International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 37, No. 4, August, 2020, pp. 661-669
Journal homepage: http://iieta.org/journals/ts

# Handwritten Gurmukhi Digit Recognition System for Small Datasets

Gurpartap Singh[1*], Sunil Agrawal[1], Balwinder Singh Sohi[2]

[1] U.I.E.T., Panjab University, Chandigarh 160017, India
[2] Chandigarh University, Gharuan, Punjab 140413, India

Corresponding Author Email: gpsphd1990@pu.ac.in

https://doi.org/10.18280/ts.370416

**Received:** 6 May 2020
**Accepted:** 10 August 2020

***Keywords:***
*DCT, DWT, support vector machine, deep convolutional neural networks, Gurmukhi handwritten digit recognition*

## ABSTRACT

In the present study, a method to increase the recognition accuracy of Gurmukhi (Indian Regional Script) Handwritten Digits has been proposed. The proposed methodology uses a DCNN (Deep Convolutional Neural Network) with a cascaded XGBoost (Extreme Gradient Boosting) algorithm. Also, a comprehensive analysis has been done to apprehend the impact of kernel size of DCNN on recognition accuracy. The reason for using DCNN is its impressive performance in terms of recognition accuracy of handwritten digits, but in order to achieve good recognition accuracy, DCNN requires a huge amount of data and also significant training/testing time. In order to increase the accuracy of DCNN for a small dataset more images have been generated by applying a shear transformation (A transformation that preserves parallelism but not length and angles) to the original images. To address the issue of large training time only two hidden layers along with selective cascading XGBoost among the misclassified digits have been used. Also, the issue of overfitting is discussed in detail and has been reduced to a great extent. Finally, the results are compared with performance of some recent techniques like SVM (Support Vector Machine) Random Forest, and XGBoost classifiers on DCT (Discrete Cosine Transform) and DWT (Discrete Wavelet Transform) features obtained on the same dataset. It is found that proposed methodology can outperform other techniques in terms of overall rate of recognition.

## 1. INTRODUCTION

Accurate recognition of handwritten characters is one of the most challenging yet essential tasks when it comes to interfacing with an AI system. Due to unpredictable variations in handwritten characters, shallow machine learning techniques are not able to give satisfactory performance. Since the last decade the field of deep learning has gained immense popularity for handwriting recognition. Several improvised deep learning algorithms [1-4] have been reported in the literature, which exhibit state of the art performance in most applications. But still, there are some issues with these networks which need to be addressed. A detailed survey [5] on deep learning algorithms reflects number of issues and challenges associated with applications. For instance, time consumed in training the network is considerably large and also these algorithms have barely at par performance with other shallow networks when used with small datasets. Hence the only choices are either to collect large datasets every time or to train network again and again with same dataset which might lead to overfitting. In fact several attempts have been made to resolve these issues [6-8] but they all have led to comparatively complex architectures. Moreover, there are around 4000 languages that have written formats completely different from each other; hence same approach may not be able to give satisfactory performance for all of them. This can be observed from the modifications done in deep learning algorithms to customize them to recognize some regional languages [9-12]. Therefore it is always desirable to design a deep learning based methodology which can be trained with small dataset for accurate recognition of specific language.
Before the advent of deep learning algorithms, the most undesirable and time consuming task in handwritten digit recognition was extraction of features. One has to analyse the data thoroughly again and again to make sure that the extracted features are good enough to build an accurate classifier. Researchers had to spend months on just this part causing the focus being diverted from the actual problems in hand. But with the breakthrough made by Hinton et al. [13], a new methodology came into existence addressing the issue of feature extraction in a quite efficient way. Instead of manually extracting the features one can exploit the benefit of an unsupervised learning algorithm, by training it layer by layer and let it learn a deeper representation of the data.

Basically, a deep network is a network connected by highly varying functions which adapt continuously during the training unless they attain an optimal representation. The biggest benefit of using deep learning is that network is free to learn the best representation of data. Hence, instead of making estimation, these deep learning algorithms rely upon mathematical equations to converge at a point which humans cannot reach manually. Therefore deep learning algorithms can offer a promising solution for problems involving recognition of handwritten characters. For this purpose, a deep learning based methodology should be devised which can be trained efficiently even with small dataset for accurate recognition of specific language.

### 1.1 Contributions

The contributions made by this paper are mentioned as follows:

1. Comparative analysis of shallow networks based on DCT and DWT features followed by polynomial SVM classifier with DCNN network which are used to extract highly accurate deep features followed by softmax classifier. The dataset used to train the DCNN is a very small dataset consisting of 960 images only.
2. Initially a small dataset has been used for implementing DCNN. But DCNN generally give exceptional result for larger datasets. The original dataset is expanded by introducing shear to get large dataset. The reason for introducing shear is quite logical. When a person writes a specific character again, then some rotation and size variation of character can be easily observed. This ensures that the data generated is as close to real time data as possible.
3. Finally, a selectively cascaded XGBoost Classifier has been used along with softmax classifier on the features extracted from the DCNN to improve the recognition rate.
4. This work presents a new approach to increase the accuracy of an algorithm on a very small dataset without making the network more complex and time consuming.

