Hybrid Classifier-Based for Offline HandwrittenKannada Digit Recognition

Ramesh G
Department of Computer Science and Engineering , SVCE, Bengaluru, Karnataka.
Email id: ramesh.g_cse@svcengg.edu.in

P. N. Sharada
Department of Computer Science and Engineering , SVCE, Bengaluru, Karnataka.
Email id: sharada.pn_cse@svcengg.edu.in

B. Padmavathy
Department of Computer Science and Engineering , SVCE, Bengaluru, Karnataka.
Email id: padmavathy.b_cse@svcengg.edu.in

Abstract—

The field of pattern recognition has many applications, and handwritten digit recognition is one of them. Sorting postal mail, processing bank checks, data entry forms, and other tasks are just a few examples of how handwritten digit recognition is used. This displays the data in a digitized format. We are releasing a new handwritten digit dataset for the Kannada script called Kannada MNIST (Modified National Institute of Standards and Technology), which may be used to directly replace the original MNIST dataset. This is made up of the digits 0 through 9. The appropriate parameters partition the Kannada MNIST dataset into training and testing. For Handwritten Digit Recognition, there are primarily two steps: feature extraction and digit recognition (HDR). The primary base for digit recognition is a set of categorization algorithms. Convolutional neural networks (CNNs) were used as feature extractors in the ongoing study. On the simple MNIST dataset, CNN is implemented using the Deep Learning Python framework, which provides an accuracy of 99.6%.We are adding a couple of extra classifiers to the CNN output to see which approach best supports the CNN Model for these digits. Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and XG Boost are some of the classifiers we used. The output of the CNN is independently added with each of these classifiers. The output of the CNN model is feature extraction, which is fed to these classifiers to improve prediction accuracy. The major goal of this work is to develop a higher accuracy classifier by combining CNN and other classifiers.

Index Terms—Convolutional Neural Network, Kannada Digit, Random Forest, Support Vector Machine, K-Nearest Neighbor, XG Boost

DOI Number: 10.14704/nq.2022.20.13.NQ88327 Neuro Quantology 2022; 20(13):2601-2615

I. INTRODUCTION

Handwritten in the domain of handwriting and pattern recognition, digit recognition is an important area of research. Based on its functionality and excellent recognition accuracy, various handwritten digit recognition systems are used in real- time applications nowadays. Different classifiers are managed to improve digit identification accuracy on a fundamental level. In general, HDR presents a difficult problem for the development of novel solutions that increase the complexity, process speed, and recognition accuracy of working scenarios. Numerous studies and various methodologies, such as NN (Neural Networks), CNN (Convolutional Neural Networks), RF (Random Forest), SVM (Support Vector Machine), and KNN (K Nearest Neighbors), [1] have been utilised to tackle this challenging challenge. Compared to DNN (Deep Neural Network).

CNN [2] works well since it just needs a small numberof hidden layers [2] and because it is a truly remarkable instrument, less variables are needed. Varied implementation frameworks

produce useful results from various input datasets. The preparation of handwritten input dataset' raw images using HDR is unique. The Kannada MNIST dataset has been pre-qualified, and image segmentation is available.

The technique of converting handwritten Kannada digits in the paper format to digital format has many issues. There are many solutions for the recognition of printed documents of different languages, but very minimal solutions for the recognition of handwritten documents, which result in a more difficult task. This is because the various writing styles, size and shape in every human being. From the perspective of the recognition system, Kannada has 10 digits as other languages, where zero (0) will be common in some of the languages. Samples refer [3] to the digits which are written in a horizontal format like English from left to right. The common challenging task for Kannada digit recognition is that Kannada numeral's shapes have more curves. There will be more differences between the printed digits and handwritten digits. The segmentation process will be easy if the numbers are not connected often in various writing styles. The problem will be more when the individual digits have maximum variations, so some of the techniques are implemented to identify the problem of individual digits. Many algorithms or techniques will vary with the working but many follow the similar way from document to UNICODE format [4]. Digit recognition imposes feature classifications and feature extraction, which are the two major functionalities.Kannada is the official language of Karnataka, and even the numbers 0 through 9 are written in this language [5]. Finally, the project is suggested using the CNN features and the Python Framework. By various classifiers like SVM [6], RF, KNN and XGB individually for HDR [7]. Even this shows the comparison of different classifiers based on its working and efficiency used in the work. By using CNN to extract features from the Kannada MNIST, the work is then done. For a better classification process, classifiers are then used to the extracted feature.

The Motivation of this work is :

As indicated above, a lot of scholars have worked extremely hard to find a solution to the problem of Indian regional languages and numerical identification. All of the previous works either carried out or increased the recognition rate of popular English digits in order to make them more widely used. India has many regional languages with their own history and better literature, hence even to improve the regional languages we opted for Kannada digits for digit recognition

M1: All the existing digit recognition systems have been implemented in the English digits, with a maximum of one or two models of recognition.

M2: Here we use digit recognition of Kannada language with one particular Model for feature extraction and various other models or algorithms for classification.

M3: So the above work is done to improve the recognitionrate of the digits.

M4: Further CNN is used for the extraction and classification is done with the help of various classifiers,they are: Support Vector Machine, Random Forest, K-Nearest Neighbour, XG Boost

M5: As there are many online systems for the recognition, this proposed work is the offline scheme for the recognition system.

This work makes the following contributions:

1. To use a variety of techniques to create a Kannada Handwritten digit classifier that can be trained to maintainhigh accuracy in classes.
2. The main goal of the enhancement is to extract features from the handwritten Kannada MNIST input dataset using CNN. These extracted features are fed into the various classifiers to improve accuracy.
3. The proposed work's effect is calculated by comparing it to Convolution methods in Python Framework with ReLu activation.

The paper is organised in such a way that Section II of the paper contains records of related work, followed by Section III, which describes the Problem Statement and Objective of our work. The fourth section describes the proposed methodology. The fifth section describes CNN's architecture and various Classifiers. Section VI contains a collection of MNIST datasets. Section VII describes the experimental results, confusion

matrix, and discussion, followed by the conclusion in Section VIII.

1. To apply a number of strategies to develop a Kannada Handwritten Digit Classifier that can be trained to retain high accuracy in courses.
2. In order to improve the handwritten Kannada MNIST input dataset, the major objective of the upgrade is to extract features using CNN. To increase accuracy, the different classifiers are supplied with these collected features.
3. The impact of the proposed work is determined by contrasting it with Convolution techniques in the Python Framework with ReLu activation. The paper is set up so that Section II of the paper lists examples of prior work, and Section III of the paper outlines the Problem Statement and Objective of our investigation. The methodology is discussed in the fourth part. A description of CNN's design is provided in the fifth section. The architecture of CNN and its many Classifiers are covered in the fifth section. A selection of MNIST datasets can be found in Section VI. The experimental findings, confusion matrix, and discussion are presented in Section VII, and the conclusion is presented in Section VIII.

