Offline Handwritten Gurumukhi Character Recognition System Using Deep Learning

Udit Jindal[1], Sheifali Gupta[2], Vishal Jain[3], Marcin Paprzycki[4]

[1, 2]Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India
[3]Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi, India
[4]Systems Research Institute Polish Academy of Sciences, Warsaw, Poland

[1]uditjndl@gmail.com, [2]sheifali.gupta@chitkara.edu.in

Abstract. Currently, Gurumukhi – a religion-specific language originating from India, ranks as the 14th most spoken language and the 18th most popular writing script language of the entire world. However, while there exists a large body of literature related to recognition of handwritten texts in various languages, number of publications related to recognition of Indian handwritten scripts is considerably smaller. It concerns also the case of Gurumukhi. Hence, in the current contribution, we consider Gurumukhi handwritten character recognition, to fill the existing practical gap. The proposed approach is based on deep convolutional neural networks, and has been applied to the 35 core Gurumukhi characters. Obtained results are promising, as accuracy of 98.32% has been achieved for the training data set, and 74.66%,on the test data. These results are comparable to results reported in earlier research, but have been obtained without any feature extraction or post-processing.

Keywords: Character Recognition, Deep Learning, Convolutional Neural Network, Handwritten script, Gurumukhi.

1 INTRODUCTION

Handwritten text recognition has a prominent role, among others, in business, healthcare or cultural heritage preservation. Observe that, while computer systems slowly replace handwritten documents in day to day practices of today, there is exists very large body of documents that have been handwritten and should be digitized. This has to be done, among others, to assure preservation of content as pen-and-paper documents have limited lifespan (due to normal ageing and degradation). Here, digitization has to involve not only creation of images (which is required primarily for historical manuscripts, which are work of art in their own right, and preserving them as images is required), but also extraction of their actual content/text (needed for further processing/searching/storage). For the latter, text images are usually converted into a machine-encoded form, using Optical Character Recognition (OCR). Recently, a lot of work has been devoted to recognition of printed, as well as handwritten, characters. Obviously, characters handwritten by different persons differ in both size and shape; since every individual has different handwriting, and thus different form of writing of individual characters. This, in turn, is the main source of difficulty for application of computers to handwritten character recognition [1][2]. To address this problem, recently, it was proposed to use deep learning and convolution neural networks, which tend to be substantially more accurate than the approaches used in the past.

Abundant literature exists, concerning handwriting character recognition for non-Indian languages and scripts that represent them. However, only very few articles are available related to recognition of Indian scripts, such as the Gurumukhi. The Gurumukhi is utilized primarily within the Punjabi dialect. It has been established to be the world's fourteenth spoken language (as far as number of people who use it is concerned). It is also the 18th most popular writing script language of the world [3]. Note that speakers, communicating in the Punjabi language (Gurumukhi), are not limited only to the states of Northern India, like Haryana and Punjab, and to the Delhi area. This language has also spread around the world. Because of its popularity, there exists also a very large number of written texts in this language (content of which is represented using the Gurumukhi script). Here, the body of written language includes, among others, sacred texts, books, verses, etc. Therefore, it is important to create an OCR system for such a rich, and generally utilized dialect, which may become useful in different regions [4]. Moreover, note that there is still large number of persons who prefer to take records manually (in a written form). Here, note, obviously a lot of drawbacks exist in traditional approaches to handwritten documents management. These are, among others, storage space, preservation against decay, impossibility of making backups, etc. Moreover, the searching operation is extremely difficult (and very time-consuming) in the case of

handwritten documents, existing on paper. Therefore, to eliminate these issues, recognition of handwritten characters is clearly needed for the Gurumukhi script.