### 1.2 Paper organisation

In Section 2 of this paper a detailed overview of the work done on handwritten numerals recognition with a focus on Gurmukhi digits is given in which many states of the art algorithms are mentioned along with their contributions. In Section 3, problem statement is given discussing the issues in pre-existing algorithms followed by the Section 4 in which the complete methodology is given to overcome the issues. Then in section 5 the experimental set up, implementation of algorithm and final results along with comparisons are mentioned.

## 2. LITERATURE REVIEW

Punjabi language (Gurmukhi Script) is the 10th most widely spoken language in the world. But the work done on Punjabi handwritten character recognition is comparatively less. Also the work being done in Punjabi is quite less accurate and lacks any futuristic perspective. This work is centred on Punjabi handwriting recognition so an overview of the work reported in the literature on Punjabi handwriting recognition is given below followed by the work being done recently in other languages.

Kumar and Singh [14] have elaborated that the scanned text image is a non-editable image. In this work a basis for an OCR (Optical Character Recognition) consisting of pre-processing, segmentation and then recognition is given. They have formulated an approach to segment the scanned document image. As per this approach, the whole image is initially considered as one large window, which is broken into small windows giving lines. Once the lines are identified then each window consisting of a line is used to find a word present in that line and finally on characters. Jindal and Sharma [15] have discussed the grading of writers. It has been accomplished based on statistical measures of distribution of points on the bitmap image of characters. The features that have been used for classification were based on zoning, which can uniquely grade the characters. The complete data sets used in their work consisted of hundreds of different Punjabi handwritten characters. The zoning used was diagonal, directional, intersection and open end points feature extraction techniques along with *k*-NN(*k*-nearest Neighbour), HMM(Hidden Markov Model) and Bayesian decision making classifiers for classification.

Singh et al. [16] have compared the features obtained from distance profiles, projection histograms, zonal density and BDD (Background Directional Distribution) along with and SVM (Support Vector Machine) as classifier. Authors have also used pre-processing and noise removal in order to obtain better results.

Singh et al. [17] have discussed a way of organizing the features under four strategies specific to Punjabi language. The authors have focused on distance profile features, projection histogram features and zoning densities hybrid Features. Further they have discussed the zoning of the handwritten data i.e. they considered dividing the Punjabi handwritten data into three zones namely Upper Zone, Middle Zone, and Lower Zone. They demonstrated that the wise use of features can lead to improvement in accuracy. But due to lack of sufficient datasets authors were not able to give any solid base to their theory. Deore and Pravin [18] have focused on significance of Histogram oriented gradient features and along with that they have used an ensemble of classifiers consisting of SVM, K-Nearest Neighbour(K-NN) and Neural Network (NN). This ensemble has greatly improved the recognition accuracy of the entire system

Punjabi handwritten characters were recognized using wavelet based features along with consideration of three zones. The features were obtained through DWT (Discrete Wavelet Transform) decomposition using various wavelet coefficients [19]. There are various wavelet families to choose from, so in this work authors have used several families and then compared accuracies achieved from each family and its variants. Similar method was followed by Kumar et al. [20] for performance comparison of DWT, DCT, and FFT (Fast Fourier Transforms). Similar shallow algorithms have been implemented by researchers in articles [21, 22]. After taking into consideration one can conclude the work done on Punjabi character recognition is not at par with the work being reported on other languages. It can be verified from Ahmed et al. [23] as well that if a large amount of work has been implemented for Arabic character recognition, then it is important to advance the work on other languages like Gurmukhi. Therefore it is need of the hour to investigate state-of-the-art technologies for accurate recognition of handwritten characters of Gurmukhi.

Li et al. [4] have used deep Convolutional neural networks in order to obtain high recognition rate on MNIST (Modified National Institute of Standards and Technology database) handwritten benchmark data. The authors have used multiple convolutional, normalization, Max pooling and RELU (Rectified Linear Unit) layers. Their work explored the connectivity of different layers to determine the knowledge being learnt by the algorithm. These changes remarkably increased the depth of the network thereby leading to high recognition rate. Again in O'Shea [2] have shown a further improvement in recognition rate by using restricted Boltzmann machines (A Generative network capable of learning probability distribution of data) on the same dataset. The main contribution of their work is optimized number of neurons to be used in each layer. As the number of layers and number of neurons in each layer are increased beyond certain values, then only a marginal improvement in the learning capability of the algorithm can be achieved. Similar observations were made in work done by Ciresan et al. [3]. But with increase in number of layers and neurons, more time is consumed in training and testing, thereby making them less suitable for real time applications.

After going through the existing literature on work under consideration one reaches to a conclusion that there is a dire need to improve pre-existing algorithms over various concerns. One is that earlier shallow networks were being used to classify handwritten digits and they are dependent on manually extracted features. These manually obtained features cannot give accuracy beyond a certain point. If a researcher claims to have achieved high accuracy using these features then obtained results would be somewhat biased and will not give similar results on other datasets.