II. RELATED WORK

Several authors have been convinced of the importance of handwritten digit recognition. Lopes et al., [8], presentthe application of the classifier OPF (Optimum-Path Forest) in digit handwriting recognition. According to the results presented, it appears that character detection and recognition are being carried out satisfactorily in the Manhattan distance stood out with an average accuracy of 99.53%, and get trainingtimes and test lower than other methods such as It is thecharacteristic of OPF method.

The author Ramesh et al., [9] proposed the algorithm di-rected to solve the classification or recognition pf handwritten Kannada characters. The model consists of 4 convolution layers and 2 maxpooling layers. By consolidating the data into vowels and consonants. The achieved more accuracy, thedataset consist 500 images of each character i.e total of 83500images, in which 18800 images are used for training and 4700for testing the network. They obtained accuracy upto 93.2%. The work presented by Sandeep et al., [10] and his team to recognize the Kannada handwritten words. In this work both CNN and SVM is used in order to achieve more accuracy. The dataset was generated as there was no standard dataset. Then it is segmented and provided as input to CNN. The featurethat has been extracted from CNN will be given as input to SVM. The average accuracy of word segment is 97.5%.

The authors Yawei et al., [11] improved the traditional BP neural network and conducted experiments on the MATLAB simulation platform using the MNIST data set. The experimen- tal results show that the improved network converges faster and is more accurate in classification. Not only is the network training fast, but the network recognition rate is also higher than the standard BP neural network. Because the network in this paper is fast, it can be used for real-time processing.

The technique used by Almodfer et al., [12] They presented VGG-No for HDR, inspired by the success of the very deep state-of-the-art VGGNet. VGG-No is fast and depend- able, which significantly improved classification performance. VGG-No is made up of thirteen convolutional layers, two maximum-pooling layers, and three fully connected layers. The 10-Fold Cross-Validation strategy was used for the Cross- Validation analysis, and 10-Fold classification accuracies of 99.57% and 99.69% were obtained for the ADBase and MNIST databases, respectively.

Kumar et al., [13] the handwritten Kannada words are recognised. The model is made up of two classifiers: CNN and Support vector machine. The CNN will have k filters of size n x n x q as well as layers of size m x m x r, where m m is the image's height and width and r is the number of channels. An optimal hyperplane equation is essentially evaluated by the support vector machine (SVM). Training and testing yielded an overall accuracy of 92%.

The work proposed by Ramesh et al., [14] to create a Kannada dataset for image classification from of the imagenet database, a sample of 1000 images from 1000 classes were related. The Google Cloud Translate API was used to

translate all English labels from imagenet. It helped in translating 4,25,723 images to Kannada. The results were 28.35% ac- curate, 1.47% for inaccurate , 1.38% for neutral, 68.80% for English.

While Sangheon et al., [15] present an overview of the Phase Change Memory (PCM), one of the most developed developing non-volatile memory that has gained a lot of attention recently for usage as electronic synapses in biolog- ically inspired neuromorphic systems. For the categorization of MNIST handwritten digits, resistance drift characteristics from tests are evaluated and put into a spiking neural network (SNN).

Ahmed et al., [16]present Automatic recognition of hand- written digit string with unknown length has many potential real applications. The proposed method uses a new cascade of hybrid principal component analysis network (PCANet) and support vector machine (SVM) classifier called PCASVMNet. Every PCA-SVMNet classifier is trained separately using combinations of real and synthetic touching digits. The first 1D-PCA-SVMNet stage is trained to recognize isolated hand- written digits (0 . . . 9) while forwarding non-isolated digits to the next stages. Experiments on the TP synthetic database highlight the advantages of the proposed method by achievingmore than 95% of recognition rate.

The Deep neural network technique used by Srihari W et al., [17] to recognise handwritten Kannada numericals. In this work, the CNN used as both for feature extraction and classification. The Kannada MNIST dataset contains 60000 images where 50000 is used for training and 10000 used for testing. The grayscale image of 28 x 28 is given as inputto model. This work achieved accuracy 98.24% for MNIST dataset and 86.65% in dig-MNIST dataset. Further the work ofRamesh et al., [18] aims to recognise the handwritten Kannada characters bu robust and state of aart technique. The capsule network imitate the human visual system by considering a parse true structure. Each view proposed by visual cortex is converted into parse tree structure. The image passed through 3 CNN layers of which one is capsule layer. The Accuracy obtained by this work is 98.7%.

Further down the lane, Saleh et al., [19] Deep Convolutional Neural Networks (DCNN) are currently the most widely used technique for learning visual features from images. A sequence of convolutional SOM layers trained to represent multiple lev-els of features. The two-dimensional SOM grid is commonly used for data visualisation or feature extraction. To create a new deep network, the work makes use of a high dimensional map size. In the final convolutional SOM layer, the output layer of the DCSOM network computes local histograms of each FII bank. To evaluate the robust representation of the proposed DCSOM network, a series of experiments are carried out using the MNIST handwritten digit database and all of its variants.

Further down the work presented by author Prasanna et al., [21] to recognize Kannada Handwritten digits based on PCA and SVM classifiers techniques. The system consists of 70000 samples, in which 60000 is used for training and 10000 for testing. In preprocessing the image is undergone deskewing and min max normalisation to remove noisy data. After PCA applied to reduce dimension of data. The overall accuracy of system in SVM + PCA is 99.02%.

III. PROBLEM STATEMENT AND OBJECTIVES

The idea for Kannada Handwritten Digit Recognition was motivated by numerous works on handwritten recognition of various languages. The biggest problem with Kannada numerals for recognition is that the shapes are more curved than English numerals. Because the digits are not printed, they are written by hand. Different writing styles are used in this piece. Many works have been completed in order to recognise each character. Many techniques for identifying individualdigits have been developed. While different algorithms have different implementations, most follow a similar path from document to UNICODE text.

This work's objectives are as follows:

1. To improve accuracy in training and validation datasetsacross multiple classes.
2. To compare and select the best classifier for the CNNfeature extractor that provides the highest accuracy.
3. To minimise the number of epochs required for the modelto converge.

4. By averaging the various outputs of each decision tree, RF reduces the possibility of error in the decision tree.

IV. PROPOSED METHODOLOGY

The basic operation implemented in this work is combining various classifiers separately to the output of the CNN model. Hence the accuracy rate will be more than the CNN [22] because the classifiers work with high efficiency and also the comparison is made between these classifiers. The classifiers implemented are Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN) [23], and XG-Boost are among the classifiers used (XGB). All of these classifiers were created by combining the best features of CNN for the classification of Kannada Handwritten MNIST datasets.