Our goal is to propose an approach to recognize Gurumukhi handwritten text. Based on analysis of available state of the art approaches, we have decided to apply Convolution Neural Networks (CNN), which seem to hold the highest promise (at the time of writing this contribution). Here, let us note that the earlier work, reported in [3-8], has already used machine learning based approaches. However, in this work, feature extraction has been done manually and explicitly. Hence, the accuracy of the proposed models depended on the type of features extracted (and on the quality of feature extraction itself). To avoid such issues, we have decided to apply a "global approach", where no feature extraction is applied at all, to see what level of accuracy can be achieved. Obviously, if successful, the proposed approach can be combined with feature extraction to further improve quality of results.

The remaining part of this contribution is organized as follows. Section 2 presents the main features of the Gurumukhi script. Next (in Section 3), we summarize that state of the art of handwritten Gurumukhi character recognition. Section 4 outlines the research methodology of the proposed CNN-based deep learning model. Initial results of our experiments, with the proposed architecture, are discussed in Section 5. A brief conclusion and future scope of possible research can be found in Section 6.

2 GURUMUKHI SCRIPT

Gurumukhi script is used for writing in the Punjabi language. The word 'Gurumukhi' means "from the mouth of the Guru". Gurumukhi script is 18th most popular writing script language of the world [3]. The Gurumukhi script has three vowel bearers, thirty-two consonants, six additional consonants, nine vowel modifiers, three auxiliary signs and three half characters [6], as shown in Figure 1. The Gurumukhi script is written from top to bottom and left to right, and it is not case sensitive. In the Gurumukhi script, the characters have an upper horizontal line called headline. In this script, some characters are quite similar to each other, which makes their recognition a relatively difficult task. For example, in the consonants set, two characters shown by red arrow are differentiated only with header lines. In the same way, characters shown by blue arrow are also very similar to each other (the difference being the "closing of the loop"). In this work, we have focused our attention on recognition of 35 selected characters, i.e. 32 consonants and 3 vowel bearers, because these are the core characters in Gurumukhi script.

Figure 1: Full Gurumukhi Character Set

3 STATE OF THE ART

Let us now briefly summarize main pertinent research results. Let us start from observation that various machine learning based techniques have already been applied to recognize characters written in different languages. Here, let us first note contributions, in which researchers have extracted different types of features for the classification purpose. For instance, D. Sharma and U. Jain [3] have proposed the Gurumukhi character recognition system based on application of machine learning. In their work, S-cell and C-cell has been used, in the Neocognitron technique, to extract local features. M. Kumar, M. K. Jindal and R. K. Sharma (in [6]) proposed a model based on the diagonal and transitional feature extraction, and used the K-nearest neighbors algorithm (KNN) classifier for character recognition. However, in this technique, it is hard to determine the nearest parameter value of *K*. M. Kumar, M. K. Jindal and R. K. Sharma (in [7]) have extracted various features, like diagonal, directional and zoning features, and further used the K-NN and the Bayesian classifiers, for distinguishing handwriting of different writers. N. Kumar and S. Gupta (in [8]) discussed the feature extraction techniques, like diagonal features, linear binary pattern (LBP) and transition features and used KNN and Support Vector Machines (SVM) based classifiers. Finally, G. Lehal and C. Singh (in [4]) proposed a Gurumukhi script recognition system based on multi-classifiers. In all these articles, different combinations of types of features that were to be extracted was considered (with various levels of success). However, it is clear that success of the selected feature selection approach may depend on the specific features of the dataset itself. Hence, it is worthy to consider an approach that is independent of the specific feature extraction method and see how successful can it become.

In the last few years, deep leaning demonstrated remarkable performance in the domain of machine learning, and pattern recognition, in particular. Among existing deep learning approaches, convolution neural networks (CNN) are one of the most popular models [9]. The CNN approach has an edge over other image processing techniques, as the images are processed using a highly optimized structure, while the basic idea behind its development was to imitate the human brain processing capabilities. The capacity of CNN to effectively display the informative information can be improved by varying characteristics of hidden layers, as well as parameters used for training in each layer [10].