Above mentioned remark is also valid for applications other than handwritten character recognition. There are several implementations reported by researchers on applications from different fields substantiating the validity of above mentioned remark. To state a few examples, work done on medical image classification by Neelapu et al. [24], on facial recognition done by Benkaddour and Bounoua [25] and on emotion classification done by Demircan and Örnek [26]. Hence, one ought to switch onto deep learning techniques. But in deep learning techniques there are even more complex architectures which require a large amount of data in order to learn a general representation. Moreover, there are issues like increased training/testing time and overfitting in these algorithms. In addition, the data requirement of deep learning algorithms is quite high. So there is a need to reduce the extent of overfitting and training time by making the deep neural network simpler and at the same time maintaining a high recognition rate for small datasets. Keeping these issues in view, researchers are trying to design an algorithm that can process the signal faster in comparison to the other networks [2-4] and also gives reasonably good classification accuracy for small datasets.

## 3. METHODOLOGY

In this work at first some shallow models have been implemented using DCT and DWT Features followed by SVM, Random Forest and XGBoost classifiers. The issue with shallow models is that they cannot learn the in-depth representation of data. Their performance greatly depends on features and they are as good as the features that are being extracted. Keeping this limitation in view, a deep learning network DCNN has been implemented and the results thus obtained have been compared with that of shallow networks. Later same deep learning algorithm have been implemented with some modifications so as to get satisfactory performance even for small datasets. In the subsequent sub-sections, a brief introduction is given for techniques which are part of the proposed methodology.

### 3.1 DCT features with SVM classifier

DCT features have shown great potential when it comes to handwritten character recognition task. This is the main reason that these transforms have been chosen to convert data images into its elementary frequency components. The DCT has a significant property that it can present most significant information in the image in terms of just a few coefficients. After the DCT features have been obtained, they are fed into SVM for classification of handwritten digits.

### 3.2 DWT features with SVM classifier

DWT features are the state of the art features which segregate the image into low and high frequency coefficients, a by applying sub-band decomposition and sophisticated thresholding. The real operation is of course more discrete and complex than that. For current implementation it is sufficient to understand that the image will have the noise reduced and sharpness increased, as only high frequency coefficients of the transformation are being considered. The DWT features thus obtained are used to train SVM, Random Forest, and XGBoost classifiers for classification of handwritten digits.

### 3.3 Random forest classifier

One of the good performing machine learning algorithms is decision tree based classifier. These classifiers have the ability to make decisions based on features provided to it. Random forest is an ensemble of several trees and has been designed to aggregate the result of each tree under consideration before declaring the classification result. The main reason for excellent performance of Random forest is because it uses ensemble techniques which are able to remove the overfitting to a great extent. It was proposed by Breiman [27] and since then several researchers [28-31] have reported its superior performance for handwriting classification.

### 3.4 DCNN with Softmax classifier

CNN (Convolutional neural networks) are based on recognizing visual structures. The main feature of convolutional networks is that they take advantage of local connectivity between neurons. The main advantage of convolutional neural network is that they can obtain translational invariance because the neurons with same parameters are applied on the segment of images. Then LeCun et al. [32] applied backpropagation using error gradient which resulted in a great deal of improvement in the recognition rate. The CNN networks used here consist of MaxPooling with convolutional layer along with ReLU (Rectified Linear Unit) as activation. Two of these layers have been included which are trained one by one using Greedy layer wise training approach. And then normalization is used followed by another layer consisting of 300 neurons which are finally connected to the output layer. For classification Softmax regression is used. The DCNN architecture with Softmax classifier used in this work is shown in Figure 1.

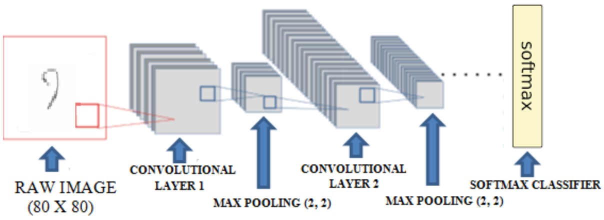

**Figure 1.** Architecture of DCNN with Softmax classifier

### 3.5 XGBoost

XGBoost algorithm is based on the greedy function approximation suggested by Friedman [33]. This tree boosting system was suggested by Chen and Carlos [34]. Gradient boosted trees have been there for a very long time but the results given by XGBoost are better in comparison to other boosted trees. The reason behind the superior performance of XGBoost is that it is able to capture the complex data dependencies using effective statistical models and secondly it has a scalable learning system that learns the appropriate model from large datasets. This system not only gives state-of-the-art recognition rate but also runs ten times faster than existing algorithms. Training loss and regularization are two objective functions in XGBoost where regularization function tries to control overfitting.

There is a large similarity between Random Forest and gradient boosted architecture with one main distinction i.e. they are trained in different ways. In XGBoost an additive training step is included which is used in fixing, what has been learned previously. Also it is not easy to train all the trees at once hence an additive strategy is used in XGBoost trees where a tree is added at a time. So additive training is what makes it so effective in recognition tasks. XGBoost also calculates the model complexity and the structure score to measure effectiveness of a tree structure. All these merits make XGBoost top of the line algorithm and for this very reason it is being used in the proposed methodology.

