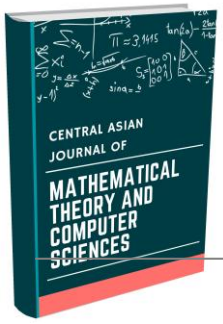


CENTRAL ASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES

https://cajmtcs.centralasianstudies.org

Volume: 04 Issue: 12 | Dec 2023 ISSN: 2660-5309

USING MACHINE LEARNING MODELS AND DEEP LEARNING NETWORKS FOR HANDWRITTEN NUMBERS AND LETTERS RECOGNITION

Sarmad Hamzah Ali

Al-Muthanna University, Iraq

Abstract

This article delves into the utilization of a multitude of classification algorithms and deep learning neural networks for the purpose of identifying handwritten letters in photographs or during manual input. A total of eight variations of recognition technology were subjected to analysis and testing, the incorporation of classifiers sourced from the Scikit-learn package, as well as the utilization of deep learning neural networks, are both integral components of the study at hand. In order to construct and train these neural networks or train classifiers, we opted for well-established and comprehensive databases, namely the base of handwritten digits MNIST, the base of handwritten letters of the Latin alphabet EMNIST, and the CoEMNIST dataset for Cyrillic characters. Two types of neural networks were taken into consideration, namely sequential and convolutional. The neural networks were trained through the utilization of varying numbers of epochs. The recognizable images were resized to 28x28 (784 cells can be represented in one dimension). The preprocessing of images (such as filtering and scaling) was undertaken utilizing the OpenCV library.

ARTICLEINFO

Article history:
Received 3 Oct 2023
Revised form 20 Nov 2023
Accepted 2 Dec 2023

Keywords: *Youth, Uganda, Self-employment, Binary and Multivariate Probit Models.*

---***---

Upon the construction of recognition models through the application of Scikit-learn classifiers or neural networks, the accuracy of recognizing all handwritten digits through the test program ranged between 98% and 100% (with one or no errors in the identification of 50 handwritten digits).The accuracy of recognition of handwritten letters of the Latin alphabet using the MLP classifier turned out to be somewhat worse - at the level of 93-96%. The accuracy of recognition of handwritten Cyrillic letters using the SVC classifier or the CNN network was lower - at the level of 90-92%.

1. Introduction

This article explores the potential of multiple classification algorithms for the identification of Latin letters and numbers in photographic images or manual input. The algorithms considered are Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), Extra Tree Classifier (EF), and various neural network variants. Optical character recognition (OCR) is a pivotal domain of inquiry in the realm of artificial intelligence and pattern recognition [1]. It encompasses a classification undertaking that involves the identification of a set of characters in an image, which is segregated into 10 categories for numeric characters or 26 categories for Latin letters of the alphabet. The segmentation and recognition of image regions that harbor handwritten or printed characters are significant quandaries due to the pervasiveness of technical utilities [2].

These systems automate the process of reading letters and numbers, storing the information obtained, and are used for tasks such as the recognition of license plates [3], the registration and recognition of the numbers of wagons and tanks [4-5] using neural network technologies. While numerous systems can identify printed text, the recognition of handwritten characters still poses a challenge in pattern recognition [2, 6].

Image preprocessing is a crucial aspect of accurate handwritten digit or letter recognition. For instance, the preprocessing and segmentation of images depicting license plates have been found to exert a significant influence on the outcomes of their recognition [7]. Incorrect character segmentation does not provide accurate results for any recognition method. As such, various pre-processing methods are utilized for training and test data, as well as images for recognition [8].

The application of pre-processing techniques involving a combination of elasticity and rotation has been observed to significantly improve the accuracy of the three networks under consideration by a maximum of 0.71%. Previous research has delved into the study of shears, rotations, as well as elastic deformations and their relevance to machine learning in diverse contexts [9,10].