In the literature survey, only very small body of work has been found on attempts at applying CNNs to the Gurumukhi character recognition [5]. Here, H. Kaur and S. Rani [11] proposed a CNN-based approach. However, it was combined with manual(!) feature extraction, preceding feeding the CNN, which is then used for the actual classification of handwritten characters. This being the case, we have decided to apply the CNN directly to the preprocessed handwritten images of Gurumukhi characters. However, it has to be stressed immediately that, as will be seen, the applied preprocessing does not involve any form of feature extraction.

4 PROPOSED APPROACH

The proposed approach consists of three main steps. First, data set to be used in the experiments is prepared. Second, it is preprocessed to be ready to be fed into the CNN. Finally, the specific CNN architecture is designed and applied to the prepared dataset. In Figure 2, we summarize the main aspects of each one of these three stages, and follow with a more detailed description of each one of them.

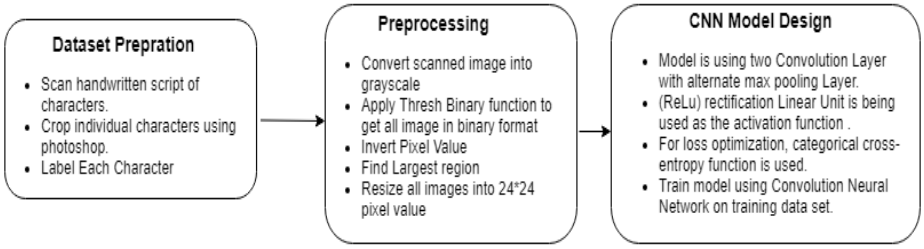

Figure 2: Summary of preparatory stages of the proposed approach.

4.1 Dataset Preparation

A dataset has been created for all 35 fundamental Gurumukhi isolated characters by creating 3,500 images (one image containing a single "version" of one character). For preparing such database of 3500 images, 50 people from various age groups and education levels, and both genders, wrote the 35 selected Gurumukhi characters on the A4 size paper. Obviously, in this way, the constructed dataset includes a wide variety of individual renditions of characters, in view of individual writing styles of persons who participated in character writing. Each "individual dataset" consists of 35 Gurumukhi characters, created by a single participant. However, each individual participant has written all 35 characters two times on separate paper sheets (resulting in 70 renditions of characters; two of each). A sample dataset is shown in Figure 3. Next, obtained sample datasets have been scanned. All individual characters have been cropped manually, from the scanned images, using Adobe Photoshop. In this way, 70 cropped images of characters, written by each individual participating in our study have been prepared. Example of cropped character images can be found in Figure 4. Each cropped image, of an individual character, was saved in a separate file, with its corresponding (distinct) label name. Overall 100 sample datasets were collected from 50 participants. Thus, the final dataset consisted of 70 x 50 = 3,500 images of 35 characters. Next, the dataset has been partitioned into two sets, the first set containing 80% of images, to be used as a training set, and the remaining 20% to be used as the test set. This is a standard approach used in machine learning experiments.