### 3.6 Proposed architecture for DCNN

Based on the study of DCNN with softmax classifier one ought to reach a conclusion that if another layer is added to the network in order to improve the overall recognition rate, the training time will increase tremendously. Hence, instead of adding an extra layer of neural network only selective cascading of old network with XGBoost algorithm has been done. This will not only increase the overall classification accuracy but also will keep in check the training and testing time. The proposed architecture is discussed in detail below.

The architecture of proposed method consists of features of both DCNN and XGBoost algorithms. It has already been established that the accuracy of DCNN with softmax classifier is very high even on the small dataset. Even some of the numerals are perfectly classified but a very few have poor classification accuracy. In case of XGBoost classifier the classification rate is relatively low, when it comes to classifying all the 10 numerals. But there are few classes in which recognition rate of XGBoost classifier is better than softmax classifier.

Whenever the softmax classifier gives an output which belongs to the highly misclassified numerals the features obtained from the DCNN will be fed into XGBoost. Then output of XGBoost is compared with that of softmax classifier in terms of recognition accuracy and better response is chosen. In this way the XGBoost is made to compensate where the softmax classifier is lacking thereby improving overall recognition rate. The architecture of selectively cascaded DCNN with XGBoost classifier is shown in Figure 2.

But it needs to be remembered that XGBoost is a tree classifier and when it is used as a cascaded classifier with more number of classes it will become less robust hence it will be difficult to maintain the accuracy.

In order to test the performance of pre-existing algorithms and proposed algorithm a large dataset is required. The dataset used in this work contains handwritten numerals in Gurmukhi Script. The detail of the dataset used in this work is described in the next sub-section.

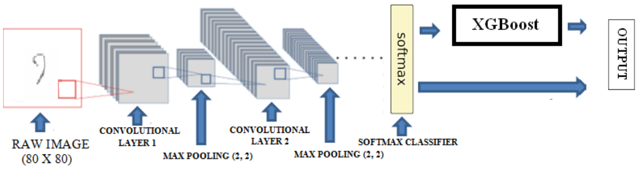

**Figure 2.** Proposed algorithm

### 3.7 Data set

The data set used in this work is collected locally from 24 individuals where each individual has written all the ten digits four times, thereby making total 960 samples. First of all the images are scanned in grey-scale format and all the numerals were cropped from the boxes, a sample of which is shown in the Figure 3.

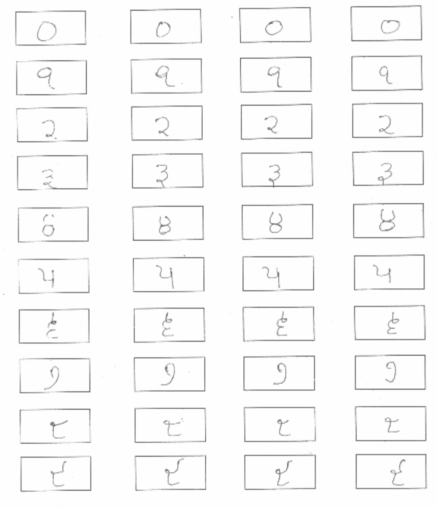

**Figure 3.** A sample of format used for data collection

Once the dataset is created, the next step is to divide this into smaller datasets and earmark them for training, validation and testing. This division is different for shallow networks and deep learning networks as shown in Table 1 and Table 2.

**Table 1.** Division of data for shallow networks

| Category of Data | Image dimensions | Appended Image Dimensions | No. of images |
|---|---|---|---|
| Training/ Validation | 80 x 80 | 1 x 6400 | 86x10=860 |
| Testing | 80 x 80 | 1 x 6400 | 10x10=100 |

**Table 2.** Division of data for DCNN

| Category of Data | Image dimensions | Appended Image Dimensions | No. of images |
|---|---|---|---|
| Training | 80 x 80 | 1 x 6400 | 76x10=760 |
| Validation | 80 x 80 | 1 x 6400 | 10x10=100 |
| Testing | 80 x 80 | 1 x 6400 | 10x10=100 |

As it is well known fact that deep architectures work well only when data available is huge so in order to increase volume of dataset, shear is introduced in the images. Each image is replicated ten times by introducing shear in them a sample of which is shown in Figure 4. In this way the new dataset is having total 960 x 11(10 images generated by introducing shear and 1 original) images. During this process of expansion, the data set is divided into three parts namely Train, Validate and Test datasets. There are 96 original images for each numeral, out of which first 76 images are taken as training images, next 10 as validation images and remaining 10 as test images. For expanded dataset, there is 76x10x11(no. of samples of each numerals x no. of numerals x sheared images) training samples and 10x10x11 validation and test samples. The dimensions of images were kept to be 80 x 80. The final dataset consists of only complement of all images, a sample of which is shown in Figure 5.

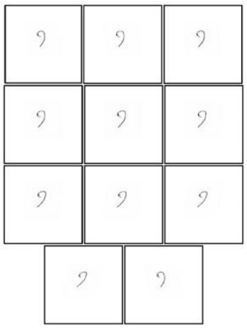

**Figure 4.** Images after shear was introduced

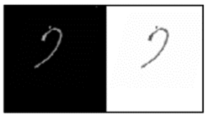

**Figure 5.** A sample of original image and its complement

The reason for taking a complemented image is that black is numerically represented as 0 and white as 255 on grey scale. It can be observed that almost all of the image consists of maximum white pixels, which can lead to increased computational complexity. Therefore in order to reduce the number of calculations the image is complemented and then fed into the algorithm. Then the complete image was appended in a single row format and written into an excel csv (Comma Separated Value) file. The final division of expanded dataset is mentioned in Table 3.

