



Article

Recognition and Classification of Handwritten Urdu Numerals Using Deep Learning Techniques

Aamna Bhatti ¹, Ameera Arif ¹, Waqar Khalid ², Baber Khan ³ , Ahmad Ali ⁴, Shehzad Khalid ² and Atiq ur Rehman ^{5,*} 

- ¹ School of Electrical Engineering and Computer Science, National University of Sciences and Technology, Islamabad 24090, Pakistan
- ² Computer Engineering Department, Bahria University, Islamabad 44000, Pakistan
- ³ Department of Electrical and Computer Engineering, International Islamic University, Islamabad 04436, Pakistan
- ⁴ Department of Software Engineering, Bahria University, Islamabad 44000, Pakistan
- ⁵ Artificial Intelligence and Intelligent Systems Research Group, School of Innovation, Design and Engineering, Mälardalen University, Högskeplan 1, 722 20 Västerås, Sweden
- * Correspondence: atiq.ur.rehman@mdu.se or atiqjadoon@gmail.com

Abstract: Urdu is a complex language as it is an amalgam of many South Asian and East Asian languages; hence, its character recognition is a huge and difficult task. It is a bidirectional language with its numerals written from left to right while script is written in opposite direction which induces complexities in the recognition process. This paper presents the recognition and classification of a novel Urdu numeral dataset using convolutional neural network (CNN) and its variants. We propose custom CNN model to extract features which are used by Softmax activation function and support vector machine (SVM) classifier. We compare it with GoogLeNet and the residual network (ResNet) in terms of performance. Our proposed CNN gives an accuracy of 98.41% with the Softmax classifier and 99.0% with the SVM classifier. For GoogLeNet, we achieve an accuracy of 95.61% and 96.4% on ResNet. Moreover, we develop datasets for handwritten Urdu numbers and numbers of Pakistani currency to incorporate real-life problems. Our models achieve best accuracies as compared to previous models in the literature for optical character recognition (OCR).

Keywords: urdu numeral recognition; convolutional neural network; SVM; GoogLeNet; ResNet



Citation: Bhatti, A.; Arif, A.; Khalid, W.; Khan, B.; Ali, A.; Khalid, S.; Rehman, A.u. Recognition and Classification of Handwritten Urdu Numerals Using Deep Learning Techniques. *Appl. Sci.* **2023**, *13*, 1624. <https://doi.org/10.3390/app13031624>

Academic Editor: José Salvador Sánchez Garreta

Received: 5 December 2022

Revised: 11 January 2023

Accepted: 20 January 2023

Published: 27 January 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

OCR technology scans printed characters to determine their shape by recognizing edge information, and then translates them into characters by process of character recognition [1]. In recent years, one of the most fascinating and difficult research areas in the fields of image processing and pattern recognition has been handwriting recognition. It has many applications such as OCR, pattern classification, postal mail sorting, bank cheque processing, form data entry, etc. Such character recognizers prove to be fruitful for humans because of their speed and accuracy. Mostly, they are based on deep learning models and solve the problem efficiently.

In the study of languages, Urdu is one of those cursive languages that is hugely popular in South Asian countries such as Pakistan, India, Bangladesh, Bhutan, and Nepal. It is the national language of Pakistan and is widely spoken in urban areas, while its adoption as a second language in rural areas is in progress. It is an amalgamation of many languages; hence, its script contains loanwords and is written in different variants all around the globe. Another important characteristic is that it is bidirectional with its numerals written from left to right while its script is written in the opposite direction [2]; therefore, this becomes a problem for OCR. It has up to 40 letters in its script and 10 numerals [3]. Urdu, Persian, and Eastern Arabic numerals are written on similar patterns; however, some of the digits differ, as shown in Figure 1, which is yet again another concern. Some other challenges that

are encountered during its OCR is blurred text, torn paper, and spacing between letters when the data is found in written form [3].

Urdu	.	۱	۲	۳	۴	۵	۶	۷	۸	۹
Persian	.	۱	۲	۳	۴	۵	۶	۷	۸	۹
Eastern Arabic	.	۱	۲	۳	۴	۵	۶	۷	۸	۹

Figure 1. Urdu, Persian, and Eastern Arabic numerals.

Our aim is to develop a classifier for our novel dataset. The motivation behind developing an Urdu numeral classifier is that majority of the work has been done on English numerals; however, no such work has been done on Urdu numerals which are distinct in features as compared to English. Data unavailability is a major obstacle in the development of Urdu handwritten character recognition [4,5]. As no dataset of Urdu numerals is available publicly for research purposes, we, therefore, present a novel dataset gathered specifically for this study. In this way, problems related to Urdu character recognition, OCR, and intelligent character recognition (ICR) can be solved as efficiently as they are solved in other languages. The previous datasets are of Latin, Arabic handwritten numerals [6], text lines of Urdu [7], integration of MNIST with Urdu numerals [8], Persian numerals [9], Bengali numerals [10], Urdu Nastalique Handwritten Dataset (UNHD) [11], Urdu Printed Text Image Database (UPTI) [12], and Urdu spoken and text words [13]. We did come across an Urdu numerals' dataset by Husnain et al. [14], but that did not incorporate variations of crumbled, torn, and ink spots on paper. So, we sought to create our novel dataset of 9800 images of handwritten Urdu numerals written by over 200 individuals with their left and right hands. The difficulty of accurately classifying handwritten characters is increased by variances in writing style, character size and form, and resemblances to other characters [5]. Hence these papers were then crumbled and torn, and some had ink spots added on them for variety. The pictures of these images were preprocessed by employing a Gaussian filter and connected component labeling. These images were then fed to machine learning and deep learning models for classification purpose. Deep learning models find out complex structures in massive datasets by using the backpropagation algorithm to indicate how a machine can change its internal parameters in each layer from representation in the previous layer [15]. We applied our proposed CNN, GoogLeNet, ResNet, and SVM on the dataset. Our proposed solution is powerful, yet simple, and results in a performance which is comparable or higher than the state-of-the-art when evaluated on our novel dataset. Since the model is deployed for real-world applications, we tested its reliability on practical applications, i.e., Pakistani currency. The chosen currency notes were 10, 20, 50, 100, 1000, and 5000. A sample of test images is shown in Figure 2.



Figure 2. Test images of Pakistani currency notes.

The main contributions of this paper are:

- Proposition of new Urdu numerals dataset that contains variations because of crumbled, torn, and ink spotted paper.
- After CNN extracts the features, we use two different activation functions: SVM and Softmax.

- The proposed CNN is compared with GoogLeNet and ResNet. The conducted experiments suggest our models' better accuracy.