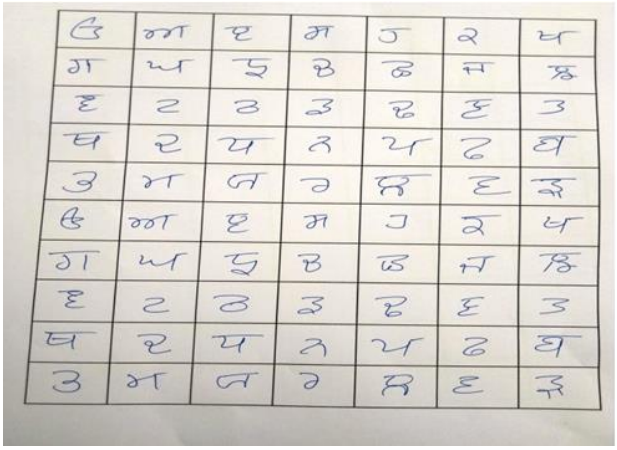

Figure 3: Sample Dataset from one participant

Figure 4: Cropped Character Images

4.2 Preprocessing

Preprocessing converts all obtained images into a common size, to feed them into the CNN. Here, the process is illustrated in Figure 5. The scanned and cropped image, of single character, is shown in Figure 5(a). Such character image is, first, converted into the grayscale image, as shown in Figure 5(b). In the grayscale image, every pixel value represents the brightness level of that pixel and it from the 0-255 range. Next, a binary function, with a threshold value of 0.7, is applied to the grayscale image, to convert it to a binary image, as shown in Figure 5(c). From that, the largest region is segmented out, to remove the noise, as shown in Figure 5(d). Before training the CNN, on the prepared images, they are resized to 24x24 pixels.

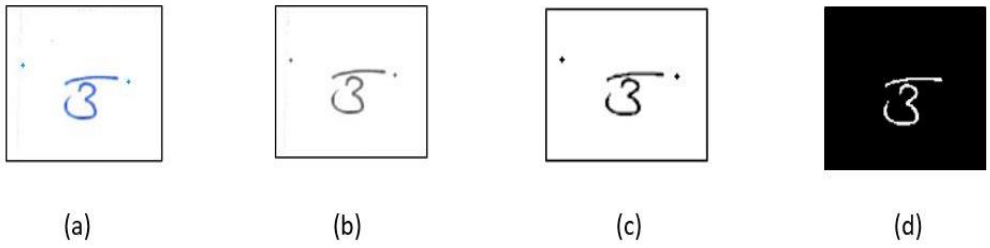

Figure 5: Phases of the image during preprocessing (a) Original Image, (b) Gray scale Image, (c) Binary Image, (d) Inverted Binary Image with largest region (without noise).

4.3 CNN Model Design

ACNN can be conceptualized using three basic layers: (a) convolution, (b) maxpooling, and (c) the final (or output) layer. In the convolution layer, features are derived from an image using a pre-defined weight filter. Depending upon the size of the weighting filter, feature maps are produced [12][13]. The convolution layer ,at the beginning of the model, merely identifies the generic features [12]. As the model's depth increases, the complexity of the extracted features also increases. The feature maps generated by the last convolution layer of the model are much closer to the problem at hand. Sometimes these feature maps are too large so, to decrease the number of trainable parameters, pooling layers are introduced. The most commonly used pooling layer form is the max pooling. Maxpooling layer is used to minimize the number of parameters, in the case when images are too large. After convolution and pooling layers, the output is formulated. Note that, the convolution and maxpooling layers are only able to derive features and decrease the number of parameters. Thus, the role of the output layer (also known as final layer), which is a fully connected layer, is to deliver an output in the form of the output classes. The architecture of the proposed CNN model, used in our work, comprises of two convolution layers, with alternate max-pooling layers, and an output layer, as shown in Figure 6.

