

Deep Learning Based Models for Offline *Gurmukhi* Handwritten Character and Numeral Recognition

Manoj Kumar Mahto^{*}, Karamjit Bhatia⁺ and R.K. Sharma[#]

^{*}Research Scholar, Gurukula Kangri Vishwavidyalaya, Haridwar, Uttarakhand, India

⁺Gurukula Kangri Vishwavidyalaya, Haridwar, Uttarakhand, India

[#]Thapar Institute of Engineering & Technology, Patiala, Punjab, India

Received August 9th, 2020; accepted 26th of December 2021

Abstract

Over the last few years, several researchers have worked on handwritten character recognition and have proposed various techniques to improve the performance of Indic and non-Indic scripts recognition. Here, a Deep Convolutional Neural Network has been proposed that learns deep features for offline *Gurmukhi* handwritten character and numeral recognition (HCNR). The proposed network works efficiently for training as well as testing and exhibits a good recognition performance. Two primary datasets comprising of offline handwritten *Gurmukhi* characters and *Gurmukhi* numerals have been employed in the present work. The testing accuracies achieved using the proposed network is 98.5% for characters and 98.6% for numerals.

Key Words: Deep convolutional neural network, Deep learning, *Gurmukhi* character recognition, *Gurmukhi* numeral recognition.

1 Introduction

Convolutional Neural Networks (CNNs) have extensively been used for visual image analysis [1]. Face recognition [2], image segmentation [3], object detection [4]–[6], pattern recognition, natural language processing, robotics, spam detection, speech recognition, topic categorization, video analysis, regression analysis and image classification [7] are some of the examples where CNNs have successfully been applied. Images are used directly by CNN as input. The CNN learns several features from these images which are used in classification task. An input is convoluted with some learned kernels repeatedly to obtain desired set of features. These features are invariant to various transformations like translation and distortion. The accuracies of Deep Convolutional Neural Networks in above listed fields, including handwritten characters and numerals recognition, have reached close to perfection.

Present day Handwritten Character and Numeral Recognition (HCNR) systems, broadly categorized as online and offline, are intelligent enough but not sufficient to recognize characters and numbers when compared with human identification ability. An HCNR system is an essential part of various applications such as data entry [8], office automation [9], postal address reading [10], printed postal code [11], cheque verification systems [12] *etc.* Writer's handwriting is sharply varied due to several influencing factors like mood, environment, time, work pressure *etc.* which enhances the error rates of the HCNR system. Therefore, to deal with such complexities and challenges, CNN is expected to be an appropriate choice for image classification. Feature extraction is a relevant key task for a fruitful recognition. In CNN feature extraction is automatically done from a trainable dataset [13]. Specially, for Roman character and digit recognition substantial achievements have been made due to good availability of datasets like CENPARMI [14], CEDAR

Correspondence to: manojkr.bit@gmail.com

Recommended for acceptance by Ana Belén Moreno Diaz

<https://doi.org/10.5565/rev/elcvia.1282>

ELCVIA ISSN: 1577-5097

Published by Computer Vision Center / Universitat Autònoma de Barcelona, Barcelona, Spain

[15], and MNIST [16]. However, for offline *Gurmukhi* HCNR, significant work on CNN is reported in literature.

The present work aims to address the problem of offline *Gurmukhi* HCNR from the perspective of deep learning and adjudge the efficacy of CNN model for the same. The outcome of the study is a model relied on the Deep Convolutional Neural Network (DCNN) for offline *Gurmukhi* handwritten characters and numerals classification.

The remaining part of this contribution is organized as follows. Section 2 provides a literature review. Section 3 deals with the overview of Convolutional Neural Network, CNN architecture and parameter setup. Section 4 presents the dataset description, implementation strategy and results obtained. A conclusion and future scope of research can be found in section 5.

2 Literature Review

Over the past few decades wide research has been conducted on HCNR systems. However, more recently, deep learning architecture like CNN has been used extensively in character recognition problematic domain.

Aggarwal *et al.* [17] proposed two-way feature extraction using gradient information for offline *Gurmukhi* handwritten character and numeric recognition. They extracted features by accumulating gradient information from an image by dividing into sub-blocks and then concatenating obtained gradient features from each block to form a feature vector with dimensionality 200. They achieved an accuracy of 97.4% for character and accuracy of 99.6% for numerals using SVM classifier with RBF kernel on 5-fold cross validation architecture of 7000-character sample and 2000 numerals sample of binary images dataset respectively. Kumar *et al.* [18] proposed the feature extraction technique based on boundary extents of a character image. They used PCA based feature selection technique to reduce the length of the feature vector and employed three different classifiers namely, k -NN (where $k = 5$), MLP and SVM with four different kernels, namely, Linear, Polynomial, Sigmoid and RBF. They achieved the best recognition accuracy of 93.8% using SVM-RBF classifier on 5-fold cross validation technique of 7000 isolated character samples collected from 200 different writers of *Gurmukhi* script. Kumar and Gupta [19] proposed deep learning approach using three types of feature extraction as; Local Binary Pattern (LBP) features, directional features and regional features. They extracted total of 117 features (59-features from LBP, 54-features from directional feature and 4-features from regional feature) for each image for classification. They achieved an accuracy of 99.3% using deep neural network on 2700-character samples of offline handwritten characters including the broken characters dataset. Kaur and Rani [20] proposed projection-based feature extraction using Radon transform for offline *Gurmukhi* handwritten character recognition. The Radon transform was used to find out projection angle of curved character. They achieved an average accuracy of 95.2% using PCA on 525-character samples written by 15 writers of *Gurmukhi* script dataset. Kumar *et al.* [21] have proposed various transformations techniques, namely, discrete wavelet transformations (DWT2), discrete cosine transformations (DCT2), Fast Fourier Transformations (FFT) and Fan Beam Transformations (FBT), for offline *Gurmukhi* handwritten character recognition. They also considered DWT2 with three different types, namely, Haar wavelet, Daubechies (db) 1 wavelet and Daubechies (db) 2 wavelet transformations. They used SVM classifier with linear and polynomial kernels. They achieved an accuracy of 95.8% with the help of 5-fold cross validation technique using DCT2 coefficients as features and SVM-linear classifier on 10,500-character samples of isolated *Gurmukhi* handwritten character dataset. Kumar *et al.* [22] have proposed various feature extraction techniques like zoning, discrete cosine transformations, gradient features and different combinations of these features. They employed four classifiers, namely, k -NN, SVM, Decision Tree, Random Forest individually and combination of these four classifiers with the voting scheme. They explored adaptive boosting and bagging technique to improve the recognition results. They achieved maximum accuracy of 95.9% using adaptive boosting technique and a combination of four different classifier on 150 documents of *Gurmukhi* manuscripts from different books of Medieval handwritten *Gurmukhi* manuscript dataset.