So, the major advantage of our proposed work is that our data includes noise added by environment as compared to the previous datasets that were collected in simple situations. Now, our models also learn these distinctions and, hence, perform relatively well on real life examples, which was yet again missing in previous work.

The paper is divided into five sections. Section 2 elaborates the state-of-the-art techniques, our dataset collection, and other datasets available for Urdu language classifiers. The proposed model and technical details are discussed in Section 3. Section 4 provides a review of the classification results. In Section 5, the details of the experiments and their corresponding results are explained, while last section concludes the paper and presents a direction for future work.

2. Literature Review

Handwriting recognition has been around in the field of computer science for almost half a century now. In [1], the oldest techniques that have been in use for character recognition since 1959 are discussed. It originates from the work suggested by Eden in 1968 known as analysis-by-synthesis. This is the basis for syntactic approaches in character recognition. K. Gaurav and Bhatia P. K. [2] discuss the advancements in preprocessing techniques when input data ranges from simple handwritten documents to deformed images, or images having viewpoint variation and background clutter. They concluded that applying only one preprocessing technique is never enough to obtain high accuracies, but rather a mixture of preprocessing techniques is applied to obtain reliable results. Basically, two types of recognition systems exist; online systems and offline systems [16,17]. Online character recognition works when user writes on a special writing surface, computer recognizes it and converts it into codes with respect to time, while offline recognition systems are images or documents fed as input with text written on them and are converted into digital form. Offline recognition works in phases where images are first segmented, cropped, and resized, their features are extracted, and then they are classified [18]. This paper presents work on Urdu numerals using offline recognition. Table 1 presents a summary of the accuracies achieved for other algorithms using Urdu datasets. All these papers used different datasets and since those datasets are not available publicly, it was difficult for us to compare them with our techniques. Additionally, these datasets were collected on different lines and solved various problems, but underlying concept is same that they work on variations of Urdu language datasets; hence, their comparison is necessary.

Table 1. Accuracy reported on different techniques.

Article Reference	Techniques Applied	Accuracy Achieved
[19]	Back propagation neural network	90%
[8]	Kohonen self-organization maps	91%
[20]	Shape context-based digit recognition computation	93%
[21]	Fuzzy rule	97.4%
[22]	Capsule-Net	98.5%

In [23], N. Gautam, R. S. Sharma, and G. Hazrati state work done on Eastern Arabic numerals through OCR. S. Abdelazeem et al. [6] compared the problems encountered in Latin and Arabic handwritten numerals by using the Arabic Handwritten Digits Database (ADBase) [24]. H. Kour and N. K. Gondhi proposed a recognition system [19] based on approaches of segmentation for feature extraction, slant analysis for slant removal, and dictionary search for classification. It resulted in a recognition rate of 93% for isolated characters and the same for numerals. In another study [25], J. Memon, M. Sami, and R. A. Khan provided an in-depth review of statistical, kernel, artificial neural network (ANN), template matching, and structural methods for classification of OCR. In a very interesting work presented by Ahmed, S.B., Naz, S., Swati, S. et al. [7], a 6.04–7.93% error rate on

700 unique text lines was achieved (including Urdu numerals and Urdu handwritten samples) after applying 1-D bidirectional long short-term memory (BLSTM) networks. In [8], L. Javed, M. Shafi, M. I. Khattak, and N. Ullah presented the utilization of Kohonen self-organization maps on 6000 handwritten Urdu numerals and obtained an efficiency of 91%. A work similar to this paper was conducted by Saad Ahmad in [7], where Urdu text was integrated with the modified National Institute of Standards and Technology dataset (MNIST) to learn the similar nature of patterns. They used CNN and multidimensional long short-term memory (MDLSTM) on UNHD samples by pretraining the network on MNIST. Their results showed the highest recall of 0.84 and 0.93 for precision. With their roots in statistical learning theory, SVMs have been used widely for image classification and character recognition tasks, so we studied their uses in different languages. In [26], Ebrahimzadeh, R. et al. employed a linear SVM as a classifier for the MNIST dataset to obtain a 97.25% accuracy rate. Das et al. [27] extracted local features of a handwritten Bangla digits dataset using a genetic algorithm which were then fed to an SVM. It gave promising results of 96.7%. Abu et al., in [10], discussed a task-oriented model that make use of densely connected neural networks termed Bengali handwriting digit classification (BDNet). The ISI Bengali handwritten numeral dataset was used to train it. In [28], Duddela et al. discussed the task of image classification by employing NN and CNN on Devangari script. Fatemeh et al. [29] proposed a novel approach that stacks ensemble classifiers to identify handwritten numbers. They employed CNN and BLSTM that takes the probability vector of the image class as an input. The model has been tested on Arabic and Persian numerals. In [9], Savita et al. proposed a hybrid model that combines CNN that serves as a feature extractor and SVM that acts as a binary classifier.

Finally, the recent notable technique MetaQNN discussed in [30] which was put forward in 2018. It relies on reinforcement learning for the design of CNN architectures and has its roots in the neuroevolution of committees of CNN. It has an error rate of 0.44% and 0.32% when using an ensemble of the most appropriate found neural networks. Lastly, data collection covers a huge portion of this paper and major work was performed on these preprocessing of images. Ahmed, R. et al., in [31], provided the insight on how to go forward with the data collection and preprocessing. They implemented the algorithms of binarization, dots removing, and thinning which are used for our feature extraction phase.

In deep learning, the next step for an OCR-based problem is the dataset where standardization is essential to obtain exemplary results. During exploration and definition of the problem statement, no standard data set was found. We did come across a dataset containing Urdu spoken and text words by EMILLE (Enabling Minority Language Engineering) [13], which is a collection of 67 million word corpus of South Asian languages. Another ligature corpus by the Centre for Language Engineering (CLE) [18] in Pakistan has been extracted from a 19.3 million corpus gathered from different domains such as sports, news, finance, culture, and consumer information. A similar ligature corpus presented by Sabbour and Shaifat [12] is called Urdu Printed Text Image Database (UPTI) which was created along similar lines as the Arabic Printed Text Image (APTII), proposed by Slimane et al. [32]. UPTI contains 10,063 synthetically generated text lines and ligature images. Lastly, an offline dataset by Ahmed et al. [11] for Urdu text by the name of Urdu Nastalique Handwritten Dataset (UNHD) was found. It was created in 2013 by collecting samples of 8 Urdu text lines having few Urdu numerals to produce 312,000 words with 10,000 text lines.