The volume and quality of the training set significantly affect the performance of the recognition system. A simple technique was proposed in [11], involving the use of elastic distortions to expand the training set significantly. The MNIST dataset was extended four times [12] using affine transformations and pre-setting the properties of invariant transformations. These distortions improved the recognition and classification results of the MNIST dataset.

The choice and improvement of algorithms for classifying and recognizing images of letters and numbers is the subject of research in many works.

2. Purpose and objectives of the study.

Numerous research results are known on the recognition of handwritten digits and Latin letters based on the MNIST and EMNIST datasets. A relatively recent analysis of the current state of the issue is presented .

Comparative studies of Cyrillic letter recognition using known datasets, various classifiers and libraries are fragmentary.

3. Methodology and results of computer experiment

3.1. Equipment and data set

Calculations were performed on a computer running Ubuntu 21.10, 9th generation i5 processor, GeForce GTX 1650 Mobile graphics card, 8 GB of RAM.

To construct and educate a model for the identification of numerical digits, a highly-regarded and thoroughly comprehensive database of manually-written digits, recognized as MNIST, was elected.

The EMNIST database consists of six datasets. In this work, we used the EMNIST Letters data set (124,800 trains, 20,800 test samples). It includes images of all uppercase and lowercase letters of the Latin alphabet to form a balanced classification problem of 26 classes.

The CoMNIST dataset was used to recognize Cyrillic characters.

3.2. Image pre-processing

To identify regions within images that possess identifiable numerical values, we employed the functionalities inherent to the OpenCV library. To extract the contours of digits, the findContours function or the algorithm for extracting the most stable extreme region (mser) were used.

The algorithm for image preprocessing and selection of an area containing numbers or letters included the following steps [13]:

1) The noise level was effectively reduced through the implementation of image filtering, whereby a Gaussian filter was utilized via the cv2 GaussianBlur function.
2) To eliminate extraneous noise, image binarization was employed by means of the cv2 threshold function, which was strategically configured to ensure the accurate selection of digit contours.

3.3. Implementation of the considered algorithms

To measure the precision of algorithms in terms of their execution, complexity, number of epochs (in cases of deep learning algorithms) and performance accuracy, this written piece employs a variety of classifiers such as SVM, KNN, RF, Extra Tree Classifier, and MLP.

The implementation of support vector machine algorithms utilized the sklearn.svm module. Scikit-learn offers three classes for binary and multiclass classification, which include SVC, NuSVC, and LinearSVC. Although SVC and NuSVC are similar in methodology, they differ slightly in their parameter sets and mathematical formulations. On the other hand, LinearSVC is a faster SVC implementation designed for the linear kernel case.

The random forest classifier is a meta-estimator that fits various decision tree classifiers to different subsamples of the dataset. This approach uses averaging to improve the accuracy of predictions and to control overfitting. The size of the subsample is determined by the max_samples parameter, which is controlled by the conditional "trees" or the entire dataset if bootstrap = True (default). The tuning of the classifier involved selecting the optimal value of the max_samples parameter.

Scikit-learn offers two nearest neighbor classifiers: KNeighbors Classifier, which learns based on the k nearest neighbors of each query point, where k is an integer value specified by the user. The work in this article utilized the KNeighbors Classifier by selecting the number of neighbors needed for each sample.

Extra Tree Classifier with Bagging Classifier was used without changing the default settings.

The implementation of the MLP classifier was carried out through the utilization of the scikit-learn package. As evidenced by the experience of setting up the MLP classifier, the accuracy of character recognition is primarily influenced by the following parameters:

- ✓ The hidden_layer_sizes parameter, which denotes the number of neurons in the i-th hidden layer.
- ✓ The solver parameter, which is responsible for weight optimization.
- ✓ The learning_rate_init parameter, which is the initial learning rate used to control the step size when updating weights (only utilized when solver = 'sgd' or 'adam').

Through experimental analysis, it was ascertained that optimal precision outcomes can be achieved through the selection of an ample quantity of neurons in the concealed strata, as well as the implementation of sgd or adam optimization methodologies, alongside a judicious initial rate of learning.