The main advantage of the CNN, which consists of various associated layers, is supervised learning. The CNN [24], like humans, works to design and learn various local features. The main two functions required for any recognition system are feature extraction and feature classification. Feature extraction is the process of dimensionality reduction of the given input image pixels and converting it into interesting sub parts of the same image efficiently [25]. This interesting part helps to match with the original features of the image. Once the feature extraction is done, a classification algorithm should be implemented to classify the digits based on the extracted features [26] and original features. Without classification this recognition cannot be achieved. Therefore the extraction is done by the CNN as it is strong in the feature extraction process. To increase the accuracy of the classification process, different classifiers are implemented. Many alternative algorithms were employed for feature extraction and classification individually prior to the invention of CNN. It was comparable to onset prefound learning after CNN because it does not call for separate classification and feature extraction techniques. [27].

The SVM is applied to the labeled training dataset, they have the ability to classify the classes. Further, this helpsto digit classification problems. This classifier is chosen because the time complexity of SVM is more 8 Department. of CSE, UVCE June/July 2021 efficient for the large dataset. In modern days, the techniques of multilevel implementation to train data have been developed to utilize the time for training huge data. As regular, SVM operates with one time consuming and single optimization step.

The hyperplane is used as a class separator in Random Forest (RF), and it directs how to view multidimensional datasets. The RF achieves this by reducing prediction error and incomplete data. Even though RF is an excellent classifier, it struggles with noisy data. Furthermore, theRF faces numerous challenges, such as shallow design for learning deep features. KNN is one of the simple algorithms used in recognition applications or in Machine learning applications for classification problems. KNN is achieved by the data trained and classified by the data based on similar features (Extracted features). Classification is proposed by the majority vote to the neighbors, KNN works by the queryof different variations in the data and the query which selects closest to a specific number. After that, the majority votes with the most frequently labeled results in the classificationof a specific digit.

XGB is the most popular and efficient implementation for algorithms like gradient boosted tree, which is open source as well. In modern days XGB is implementedin Machine learning applications as it deals with either structured datasets or with a tabular dataset, based on prediction of the classification problems. The speed and performance are better with the XGB as it implements a decision tree design. This has much application which solves regression, ranking, classification, and solution for user- defined prediction, because of this reason many use this algorithm.

V. CONVOLUTION NEURAL NETWORK AND CLASSIFIER

A. CNN Architecture

Image processing is achieved through one of the functions known as image classifier, which is built using the best-suited algorithm known as CNN. CNN's architecture employsa multilayer neural network design that requires minimal

processing. Because the input dataset has been pre-qualified through pre-processing and data cleaning, there is no need to repeat the operation on CNN [28]. CNN can extract all of the available features that are shared by the group of input images used for learning. The classification of input images is performed using the extracted features, as shown in Fig [1]. CNN is null without input data and performs better with input image dependence. The methods for developing a better CNN algorithm with a high computational mean include various stages of resizing. Initially, only a few features are extracted for training, which is done only for a small number of data points because the process is [29]. Later on, the size of the data grows in stages. To improve feature extraction by replacing the initial convolutional layer with a new layer, a new dataset is required. This technique aids in the prevention of some issues such as overfitting and so on. Maxpooling is a popular algorithm for resizing images from larger to smaller model required sizes [30] this procedure contributes to faster computation.

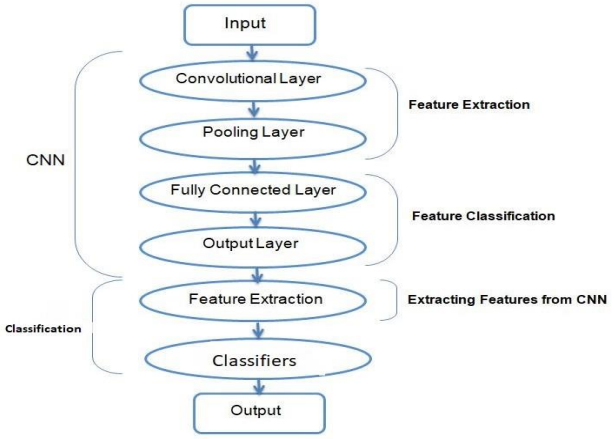

Fig. 1. Classification process of different classifiers

1) ***Convolutional Neural Networks Model****:* The proposed work is represented in Figure [2], which is made up of two convolutional layers and a maxpooling layer. The functionality of the convolution and maxpooling layers can be used to reduce parameter overflow after each convolutional layer. The convolutional layer is achieved by using 5 x 5 input kernels, and these combinations are repeated in the output of the maxpooling layer, which is directed to flatten layer over denselayer [33]. The flattening layers are used to reduce the number of data flow channels. The dense layer contribution is obtained after the Flattening layer is computed. The output is directed to specific nodes in a dense layer, from which the output is dropped [34], the next dense layer that produces the output.

In the implementation of the convolutional neural network, a convolution layer is the function where the set of multiplication of weights is involved with the input by linear operation, which is similar to the normal neural network. This convolutional layer is designed with the technique of accepting two dimensional arrays. A filter or kernel is used in this layer which is used to achieve multiplication between the weight of two-dimensional array and input array. [35]. The feature map which is covered by the filter is extractedby the pooling operation, in which the maximum element is selected from the same region known as Maxpooling. Hence, the maxpooling layer output is the final feature map which contains the most important features [36] compared to the previous mapping. The neural network with feed forward is simply called a fully connected layer, which is in the last few layers. The output of the convolutional layer or the maxpooling layer is the input for the fully connected layer [37], in which the flattened takes place and then fed to the fully connected layer. The Dense layer is also achieved in the last few layers of the neural network, where each and every neuron gets input from the layer of previous all neurons, hence called connected densely [38].