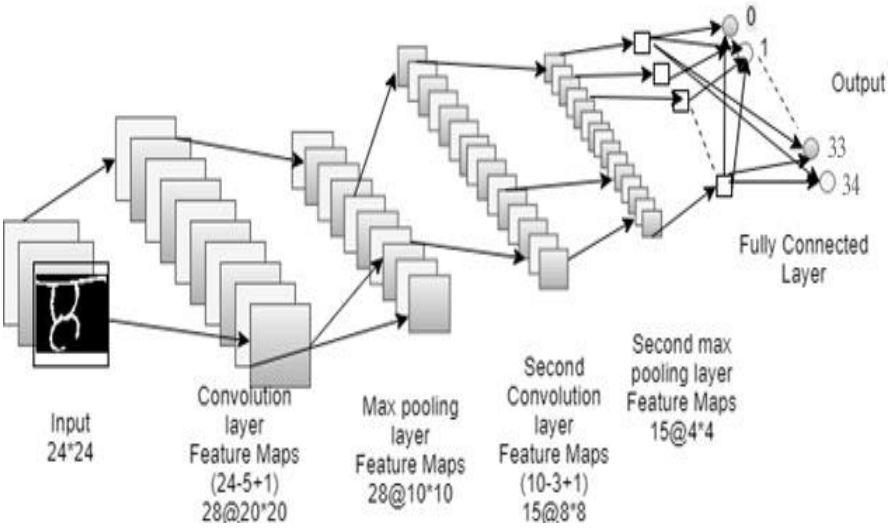

Figure 6: Layer diagram of the proposed CNN model.

In our work, the first convolution layer uses 28 weight filters of size (5*5) on (24*24) images, from which 28 feature maps, of size 20*20, are obtained. From the first layer, the corresponding 28 feature maps are passed to the maxpooling layer with the filter size of (2*2). The maxpooling layer produces 28 feature maps of size (10*10), which are again passed through a second convolution layer, with 15 weight filters of size (3*3). This creates 15 feature maps of the size of (8*8). Resulting features are then passed through a maxpooling layer with the same filter as the last maxpooling layer. The final output from these layers is passed to a final layer, i.e. a fully connected layer, which uses the Softmax as the activation function. This fully connected layer transforms the feature maps to the 35 classes. As a result, the designed CNN model is classifying the images of the corresponding 35 Gurumukhi characters [14][15][16]. All layers in this model, except the pooling layers, use the Rectified Linear Unit (ReLu) as an activation function. In summary, the steps used to apply the proposed CNN architecture to solve the problem of Gurumukhi script recognition are as follows.

i. Input:

Scan the handwritten manuscript.

Label each character. [3,500 images written by 100 participants]

ii. Preprocessing:

a) Convert all character images into the gray-scale format and then invert all pixel value changing them into binary form.

/**Rgb2gray converts: Change over RGB esteems to gray-scale esteems by framing a weighted sum of the R, G, and B value of pixel: **/

$$Y = 0.2989 * R + 0.5870 * G + 0.1140 * B \quad (1)$$

b) Resize all images into 24*24-pixel value matrices.

iii. Splitting Dataset:

Split all image data set into 80% of training and 20 % of testing data at random state 45.

iv. Use CNN model for character recognition using two convolutional layers with alternate max pool layer.

a) Apply ReLu (Rectified Linear Unit) activation function on the Convolutional Layer.

$$f(x) = \begin{cases} 0 \ for x \leq 0 \\ x for x \geq 0 \end{cases} \quad (2)$$

b) Apply max-pooling layerwith (2*2) pool-size (filter)

- Input a volume of an image W1×H1×D1
- Output:

$$W2 = (W1 - F)/S + 1 \quad (3)$$

$$H2 = (H1 - F)/S + 1 \quad (4)$$

$$D2 = D1 \quad (5)$$

Here W1, H1, and D1are the width, height and channel of an input image. W2, H2 and D2 are the width, height and channel of an output image. F is the size of Filter. S is the stride, i.e. the step size of the convolution operation.

c) Apply Softmax Regression (often used in the final layer of the neural network) to handle multiple classes classifications.

$Y^{(i)} = \{1, \ldots, K\}$ //K ->number of classes

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{K=1}^{K} e^{z_k}} \quad (6)$$

Where $\sigma(z)$ = Previous layer weighted sum of output.

d) In the end, use 'Categorical Cross-entropy' function to determine loss optimization. Specifically, use the following function:

$$H(T,q) = -\sum_{i=1}^{N} \frac{1}{N} \log_2 q(x_i) \quad (7)$$

/**Where the size of the test set is N and the probability of event x evaluated from the training set is q(x). This is also known as Monte Carlo estimation**/

v. **Output:**

This experiment results, for the Gurumukhi character recognition, results in accuracy, for the training data, of 98.32%. Furthermore, the testing data accuracy is 74.66% (see, the next Section).