3. Proposed Model

In this section, we present our CNN model which is trained to learn Urdu numerals. The network is trained on raw image pixels having preprocessed and cropped images of Urdu numerals. It classifies the dataset into 10 feature-mapped classes. Figure 3 elaborates these steps in the form of a block diagram and the following subsections give an insight into these steps. We start with preprocessing the set of images to get them into desired shape for all models. Our custom built CNN is applied to the final set of images. The features extracted from our CNN are fed to the Softmax activation function and SVM classifier in

parallel to obtain an in-depth review of our results. GoogLeNet and ResNet architectures are also applied on the set of images using transfer learning to compare results with our base architecture.

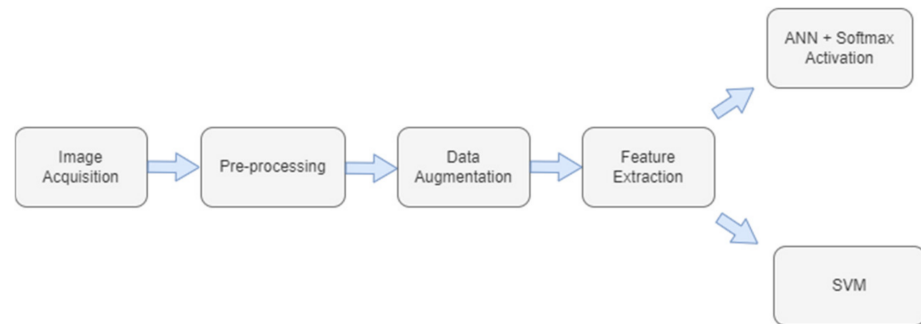


Figure 3. Block diagram.

3.1. Image Acquisition

In order to continue with our research work, we sought to make our own unique dataset as all the available datasets did not match our paper's requirements. Our dataset contains a total of 9800 images of 10 Urdu numerals written with left and right hands to create diversity. Each person wrote 0 to 9 numerals four times. These were people belonging to various age groups and different fields of life. This was done subconsciously to include people with diverse writing styles so as to bring variety to our dataset.

3.2. Preprocessing

The scanned pages of handwritten data are used for preprocessing to remove any redundant information that could be misclassified by employing connected component labeling. In order to contain different types of noise in our dataset, we crumbled some pages as shown in Figure 4, added additional dots on the page, and dropped ink spots so the classifier does not have a simple version of the dataset but instead has complex samples. These modified pages are then filtered and thresholded to maintain their maximum information. Firstly, the noise is suppressed by applying a Gaussian filter which not only subdues the effect of noise but also maintains sharp edges. A Gaussian filter with different sigma values is applied. The ideal sigma value is found to be 3, which is checked in accordance with thresholding as shown in Figure 5.



Figure 4. Actual crumbled paper.

Finally, the images are thresholded to obtain binary images. With different experiments, we found out that if the threshold value is kept low, i.e., 50, it removes the background noise completely. However, the problem is that it also lightens the boundaries of the handwritten numbers which is a great failure as shown in Figure 6. On the other hand, if a large threshold value is set, i.e., 150, it incorporates all noise in the image. The noisy dots which are closer to handwritten numbers when joined with them got mixed. This is illustrated in Figure 7 where it is difficult to distinguish the two 2s. This can result in a loss

for our data because the model would be unable to classify such numbers. So, after intense experimentation, we found that the appropriate thresholding level is 120 which retains an accurate amount of information according to Figure 8. The images are cropped in a way so as to remove maximum background and obtain images similar to MNIST. MNIST [33] is a benchmark dataset for numerals, and its images are normalized and centered in a fixed-size image. Finally, the images are resized to 32 by 32 pixels and saved in their respective folders for ease of labeling.



Figure 5. Application of Gaussian filter.

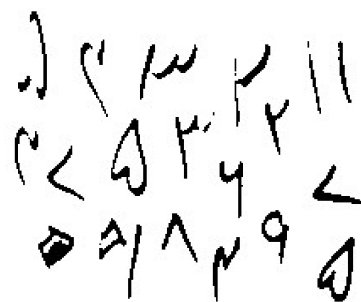


Figure 6. Low threshold level.

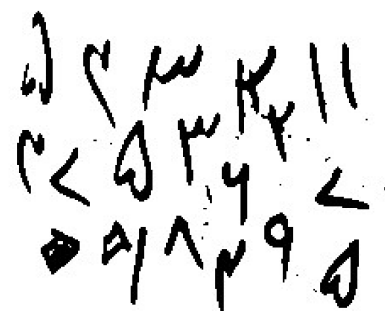


Figure 7. High threshold level.

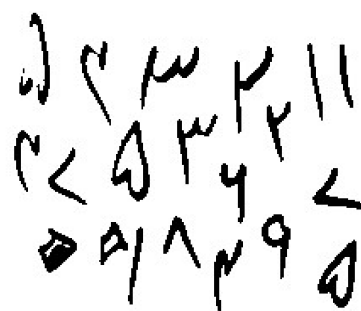


Figure 8. Ideal threshold level.

3.3. Augmentation

Data augmentation is a technique to create new data artificially so as to bring variation into the original dataset. Different techniques of augmentation (such as flipping, rotating, and adding noise) are applied on separate numbers because some numerals upon these transformations change into another number and could not preserve their respective labels [34]. For instance, the numbers zero, one, and five are flipped and labelled as is because they are unaffected by flipping.

However, when digit eight is rotated in a 45° counterclockwise direction it becomes Urdu digit seven. Thus, it is then labelled as seven. Similarly, digit seven is rotated 45° in clockwise direction to resemble digit eight as shown in Figure 9a. As Urdu digit two and six are similar in terms of shape and are counter flips of each other, the digit two is flipped and labelled as six, while digit six is flipped to be labelled as two. Deep research is performed regarding these transformations as a slight error could change the class of data. For example, digit six could change into digit two upon a vertical flip as shown in Figure 9b. For rest of the numbers, three, four, and nine, noise is added to them which is depicted in Figure 9c. As a result of augmentation, our dataset increased to almost 13,000.

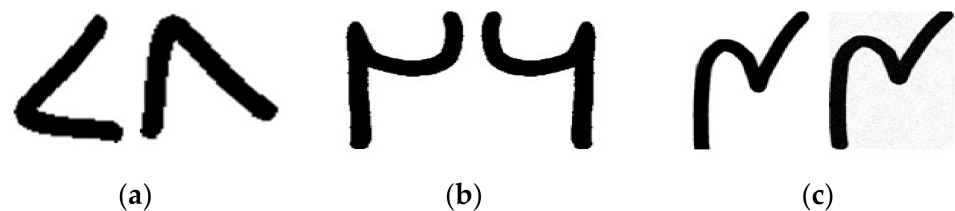


Figure 9. Sample of (a) rotated image, (b) flipped image, and (c) noisy image.