**Table 3.** Division of dataset after introducing shear

| Category of Data | Image dimensions | Appended Image Dimensions | No of images |
|---|---|---|---|
| Training | 80 x 80 | 1 x 6400 | 8360 |
| Validation | 80 x 80 | 1 x 6400 | 1100 |
| Testing | 80 x 80 | 1 x 6400 | 1100 |

## 4. EXPERIMENTAL SETUP, EXECUTION AND RESULTS

### 4.1 Experimental setup

The algorithms mentioned in section - 4 were implemented on a computer with **i7**-4770, 3.4GHz processor, 12GB of DDR3 RAM and 500 GB hard disk with Ubuntu 15.04 operating system platform. The language used for programming is Python.

### 4.2 Executing shallow networks and DCNN on original dataset

Initially the algorithms are implemented on original dataset consisting of 960 images. The dataset details are mentioned in Table 1 and Table 2 and results are summarised in Table 4.

From this Table, it can be observed that accuracy of SVM with DWT features is higher than SVM with DCT features, henceforth work only with DWT features will be considered for comparison. Among SVM, Random Forest and XGBoost accuracy of Random forest is inferior, therefore it will also be not considered for further analysis. Main observation here is that even for a small dataset the accuracy of DCNN is highest but with only a small margin. The reason behind this is the lack of a larger dataset, which could have helped in training the network up to a point of convergence thereby producing a more generalised representation of data. So in order to increase the accuracy a detailed analysis of DCNN for expanded dataset is given in the upcoming sections so that better recognition accuracy can be obtained.

**Table 4.** Recognition accuracy for various techniques

| Classification Technique<br>Performance Parameters | SVM with DCT | SVM with DWT | Random Forest with DWT | XGBoost with DWT | DCNN |
|---|---|---|---|---|---|
| Training Accuracy(%) | 97.45 | 98.49 | 100 | 100 | 100% |
| Validation Aaccuracy(%) | 94.31 | 96.52 | 93.26 | **94.27** | **97.91%** |
| Testing Aaccuracy(%) | **91.375** | **94.475** | **92.25** | **94.5** | **94.8%** |
| Training Time(s) | 852 | 721 | 245 | 311 | 1827 |
| Testing Time(ms) | 7.0 | 7.0 | 1.1 | 1.2 | 7.0 |

### 4.3 Executing the DCNN algorithms over expanded dataset

In order to increase recognition accuracy initially collected data is expanded by introducing shear into them, as already described in previous sections. After introducing shear the training data with dimensions 8360 x 6400 and validating data with dimensions 1100 x 6400 are fed into the algorithm. Initially the features will be obtained by training DCNN. As DCNN uses an unsupervised learning model for training, the network can learn highly varying non-linear features i.e. in depth representation of data. Here in depth representation of data actually refers to the count of composition layers for learning non-linearity in the functions. But it must also be taken into consideration that higher order of nonlinearities may also lead to higher extent of overfitting. Therefore an equivalent representation is to be learned so that when new entries/data occur they can also fit suitably in the already trained model. Keeping this point in focus, training data, validation data and test data all are made mutually exclusive by using handwritten characters from different persons. Also this property of data is maintained by introducing shear in the images only after separating the original 960 images into three parts namely training, validation and testing sets.

The DCNN used in this work have two hidden layers. More hidden layers can be added but it will only increase the training/testing time and complexity of the network as well as overfitting which are most undesirable parameters in a deep learning algorithm. The detailed architecture of DCNN is mentioned in the Table 5.

**Table 5.** Detailed architecture of DCNN

| Parameter<br>Stage | Batch Size | Neurons | Activation | MaxPooling |
|---|---|---|---|---|
| Layer 1 | 209 | 30 | RELU | 2 x 2 |
| Layer 2 | 209 | 30 | RELU | 2 x 2 |
| Dense Layer | 209 | 500 | RELU | - |

The data is presented to the DCNN in batches of size 209 and this process is repeated 10 times (10 epochs). First hidden layer consists of 30 neurons and kernel of variable size is swept across the whole image after that activation function RELU is applied followed by 2 x 2 max pooling. The same processing steps are applied on a second hidden layer consisting of 30 neurons. Also a part of validation data (100 out of 1100 images) is included with the training data during learning procedure to investigate the extent of overfitting. The same accuracies on validation data and the test data ensures that network has learned perfect equivalent representation of handwritten Gurmukhi numerals.

Once the network completes its training the initial 6400 features are reduced to only 500, to reduce the computation complexity during validation and testing. Now with these convolved deep features, validation and test data are presented to a softmax classifier for calculation of respective recognition accuracy. The impact of kernel size on total training time and recognition accuracy is also investigated, a summary of which is presented in Table 6. It is found that in all cases the recognition rate on training data remains to be 100%. A comparison of recognition rate with kernel size is mentioned in the next part of this section. The reason behind varying kernel size is to reduce the extent of overfitting.