Moreover, more intricate variations of the MLP and CNN classifiers were established using the TensorFlow package, equipped with a Keras frontend in the Python programming language.

The MLP model utilized for recognizing digits or letters was comprised of the layers input and output, in addition to more than one layers are hidden.

The layer denoted as the output layer is comprised of nodes that utilize the tf.nn.softmax function (10 nodes for digit recognition, or 26 nodes for Latin letters recognition), which returns an array of ten probability scores whose sum is 1. Each node contains a score , which indicates the probability that the current image belongs to one of the 10 classes for numbers or 26 (33) classes for letter recognition.

Various configurations of neural networks for deep learning, utilizing the Keras framework, and employed for recognition purposes, are depicted in Figure 1.

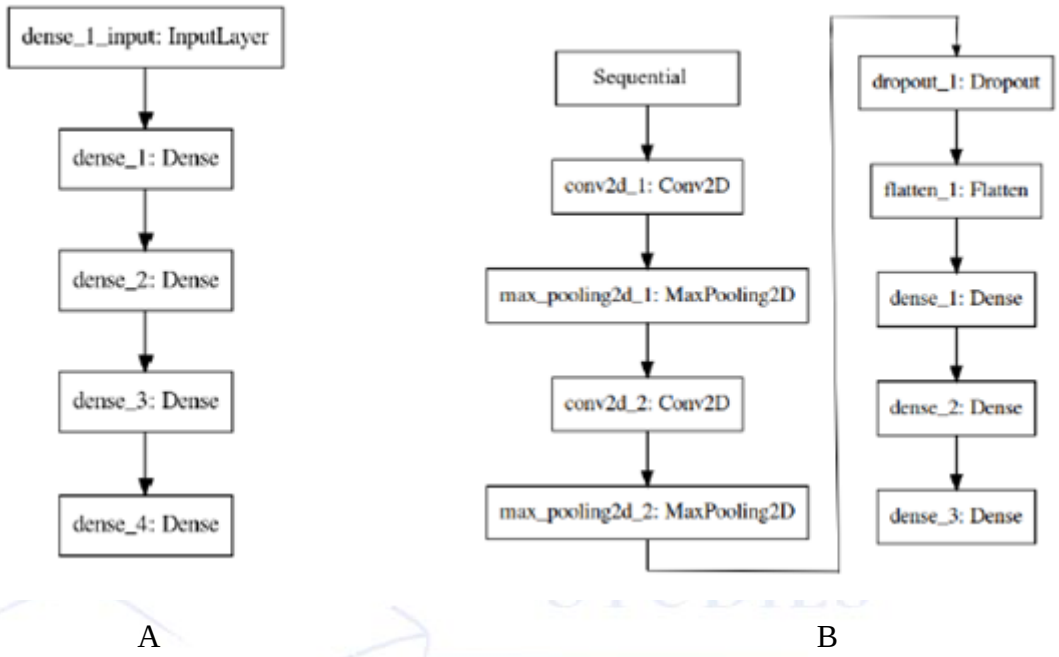


Fig. 1. Various adaptations of neural network architectures for deep learning have been developed utilizing the Keras framework. Their primary purpose is to facilitate the recognition of numerical or alphabetical characters.

A simplified iteration of a fully connected neural network (depicted in Figure 1a) demonstrated a recognition error rate of 1.3–1.5% in the test sample, however, in real-world scenarios, the accuracy of recognition did not surpass 70%. To attain more precise recognition of ROI digits, a CNN (refer to Figure 1b) was implemented. Nonetheless, the drawback of these networks is the significantly prolonged training duration.

The training of a CNN to recognize letters, specifically from a given set, necessitates the incorporation of CUDA technology as the letters' data is exceedingly intricate.

The phases involved in the development and instruction of a model, along with the identification of practical examples, can be readily segregated owing to the model's exportability.