3.4. Feature Extraction

3.4.1. Convolutional Neural Network

We propose a dataset of Urdu handwritten numerals with 10 labels and 13,000 images in all. Initially, all the images are standardized by dividing current pixel value by sum of all pixels. This represents image pixels in range of 0 to 1. Then, each image is resized to new dimensions while ensuring that no information is lost. The CNN model consists of four core substructures which are used repeatedly with different nonlinearities to bring the best results.

1. The input layer contains raw pixel values, and, in this case, each image of the size $32 \times 32 \times 3$ pixels is fed to the CNN. Here, 32 represents the width and height of the image while 3 is the color channels—red, green, and blue.
2. The convolution layer connects local receptive field of the input with neurons in the next layer. This is achieved through a simple dot product of kernel and input image. A kernel size of 3×3 is maintained throughout the model, whereas padding is set to 1. It is followed by batch normalization of convolution layer. Each output of convolution layer uses the ReLU activation function followed by pooling layer. ReLU activations work better than sigmoid function in terms of gradient vanishing problems. ReLU was picked out of other nonlinearities (e.g., tanh, sigmoid) after comparing their results in our CNN model. Batch normalization is applied after each convolution layer to improve generalization [35].
3. The pooling layer down samples input along spatial dimensions. One of the most famous pooling layers is ‘Max Pooling’ which is used here to extract the highest pixel value in the current space. These extracted features are then fed to the classifiers which are discussed further. Figure 10 depicts the architecture of our proposed model and Table 2 provides an analysis of required computation resources and learning parameters.

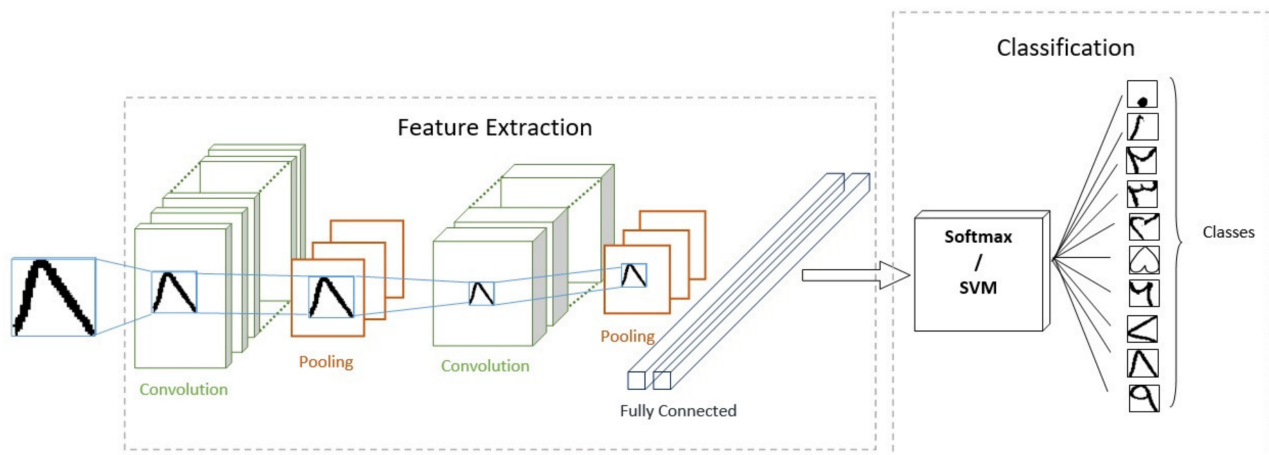


Figure 10. Architecture of CNN for Urdu handwritten numeral recognition.

Table 2. Summary of proposed CNN model.

Layer	Output Shape	Number of Parameters
Conv	$32 \times 32 \times 256$	7168
Batch Norm	$32 \times 32 \times 256$	1024
MaxPool	$16 \times 16 \times 256$	0
Conv	$16 \times 16 \times 128$	295,040
Batch Norm	$16 \times 16 \times 128$	512
MaxPool	$8 \times 8 \times 128$	0
Flatten	8192	0
Dense	90	737,370
Dense	64	5824
Dense	10	650
Total parameters		1,047,588
Trainable parameters		1,046,820
Nontrainable parameters		768

3.4.2. GoogLeNet

GoogLeNet [36] is based on the idea of an inception layer that covers a large area but maintains fine resolution on a dataset for small information. Because GoogLeNet achieved the top 5 error rate of 6.67%, we used it to train the Urdu numeral dataset. A major task was to tune the three main parameters learning rate, number of epochs, and batch size. A batch size of 32 with a learning rate of 0.001 gave us the best results for 30 epochs. The learning curve for GoogLeNet is shown in Figure 11 which does not show underfitting or overfitting as both losses reached a point of stability. This is an exceptionally good result for a novel dataset like ours.

3.4.3. ResNet

The intuition behind ResNet is that deep neural networks are hard to train due to their huge number of layers, especially where the problem of vanishing gradient occurs [37]. ResNet50 is the variant that is used in this paper. It is 50 layers deep as the name indicates and its pretrained version from the ImageNet dataset is used. Images of size 224×224 with 3 color channels were used. For ResNet, batch size and epochs were used as tunable parameters. Here, a batch size of 16 with a learning rate of 0.001 and 40 epochs achieved the best results. The learning curves show a good generalization between training and validation data on the Urdu numeral dataset. They are plotted in Figure 12.

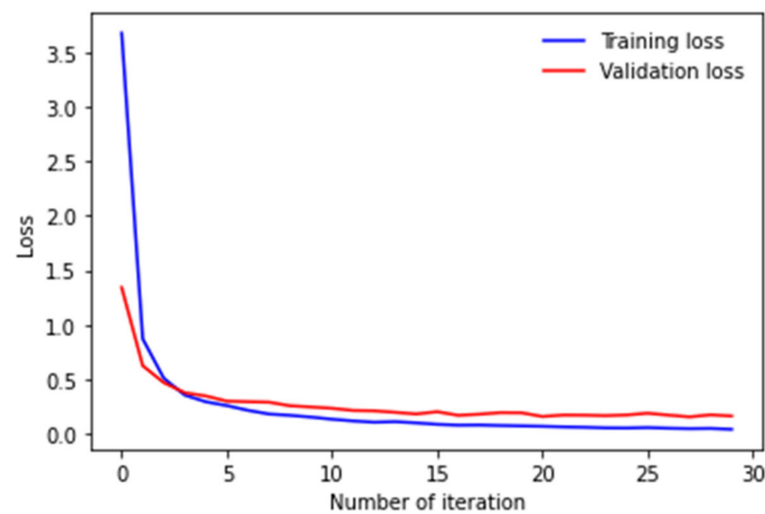


Figure 11. Training and validation learning curves on GoogLeNet.