From Table 6, it is clear that validation rate is highest for kernel size 20 x 20 but at the same time testing accuracy is low giving an indication of more overfitting. Whereas, minimum overfitting is achieved for kernel size 25 x 25 and at the same time highest testing accuracy is observed, therefore this kernel size can be declared as optimum size. As it can be clearly observed that the optimal kernel size is far greater than kernel sizes (3x3 and 5x5) considered in majority of the work reported in the literature. But the values of testing accuracy and overfitting are far away from the acceptable range.

After the optimal kernel size has been identified the classification results are imported in the form of confusion matrix for investigation of class-wise accuracies. These confusion matrices are shown in Table 7(a) and 7(b).

From the Table 7(a) it can be observed that for validation data overall correctly classified numerals are 1083 out of 1100 which means the recognition accuracy is 98.45% and the time spent in classification is 78 seconds.

For test data overall correctly classified numerals are 1066 out of 1100 which means the recognition accuracy is 96.91% and misclassified numerals are 34 out of 1100 this means that error rate is 3.09% and the time spent in classification is 78 seconds.

Also the difference in recognition rate of validation data and test data is 1.54% which can be attributed to the fact that overfitting could not be removed completely while taking optimum kernel size. The reason for high recognition rate in case of validation data is that a part of validation data was fed to algorithm during the training phase hence the network learned the data specific representation instead of a generalized representation.

It can also be observed from the confusion matrix in Table 7(b) that the highest rate of misclassification is among the numerals 1 and 2 of Gurmukhi script and both are being misclassified as numeral 7. Therefore in order to improve classification accuracy among misclassified classes proposed algorithm will be executed. As per proposed methodology, XGBoost is trained along with the features obtained by DCNN. Due to this additional training step, the training time of proposed method is more than that of DCNN.

Then the same procedure for classification is repeated with a minor change i.e. whenever the softmax predicts that the numerals is either 1 or 2 or 7 the features obtained by DCNN are fed into the XGBoost classifier to crosscheck this prediction. Now XGBoost has to classify these highly misclassified numerals from the 330 validation and test images. The recognition rates for numerals 1, 2 and 7 before and after applying the proposed algorithm are summarised in Table 8 for comparison purpose.

This table clearly indicates an increase of 6.66% in the recognition rate for 1, 2 and 7 test numerals thereby improving the overall recognition rate. The comparison of recognition rates for all numerals is given in Table 9.

From Table 9, it can be clearly observed that the testing accuracy has improved for proposed algorithm with marginal increase in testing time. Also the increase in accuracy of DCNN for increased dataset is quite large when compared to that of shallow networks. Therefore the proposed algorithm is able to outperform pre-existing DCNN with softmax classifier and other conventional techniques like shallow networks.

**Table 6.** Variation in validation/testing accuracy, overfitting and Training/Testing time

| Kernel Size | Training Accuracy | Validation Accuracy | Testing Accuracy | Extent of Overfitting | Training Time(seconds) | Validation/Testing Time(seconds) |
|---|---|---|---|---|---|---|
| 5 x 5 | 100 % | 94.55% | 88.73% | 5.82% | 2374 | 13 |
| 10 x 10 | 100 % | 94.91% | 91.27% | **3.64%** | 4146 | 23 |
| 15 x 15 | 100 % | 98% | 91.73% | 6.27% | 7231 | 38 |
| 20 x 20 | 100 % | **99.18%** | 92.82% | 6.36% | 10607 | 55 |
| **25 x 25** | **100 %** | **98.45%** | **96.91%** | **1.55%** | **20087** | **78** |
| 30 x 30 | 100 % | 97.73% | 91.27% | 6.46% | 32317 | 106 |

**Table 7(a).** Confusion matrix for validation data

| | | Predicted Numeral Class | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Actual Numeral Class** | **0** | 110 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | **1** | 0 | 110 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | **2** | 0 | 0 | 102 | 0 | 0 | 0 | 0 | 8 | 0 | 0 |
| | **3** | 0 | 0 | 7 | 103 | 0 | 0 | 0 | 0 | 0 | 0 |
| | **4** | 0 | 0 | 0 | 0 | 110 | 0 | 0 | 0 | 0 | 0 |
| | **5** | 0 | 0 | 0 | 0 | 0 | 110 | 0 | 0 | 0 | 0 |
| | **6** | 0 | 0 | 0 | 0 | 2 | 0 | 108 | 0 | 0 | 0 |
| | **7** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 110 | 0 | 0 |
| | **8** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 110 | 0 |
| | **9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 110 |

**Table 7(b).** Confusion matrix for test data

| | | Predicted Numeral Class | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Actual Numeral Class** | **0** | 110 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | **1** | 0 | 97 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 |
| | **2** | 0 | 0 | 90 | 1 | 0 | 0 | 0 | 19 | 0 | 0 |
| | **3** | 0 | 0 | 0 | 110 | 0 | 0 | 0 | 0 | 0 | 0 |
| | **4** | 0 | 0 | 0 | 0 | 110 | 0 | 0 | 0 | 0 | 0 |
| | **5** | 0 | 0 | 0 | 0 | 0 | 110 | 0 | 0 | 0 | 0 |
| | **6** | 0 | 0 | 0 | 0 | 0 | 0 | 110 | 0 | 0 | 0 |
| | **7** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 110 | 0 | 0 |
| | **8** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 110 | 0 |
| | **9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 109 |