Arabic handwritten digit recognition has been proposed using subtle CNN with SoftMax layer to minimize the error and recognition success of 95.7% obtained on self-created 46,000 sample digit's dataset [23]. The Deep Belief Network and Convolution Neural Networks (with and without dropout, with filters such as Gaussian and Gabor) have been reported to identify Bangla digit and accuracy of 98.7% is attained on CMATERdb 3:1:1 dataset using CNN with Gabor features and dropout [5]. A Deep CNN have been proposed for Bangla digit recognition using large and unbiased NumtaDB dataset and an accuracy of 92.7% is achieved [24]. Chen *et al.* implemented four networks, specifically, CNN, Deep Residual Network (DRN), Dense Convolutional Network (DCN) and Capsule Network (CapsNet) and evaluated their performance on the MNIST dataset. The CapsNet exhibited overall highest recognition accuracy of 99.7% [25]. Also, a CNN based new architecture have been proposed with three layers, specifically for, feature extraction, feature dimension transposition and output for handwritten digit string recognition and recognition accuracies of 92.2% and 94.0% achieved on ORAND-CAR-A and ORAND-CAR-B datasets [26]. Recently, CNN based feature extraction scheme has been proposed for handwritten digit recognition that gave accuracy of 98.8% on the MNIST database with a valid parameter setup [27].

Bhalerao *et al.* [28] designed a Devanagari character recognition model using Gaussian filter, the strength, angle and histogram of gradient (SOG, AOG, HOG) feature and combination of quadratic and SVM classifiers. They obtained the highest accuracy of 95.8% using 3-fold cross validation on self-created size of 29,440 dataset. Sethy *et al.* [29] extracted various statistical features from a 2-level DWT sub-bands and applied PCA for fine most significant features Odia character recognition. They tested their model over a dataset containing 150 sample per class using BPNN classifier and achieved 94.8%. He *et al.* [30] introduced a ClothFace technology, a UHF RFID-based handwritten recognition platform integrated into a cotton fabric to recognize handwritten numbers 0-9. The images of digits written on platform can be classified by deep learning and achieved an accuracy of 94.6%. Li *et al.* [31] proposed a new CNN-based matching network technique to learn similarity between template character and handwritten character for large-scale HCCR involving 3755 classes of Chinese character. They achieved an accuracy of 95.5% on CASIA-HWDB dataset. Raj and Abirami [32] proposed Junction Point Elimination technique using Prism and Quad tree feature extraction approach for Tamil handwritten character recognition. They achieved 92.5% accuracy on HP-India lab dataset. Cao *et al.* [33] proposed a novel zero-shot hierarchical decomposition embedding method using CNN based framework and achieved an accuracy of 97.1% on CASIA-HWDB 1.0 database. Narang *et al.* [34] proposed Scale-invariant feature transfer (SIFT) and Gabor feature extraction for Devanagari ancient handwritten character recognition using SVM classifier with polynomial kernel on 5484 samples of character collected from various ancient manuscripts and achieved an accuracy of 91.3% on 10-fold cross validation technique. Bora *et al.* [35] have proposed OCR by combining Convolutional Neural Network and Error Correcting Output Code (ECOC) classifier on NIST handwritten character image dataset and achieved an accuracy of 97.7%. Das *et al.* [36] proposed a new feature learning multi-objective Jaya convolutional network (MJCN) for handwritten optical character recognition on different datasets. They evaluated the proposed model on various benchmark dataset an achieved an accuracy of 98.8% on NITR Bangla, 97.5% on ISI Bangla, 97.7% on ISI Odia, 98.3% on MNIST, 97.7% on OHCSv1.0 datasets.

3 Proposed work

3.1 Deep Neural Network

The application of machine learning proved to be a milestone in developing solution strategies for a wide range of problems. With the use of deep learning the performance of the systems became even better which, in turn, increased its applicability to several other new problem domains.

Convolutional Neural Network, Stacked Auto-Encoder, Deep Belief Network, *etc.* are typically included in the Deep Neural Network (DNN). The layers used for feature extraction and classification are combined into DNN for the purpose of efficient learning. Generally, feature extraction techniques are not required in

deep learning processes which take normalized images directly as input. Thus, image normalization is required before providing images as input to DNN. For automatic feature extraction, the initial and middle layers are used, the high-level layer is used for the purpose of classification based on the previously extracted features, and the last layer uses the SoftMax function [5]. All-important modules are structured within a network as a uniform integrated framework. Consequently, this framework provides improved precision as compared to training of each module independently.

3.2 Convolutional Neural Network

Fukushima introduced Convolutional Neural Network structure in 1980 [37]. However, due to the high complexity of training algorithms, it was not widely adopted. In 1990s, the application of gradient-based learning algorithms [1] provided encouraging results and further improvements in the arena of pattern recognition yielded even better results.

A CNN is a specific kind of ANN consisting of several hidden layers with one input and one output layer where many hidden layers comprise convolutional and pooling layers replicated before one or more fully joined layers [27]. A CNN works as a feature extractor, a type of neural network, and comprises of repetitive convolutional and pooling layers. In classification, weights are determined through training process [38]. CNN has some additional advantages over DNN: it targets to mimic human visual processing, and it do have extremely optimized structures for understanding the feature extraction. The max-pooling layer of the CNN is very influential in capturing shape variations. Also, CNN requires much smaller amounts of parameters than a similar sized fully connected network as it has sparse connections with tiled weights [5].