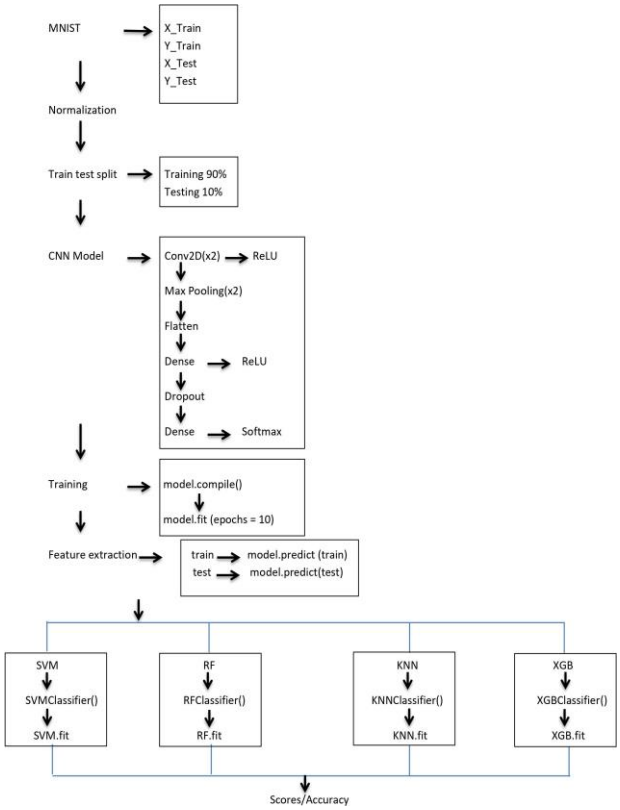

Fig. 2. Proposed System Architecture

Rectified linear activation function advantages The rectifiedlinear activation function has swiftly evolved into the standardactivation function for most types of neural networks.

1. Simplicity of Computers newline The max() function is all that is necessary to implement the rectifier function. The tanh and sigmoid activation functions, on the other hand, call for an exponential computation. Because activations do not require computing the exponential function, computations are also less expensive.

2. Representational Sparsity

The rectifier function's ability to generate a real zero valueis a benefit. The tanh and sigmoid activation functions, in contrast, learn to approximate a zero output, that is, a number that is nearly zero but not quite zero. By extension, hidden layers in neural networks can be triggered with one or more true zero values when negative inputs result in true zero values. A sparse representation is what is referred to as desirable in representational learning because it speeds up learning and makes the model simpler.

3. Linear Behavior

The rectifier function behaves and looks like a linear activation function most of the time. The behaviour of a neural network is typically simpler to optimise when it is linear or nearly linear. The concept behind rectified linear units is that models with more linear behaviour are simplerto optimise. It is crucial to this attribute because networks trained with this activation function almost entirely avoid the issue of disappearing gradients since the gradients continue to be proportionate to the node activations.

4. Train Deep Networks

Importantly, it was possible to successfully train deep multi- layered networks with a non-linear activation function using backpropagation by rediscovering and utilising the rectified linear activation function. Boltzmann machines, one of the more complex networks, as well as layer-wise training and unlabelled pre-training, two of the more complex training methods, may thus be forgotten.

B. Softmax

Softmax can be considered as a relaxed rendition of the argmax work that profits the record of the biggest worth in a run down. Step by step instructions to carry out the softmax work without any preparation in Python and how to change over the yield into a class name. A vector of K true qualities can be transformed into a vector of K true qualities that total to 1 by the softmax work. The softmax transforms the information values, which can be positive, negative, zero, or higher than one, into values between 0 and 1, allowing them to be understood as probabilities. The softmax work is in some cases called the softargmax work, or multi-class strategic relapse. This is on the grounds that the softmax is a

speculation of strategic relapse that can be utilized for multi- class grouping, and its equation is basically the same as the sigmoid capacity which is utilized for calculated relapse. The softmax capacity can be utilized in a classifier just when the classes are totally unrelated. Thus it is normal to annex a softmax work as the last layer of the neural organization.

optimizer = "adam", metrics = [accuracy])

Model summary.

13. model.summary()

Fit a Model.

model.fit(train x, train y, epochs, batch size, verbose, validationdata(val x, val y))

C. Algorithm : Procedure of CNN layers with trainingprocess

INPUT : train x, train y,: features(X) and labelled(Y)training dataset.

train x, train y,: features(X)and labelled(Y) training dataset.

OUTPUT : Accuracy : val c, train acc.

Loss : val loss, train loss.

1. Procedure CNN Model(train x, train y)# Parameters with arguments.
2. kernel size = 5; pool_size = 2; epochs = 10;
3. verbose = 1; batch_size = 512; classes = 10;
4. units = 1024;

Build a CNN model.

5. model = sequential();

Convolution Layer.

6. model.add(conv2D(filter_size = 32, kernel_ size,Padding = ”same”, activation = ”ReLu”))

Max-Pooling Layer.

7. model.add(MaxPooling2D(pool size))
8. model.add(conv2D(filter_size = 64, kernel_ size,Padding = ”same”, activation = ”ReLu”))
9. model.add(MaxPooling2D(pool size))

Flatten Layer.

10. model.add(Dense(units, activation = ”ReLu”))

Softmax Layer.

11. model.add(Dense(classes, activation = ”softmax”)

Compile function.

12. model.compile(loss=”categorial crossentrophy”,

15. model.evaluate(val x, val y, batch size, verbose)

train acc and train loss.

16. model.evaluate(train x, train y, batch size, verbose)

17. End Procedure.
18. Procedure extracting features.# Extraction.
19. Model(input=model.input,output=model.get layer(”dense”).output)

Training features.

20. model.predict(train x)

Validation features.

21. model.predict(val x)

22. End Procedure.

D. Support Vector Machine

Support Vector Machine (SVM) is designed by the strategy of sampling to maximize the SVM architecture space coverage, and build the constraints feasibly. The determination of the feasibility of the input process is done by the black box simulator on successful failure convergence. The separator called hyperplane can be determined by giving prior to the classificatioThe in-feasibility of the sample design can be made a feasible design by activating the SVM after some initial stages. Sometimes the classes cannot be able to separate linearly, at that time kernel function is used by the SVM to the implicit transformation of the input data,by which SVM using hyperplane allows to separate the non-linearly vectors. This SVM is implemented in our workto train the high features for various users behavior like GUI based and detection

of masquerade. Support vector is the values that is closest to the margin of classification, so the margin maximization between support vector and hyperplane is the main goal of SVM. The SVM supports only binary classification technique, the multi-class classification is achieved.

E. Random Forest

The features used by numerous algorithms were extracted and classified in an earlier implementation. It is not required to define explicitly because feature categorization is oneof CNN's applications for DL (Deep Learning). A multi- dimensional image collection is represented using RF, with each image belonging to a separate class and the hyperplane serving as a divider. RF employs decision trees to improve prediction and classification performance. The classification process is carried out by generating a large number of decision trees and generating a simple bootstrap for data training. RF searches for a subset of random variables to achieve a split at every node. For classification, the RF input vector is directed to each decision tree and considers the tree's vote for each class. Therefore, the classifier chooses a maximum number of votes for a particular class. The generalization error and invisible data are reduced by the RF classifier. RF describes that it has a shallow type of architecture that has difficultiesin deep feature learning.