**Table 8.** Comparison of accuracy for numerals 1, 2 and 7

| **Classification Technique** | **DCNN** | **Proposed Algorithm** |
|---|---|---|
| **Performance Parameters** | | |
| Training Accuracy | 100% | 100% |
| Validation accuracy | 97.57% | 99.1% |
| Testing accuracy | **90%** | **96.66%** |

**Table 9.** Comparison of overall accuracy

| **Classification Technique** | **SVM with DWT features** | **XGBoost with DWT features** | **DCNN** | **Proposed Algorithm** |
|---|---|---|---|---|
| **Performance Parameters** | | | | |
| Training Accuracy(%) | 99.06 | 100 | 100 | 100 |
| Validation accuracy(%) | 97.3 | 97.5 | **98.45** | **100** |
| Testing accuracy(%) | **95.46** | **96.18** | **96.91** | **98.91** |
| Training Time(s) (%) | 7964 | 2984 | 20087 | 21332 |
| Testing/Validation Time per digit(ms)(%) | 7.0 | 1.2 | 7.0 | 7.1 |

## 5. CONCLUSION

The proposed methodology has shown significant improvement in the classification accuracy in comparison to that of the pre-existing algorithm and that too with negligible increase in training/testing time. It is important to note here that this improvement is obtained for a comparatively small dataset. The results mentioned in this work have clearly established that even for smaller dataset deep learning based techniques can perform better that shallow algorithms. All we need is an appropriate synthetic method for data replication for reliable expansion of small dataset. Therefore it can be concluded that proposed methodology has performed extremely well surpassing SVM, Random Forest and XGBoost with DCT and DWT features. This work also presents the comprehensive analysis for finding optimal kernel size so as to have the best trade-off between accuracy and testing time. One of the factors contributing to good accuracy can be high resolution images in the dataset along with large kernel size. A significant decrease in overfitting is also observed which has led to an increase in overall accuracy.

## ACKNOWLEDGMENT

This Publication is an outcome of the R&D work undertaken under the project Visvesvaraya Ph.D. scheme of the Ministry of Electronics & Information Technology, Government of India, being implemented by Digital India Corporation.

## REFERENCES

[1] Bengio, Y. (2009). Learning Deep Architectures for AI. Now Publishers Inc.

[2] O'Shea, K. (2015). Massively deep artificial neural networks for handwritten digit recognition. arXiv: 1-2. arXiv:1507.05053.

[3] Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. Neural Computation, 22(12): 3207-3220. http://dx.doi.org/10.1162/NECO_a_00052

[4] Li, Y., Li, H., Xu, Y., Wang, J., Zhang, Y. (2016). Very deep neural network for handwritten digit recognition. Springer International Publishing, IDEAL, LNCS: 174-182. https://doi.org/10.1007/978-3-319-46257-8_19

[5] Wason, R. (2018). Deep learning: Evolution and expansion. Cognitive Systems Research, 52: 701-708. https://doi.org/10.1016/j.cogsys.2018.08.023

[6] Mrutyunjaya, P., Vihar, V. (2016). Towards the effectiveness of deep convolutional neural network based fast random forest classifier. arXiv: 1-11. arXiv:1609.08864

[7] Pang, S., Yang, X. (2016). Deep convolutional extreme learning machine and its application in handwritten digit classification. Computational Intelligence and Neuroscience, 2016: 3049632. https://doi.org/10.1155/2016/3049632

[8] Chandra, B., Sharma, R.K. (2016). Fast learning in deep neural networks. Neurocomputing, 171: 1205-1215. https://doi.org/10.1016/j.neucom.2015.07.093

[9] Singh, P., Verma, A., Chaudhari, N. (2016). Deep convolutional neural network classifier for handwritten Devanagari character recognition. Information Systems Design and Intelligent Applications, AISC, 434: 551-561. https://doi.org/10.1007/978-81-322-2752-6

[10] Elleuch, M., Zouari, R., Kherallah, M. (2016). Feature extractor based deep method to enhance online Arabic handwritten recognition system. International Conference on Artificial Neural Networks, LNCS, Springer, 9887: 136-144. https://doi.org/10.1007/978-3-319-44781-0_17

[11] Acharya, S., Pant, A.K., Gyawali, P.K. (2015). Deep learning based large scale handwritten Devanagari character recognition. 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Kathmandu, pp. 1-6. http://dx.doi.org/10.1109/SKIMA.2015.7400041

[12] ElAdel, A., Ejbali, R., Zaied, M., Amar, C.B. (2015). Dyadic multi-resolution analysis-based deep learning for Arabic handwritten character classification. 27th International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, Vietri sul Mare, Italy, pp. 807-812, http://dx.doi.org/10.1109/ICTAI.2015.119

[13] Hinton, G.E., Osindero, S., The, Y. (2006). A fast learning algorithm for deep belief nets. Neural Computation Journal, 18(7): 1527-1554. https://doi.org/10.1162/neco.2006.18.7.1527

[14] Kumar, R., Singh, A. (2010). Detection and segmentation of lines and words in Punjabi handwritten text.2nd International Advance Computing Conference (IACC), IEEE, Patiala, India, pp. 353-356. https://dx.doi.org/10.1109/IADCC.2010.5422927

[15] Jindal, M.K., Sharma, R.K. (2011). Classification of characters and grading writers in offline handwritten Punjabi script. International Conference on Image Information Processing, ICIIP, Shimla, India, pp. 1-4, https://doi.org/10.1109/ICIIP.2011.6108859

[16] Singh, K., Renu, S., Rani, D.R. (2011). Handwritten Gurmukhi numeral recognition using different feature sets. International Journal of Computer Applications, 28(2): 20-24.