4. Results

The findings of the support vector classifier (SVC) incorporated in the scikit-learn software package for the MNIST dataset are presented in Figure 2. It is evident from the outcomes depicted in Figure 2 that the optimal accuracy in image recognition of the MNIST sample utilizing the SVC classifier is attained for the "rbf" kernel and a regularization parameter of no less than 50. This outcome persisted even after modifying the recognition of the EMNIST dataset.

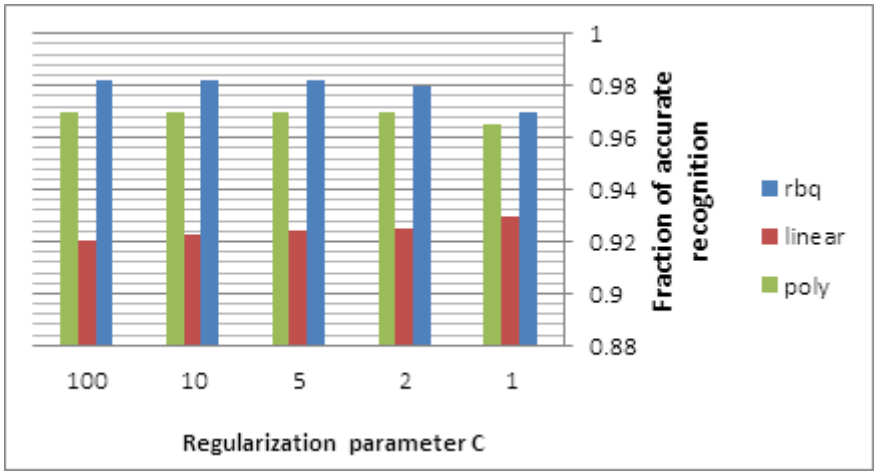


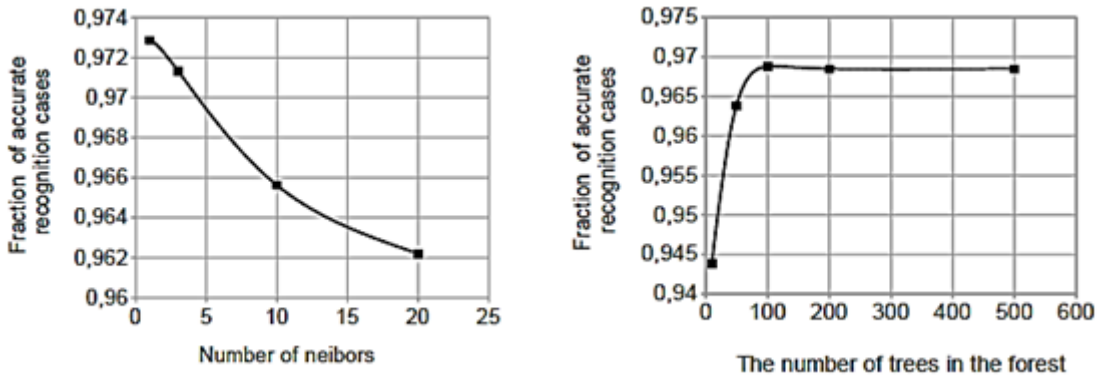
Fig. 2. The present table outlines the outcomes of the evaluation of the accuracy of recognition for the Support Vector Classification (SVC) classifier with varying calibration parameters. Furthermore, the table compares recognition results amongst different kernels, specifically utilizing the "rbq," "linear," and "poly" kernels.

Analogous investigations were conducted for alternative classification alternatives such as KNN, RF, and MLP. The outcomes of the scrutiny of the influence of the KNN and RF classifiers' settings on the MNIST dataset can be observed in Figure 3.

The presented data in Figure 3 unambiguously illustrates that modifying the number of nearest neighbors for the KNN classifier or augmenting the number of trees in the forest for the RF classifier engenders a marked escalation in recognition accuracy. Nevertheless, neither of the aforementioned classification algorithms exhibits any capacity to enhance recognition accuracy by means of modifying the settings.

The configuration of the MLP model was found to exert a significant influence on the level of accuracy achieved in recognition tasks.