5 EXPERIMENTAL RESULTS AND DISCUSSIONS

The proposed model has been experimented with using the dataset, which was prepared as described in Section 4.1. As noted, for training, 80% of the dataset was used whereas the remaining 20% was used for testing purposes. This was done in such way that each of the 35 core Gurumukhi characters has been split into 80%-20% groups, to assure balance between "character classes" in the training and the testing datasets.

The CNN model was implemented in Python, using machine learning libraries Tensorflow and Keras. The two parameters that impact the efficiency of the CNN model are the learning rate and batch size. The learning rate (LR) of a model implies how rapidly the weights of the neurons will be adapted. Here, three different LR shave been tried: 0.0005, 0.0007 and 0.0009. The test set accuracy graph for different LR's, at different iterations, is shown in Figure 7.

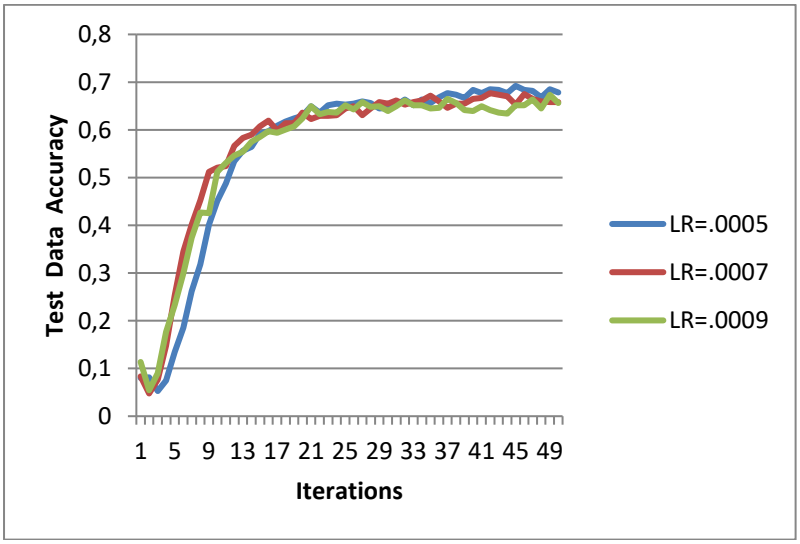

Figure 7: Test data accuracy for each iteration

From Figure 7 it can be observed that, for more than 30 iterations, use of LR=0.0005 results in slightly higher accuracy of character recognition (for the test dataset) than the remaining two LR's. However, the results are relatively close to each other. More precise data can be found in Table 1. Here, the training set accuracy and the test dataset accuracy for three different LR's are presented. It can be seen that the highest accuracy for LR=0.0005 (after 50 iterations) has been achieved for both the training and test dataset accuracy. Hence, LR=0.0005 has been used in subsequent experiments.

Table 1: Training and test data accuracy at different LR's after 50 iterations

Learning rate	Training dataset accuracy	Test dataset accuracy
0.0005	0.9788	0.678571
0.0007	0.9778	0.658163
0.0009	0.9744	0.656463

Note that the 3,500 resized images (24*24 pixels), used in the experiments, cause a memory overflow when considered all at once. Hence, a batch size becomes also a parameter for evaluating the performance of the proposed model. Test dataset error has been analyzed for three different batch sizes (BS) of 50, 60 and 70. Figure 8 demonstrates the impact of BS on testing dataset error for different iterations. From Figure 8, it is clear that the test dataset accuracy is highest at BS=60 for iterations 16-50. Hence batch size of 60 is preferable when larger number of iterations can be applied.