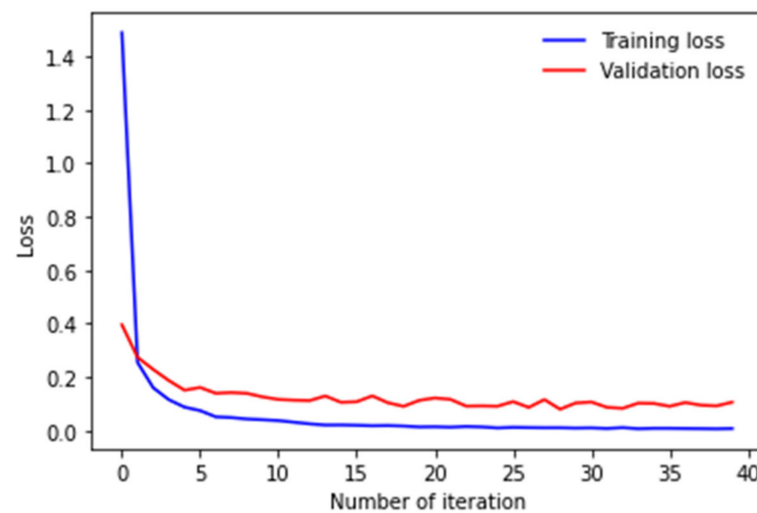


Figure 12. Training and validation learning curves on ResNet.

4. Classification

For classification purposes, SVM and Softmax activation functions are used. The features extracted from CNN are fed to these classifiers separately to manipulate their different results. First, we apply the Softmax activation function on the features extracted from the CNN which classifies the output into a probability distribution of 10 classes. Its function is given as follows:

$$\text{softmax}_j = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (1)$$

It computes the exponential of the input parameter and the sum of the exponential parameters of all existing values in the inputs while giving output in the ratio of the exponential of the parameter and the sum of the exponential parameter. The learning curve for our CNN model in Figure 13 does not show either underfitting or overfitting. The training curve shows how well the model is learning while the validation curve shows its rate of generalization. The loss is lower on the training set as compared to the validation set. It can be concluded that it is a good fit as both losses decrease to a point of stability and the gap between them is negligible.

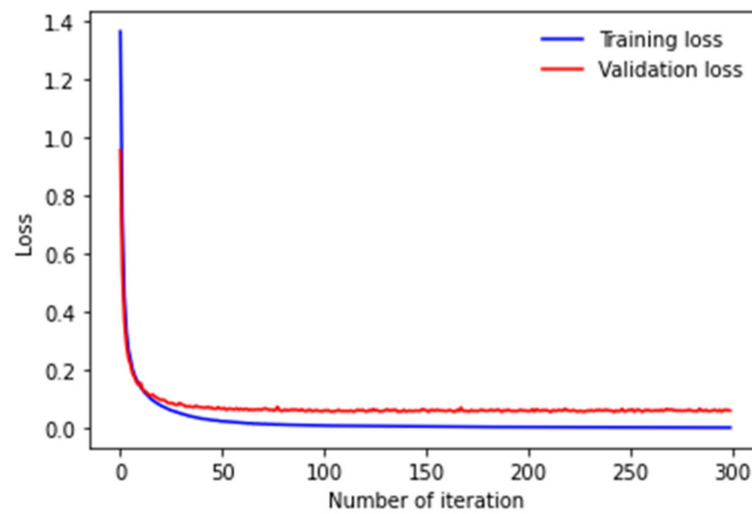


Figure 13. Training and validation learning curves for Softmax activation function.

Then, the SVM is applied to the same extracted features. The SVM is a supervised classification algorithm that works on features extracted from images rather than raw images. Its training equation is given as: set of attributes–label pairs (x_i, y_i) , $i = 1, 2, \dots, l$:

$$\begin{aligned} & \text{minimize} \quad w^T w + C \sum_{i=1}^m \xi_i^2 \\ & \text{subject to} \quad y_i (x_i^T w + b) \geq 1 - \xi_i, \quad (i=1, \dots, m) \end{aligned} \quad (2)$$

Studies have shown that it works better for classification as compared to the traditional Softmax function but fails for multiclass problems. This is validated in our results as shown in Figure 14 which is a good generalization curve.

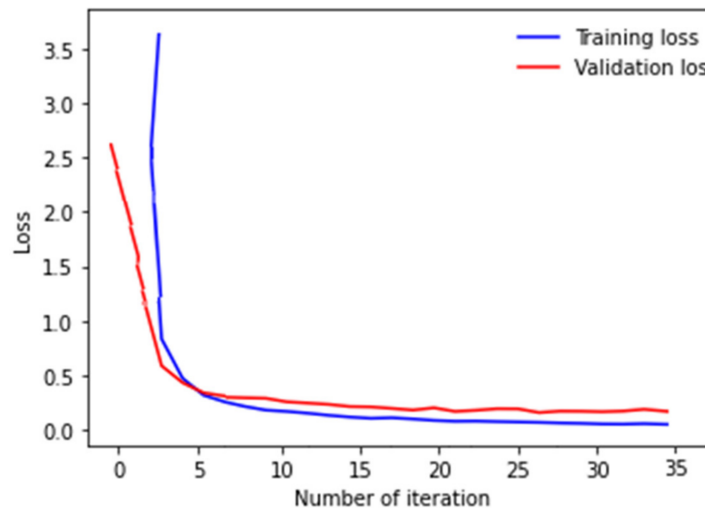


Figure 14. Training and validation learning curves on SVM.

5. Experiment and Discussion

To validate the accuracy of our proposed model and two architectures (GoogLeNet and ResNet), we used our novel dataset of Urdu numerals. They were compared for performance on different datasets and a different number of labels. For all our models, we used two types of datasets.

1. One is the original one that we proposed initially, and we evaluated it by splitting it into 85–15 ratio.

2. The other one is made as a separate set consisting of Pakistani currency note images which are used for testing only but are trained using the Urdu dataset. This is done to test our models on real-life scenarios.