The findings derived from the computational experiment are visually presented in figure 4. The results of the computational experiment for the Bagging Classifier with an additional classifier Extra Tree Classifier are shown in Fig. 5.



a) Results for KNeighbors Classifier b) Results for RF Classifier

Fig. 3. Results of estimation of recognition accuracy for KNeighborsClassifier and RF Classifier with different parameters calibration

The graphical representations presented in Figure 4 were generated through the utilization of the MLP Classifier methodology from the scikit-learn software package for the MNIST dataset. Results comparable to those achieved through the use of the Keras package were obtained, with respect to the impact of the

quantity of hidden layers and neurons within a layer on recognition accuracy, it is noteworthy to mention the significance of a more resilient classifier.

Figure 5 reveals that amplifying the number of estimators for the Bagging Classifier, coupled with an Extra Tree Classifier, results in a higher level of recognition accuracy. This classifier offers the shortest duration for tuning completion when compared to all other classifiers examined, while still maintaining an acceptable level of recognition accuracy.

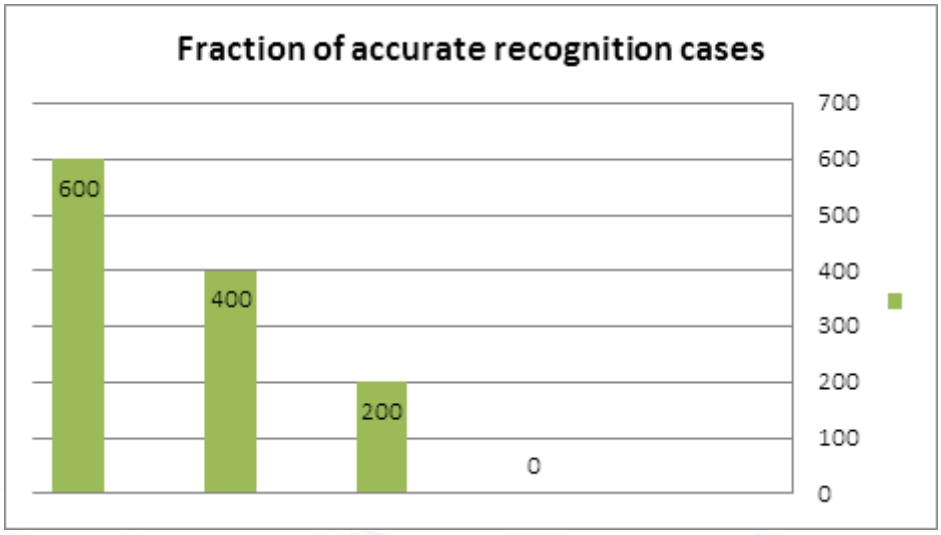


Fig. 4. The present study presents the outcomes of the assessment of the recognition accuracy of the MLP Classifier, utilizing the scikit-Learning package, while employing various parameters.