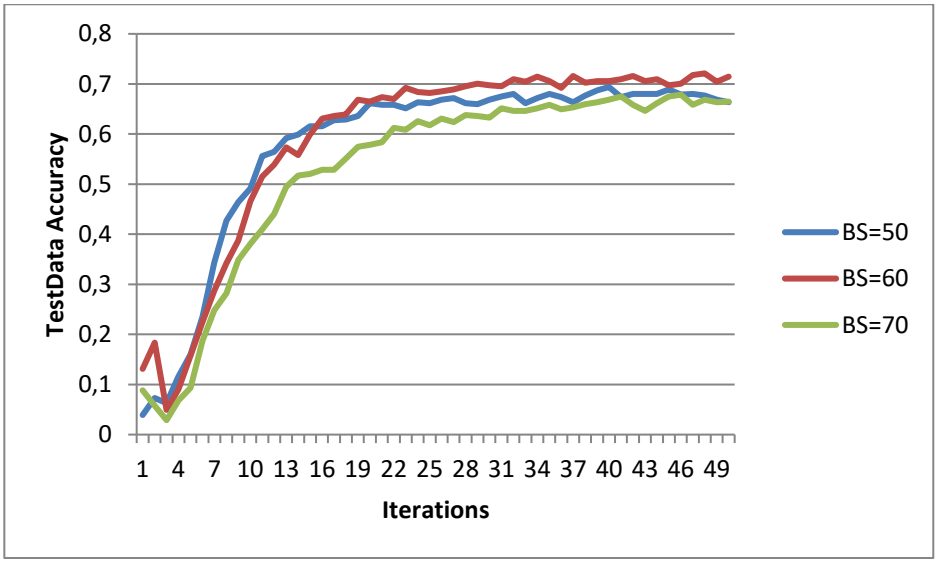

Figure 8: Effect of Batch Size on Test Dataset accuracy at LR=0.0005

Table 2 shows the training and test data set accuracy for different batch sizes at LR=0.0005.

Table 2: Performance at different batch Size at LR = 0.0005

Batch size	Test dataset accuracy	Training dataset accuracy
50	66.32	94.04
60	71.42	97.83
70	66.49	98.00

Finally, in Table 3 we compare test data accuracy for all learning rates and batch sizes. This table is collectively showing the effect of learning rate and batch size on test dataset accuracy.

Table 3: Test data accuracy of CNN model at different LR and BS

Learning Rate	Batch Size	Test set accuracy (%)
0.0005	50	**74.66**
	60	71.42
	70	66.49
0.0007	50	**74.32**
	60	69.39
	70	69.90
0.0009	50	**74.51**
	60	72.96
	70	69.39

The simulations have been done for different combinations of learning rate and batch size with Gaussian filter and it is observed that the highest accuracy of 74.66% was achieved for learning rate of 0.0005 and batch size 50. It can also be observed, from this Table, that the system is always giving better accuracy for the batch size of 50 for all the three learning rates. Similarly said, system performs comparatively better for learning rate of 0.0009. So if batch size is kept small and learning rate is high, the best system performance is observed.

6 CONCLUSION AND FUTURE SCOPE

In comparison to feature extraction, Convolution Neural Networks (CNN) can directly draw visual patterns from the pixelated images. Hence, in this study, a CNN model was applied to Gurumukhi character classification. A handwritten dataset of 35 fundamental Gurumukhi characters comprising of 100 images for each character written by 50 different persons was prepared to represent different writing styles of people. It was established that, for the proposed CNN model, the best learning rate is 0.0005 and the batch size of 50. Accuracy obtained for the training data was 98.32%, while the testing data accuracy was 74.66%. In the future, we plan to (a) apply the proposed model to the complete set of all Gurumukhi characters, (b) develop automatic feature extraction approach(es) and combine them with the CNN, and (c) apply the proposed approach to the complete scripts / words in the Gurumukhi language.