[17] Singh, G., Kumar, C.J., Rani, R., Dhir, D.R. (2013). Feature extraction of Gurmukhi script and numerals: A review of offline techniques. International Journal of Advanced Research in Computer Science and Software Engineering, 3(1): 257-263. https://dx.doi.org/10.23956/ijarcsse

[18] Deore, S.P., Pravin, A. (2017). Ensembling: Model of histogram of oriented gradient based handwritten Devanagari character recognition system. Traitement du Signal, 34(1-2): 7-20. https://doi.org/10.3166/TS.34.7-20

[19] Kaur, A., Malhotra, S. (2016). Punjabi handwritten character recognition using wavelet based features. International Journal of New Technologies in Science And Engineering, 2(4): 217-226.

[20] Kumar, M., Jindal, M.K., Sharma,R.K. (2017). Offline handwritten Gurmukhi character recognition: Analytical study of different transformations. Proceedings of the National Academy of Sciences, India Section A: Physical Sciences, 87: 137-143. https://doi.org/10.1007/s40010-016-0284-y

[21] Singh, G., Sachan, M. (2015). Offline Gurmukhi script recognition using knowledge based approach & Multi-Layered Perceptron neural network. International Conference in Signal Processing, Computing and Control (ISPCC), Waknaghat, pp. 266-271. https://dx.doi.org/10.1109/ISPCC.2015.7375038

[22] Kumar, M., Sharma, R.K., Jindal, M.K. (2014). A novel feature extraction technique for offline handwritten Gurmukhi character recognition. IETE Journal of Research, 59(6): 687-691. https://doi.org/10.4103/0377-2063.126961

[23] Ahmed, R.A., Kia, D., Mandar, G., Ali, R., Rui, Z., Kaizhu, H., Ahmad, H., Ahsan, A., Amir, H. (2020). Offline Arabic handwriting recognition using deep machine learning: A review of recent advances. Advances in Brain Inspired Cognitive Systems, 11691: 457-468. https://doi.org/10.1007/978-3-030-39431-8_44

[24] Neelapu, R., Devi, G.L., Rao, K.S. (2018). Deep learning based conventional neural network architecture for medical image classification. Traitement du Signal, 35(2): 169-182. https://doi.org/10.3166/TS.35.169-182

[25] Benkaddour, M.K., Bounoua, A. (2017). Feature extraction and classification using deep convolutional neural networks. PCA and SVC for face recognition, Traitement du Signal, 34(1-2): pp. 77-91. https://doi.org/10.3166/TS.34.77-91

[26] Demircan, S., Örnek, H.K. (2020). Comparison of the effects of Mel coefficients and spectrogram images via deep learning in emotion classification. Traitement du Signal, 37(1): 51-57. https://doi.org/10.18280/ts.370107

[27] Breiman, L. (2001). Random forests. Machine Learning, 45: 5-32. https://doi.org/10.1023/A:1010933404324

[28] Zhao, H., Liu, H.M. (2019). Multiple classifiers fusion and CNN feature extraction for handwritten digits recognition. Granular Computing, 5: 411-418.

https://doi.org/10.1007/s41066-019-00158-6

[29] Alghazo, J.M., Latif, G., Alzubaidi, L., Elhassan, A. (2019). Multi-language handwritten digits recognition based on novel structural features. Journal of Imaging Science and Technology, 63: 20502-1-20502-10. https://doi.org/10.2352/J.ImagingSci.Technol.2019.63.2.020502

[30] Devi, D., Ramya, R., Dinesh, P.S., Palanisamy, C., Kumar, G.S. (2020). Design and simulation of handwritten recognition system. Materials Today: Proceedings. https://doi.org/10.1016/j.matpr.2020.02.720

[31] Chhajro, M.A., Khan, H., Khan, F., Kumar, K., Wagan, A.A., Solangi, S. (2020). Handwritten Urdu character recognition via images using different machine learning and deep learning techniques. Indian Journal of Science and Technology, 13(17): 1746-1754. https://doi.org/10.17485/IJST/v13i17.113

[32] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D. (1989). Backpropagation applied to handwritten zip code recognition. Neural Computation, 1(4): 541-551. https://doi.org/10.1162/neco.1989.1.4.541

[33] Friedman, J.H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5): 1189-1232.

[34] Chen, T., Carlos, G. (2016). Xgboost: A scalable tree boostingsystem. arXiv, pp. 1-13. arXiv:1603.02754