Batch size, number of epochs, and learning rate are used for hyperparameter tuning. Different batch sizes of 16, 32, 64, and 128 are checked for each of the three models with different types of regularization. For our own CNN and ResNet, batch sizes of 16 gave the best results, while for GoogLeNet, 32 gave optimal results. A learning rate of 0.001 is used for CNN, SVM, and ResNet, while a 0.003 learning rate is used for GoogLeNet. Stochastic gradient descent (SGD) with momentum 0.9 worked best as compared to other types of optimizations such as Adamax, Adam. In CNN, it is observed that training deep neural networks for more layers brought sensitivity to weights and settings of the learning algorithm. To solve this, batch normalization is employed to standardize the layers for each batch. This also caused the number of epochs to reduce which in turn stabilized the learning process [36]. However, it also brought an increase in training time because of the use of an optimizer—SGD. Both batch normalization and dropout are used in the CNN after much hyperparameter tuning. It is concluded that combinations of the above values of hyperparameters produced maximum accuracy with the Urdu numeral dataset [38]. We tried to use dropout too, but it reduced the accuracy, so we relied on batch normalization only. On the other hand, a dropout rate of 0.5 was used for SVM to obtain the best results.

The performance graphs indicate that the increased accuracy is because of more neurons in more layers. These neurons helped in choosing the features for a dataset in a deep manner. It is quite noteworthy that our proposed CNN gave the best results equivalent to ResNet and GoogLeNet as shown in Figure 15. The combination of CNN with SVM gave the best accuracy of 99.96% with a promising validation accuracy of 99%. These are ground breaking results for a dataset that is unique. In its comparison training, the accuracy of the CNN with the Softmax classifier is 98.89% which is equally promising. GoogLeNet showed a training accuracy of 99.06%, ResNet showed a training accuracy of 99.88%. The validation accuracy of CNN with softmax classifier is 98.41% and 96.4% for ResNet, while that of GoogLeNet is comparatively lower at 95.61%. These accuracies conclude that our custom built CNN with SVM outperformed all the available models.

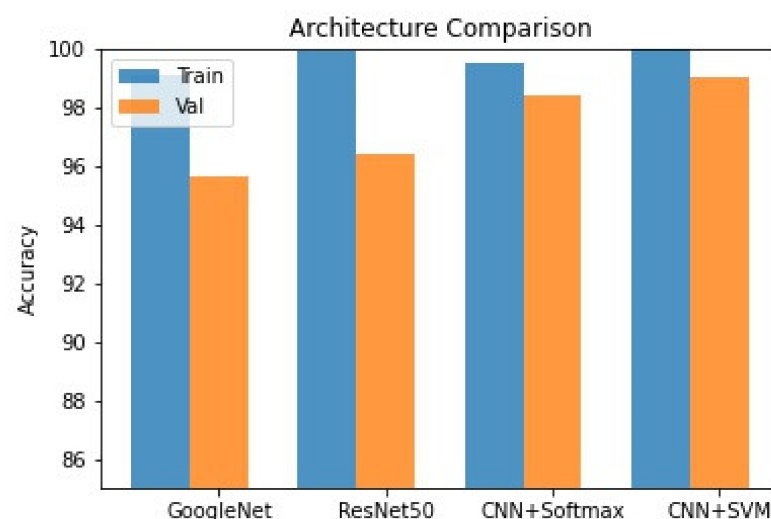


Figure 15. Comparison of accuracies on 4 proposed architectures.

5.1. Comparison with Existing Methods

In order to authenticate our results, we compared the performance of our data with the previously published model by Husnain, M. and Saad Missen et al. [39]. We tried 16, 32, 40, 60, and 80 neurons in fully connected layers with SGD and batch sizes of 32, 64, and 128. Momentum was varied between 0.7 and 0.99. Adamax with batch sizes of 32, 64, and 128 was checked to obtain the best results. We achieved an accuracy of 95.7% on their

model and its variants. However, we achieved the final test accuracy of 99.0% using SVM as a classifier on our dataset as compared to their test accuracy of 98.41% on their Urdu numeral dataset. This proves that classification using SVM results in better accuracies for the features extracted for Urdu numerals. A brief comparison of accuracies of previously published papers is given in Table 3. As is evident from the results, our models beat the previous techniques with remarkable accuracies.

Table 3. Comparison of handwritten Urdu numerals recognition on different classifiers.

Systems	Dataset	Classifier	Accuracy Achieved
[11]	UNHD(Urdu characters and ligatures)	BLSTM	93.96%
[40]	Sindhi handwritten numbers	Self-organizing map neural network	86.89%
[21]	Handwritten Urdu numerals	Rule based technique, HMM	97.4% (Rule based technique), 96.2% (HMM)
[41]	Handwritten Urdu numerals	Daubechies wavelet	92.05%
[42]	Urdu handwritten characters and numerals	Convolutional neural network	98.3%
[26]	Urdu handwritten characters and numerals	Convolutional neural network	98.3%
Our Approach	Handwritten Urdu numerals	GoogLeNet and ResNet	95.7%
Our Approach	Handwritten Urdu numerals	Feature extraction using convolution layer and Softmax Activation for classification	98.41%
		Feature extraction using convolution layer and SVM for classification	99.3%

Additionally, in comparison to the previous paper [33], in terms of test accuracy, our approach achieved 99.0% as compared to their accuracy of 98.3%, which is optimal considering that the Urdu numeral dataset is novel. This dataset shows promising training and validation accuracies on GoogLeNet, ResNet, and the proposed approach as shown in Figure 15.

Additionally, we experimented with our model on a novel Pakistani currency dataset and achieved a test accuracy of 89.41%, which further validates the performance and robustness of our model on real-world problems.

5.2. Expanded Testing Set

As our model is tested on real-life examples thus, it is deemed necessary to validate our results with a separate dataset. This dataset consists of 90 images captured with the same tools as were used for the training data. The preprocessing techniques similar to training data were applied to it. The only difference here is that it contained 4 classes of zero, one, two, and five instead of 10 classes. So, during training and testing of the currency dataset, the training data for Urdu numerals was minimized to the same 4 classes. The same CNN network was used as proposed in Figure 10. An accuracy of 97.97% was achieved on the training set along with 89.41% on the test set. To obtain the best results, we retuned the hyperparameters of the model. We chose different batch sizes ranging from 16 to 128 to find the one where we achieved the best results while keeping the learning rate at 0.001. Furthermore, the number of epochs was varied accordingly which resulted in getting momentum 0.7 with SGD.

Along with CNN, ResNet, GoogLeNet, and SVM were also applied to this expanded test set to obtain an insight into their accuracies and losses. The parameter batch size was kept to 32 for both ResNet and GoogLeNet. The bar graph in Figure 16 shows that our proposed CNN model with the SVM worked best on a real-life dataset with a validation accuracy of 91%. On the other hand, the validation accuracies of GoogLeNet (53.47%), ResNet50 (64.24%), and Softmax (89.41%) are worth considering. As we worked

on particular classes instead of all the 10 classes and the data within these 4 classes was also less in quantity, so this caused a reduction in accuracies for real-life data.