3.3 Convolution Layer

A convolutional layer is used to apply sliding filters through the image vertically and horizontally. During input image, scanning features are learned for its all-region through the convolutional layer. For each region, bias is added to the computed scalar product of filter values and image region values. In the convolutional layer, preceding layer's feature maps are convolved with kernels, which can be learned. The kernel output performs linear or non-linear activation function to yield an output feature map. The convolution operation as eqn.(1) [5] is given by

$$x_j^l = f(\sum_{i \in M_j} x_i^{l-1} k_{ij}^l + b_j^l) \quad (1)$$

where, x_j^l is current layer's output, x_i^{l-1} is output of previous layer, k_{ij}^l is present layer's kernel and b_j^l is current layer's bias, M_j is a selection of input maps. For individual output map, b is also specified. The input maps are convolved with different kernels to create the consistent output maps [5]. The simplified form of the activation function ReLU, a piecewise linear function, is $f(x) = \max(x, 0)$ [39]. It reduces negative portion of activation to zero and holds only the positive portion, the maximum operator helps in faster computation [40].

3.4 Pooling Layer

This layer acts to perform down sampling on the input map. The input and output feature maps in this layer are unchanged. The effect of down-sampling operation is to reduce each dimension of the output map and is represented as eqn.(2) [10].

$$x_j^l = \text{down}(x_j^{l-1}) \quad (2)$$

where, $\text{down}()$ indicates the pooling function. The average and max-pool operations are performed mainly in this layer.

3.5 Fully Connected Layer

Fully-connected layers, consisting of separate neurons for each pixel alike in standard neural network, follow the previously defined steps viz., convolutional and pooling. Here, in final layer, neurons equal to the number of predicted classes are used. The most advanced and effective regularization method “Dropout” is used to decrease overfitting at fully-connected layer [41]. Dropout is a technique for regularization by employing random removal of some neurons, which made the learned weight of other neurons insensitive to them.

3.6 CNN architecture

In the present problem domain, the overall CNN structure has two prime tasks: feature extraction and classification [42]. Each node of convolutional layer performs feature extraction from the input image through a convolution operation [10]. The architecture of CNN comprises of three layers, namely, convolutional, pooling and fully connected (FC) layers apart from I/O layers. The number of convolutional, pooling and fully connected layers in architecture of CNN is dependent on the image size and application. Determination of the stopping criteria of training the network do play a substantial role in overall performance of the system as too many epochs can lead to over-fitting of the training dataset, whereas too few may result in an under-fitting of model. Training on the training dataset is allowed to continue till the point from where the performance of system starts to degrade.

The proposed CNN architectures for offline *Gurmukhi* HCNR system are shown in Figure 2 and Figure 3, respectively. The suggested architecture of CNN comprises of an input layer, twice repeated convolutional and max- pooling layer sequence for feature extraction task, two fully connected layers for classification task and one output layer. The nodes of the network are connected to each input image in the upper layer through a small region. Each layer of the network takes the output of the previous layer as input and passes its output as input to the next layer.

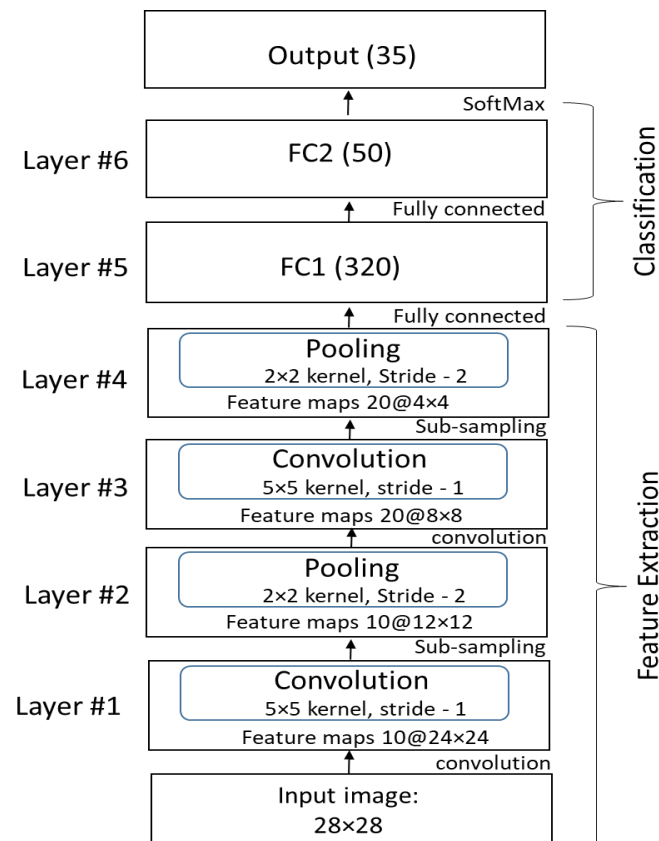


Figure 2: Architecture of CNN for classification of offline *Gurmukhi* handwritten characters.

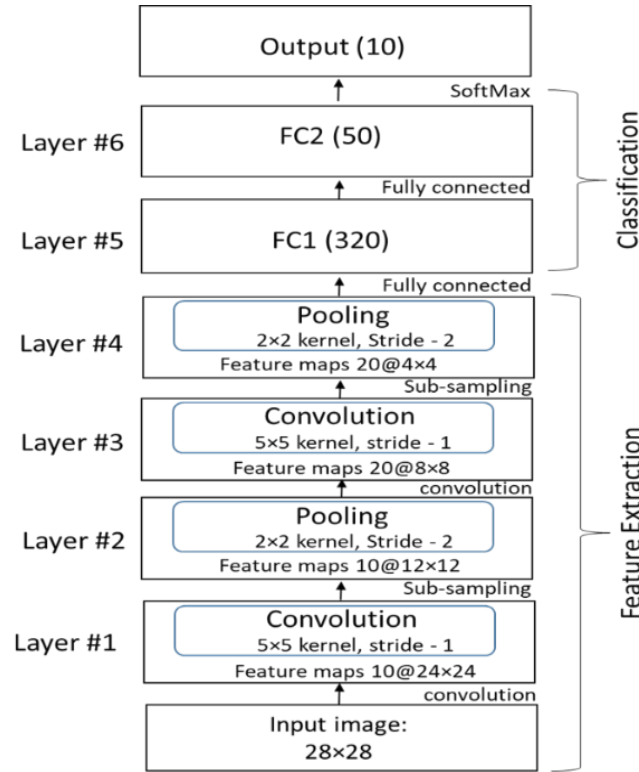


Figure 3: Architecture of CNN for classification of offline *Gurmukhi* handwritten numerals.