F. K - Nearest Neighbour

The K-Nearest Neighbors algorithm is very old technique for the pattern recognition applications and classification process, in which some testing dataset is initialized to compare or check with the similar type of trained dataset. The better choice of k depends on the input data normally, the higher value of k decreases the effect of the wrong prediction in the classification. KNN also makes boundaries between each class that are distinct. Using the technique called heuristic the best value for k can be selected. Then the special case of class in which it is predicted to be the closest training variable of the class called Nearest Neighbors algorithm. Then the work is improved by the development of the algorithm by fuzzy version, which is done by introducing the fuzzy set theory into the KNN technique. As the improvement of the fuzzy version, it has many advantages compared to the old traditional technique as the advanced technology leads to the minimum error rate. The KNN algorithm was used alongside the CNN model as a combination of both to yield a better results for the classification process and also to achieve the correct prediction rate.

G. XG Boost

A gradient boosting framework is used in the decision tree-based machine learning calculation known as XG Boost. In predicting problems involving unstructured information, artificial neural systems frequently exceed all other algorithms or frameworks (pictures, content, etc.). Tree-based calculations are now regarded as best in class for small to medium-sized structured or tabular data. Both XGB and GBM are gathering tree systems that support weak learners by utilising the slope plummet design. On the other side, XG Boost enhances the fundamental GBM system by algorithmic upgrades and framework optimization.

VI. MNIST DATASET COLLECTION

To stay away from the sort of vulnerabilities and random data encompassing, we have chosen MNIST dataset to detail all parts of the information assortment measures. Further, we chose to open source crude output pictures, in order to work with start to finish experimentation with divergent sign preparing pipelines. In this segment, we will cover the subtleties of making the following two datasets:

1. The principle Kannada - MNIST dataset comprises set of 60000 28 x 28 dark scale test pictures and a test set of 10000 test pictures consistently dispersed across the 10 classes. This dataset is dependent on the endeavors of 65 volunteers from Bangalore, India, who are local speakers and clients of the Kannada language. This was curated to fill in as a direct oneto one drop in trade for the first MNIST dataset.

2. The Dig - MNIST dataset that comprises of 10240 28 X 28 dim scale pictures, that were curated determined to give a more testing dataset that was curated around their CA with the assistance of volunteers. A considerable lot of who were experiencing the Kannada script interestingly and had reasonable trouble in duplicating the state of the glyphs. This test dataset, we expectation will work with space variation tests.

VII. EXPERIMENTAL RESULTS AND DISCUSSION

A. Results

The accuracy rate is increased by 99.6% when the Kannada MNIST is trained using the CNN architecture, which uses two layers of convolutional and maxpooling layers, followed by flattening and dense layers. The RF classifier receives the extracted features from the CNN and improves accuracy by 99.65% percent. The SVM classifier receives the retrieved features from the CNN and increases accuracy by 99.66% percent. then with both KNN and XGB classifier it achieved the score of 99.63% each as shown in Fig [3] and Table I.

The above shows the validation scores of all classifiers. Here is the training score of every classifier i.e SVM is achieved with 99.93%, RF achieved with 100%, KNN achieved with 99.9% and finally XGB achieved with 100% of training accuracy Fig [4].

Accuracy is a metric that can be applied to characterization assignments as it were. It exactly which levels of your test information are characterized effectively. As an illustration Accuracy comes out to 0.98, or 98% (98 right expectations out of 100 all out models). That implies our proposed work or classifier is working really hard of recognizing gather names. When working with a class-imbalanced informative collection like this one, where there is a significant difference between the number of positive and negative names, accuracy alone doesn't tell the whole story. An crucial part of neural networks is the loss function. A prediction mistake in a neural network is all that loss is. And the Loss Function is the name of the formula used to determine the loss. Simple

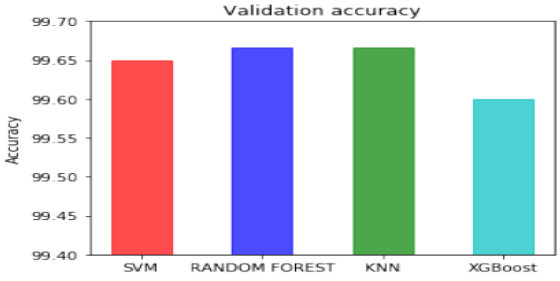

Fig. 3. Comparison of classifiers with validation score

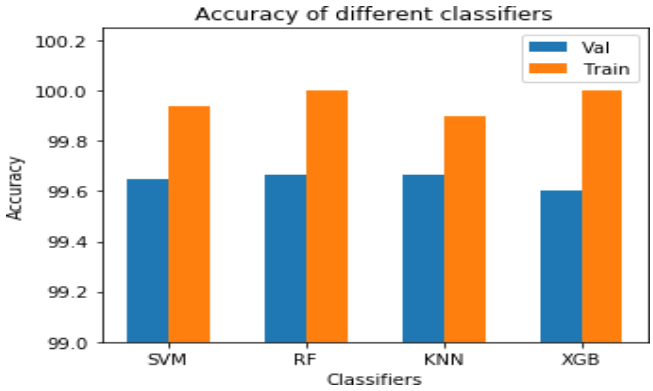

Fig. 4. Comparison of classifiers

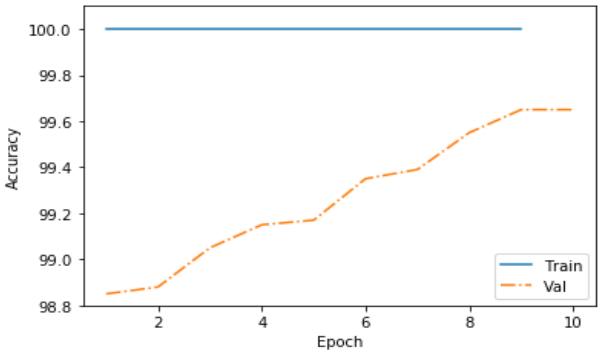

Fig. 5. CNN - Random Forest accuracy

put, the gradients are computed using the Loss. The weightsof the Neural Net are also updated using gradients. In thisway, a neural network is trained. For a number of goals, Keras and Tensorflow include built-in loss functions. How accurately you forecast classes in your classification problem will determine the loss. Using probabilities, for instance, your model could forecast binary class cat or non-cat values between 1 and 0. The chance of a non-cat is therefore 0.4 ifthe likelihood of a cat is 0.6. In this instance, the picture is categorised as a cat. The difference between the real classof the test picture's projected probability and 1 will be the loss. The training accuracy and validation accuracy components of the Random Forest Classifier graph are also included. Be- cause the training accuracy is familiar with the characteristics that are chosen for training and validation is like prediction using the taught features, training accuracy is better than validation accuracy. Random forest builds numerous decision trees and combines them to produce a more accurate and steady forecast. A decision tree or a bagging classifier's hyper-

parameters are quite similar to those of a random forest. The random forest increases the model's randomness as the trees grow. The accuracy for training is 100% and for validation is 99.66%, as shown in Fig [5]. Only classification tasks allow

for the application of accuracy measures. It provides a detailed breakdown of the percentage of testing data that are accurately predicted based on training. Scores for validation and training are computed using a percentage determined by each epoch's step. The accuracy will remain the same even if the number of epochs is increased see Fig. [6].