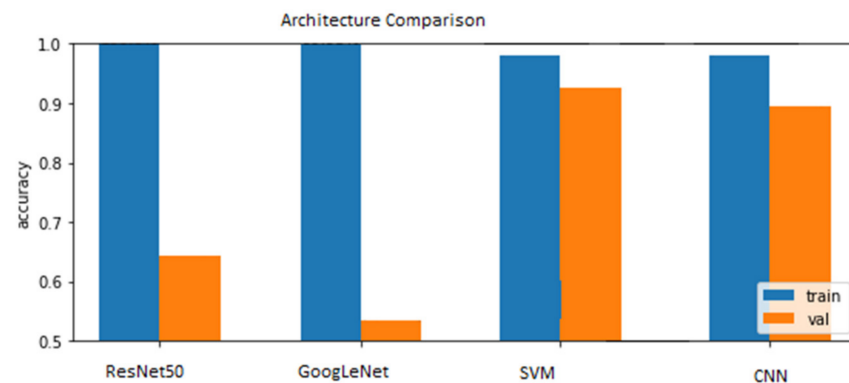


Figure 16. Bar plot for comparison of accuracies on real-life example.

6. Conclusions and Future Work

In this paper, we proposed two approaches to classify novel datasets of Urdu numerals. The first approach extracts features using convolution layers and uses Softmax activation followed by fully connected layers for classification. The second approach applies an SVM classifier to the features extracted from the convolution layer. All the models give best results in terms of accuracy where the first approach provides a validation accuracy of 98.41%, while an accuracy of 99.0% is achieved by the second approach. The accuracy of 96.4% on ResNet and 95.61% on GoogLeNet is achieved on this novel dataset. We tested these models on Pakistani currency to see their reliability in real-world application after being trained on our dataset. To implement this, we developed another dataset from Pakistani currency notes and evaluated our proposed models with it. Our handwritten Urdu numerals dataset is unique and any such dataset is not available publicly. This hinders research in the domain of the Urdu language. Moreover, our dataset is refined and is collected along the lines of the MNIST dataset, so it provides the best results with real-life problems as shown in our paper.

In the future, we plan on increasing the Urdu numeral dataset and then making it publicly available so as to motivate researchers to work in this field. Increasing this dataset will also increase the accuracies of all models. Additionally, our dataset and CNN can help develop a system to identify and count currency notes. Since the performance of deep learning algorithms in real-world applications is of utmost importance, we plan on testing it on other applications such as recognizing the Surah numbers of The Holy Quran and numbers on Pakistani postage stamps. The sole motivation of this paper is to bring our mother language Urdu to a competitive level with all the latest research.

Author Contributions: Conceptualization, W.K. and S.K.; methodology, A.A. (Ameera Arif) and A.B.; software, A.B.; validation, W.K., A.u.R., B.K. and A.A. (Ahmed Ali); data curation, A.B., A.A. (Ameera Arif) and W.K.; writing—original draft preparation, A.B.; writing—review and editing, A.A. (Ameera Arif), W.K., A.u.R. and B.K.; visualization, A.A. (Ameera Arif) and S.K.; supervision, A.A. (Ahmed Ali) and S.K.; funding acquisition, A.u.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Singh, R.; Mishra, R.K.; Bedi, S.; Kumar, S.; Shukla, A.K. A Literature Review on Handwritten Character Recognition based on Artificial Neural Network. *Int. J. Comput. Sci. Eng.* **2018**, *6*, 753–758. [CrossRef]
2. The Online Encyclopedia of Writing Systems and Languages. Available online: <https://www.omniglot.com/writing/urdu.htm> (accessed on 9 June 2020).
3. Spitz, A.L.; Andreas, D. Document Analysis Systems. In *Proceedings of the International Association for Pattern Recognition Workshop*; World Scientific: Singapore, 1995; pp. 237–292.
4. Sharif, M.; Ul-Hasan, A.; Shafait, F. Urdu Handwritten Ligature Generation Using Generative Adversarial Networks (GANs). In *Proceedings of the Frontiers in Handwriting Recognition: 18th International Conference, ICFHR 2022, Hyderabad, India, 4–7 December 2022*; Springer-Verlag: Berlin/Heidelberg, Germany, 2022; pp. 421–435. [CrossRef]
5. Misgar, M.M.; Mushtaq, F.; Khurana, S.S.; Kumar, M. Recognition of offline handwritten Urdu characters using RNN and LSTM models. *Multimed. Tools Appl.* **2022**, *82*, 2053–2076. [CrossRef]
6. Gautam, N.; Sharma, R.S.; Hazrati, G. Eastern Arabic Numerals: A Stand out from Other Jargons. In *Proceedings of the International Conference on Computational Intelligence and Communication Networks (CICN)*, Jabalpur, India, 12–14 December 2015; pp. 337–338. [CrossRef]
7. Memon, J.; Sami, M.; Khan, R.; Uddin, M. Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). *IEEE Access* **2020**, *8*, 142642–142668. [CrossRef]
8. Khan, S. A Mechanism for Offline Character Recognition. *Int. J. Res. Appl. Sci. Eng. Technol.* **2019**, *7*, 1086–1090. [CrossRef]
9. Haghighi, F.; Omranpour, H. Stacking ensemble model of deep learning and its application to Persian/Arabic handwritten digits recognition. *Knowl. Based Syst.* **2021**, *220*, 106940. [CrossRef]
10. Das, N.; Sarkar, R.; Basu, S.; Kundu, M.; Nasipuri, M.; Basu, D.K. A genetic algorithm based region sampling for selection of local features in handwritten digit recognition application. *Appl. Soft Comput.* **2012**, *12*, 1592–1606. [CrossRef]
11. Slimane, F.; Kanoun, S.; Hennebert, J.; Alimi, A.; Ingold, R. A study on font-family and font-size recognition applied to Arabic word images at ultra-low resolution. *Pattern Recognit. Lett.* **2013**, *34*, 209–218. [CrossRef]
12. Center for Language Engineering Urdu Ligatures from Corpus Page. Available online: http://www.cle.org.pk/software/ling_resources/UrduLigaturesfromCorpus.htm (accessed on 11 June 2020).
13. Ahmed, R.; Musa, M. Preprocessing Phase for Offline Arabic Handwritten Character Recognition. *Int. J. Comput. Appl. Technol. Res.* **2016**, *5*, 760–763. [CrossRef]
14. Borse, R.; Ansari, I.A. *Offline Handwritten and Printed Urdu Digits Recognition using Daubechies Wavelet*; ER Publication: New Delhi, India, 2015.
15. Kumar, G.; Bhatia, P.K. Analytical Review of Preprocessing Techniques for Offline Handwritten Character Recognition. In *Proceedings of the 2nd International Conference on Emerging Trends in Engineering Trends in Engineering and Management ICETEM*, Rohtak India, 21–22 July 2013.
16. Akhtar, P. An Online and Offline Character Recognition Using Image Processing Methods—A Survey. *Int. J. Commun. Comput. Technol.* **2016**, *4*, 102. [CrossRef]
17. Liu, C.; Yin, F.; Wang, D.; Wang, Q. Online and offline handwritten Chinese character recognition: Benchmarking on new databases. *Pattern Recognit.* **2012**, *46*, 155–162. [CrossRef]
18. Baker, P.; Hardie, A.; McEnery, T.; Cunningham, H.; Gaizauskas, R.J. EMILLE, A 67-Million Word Corpus of Indic Languages: Data Collection, Mark-up and Harmonisation. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02) LREC*, Las Palmas, Canary Islands Spain, 29–31 May 2002.
19. Javed, L.; Shafi, M.; Khattak, M.I.; Ullah, N. Hand-written Urdu Numerals Recognition Using Kohonen Self Organizing Maps. *Sindh Univ. Res. J. SURJ* **2015**, *47*, 403–406.
20. Razzak, M.I.; Hussain, S.A.; Belaid, A.; Sher, M. Multi-font Numerals Recognition for Urdu Script based Languages. *Int. J. Recent Trends Eng.* **2009**.
21. Kour, H.; Gondhi, N.K. Machine Learning approaches for Nastaliq style Urdu handwritten recognition: A survey. In *Proceedings of the 6th Communication International Systems Conference (ICACCSon) Advanced*, Coimbatore, India, 23 April 2020; pp. 50–54. [CrossRef]
22. Yusuf, M.; Haider, T. Recognition of Handwritten Urdu Digits using Shape Context. *INMIC* **2004**. [CrossRef]
23. Iqbal, T.; Ali, H.; Saad, M.M.; Khan, S.; Tanougast, C. CapsuleNet for Urdu Digits Recognition. In *Proceedings of the 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, Metz, France, 18–21 September 2019.
24. Abdelazeem, S. Comparing Arabic and Latin Handwritten Digits Recognition Problems. *Int. J. Comput. Inf. Eng.* **2009**, *3*, 1583–1587. [CrossRef]
25. Abdelazeem, S.; El-Sherif, E. The Arabic Handwritten Digits Databases ADBase & MADBase. Available online: <http://datacenter.aucegypt.edu/shazeem/> (accessed on 14 May 2020).
26. Ahmed, S.B.; Hameed, I.A.; Naz, S.; Razzak, M.I.; Yusof, R. Evaluation of Handwritten Urdu Text by Integration of MNIST Dataset Learning Experience. *IEEE Access* **2019**, *7*, 153566–153578. [CrossRef]
27. Ebrahimzadeh, R.; Jampour, M. Efficient Handwritten Digit Recognition based on Histogram of Oriented Gradients and SVM. *Int. J. Comput. Appl.* **2014**, *104*, 10–13. [CrossRef]