3.7 Parameter setup for CNN

Network parameters play a prominent role in assessing complexity of architecture. In this work, CNN uses twice repeated sequence of convolution and pooling layers for feature extraction task and two fully-connected layers for classification task. The first convolutional layer and second convolutional layer consist of 10 and 20 output mappings, respectively. The 28×28 input image is fed to first convolutional layer. The output feature map's dimension is computed through dimension of input feature maps (N), dimension of filter (F) and stride (S) in the convolution as $((N - F) / S) + 1$ [10]. Further, learnable parameters in I^{th} convolution layer is computed based on size of filter (F), feature map of previous layer (FM_{i-1}), feature map of current layer (FM_i) as $P_i = ((F \times F) + 1) \times FM_{i-1} \times FM_i$ [43]. Here, 1 is the bias for each feature map.

For a (28×28) dimensional input image, filter size 5×5 and stride 1 for convolution layer, the out-dimension for convolutional layer comes out to be 24. In pooling layer no parameters are trainable. The output of convolutional layer is 24×24 with 10 feature maps and the size of filter mask is 5×5 .

Thus, with 1 bias, parameters used to learn in the first convolutional layer is $(5 \times 5 + 1) \times 10 = 260$ and connections are $24 \times 24 \times (5 \times 5 + 1) \times 10 = 1,49,760$. To maintain non-linearity, ReLU activation function is employed in convolutional layer. The parameters and output size of pooling layers is 0 and 12×12 with 10 feature maps, respectively.

Likewise, learning parameters in the second convolutional layer are $((5 \times 5 + 1) \times 10) \times 20 = 5,200$ for convolution and 0 for max-pooling. Further, the pooling layers' parameters are also estimated by using ReLU activation function for non-linearity at the remaining convolutional layer. In fully connected layer, feature maps chosen is $(4 \times 4 \times 20) = 320$, verified experimentally, from the output with 20 maps provided by max-pooling layer which have been used previously and the size of output is 5×5 for each input. In first and second fully-connected layers, parameters are $320 \times 20 \times (5 \times 5 + 1) = 1,66,400$, and $50 \times (320 + 1) = 16,050$ and the final layers are $10 \times (50 + 1) = 510$. For a total of 10 numerals, each output node signifies a specific numeral

and the node's desired value is defined as 1 (and rest 9 node's value as 0) for the any input pattern. CNN training is performed to reduce error (E) [44] as given in eqn. (3).

$$E = \frac{1}{2} \frac{1}{pO} \sum_{p=1}^p \sum_{o=1}^O (d_o(p) - y_o(p))^2 \quad (3)$$

where, p is the number of patterns (1500 for numeral recognition); O is number of output nodes (*i.e.*, 10); d_o is expected output and y_o is actual output of a node for a specific pattern p .

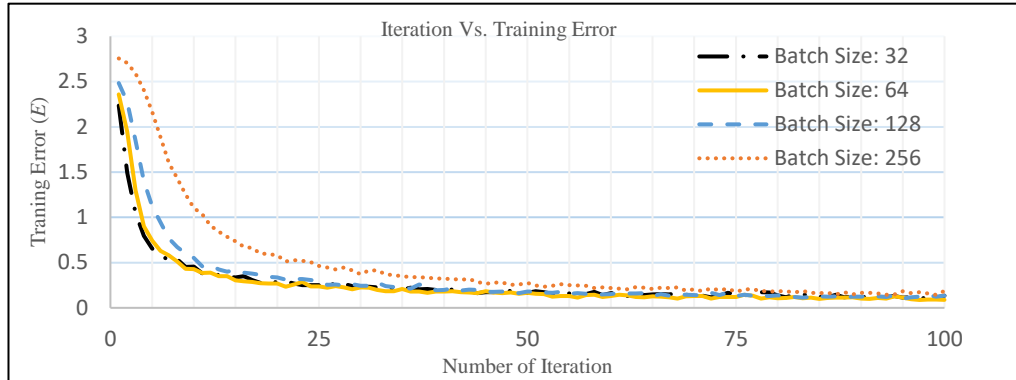


Figure 4: Error (E) for varying iterations on training of dataset

In the training process, kernel's value and weight of hidden layer's output are updated by deviation in each convolutional layer. Figure 4 shown the CNN performance in terms of error (E) for varying iterations on training of dataset.

The parameters of concerning layers for *Gurmukhi* handwritten numerals recognition are given in Table 1.

Table 1: CNN parameter setup for offline *Gurmukhi* handwritten numerals recognition

Layer	Operation in Layer	Feature map (in number)	Feature map size	Size of Window	Stride	No. of parameters
C ₁	Convolution	10	24×24	5×5	1	260
S ₁	Max-pooling	10	12×12	2×2	2	0
C ₂	Convolution	20	8×8	5×5	1	5,200
S ₂	Max-pooling	20	4×4	2×2	2	0
F ₁	Fully connected	320	1×1	N/A	-	1,66,400
F ₂	Fully connected	50	1×1	N/A	-	16,050
Output	SoftMax	10	1×1	N/A	-	510

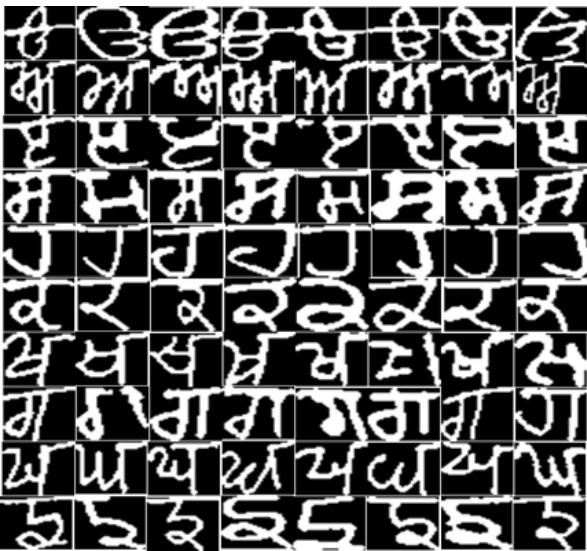
Similarly, the number of parameters required to be optimized for training the *Gurmukhi* handwritten character recognition system can be obtained. It is worth to note here that the architecture of character recognition model differs from the numeral recognition model in the output layer only. This layer will contribute some extra parameters as there are 35 output classes instead of 10 output classes. The total number of parameters for character recognition model that needs to be optimized is 1,89,695.