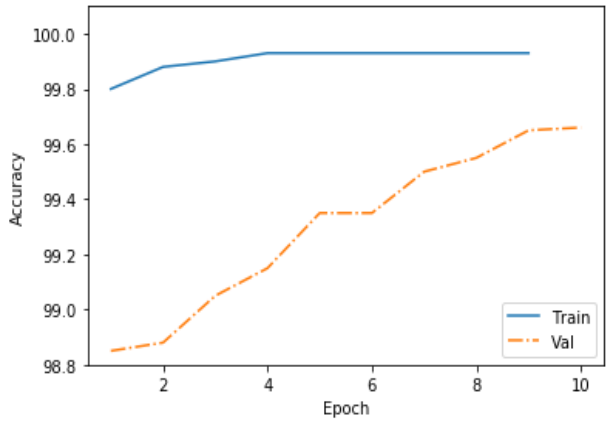

Fig. 6. CNN Validation accuracy

The error rate that is unable to anticipate the true value is referred to as loss. Even the loss score is specified in each epoch's step. The loss rate in the first epoch will be largerthan in the previous epoch. Loss is alternatively described as the total of incorrectly predicted as Figure [7] .

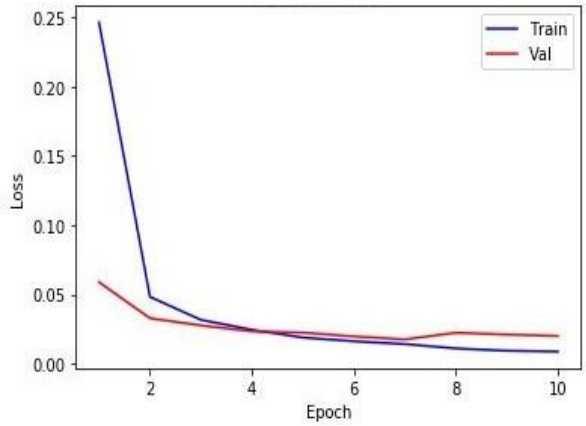

Fig. 7. CNN Validation Loss

The two components of the SVM graph that are imple- mented are training accuracy and validation accuracy. The term "test" or "testing" accuracy is frequently used to refer to the validation accuracy, which is the accuracy you determine usinga data set that was not used for training but was instead used during training to validate your model's generalizability or to "early stop." To address two-group classification issues, super- vised machine learning models called support vector machines (SVM) use classification methods. After being provided setsof labelled training data for each category, an SVM model may categorise new text. According to Fig [8], the training accuracy is 99.93% and the validation accuracy is 99.65%. The classification method known as K-Nearest Neighbor is

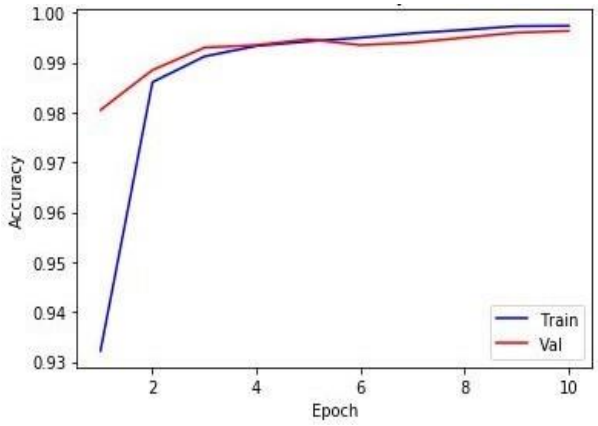

Fig. 8. CNN - SVM accuracy

shown as a graph with training and validation scores. As shown in Fig [9], the training accuracy is 99.91% and the validation accuracy is 99.66%. The training phase of the K- nearest neighbour methodology is significantly quicker than that of earlier classification systems. Because there is no need to train a model for generalisation, KNN is known as the simple and instance-based learning method. KNN produces predictions by looking at the k nearest neighbours of a casex to predict its y, so that's fine. In particular, the KNN model basically consists of its training cases - but that's the crossvalidation procedure doesn't care about at all.

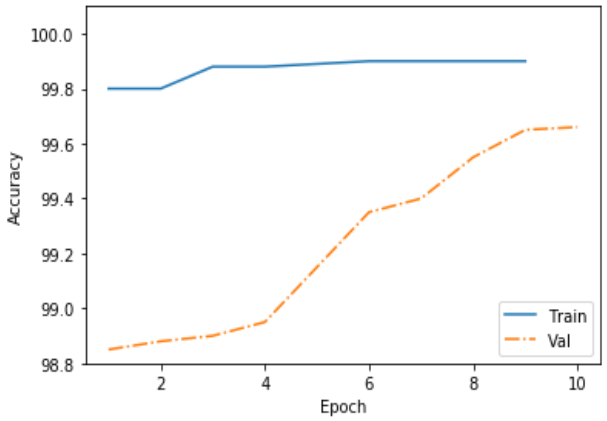

Fig. 9. CNN - KNN accuracy

XGB is the the classification technique is represented in the graph format which contain Trainings and validation scores. A wrapper class from XGBoost enables models to be used in the scikit-learn framework as classifiers or regressors. This means that XGBoost models can use the entire Scikit-Learn package. This approach should be used primarily because itis accurate, effective, and feasible. On a single machine, a linear model and a tree learning algorithm carry out concurrent computations. Additionally, it provides features for cross-validation and determining the relevance of features. The accuracy for training is 100% and for validation is 99.60% Fig [10].