28. Sufian, A.; Ghosh, A.; Naskar, A.; Sultana, F.; Sil, J.; Hafizur Rahman, M.M. BDNet: Bengali Handwritten Numeral Digit Recognition based on Densely connected Convolutional Neural Networks. *J. King Saud Univ. Comput. Inf. Sci.* **2020**, *34*, 2610–2620. [CrossRef]
29. Prashanth, D.S.; Mehta, R.V.K.; Sharma, N. Classification of Handwritten Devanagari Number An analysis of Pattern Recognition Tool using Neural Network and CNN. *Procedia Comput. Sci.* **2020**, *167*, 2445–2457. [CrossRef]
30. Ahlawat, S.; Choudhary, A. Hybrid CNN-SVM Classifier for Handwritten Digit Recognition. *Procedia Comput. Sci.* **2020**, *167*, 2554–2560. [CrossRef]
31. Baldominos, A.; Saez, Y.; Isasi, P. Evolutionary Convolutional Neural Networks: An Application to Handwriting Recognition. *Neurocomputing* **2018**, *283*, 38–52. [CrossRef]
32. Sabbour, N.; Shafait, F. A segmentation-free approach to Arabic and Urdu OCR. In Proceedings of the SPIE 8658, Document Recognition and Retrieval XX, 86580N, Burlingame, CA, USA, 3–7 February 2013. [CrossRef]
33. Ahmed, S.; Naz, S.; Swati, S.; Razzak, M. Handwritten Urdu character recognition using one-dimensional BLSTM classifier. *Neural Comput. Appl.* **2017**, *31*, 1143–1151. [CrossRef]
34. LeCun, Y. The MNIST DATABASE of handwritten digits. Available online: <http://yann.lecun.com/exdb/mnist/> (accessed on 14 June 2020).
35. Shorten, C.; Khoshgoftaar, T. A survey on Image Data Augmentation for Deep Learning. *J. Big Data* **2019**, *6*, 60. [CrossRef]
36. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
37. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9. [CrossRef]
38. Prabhu, R. CNN Architectures—LeNet, AlexNet, VGG, GoogLeNet and ResNet. Available online: <https://medium.com/@RaghavPrabhu/cnn-architectures-lenet-alexnet-vgg-googlenet-and-resnet-7c81c017b84> (accessed on 14 June 2020).
39. Garbin, C.; Zhu, X.; Marques, O. Dropout vs. batch normalization: An empirical study of their impact to deep learning. *Multimed. Tools Appl.* **2020**, *79*, 12777–12815. [CrossRef]
40. Husnain, M.; Missen, M.M.S.; Mumtaz, S.; Jhanidr, M.Z.; Coustaty, M.; Muzzamil Luqman, M.; Ogier, J.M.; Choi, G.S. Recognition of Urdu Handwritten Characters Using Convolutional Neural Network. *Appl. Sci.* **2019**, *9*, 2758. [CrossRef]
41. Chandio, A.A.; Jalbani, A.H.; Leghari, M.; Awan, S.A. Multi-Digit Handwritten Sindhi Numerals Recognition using SOM Neural Network. *Mehran Univ. Res. J. Eng. Technol.* **2017**, *36*, 8. [CrossRef]
42. Malik, S.; Khan, S.A. Urdu online handwriting recognition. In Proceedings of the IEEE Symposium on Emerging Technologies, Islamabad, Pakistan, 18 September 2005. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.