4 Experimental results and discussion

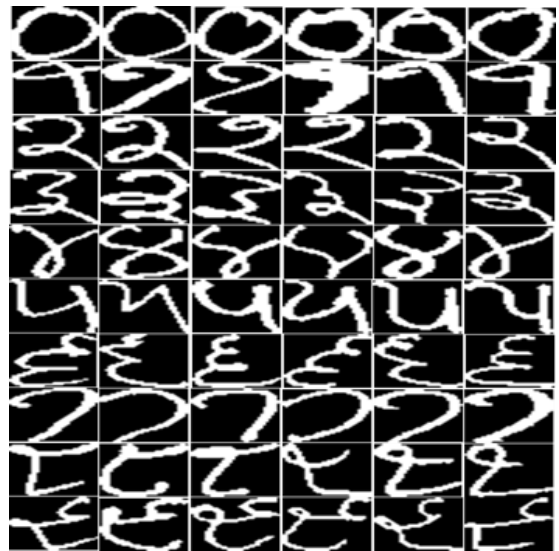
In this paper, handwritten images of *Gurmukhi* characters and numerals have been used for the extraction of features using deep learning. The *Gurmukhi* character and numeral recognition from the handwritten images is attempted through CNN.

4.1 Dataset description

For this work, we used a self-created dataset of handwritten *Gurmukhi* character and numeral images. The character samples have been collected from 100 persons from different age groups, different associations and with changing environment conditions. Each participant wrote two sets of patterns of *Gurmukhi* script including characters and numerals on plain white A4 size paper. As each writer contributed two sets of 35 characters each, a data set of 7000 characters of the *Gurmukhi* script is formed. This dataset is named as Database1 (Db1) in this work. After data collection phase, samples were scanned with 300 dpi resolutions to extract image samples, then categorized and bounded by bounding boxes using MATLAB. Similarly, for *Gurmukhi* numeral dataset, two sets of 10 *Gurmukhi* numerals have been collected from the same. As such, the count of samples in individual class is 200 and the aggregate number of samples in the numeral dataset is 2000. This dataset has been named as Database2 (Db2). The input image in each class has been normalized to the size of 28×28 pixels. In Figure 1, few sample imageries from these datasets are shown. There is no visible noise in this dataset.



(a) *Gurmukhi* character samples from Db1 dataset



(b) *Gurmukhi* numeral samples from Db2 dataset

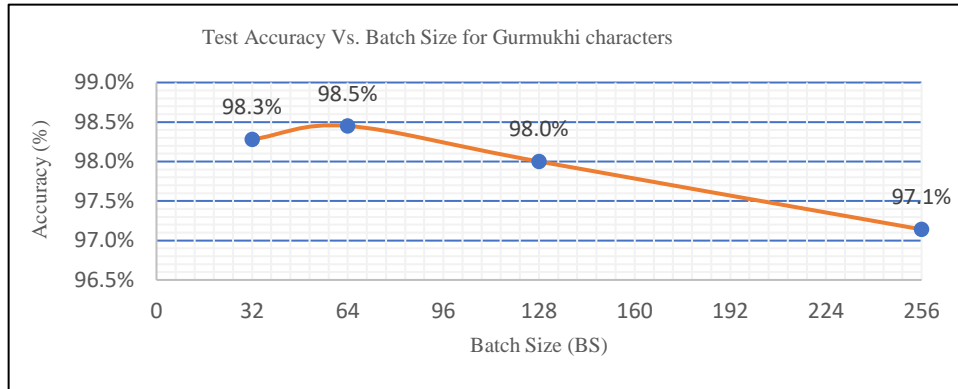
Figure 1: Samples of handwritten *Gurmukhi* characters and numerals

4.2 Performance evaluation

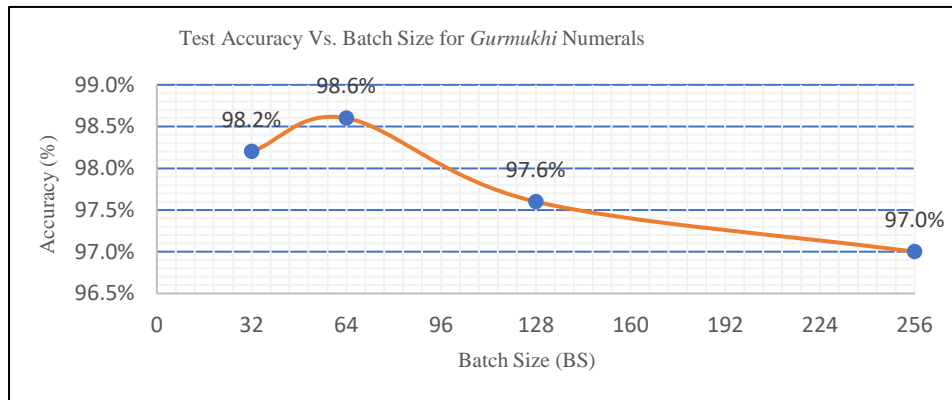
The proposed offline *Gurmukhi* HCNR system's results are based on the datasets Db1 and Db2. Out of 200 samples for a given character, 150 samples are arbitrarily selected for training and left over 50 for testing. Thus, for character recognition, training and testing sets contained 5250 and 1750 samples, respectively. Similarly, for numeral recognition, 1500 samples per character are taken in training set and remaining 500 samples are reserved for testing. Training samples are distributed evenly across the underlying 35 classes (for character recognition) and 10 classes (for numeral recognition). The performance results are based on test set accuracy.

Due to the large size training set, the batch-wise training has been conducted in experiments. Each input image has been resized to 784 (28×28) pixels in the training. Consequently, complete training set might have resulted into memory overflow, thereby inspiring to adopt batch-wise training of the network. In the experiments, four batch sizes (BS) have been considered. For a batch of images, the weights of CNN have been updated once [44].