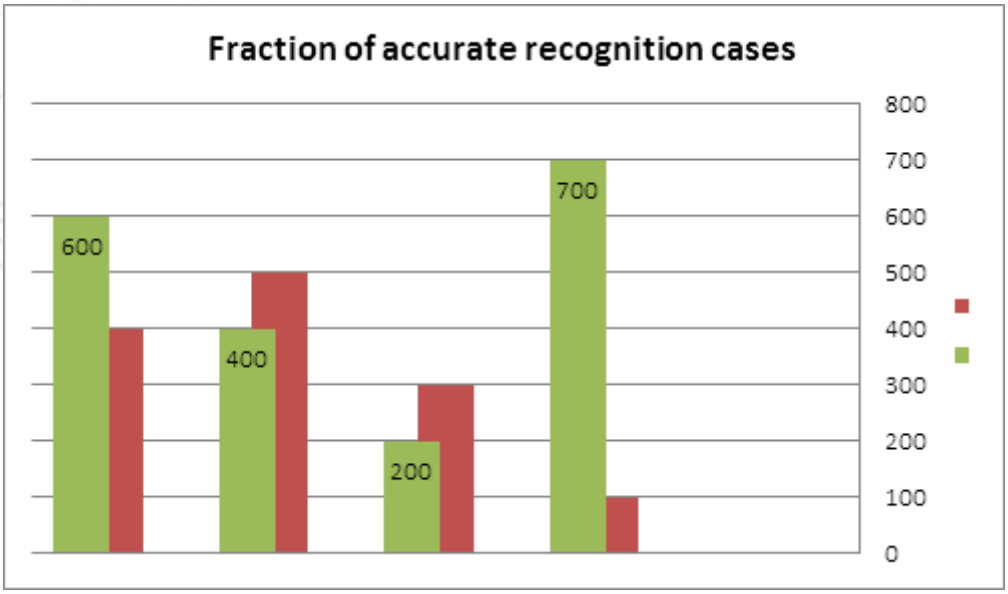


Fig. 5. Results of estimation of recognition accuracy for Extra Tree (scikit-learn package) with different parameters on EMNIST and CoMNIST datasets

As is evident from the displayed graphs, the precision of identifying the numerical quantities of digits within the trial sample escalates in proportion to the augmentation of neurons within the concealed strata (under the assumption of homogeneity across all layers) and the amplification of the number of neurons in the hidden layers.

Comparison of the results for the two datasets showed the same direction of the impact of changes in the number of estimates.

5. Discussion of results

When conducting model optimization on image recognition tasks such as MNIST, CoMNIST, and EMNIST datasets, it was observed that the recognition accuracy exhibits a marginal improvement with a rise in the quantity of concealed strata. However, it is crucial to note that the choice of optimizer type and its associated parameters, as well as the number of training epochs, have a more significant impact on the efficacy of model tuning.

The results of assessing the classification accuracy of MNIST, CoMNIST or EMNIST Letters datasets by various methods are presented in Table 1.

CoMNIST	EMNIST	MNIST	Classifier
84.8	92.0	98.2	C-Support Vector Classificator (SVC)
80.3	88.5	96.8	Extra Tree Classifier with Bagging Classifier
78.0	87.8	96.8	RF Classifier
75.6	85.8	96.9	KNeighbors Classifier
79.9 (5 layers)	89.7 (3 layers)	98.5	(Scikit-Learning)

Table 1. Results of assessing classification accuracy for different data sets

By appropriately adjusting all the aforementioned models, which range from K-Nearest Neighbors to Convolutional Neural Networks, as their complexity increases, it was determined that the estimated accuracy of the MNIST test set falls within the range of 97.5%-98.5%.However, it is noteworthy that this accuracy is slightly suboptimal in comparison to the optimal outcomes attained through the utilization of pre-tuned or convolutional neural networks.

Classifying an EMNIST dataset is significantly more difficult than classifying an MNIST dataset due to the significantly larger amount of data in the dataset. Therefore, the classification accuracy is somewhat lower - within 86-92%.

For the classification of the CoMNIST dataset, the achieved classification accuracy is in the range of 75.6-84.8%. The original set of images contains letters of different sizes depending on the area of interest. Some of them are offset to the edges.

The conversion of the original set of images to the MNIST format was carried out in two ways - directly or with centering and fitting images to the same size. The results of classifier tuning are slightly different (the table shows the data for the second option).

For all considered classifiers, the result of setting up a centered data set is better than for the original one. For example, for the SVC classifier for a data set without centering, the accuracy was 78.2%, for a sample with centering and fitting - 84.8%.

The classification accuracy of both datasets depends on the method settings. For the CNN neural network, the accuracy of recognizing Cyrillic letters was the highest - 89.7%. In this case, preliminary preparation of the data set did not improve recognition accuracy.

An important advantage of using deep learning neural networks based on the Keras/TensorFlow package is the ability to parallelize the learning process and significantly speed it up using CUDA technology.

Due to the use of CUDA, the CNN-based recognizer setup time turned out to be slightly less than for most classifiers from the Scikit-Learning package.