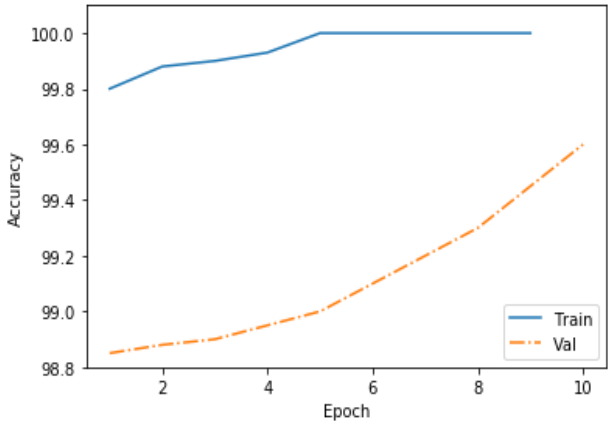

Fig. 10. CNN - XG Boost accuracy

TABLE I

COMPARISION OF DIFFERENT CLASSIFIERS WITH CNN

CNNwith CLASSI-FIERS	VALIDATION ACCURACY	TRAINING AC-CURACY
CNN + SVM	99.66	99.93
CNN + RF	99.65	100
CNN + KNN	99.63	99.21
CNN + XGB	99.63	100

TABLE II

PRECISION COMPARISON OF PREVIOUS WORKS

AUTHOR	METHOD	ACCURACY
Proposed work	**CNN Hybrid Classifier**	**99.60**
Zeeshan et al.,2020 [40]	HDR(cloud-based) + DL4J	99.41
Savita A et al.,2020 [41]	CNN + SVM	99.28
Ali S et al., 2019 [42]	CNN + DL4J	99.21
Dutt A et al.,2017 [43]	CNN(Keras + Theane)	98.70

VIII. CONCLUSION AND FUTURE WORK

This work with a hybrid model of CNN and 4 Classifiers is proposed for handwritten digit recognition. The work entails automatically creating features with CNN and predicting out-put with classifiers. Support Vector Machine (SVM), RandomForest (RF), K-Nearest Neighbors (KNN), and XG Boost are the classifiers utilised (XGB). The model combines the benefit

of CNN and each of the four classifiers separately for identify- ing the handwritten digits in kannada. ReLU (Rectified Linear Unit) activation is combined with the Python framework to

create a system in the current study. The recommended CNN architecture is effectively assisted with the right settings for the high accuracy of Kannada MNIST digit categorization.The number of CNN layers is also altered to further confirm the system's accuracy. The work just uses CNN to get a 99.60% accuracy rate. This is accomplished by training and testing using the 54000-item main dataset, 6000 test images and 28 x 28 grayscale training images.The test images are termed as validation. Then the scores with the combination of each classifier i.e. SVM, RF, KNN, XGB with CNN acquired 99.66%, 99.65%, 99.63%, 99.63% respectively.

The future work can be implemented with the following ways:

- The current work deals with the combination of CNN and any one classifier i.e among the four classifiers, a single classifier is used by CNN at a time. This can be improved by the combination of a strong feature extraction model like CNN with more than 2 classifiers at a time.

- This work can be implemented with the digits of other Languages, which may vary in the feature extraction mapping. This can find a better technique for the poor recognition rate.

- The proposed work is implemented only with the digits not with the other characters. The future work can make use of every characters in that particular language for the recognition.

- The layers of the CNN model can be increased for the better mapping of feature extraction and the nodescan be increased in the fully connected layer for best classification.

-

REFERENCES

[1] H. Basly, W. Ouarda, F. E. Sayadi, B. Ouni, and A. M. Alimi, "Cnn-svm learning approach based human activity recognition," in *International Conference on Image and Signal Processing*. Springer, 2020, pp. 271–281.

[2] H. E. Nouri, "Handwritten digit recognition by deep learning for automatic entering of academic transcripts," in *Proceedings of the Computational Methods in Systems and Software*. Springer, 2020, pp. 575–584.

[3] A. S. A. Rabby, S. Abujar, S. Haque, and S. A. Hossain, "Bangla handwritten digit recognition using convolutional neural network," in *Emerging Technologies in Data Mining and Information Security*. Springer, 2019, pp. 111–122.

[4] A. Ashiquzzaman and A. K. Tushar, "Handwritten arabic numeral recog- nition using deep learning neural networks," in *2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*. IEEE, 2017, pp. 1–4.

[5] A. S. Rao, S. Sandhya, K. Anusha, C. N. Arpitha, and S. N. Meghana, "Exploring deep learning techniques for kannada handwritten character recognition: A boon for digitization," 2020.

[6] E. Tuba, R. C. Hrosik, A. Alihodzic, R. Jovanovic, and M. Tuba, "Support vector machine optimized by fireworks algorithm for hand- written digit recognition," in *International Conference on Modelling and Development of Intelligent Systems*. Springer, 2019, pp. 187–199.

[7] Y. Nanehkaran, D. Zhang, S. Salimi, J. Chen, Y. Tian, and N. Al- Nabhan, "Analysis and comparison of machine learning classifiers and deep neural networks techniques for recognition of farsi handwritten digits," *The Journal of Supercomputing*, vol. 77, no. 4, pp. 3193–3222, 2021.

[8] G. S. Lopes, D. C. da Silva, A. W. O. Rodrigues, and P. P. Rebou- cas Filho, "Recognition of handwritten digits using the signature features and optimum-path forest classifier," *IEEE Latin America Transactions*, vol. 14, no. 5, pp. 2455–2460, 2016.

[9] G. Ramesh, G. N. Sharma, J. M. Balaji, and H. Champa, "Offline kannada handwritten character recognition using convolutional neural networks," in *2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*. IEEE, 2019, pp. 1–5.

[10] G. Ramesh, S. K. N, and H. Champa, "Ohkwr: Offline handwritten kannada words recognition using svm classifier with cnn," *International Journal of Innovative Technology and Exploring Engineering*, vol. 9,no. 10, pp. 2278–3075, 2020.

[11] Y. Hou and H. Zhao, "Handwritten digit recognition based on im- proved bp neural network," in *Chinese Intelligent Systems Conference*. Springer, 2017, pp. 63–70.

[12] R. Almodfer, S. Xiong, M. Mudhsh, and P. Duan, "Very deep neural networks for hindi/arabic offline handwritten digit recognition," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 450–459.

[13] G. Ramesh, S. Kumar *et al.*, "Recognition of kannada handwritten words using svm classifier with convolutional neural network," in *2020 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 2020, pp. 1114–1117.

[14] G. Ramesh, W. Srihari, A. M. Bharadwaj, and H. Champa, "Kannada imagenet: A dataset for image classification in kannada," in *2021 International Conference on Computer Communication and Informatics (ICCCI)*. IEEE, 2021, pp. 1–4.

[15] S. Oh, Y. Shi, X. Liu, J. Song, and D. Kuzum, "Drift-enhanced unsupervised learning of handwritten digits in spiking neural network with pcm synapses," *IEEE Electron Device Letters*, vol. 39, no. 11, pp. 1768–1771, 2018.

[16] S. Aly and A. Mohamed, "Unknown-length handwritten numeral string recognition using cascade of pca-svmnet classifiers," *IEEE Access*, vol. 7, pp. 52 024–52 034, 2019.

[17] G. Ramesh, W. S. G. Srihari, and H. Champa, "Deep convolutional neural networks for handwritten kannada numerals recognition," vol. 8, pp. 1–8, 2020.