It is interesting to observe that the test accuracy for one specific BS value (*i.e.*, 64) outperforms all other BS values. An accuracy of 98.5% have been achieved with 300 iterations for this BS value for offline *Gurmukhi* handwritten character recognition, as shown in Figure 5(a) and also in Table 2(a). For offline *Gurmukhi* handwritten numerals, the test accuracy of 98.6% for BS value 64 is best among all the test accuracies. This accuracy is achieved for epoch value 100, as shown in Figure 5(b) and also in Table 2(b)



(a) *Gurmukhi* Characters



(b) *Gurmukhi* Numerals

Figure 5: Test accuracies vs. batch size for offline *Gurmukhi* HCNR, (a) *Gurmukhi* Characters, (b) *Gurmukhi* Numerals

Table 2: Test Accuracies of different batch sizes

(a) *Gurmukhi* Characters

Epoch	Batch Size	Accuracy
300	32	98.3%
300	64	98.5%
300	128	98.0%
300	256	97.1%

(b) *Gurmukhi* Numerals

Epoch	Batch Size	Accuracy
100	32	98.2%
100	64	98.6%
100	128	97.6%
100	256	97.0%

The confusion matrix for offline *Gurmukhi* handwritten numeral recognition as obtained during the testing phase for proposed model is shown in Table 3. Among all the *Gurmukhi* numerals, six numerals are misclassified. Numeral “ੴ” is sometimes recognized as “ੳ”; numeral “ੳ” as “ੲ” or “ੴ”; numeral “ੴ”







as “ 4 ”; numeral “ 3 ” as “ 2 ”; numeral “ 7 ” as “ 1 ” and numeral “ 8 ” is recognized as “ 3 ”. In particular, variations in the writing styles increase the non-diagonal entries in this confusion matrix. However, the proposed system shows a very high performance for “ 0 ”, “ 2 ”, “ 4 ” and “ 8 ”, wherein all 50 test samples of each of these classes are correctly classified.

Table 4 includes some examples of hand written numeral images that have been misclassified by the proposed model. Owing to significant variations in writing styles, it is sometimes difficult for systems to classify these digital images. This is a challenge to develop a model that handles such type of variations as well.

Table 3: Confusion matrix of test samples for offline *Gurmukhi* handwritten numerals

Numeral	0	1	2	3	4	5	6	7	8	9	Total
0	50	0	0	0	0	0	0	0	0	0	50
1	0	49	0	0	0	0	0	0	1	0	50
2	0	0	50	0	0	0	0	0	0	0	50
3	0	0	1	48	1	0	0	0	0	0	50
4	0	0	0	0	49	1	0	0	0	0	50
5	0	0	0	0	0	0	49	0	1	0	50
6	0	1	0	0	0	0	0	49	0	0	50
7	0	0	0	0	0	0	0	0	50	0	50
8	0	0	0	0	0	0	1	0	0	49	50
	50	50	51	48	50	51	50	49	52	49	98.6

Table 4: Some misclassified numerals by proposed model

Handwritten numeral	Actual numeral	Misclassified as
	1	8
	3	2, 4
	4	5
	6	8
	7	1
	8	6

The confusion matrix as obtained during the testing phase, is shown in Table 5. From this confusion matrix, it has been observed that the character recognition of offline *Gurmukhi* script performance is quite successful. Twenty-two classes out of the thirty-five classes achieved an accuracy of 100.0% on the dataset (Db1). Five classes have 98.0% accuracy, 4 classes have 96.0% accuracy, 3 classes have 94.0% accuracy and the remaining 1 class has the recognition accuracy of 90.0% in the proposed system. An overall 98.5% accuracy is achieved for the *Gurmukhi* character recognition system in 300 iterations, when the BS value is 64.

Table 5: Confusion matrix for offline *Gurmukhi* handwritten character recognition

Char	ੳ	ਅ	ੲ	ਸ	ਹ	ਕ	ਖ	ਗ	ਘ	ਙ	ਚ	ਛ	ਜ	ਝ	ਞ	ਟ	ਠ	ਡ	ਢ	ਤ	ਥ	ਦ	ਧ	ਨ	ਪ	ਫ	ਬ	ਭ	ਮ	ਯ	ਰ	ਲ	ਵ	ੜ	Total			
ੳ	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50		
ਅ	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50		
ੲ	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50		
ਸ	0	0	0	48	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	50		
ਹ	0	0	0	0	49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	50			
ਕ	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50		
ਖ	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਗ	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਘ	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਙ	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਚ	0	0	0	0	0	0	0	0	0	0	48	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਛ	0	0	0	0	0	2	0	0	0	0	0	48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਜ	0	0	0	0	0	0	0	0	0	0	0	1	49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਝ	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਞ	0	0	1	0	0	0	0	0	0	0	0	0	0	0	49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਟ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਠ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਡ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	45	0	2	0	0	0	0	0	0	0	0	0	2	0	0	1	0	50		
ਢ	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	47	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਤ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਥ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਦ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਧ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	49	0	0	0	0	0	0	1	0	0	0	0	0	0	50	
ਨ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਪ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਫ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	50
ਬ	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਭ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਮ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਯ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਰ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਲ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ਵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
ੜ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	
	50	50	52	48	49	52	50	51	50	50	49	48	53	50	49	50	45	47	50	53	51	52	49	50	50	50	49	48	50	51	53	50	50	51	98.5			

Table 6 contains some of the handwritten characters, which have been misclassified by the proposed model. It has been observed that the offline *Gurmukhi* HCNR shows quite good performance in both the cases: character recognition and numeral recognition.

Table 6: Some misclassified characters by proposed model

Handwritten character	Actual character	Misclassified as	Handwritten character	Actual character	Misclassified as
ਸ	ਸ	ਗ, ਥ	ਙ	ਡ	ਤ, ਰ, ਝ
ਹ	ਹ	ਰ	ਦ	ਢ	ਜ, ਦ
ਚ	ਚ	ਜ	ਧ	ਯ	ਯ
ਛ	ਛ	ਕ	ਨ	ਨ	ਠ
ਜ	ਜ	ਚ	ਥ	ਥ	ੲ
ਝ	ਝ	ੲ	ਤ	ਭ	ਜ, ਤ
ਞ	ਞ	ਨ			

4.3 Comparison with the *state-of-arts*

The proposed model is compared with the *state-of-arts* approaches, proposed by researchers over a decade, of *Gurmukhi* numeral and character recognition and comparison results are presented in Table 7 and Table 8, respectively.

