journal of Ambient Intelligence and Humanized Computing, 11(11), 5445-5457.
14. Al-Hmouz, A., Latif, G., Alghazo, J., & Al-Hmouz, R. (2020). Enhanced numeral recognition for handwritten multi-language numerals using fuzzy set-based decision mechanism. International Journal of Machine Learning and Computing, 10(1), 99-107.
15. Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In 2017 international conference on engineering and technology (ICET) (pp. 1-6). Ieee.
16. Akhand, M. A. H., Ahmed, M., & Rahman, M. H. (2016, May). Convolutional neural network training with artificial pattern for bangla handwritten numeral recognition. In 2016 5th International conference on informatics, electronics and vision (ICIEV) (pp. 625-630). IEEE.
17. Ashiquzzaman, A., & Tushar, A. K. (2017, February). Handwritten Arabic numeral recognition using deep learning neural networks. In 2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR) (pp. 1-4). IEEE.
18. Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., & Masquelier, T. (2018). STDP-based spiking deep convolutional neural networks for object recognition. Neural Networks, 99, 56-67.
19. Jha, D., Yazidi, A., Riegler, M. A., Johansen, D., Johansen, H. D., & Halvorsen, P. (2020, Decer
efficient dense and convolutional l
In International Conference on Parallel and Distributed Computing: Applications and Technologies (pp. 285-296). Springer, Cham.
20. Dileep, P., Das, D., & Bora, P. K. (2020, February). Dense layer dropout based CNN architecture for automatic modulation classification. In 2020 National Conference on Communications (NCC) (pp. 1-5). IEEE.
21. Apicella, A., Donnarumma, F., Isgrò, F., & Prevete, R. (2021). A survey on modern trainable activation functions. Neural Networks, 138, 14-32.
22. Sharma, S., Sharma, S., & Athaiya, A. (2017). Activation functions in neural networks. towards data science, 6(12), 310-316.
23. Agostinelli, F., Hoffman, M., Sadowski, P., & Baldi, P. (2014). Learning activation functions to improve deep neural networks. arXiv preprint arXiv:1412.6830.
24. Balaha, H. M., Ali, H. A., Youssef, E. K., Elsayed, A. E., Samak, R. A., Abdelhaleem, M. S., ... & Mohammed, M. M. (2021). Recognizing arabic handwritten characters using deep learning and genetic algorithms. Multimedia Tools and Applications, 80(21), 32473-32509.
25. Kumar, M., Jindal, M. K., Sharma, R. K., & Jindal, S. R. (2020). Performance evaluation of classifiers for the recognition of offline handwritten Gurmukhi characters and numerals: a study. Artificial Intelligence Review, 53(3), 2075-2097.
26. Alqudah, A., Alqudah, A. M., Alquran, H., Al-Zoubi, H. R., Al-Qodah, M., & Al-Khassaweneh, M. A. (2021). Recognition of handwritten arabic and hindi numerals using convolutional neural networks. Applied Sciences, 11(4), 1573.
27. Labach, A., Salehinejad, H., & Valaee, S. (2019). Survey of dropout methods for deep neural networks. arXiv preprint arXiv:1904.13310.
28. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.
29. Ba, J., & Frey, B. (2013). Adaptive dropout for training deep neural networks. Advances in neural information processing systems, 26.
30. Srivastava, N. (2013). Improving neural networks with dropout. University of Toronto, 182(566), 7.