REFERENCES

1. R. D. Zarro and M. A. Anwer, “Recognition-based online Kurdish character recognition using hidden Markov model and harmony search,” *Eng. Sci. Technol. an Int. J.*, vol. 20, no. 2, pp. 783–794, 2017.
2. Z. Alom, P. Sidike, M. Hasan, T. M. Taha, and V. K. Asari, “Handwritten Bangla Character Recognition Using The State-of-Art Deep Convolutional Neural Networks,” *Comput. Vis. Pattern Recognit.*, pp. 1–12, 2017.
3. D. Sharma and U. Jain, “Recognition of Isolated Handwritten Characters of Gurumukhi Script using Neocognitron,” *Int. J. Comput. Appl.*, vol. 10, no. 8, pp. 975–8887, 2010.
4. G. Lehal and C. Singh, “A Gurumukhi Script Recognition System,” *Int. Conf. Pattern Recognit.*, vol. 2, no. 2, pp. 557–560, 2000.
5. N. Kumar, S. Gupta, “A Novel Handwritten Gurumukhi Character Recognition System Based On Deep Neural Networks,”International Journal of Pure and Applied Mathematics, vol. 117, no. 21, pp. 663–678, 2017.
6. M. Kumar, M. K. JindalR. K. Sharma, “k -Nearest Neighbor Based Offline Handwritten Gurumukhi Character Recognition,” Proceedings of the 2011 International Conference on Image Information Processing (ICIIP 2011).
7. M. Kumar, M. K. Jindal and R. K. Sharma, "Classification of characters and grading writers in offline handwritten Gurumukhi script," 2011 International Conference on Image Information Processing, Shimla, 2011, pp. 1-4.
8. N. Kumar, S. Gupta, “Offline Handwritten Gurumukhi Character Recognition : A Review,”International Journal of Software Engineering and its Applications vol. 10, no. 5, pp. 77–86, 2016.
9. S. Acharya, A. K. Pant and P. K. Gyawali, "Deep learning based large scale handwritten

Devanagari character recognition," 2015 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Kathmandu, 2015, pp. 1-6.doi: 10.1109/SKIMA.2015.7400041

10. S. Roy, N. Das, M. Kundu, and M. Nasipuri, “Handwritten isolated Bangla compound character recognition: A new benchmark using a novel deep learning approach,” *Pattern Recognit. Lett.*, vol. 90, pp. 15–21, 2017.
11. H. Kaur, S. Rani, “Handwritten Gurumukhi Character Recognition Using Convolution Neural Network” *International Journal of Computational Intelligence Research*13.5, vol. 90, pp. 933-943, 2017.
12. S. Naz, A. I.Umar, R. Ahmad, I.Siddiqi, S. B.Ahmed, M. I.Razzak, F.Shafait, “Urdu Nastaliq recognition using convolutional–recursive deep learning,” *Neurocomputing*, vol. 243, pp. 80–87, 2017.
13. B. Chakraborty, B. Shaw, J. Aich, U. Bhattacharya, and S. K. Parui, “Does Deeper Network Lead to Better Accuracy: A Case Study on Handwritten Devanagari Characters,” *2018 13th IAPR Int. Work. Doc. Anal. Syst.*, pp. 411–416, 2018.
14. M. M. Rahman, M. A. H. Akhand, S. Islam, P. Chandra Shill, and M. M. Hafizur Rahman, “Bangla Handwritten Character Recognition using Convolutional Neural Network,” *Int. J. Image, Graph. Signal Process.*, vol. 7, no. 8, pp. 42–49, 2015.
15. M. Mhiri, C. Desrosiers, and M. Cheriet, “Convolutional pyramid of bidirectional character sequences for the recognition of handwritten words,” *Pattern Recognit. Lett.*, vol. 111, pp. 87–93, 2018.
16. M. Z. Alom, P. Sidike, T. M. Taha, and V. K. Asari, “Handwritten Bangla Digit Recognition Using Deep Learning, arXiv:1705.02680, 2017.