Table 7: Comparison with other existing *state-of-art* approaches for *Gurmukhi* numerals

Researcher(s)	Approach/ Features	Classifier	No. of samples	Accuracy
Aggarwal <i>et al.</i> [17]	Two-way gradient information	SVM	2000	99.6%
Sharma <i>et al.</i> [45]	Directional Distance Distribution	k-NN	120	92.6%
	Zoning			90.6%
	Structural			84.0%
Singh and Dhir [46]	Gabor filter	SVM	1500	98.4%
Siddharth <i>et al.</i> [47]	Distance profile	SVM	1500	98.1%
Proposed Model	CNN		2000	98.6%

Table 8: Comparison with other existing *state-of-art* approaches for *Gurmukhi* characters

Researcher(s)	Approach/ Features	Classifier	No. of samples	Accuracy
Kumar and Gupta [19]	LBP, directional and regional features	Deep Learning	2,700	99.3%.
Kaur and Rani [20]	Radon transform, PCA		525	95.2%
Kumar <i>et al.</i> [21]	Discrete wavelet transformations (DWT2), discrete cosine transformations (DCT2), fast Fourier transformations and fan beam transformation	SVM	10,500	95.8 %
Kumar <i>et al.</i> [22]	Zoning, discrete cosine transformations and gradient features	k-NN, SVM, Decision Tree, Random Forest	1,150	95.9%
Proposed Model	CNN		7,000	98.5%

5 Conclusion

Character and numeral recognition are research field which has attracted researchers around the globe to contribute and come up with new ideas and techniques to propose systems that exhibit accuracies close to 100% for all type of documents such as printed/handwritten or written in continuous/isolated manner. A deep learning model for offline *Gurmukhi* HCNR system is proposed. The architecture and parameters required to be trained in this network is explained in this paper. The proposed systems reported excellent accuracies in both the cases: character recognition and numeral recognition. Experimental results lead to the conclusion that CNN performs well in classifying *Gurmukhi* characters and numerals on the datasets Db1 and Db2. In this study, a recognition accuracy of 98.5% for character recognition on Db1 dataset and 98.6% for numerical recognition on Db2 dataset has been achieved. The proposed method deals with isolated character and numeral recognition which is a limitation of the method as obtaining isolated characters or cleanly separated characters may not be feasible always in real world. The scope of the proposed work can be extended to the test of effectiveness of technique on dataset of *Gurmukhi* characters and numerals written in continuous manner. The proposed technique can also be applied on handwritten other Indic-scripts' recognition.

References

- [1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [2] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," *Proceedings Br. Mach. Vis. Conf. 2015*, no. Section 3, pp. 41.1-41.12, 2015.
- [3] C. Couprie, C. Farabet, L. Najman, and Y. LeCun, "Indoor Semantic Segmentation using depth information," pp. 1–8, *arXiv preprint arXiv:1301.3572* (2013).
- [4] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.

- [5] M. Z. Alom, P. Sidike, T. M. Taha, and V. K. Asari, "Handwritten Bangla Digit Recognition Using Deep Learning," *ArXiv1705.02680v1 [Cs]*, 2017.
- [6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, pp. 580–587, 2014.
- [7] O. Russakovsky et al., "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
- [8] L. V. Rasmussen, P. L. Peissig, C. A. McCarty, and J. Starren, "Development of an optical character recognition pipeline for handwritten form fields from an electronic health record," *J. Am. Med. Informatics Assoc.*, vol. 19, no. E1, 2012.
- [9] D. Deodhare, N. N. R. Ranga, and S. R. Amit, "Preprocessing and Image Enhancement Algorithms for a Form-based Intelligent Character Recog. Sys.," *Int. J. Comp. Sci. Appl.*, vol. 2, no. 2, pp. 131–144, 2005.
- [10] M. Z. Alom, P. Sidike, M. Hasan, T. M. Taha, and V. K. Asari, "Handwritten Bangla Character Recognition using Inception Convolutional Neural Network," *Computational Intell. Neurosci.*, vol. 2018, pp. 1–13, 2018.
- [11] M. K. Mahto, K. Bhatia, and R. K. Sharma, "Combined Horizontal and Vertical Projection Feature Extraction Technique for Gurmukhi Handwritten Character Recognition," in *2015 International Conference on Advances in Computer Engineering and Applications (ICACEA)*, 2015, pp. 59–65.
- [12] R. Ghosh, C. Panda, and P. Kumar, "Handwritten Text Recognition in Bank Cheques," *2018 Conf. Inf. Commun. Technol. CICT 2018*, pp. 1–6, 2018.
- [13] D. Sen Maitra, U. Bhattacharya, and S. K. Parui, "CNN based common approach to handwritten character recognition of multiple scripts," *Proc. Int. Conf. Doc. Anal. Recog. ICDAR*, vol. 2015, pp. 1021–1025, 2015.
- [14] P. J. Haghghi, N. Nobile, C. L. He, and C. Y. Suen, "A new large-scale multi-purpose handwritten farsi database," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5627 LNCS, pp. 278–286, 2009.
- [15] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: Benchmarking of state-of-the-art techniques," *Pattern Recognit.*, vol. 36, no. 10, pp. 2271–2285, 2003.
- [16] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges." [Online]. Available: <http://yann.lecun.com/exdb/mnist/>. [Accessed: 23-Jul-2020].
- [17] A. Aggarwal, K. Singh, and K. Singh, "Use of gradient technique for extracting features from handwritten Gurmukhi characters and numerals," *Procedia Comput. Sci.*, vol. 46, no. Ict 2014, pp. 1716–1723, 2015.
- [18] M. Kumar, M. K. Jindal, R. K. Sharma, and S. R. Jindal, "Offline handwritten pre-segmented character recognition of gurmukhi script," *Mach. Graph. Vis.*, vol. 25, no. 1–4, pp. 45–55, 2016.
- [19] N. Kumar and S. Gupta, "A Novel Handwritten Gurmukhi Character Recognition System Based on Deep Neural Networks," *Int. J. Pure Appl. Math.*, vol. 117, no. 21, pp. 663–678, 2017.
- [20] S. Kaur and S. Rani, "Isolated Curved Gurmukhi Character Recognition Using Projection of Gradient," *International Journal of Computational Intelligence Research*, vol. 13, no. 6, pp. 1387–1396, 2017.
- [21] M. Kumar, M. K. Jindal, and R. K. Sharma, "Offline Handwritten Gurmukhi Character Recognition: Analytical Study of Different Transformations," *Proc. Natl. Acad. Sci. India Sect. A - Phys. Sci.*, vol. 87, no. 1, pp. 137–143, 2017.
- [22] M. Kumar, S. R. Jindal, M. K. Jindal, and G. S. Lehal, "Improved Recognition Results of Medieval Handwritten Gurmukhi Manuscripts Using Boosting and Bagging Methodologies," *Neural Process. Lett.*, vol. 50, no. 1, pp. 43–56, 2019.
- [23] A. H. Alwzawy, M. H. Albehadili, S. Y. Alwan, and E. N. Islam, "Handwritten Digit Recognition Using Convolutional Neural Networks," *Int. J. Innov. Res. Comput. Comm. Eng.* vol. 4, no.2, pp. 1101–1106, 2016.
- [24] A. Shawon, M. Jamil-Ur Rahman, F. Mahmud, and M. M. Arefin Zaman, "Bangla Handwritten Digit Recognition Using Deep CNN for Large and Unbiased Dataset," *2018 Int. Conf. Bangla Speech Lang. Process. ICBSLP 2018*, pp. 1–6, 2018.
- [25] F. Chen, N. Chen, H. Mao, and H. Hu, "Assessing four Neural Networks on Handwritten Digit Recognition Dataset (MNIST)," *arXiv preprint arXiv:1811.08278*. (2018).
- [26] H. Zhan, S. Lyu, and Y. Lu, "Handwritten Digit String Recognition using Convolutional Neural Network," *Proc. - Int. Conf. Pattern Recognit.*, vol. 2018-Augus, pp. 3729–3734, 2018.
- [27] R. Jana and S. Bhattacharyya, "Character Recognition from Handwritten Image Using Convolutional Neural