[18] G. Ramesh, J. M. Balaji, G. N. Sharma, and H. Champa, "Recognition of off-line kannada handwritten characters by deep learning using capsule network," *Int. J. Eng. Adv. Technol*, vol. 8, no. 6, pp. 4767–4777, 2019.

[19] S. Aly and S. Almotairi, "Deep convolutional self-organizing map network for robust handwritten digit recognition," *IEEE Access*, vol. 8, pp. 107 035–107 045, 2020.

[20] N. Altwaijry and I. Al-Turaiki, "Arabic handwriting recognition system using convolutional neural network," *Neural Computing and Applica- tions*, vol. 33, no. 7, pp. 2249–2261, 2021.

[21] G. Ramesh, G. B. Prasanna, V. B. Santosh, C. Naik, and H. Champa, "An efficient and improved method for handwritten kannada digit recognition based on pca and svm classifier," vol. 8, pp. 1–13, 2020.

[22] H. M. Balaha, H. A. Ali, M. Saraya, and M. Badawy, "A new arabic handwritten character recognition deep learning system (ahcr-dls)," *Neural Computing and Applications*, pp. 1–43, 2020.

[23] D. Chikmurge and R. Shriram, "Marathi handwritten character recog- nition using svm and knn classifier," in *International Conference on Hybrid Intelligent Systems*. Springer, 2019, pp. 319–327.

[24] A. Hazra, P. Choudhary, S. Inunganbi, and M. Adhikari, "Bangla-meitei mayek scripts handwritten character recognition using convolutional neural network," *Applied Intelligence*, vol. 51, no. 4, pp. 2291–2311, 2021.

[25] A. Gattal, C. Djeddi, Y. Chibani, and I. Siddiqi, "Oriented basic image features column for isolated handwritten digit," in *Proceedings of the International Conference on Computing for Engineering and Sciences*, 2017, pp. 13–18.

[26] F. Hasan, S. N. Shuvo, S. Abujar, and S. A. Hossain, "Bangla hand- written math recognition and simplification using convolutional neural network," *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020, Volume 2*, p. 133.

[27] M. M. Zayed, S. N. K. Utsha, and S. Waheed, "Handwritten bangla character recognition using deep convolutional neural network: Com- prehensive analysis on three complete datasets," in *Proceedings of International Conference on Trends in Computational and Cognitive Engineering*. Springer, 2021, pp. 77–87.

[28] K. T. Islam, G. Mujtaba, R. G. Raj, and H. F. Nweke, "Handwritten digits recognition with artificial neural network," in *2017 International Conference on Engineering Technology and Technopreneurship (ICE2T)*. IEEE, 2017, pp. 1–4.

[29] A. Sahu and S. Mishra, "Offline odia

handwritten characters recognition using weka environment," *Intelligent Systems: Proceedings of ICMIB 2020*, p. 477.

[30] Y. Zhang, X. Guo, Y. Li, and S. Wang, "An off-line handwritten numeral recognition method," in *International Conference on Computer Engineering and Networks*. Springer, 2020, pp. 1271–1278.

[31] S. A. Azeem, M. El Meseery, and H. Ahmed, "Online arabic handwritten digits recognition," in *2012 International Conference on Frontiers in Handwriting Recognition*. IEEE, 2012, pp. 135–140.

[32] D. Paul, P. K. Pattnaik, and P. Mukherjee, "A robust approach with text analytics for bengali digit recognition using machine learning," in *Multimedia Technologies in the Internet of Things Environment*. Springer, 2021, pp. 157–169.

[33] R. Pramanik and S. Bag, "Segmentation-based recognition system for handwritten bangla and devanagari words using conventional classifica- tion and transfer learning," *IET Image Processing*, vol. 14, no. 5, pp. 959–972, 2020.

[34] C. Vinotheni, S. L. Pandian, and G. Lakshmi, "Modified convolutional neural network of tamil character recognition," in *Advances in Dis- tributed Computing and Machine Learning*. Springer, 2021, pp. 469– 480.

[35] Y.-K. Chen and J.-F. Wang, "Segmentation of single-or multiple- touching handwritten numeral string using background and foreground analysis," *IEEE Transactions on pattern analysis and machine intelli- gence*, vol. 22, no. 11, pp. 1304–1317, 2000.

[36] N. Rahal, M. Tounsi, T. M. Hamdani, and A. M. Alimi, "Handwritten words and digits recognition using deep learning based bag of features framework," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 701–706.

[37] C. Ma and H. Zhang, "Effective handwritten digit recognition based on multi-feature extraction and deep analysis," in *2015 12th international conference on fuzzy systems and knowledge discovery (FSKD)*. IEEE, 2015, pp. 297–301.

[38] A. F. D. S. Neto, B. L. D. Bezerra, E. B. Lima, and A. H. Toselli, "Hdsr- flor: A robust end-to-end system to solve the handwritten digit string recognition problem in real complex scenarios," *IEEE Access*, vol. 8, pp. 208 543–208 553, 2020.

[39] W. Liu, J. Wei, and Q. Meng, "Comparisions on knn, svm, bp and the cnn for handwritten digit recognition," in *2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*. IEEE, 2020, pp. 587–590.

[40] Z. Shaukat, S. Ali, C. Xiao, S. Sahiba, A. Ditta *et al.*, "Cloud-based efficient scheme for handwritten digit recognition," *Multimedia Tools and Applications*, vol. 79, no. 39, pp. 29 537–29 549, 2020.

[41] S. Ahlawat and A. Choudhary, "Hybrid cnn-svm classifier for hand- written digit recognition," *Procedia Computer Science*, vol. 167, pp. 2554–2560, 2020.

[42] S. Ali, Z. Shaukat, M. Azeem, Z. Sakhawat, T. Mahmood, and

K. ur Rehman, "An efficient and improved scheme for handwritten digit recognition based on convolutional neural network," *SN Applied Sciences*, vol. 1, no. 9, pp. 1–9, 2019.

[43] A. Dutt and A. Dutt, "Handwritten digit recognition using deep learn- ing," *Int J AdvRes Comput Eng Technol*, vol. 6, no. 7, pp. 990–997, 2017.

[44] G. Mayraz and G. E. Hinton, "Recognizing handwritten digits using hierarchical products of experts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 189–197, 2002.

[45] Z. Kayumov and D. Tumakov, "Convolution neural network learning features for handwritten digit recognition," in *2020 IEEE East-West Design & Test Symposium (EWDTS)*. IEEE, 2020, pp. 1–5.

[46] S.-B. Cho, "Self-organizing map with dynamical node splitting: Appli- cation to handwritten digit recognition," *Neural Computation*, vol. 9, no. 6, pp. 1345–1355, 1997.