Conclusion

In this work, a number of models were implemented (classifiers SVC, KNN, RF, Extra Tree, MLP in two versions and a convolutional neural network) for recognition of handwritten numbers and letters using EMNIST, CoMNIST or MNIST datasets. Models were compared mainly in terms of accuracy.

1. All surveyed algorithms for the recognition of digits, following a straightforward calibration, exhibited nearly identical levels of accuracy in recognizing handwritten digits, with variations of no more than 1%.
2. For the recognition of handwritten letters of English alphabets, the spread in the accuracy of various classifiers is greater, but fits within + 5%.
3. CNN has been found to give the most accurate results for recognizing handwritten letters or numbers. The time to set up a CNN-based recognizer using CUDA turned out to be slightly less than that of most classifiers from the Scikit-Learning package.
4. When using the CoMNIST data set, pre-image preparation has an ambiguous effect on the result. For all considered classifiers, the result of setting up a centered data set is better than for the original one.

References

1. R. Tokas and A. Bhadu, "A comparative analysis of feature extraction techniques for handwritten character recognition," *International Journal of Advanced Technology & Engineering Research*, vol. 2, no. 4, pp. 215–219, 2012.
2. I.A. Jannoud, "Automatic Arabic hand written text recognition system," *American Journal of Applied Sciences*, vol. 4, no. 11, pp. 857–864, 2007.
3. A. rani and D. Singh, "Handwritten English character recognition using neural network," *International Journal of engineering science & Communication*, vol. 1, no. 2, pp. 141–144, 2010.
4. F. Stark, C. Hazirbas, R. Triebel, D. Cremers, CAPTCHA Recognition with Active Deep Learning, In: GCPR Workshop on New Challenges in Neural Computation, 2015. URL: https://vision.in.tum.de/_media/spezial/bib/stark-gcpr15.pdf
5. W. Liu, J. Wei, and Q. Meng, "Comparisions on KNN, SVM, BP and the CNN for handwritten digit recognition," in *Proceedings of the 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, pp. 587–590, IEEE, Dalian, China, August 2020.
6. F. Siddique, S. Sakib, M.A.B. Siddique, Handwritten Digit Recognition using Convolutional Neural Network in Python with Tensorflow and Observe the Variation of Accuracies for Various Hidden Layers. Preprints 2019, 2019030039. doi: 10.20944/ preprints201903.0039.v1.
7. O. Vovchuk, M. Kyrychenko, Y. Kazan, Recognition of Handwritten Cyrillic Letters, 2019.URL: https://www.researchgate.net/publication/336987544_Recognition_of_Handwritten_ Cyrillic_Letters_using_PCA
8. Alom, M.Z.; Sidike, P.; Hasan, M.; Taha, T.M.; Asari, V.K. Handwritten bangla character recognition using the state-of-the-art deep convolutional neural networks. *Comput. Intell. Neurosci.* 2018, *2018*, 6747098.
9. Sutradhar, S. Old English Character Recognition Using Neural Networks 2018. Electronic Theses and Dissertations, Georgia Southern University. Available online: https://digitalcommons.georgiasouthern.edu/etd/1783/ (accessed on 28 February 2023).
10. Sazal, M.M.R.; Biswas, S.K.; Amin, M.F.; Murase, K. Bangla handwritten character recognition using deep belief network. In Proceedings of the 2013 International Conference on Electrical Information and Communication Technology (EICT), Khulna, Bangladesh, 13–15 February 2014.

11. Baldominos, A.; Saez, Y.; Isasi, P. A survey of handwritten character recognition with MNIST and EMNIST. *Appl. Sci.* 2019, *9*, 3169.

12. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. Pattern Recognit. 2018, 77, 354–377.

13. Bala, R.; Singh, C. An optimized CNN-based handwritten gurmukhi character recognition from punjabi script image. Int. J. Sci. Res. Comput. Sci. Appl. Manag. Stud. 2020, 9, 1–10.