- Networks,” in *Recent Trends in Signal and Image Processing*, 2019, pp. 23–30.
- [28] M. Bhalerao, S. Bonde, A. Nandedkar, and S. Pilawan, “Combined classifier approach for offline handwritten devanagari character recognition using multiple features,” *Lect. Notes Comput. Vis. Biomech.*, vol. 28, pp. 45–54, 2018.
- [29] A. Sethy, P. K. Patra, and D. R. Nayak, “Off-Line Handwritten Odia Character Recognition Using DWT and PCA,” *Adv. Intell. Syst. Comput.*, vol. 563, pp. 187–195, 2018.
- [30] H. He et al., “ClothFace: A Batteryless RFID-Based Textile Platform for Handwriting Recognition,” *Sensors (Basel)*, vol. 20, no. 17, 2020.
- [31] Z. Li, Q. Wu, Y. Xiao, M. Jin, and H. Lu, “Deep Matching Network for Handwritten Chinese Character Recognition,” *Pattern Recognit.*, vol. 107, 2020.
- [32] M. A. R. Raj and S. Abirami, “Structural representation-based off-line Tamil handwritten character recognition,” *Soft Comput.*, vol. 24, no. 2, pp. 1447–1472, 2020.
- [33] Z. Cao, J. Lu, S. Cui, and C. Zhang, “Zero-shot Handwritten Chinese Character Recognition with hierarchical decomposition embedding,” *Pattern Recognit.*, vol. 107, 2020.
- [34] S. R. Narang, M. K. Jindal, S. Ahuja, and M. Kumar, “On the recognition of Devanagari ancient handwritten characters using SIFT and Gabor features,” *Soft Comput.*, vol. 0123456789, 2020.
- [35] M. B. Bora, D. Daimary, K. Amitab, and D. Kandar, “Handwritten Character Recognition from Images using CNN-ECOC,” *Procedia Comput. Sci.*, vol. 167, no. 2019, pp. 2403–2409, 2020.
- [36] D. Das, D. R. Nayak, R. Dash, and B. Majhi, “MJCN: Multi-objective Jaya Convolutional Network for handwritten optical character recognition,” *Multimed. Tools Appl.*, pp. 1–20, 2020.
- [37] F. Kuniyoshi, “Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position,” *Nature*, vol. 36, pp. 193–202, 1980.
- [38] J. Gilewski, P. Phillips, S. Yanushkevich, and D. Popel, “Education Aspects : Handwriting Recognition-Neural Networks- Fuzzy Logic,” *Proceedings of the IAPR International Conference on Pattern Recognition and Information Processing*, vol. 1, pp. 1–8, 1997.
- [39] P. P. Nair, A. James, and C. Saravanan, “Malayalam handwritten character recognition using convolutional neural network,” *Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2017*, pp. 278–281, 2017.
- [40] W. Rawat and Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review,” *Neural Comput.*, vol. 29, pp. 2352–2449, 2017.
- [41] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [42] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning,” *MIT Press*, 2016. [Online]. Available: <https://www.deeplearningbook.org/>. [Accessed: 23-Jul-2020].
- [43] M. Z. Alom, P. Sidike, T. M. Taha, and V. K. Asari, “Handwritten Bangla Digit Recognition Using Deep Learning,” *arXiv preprint arXiv:1705.02680* (2017).
- [44] M. M. Rahman, M. A. H. Akhand, S. Islam, and P. C. Shill, “Bangla Handwritten Character Recognition using Convolutional Neural Network,” *Int. J. Image, Graph. Signal Process.*, vol. 7, no. 8, pp. 52–59, 2015.
- [45] D. Sharma, G. Lehal, and P. Kathuria, “Digit extraction and recognition from machine printed Gurmukhi documents,” *Proc. Int. Work. Multiling. OCR*, 2009.
- [46] S. Singh and R. Dhir, “Recognition of Handwritten Gurmukhi Numeral using Gabor Filters,” *Int. J. Comput. Appl.*, vol. 47, no. 1, pp. 7–11, 2012.
- [47] K. S. Siddharth, R. Dhir, and R. Rani, “Comparative Recognition of Handwritten Gurmukhi Numerals Using Different Feature Sets and Classifiers,” in *International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT2012)*, 2012, pp. 20–24.