Chapter 14
Implementation of Residual Network (ResNet) for Devanagari Handwritten Character Recognition

Mandar Mhapsekar, Prathamesh Mhapsekar, Aniket Mhatre and Vinaya Sawant

1 Introduction

The study in the field of optical character recognition can be traced back to mid-1940s and has ever since gaining the attention of various industries and sectors. Optical character recognition has been highly used in banks, post offices, libraries, and publishing houses. The main challenge in OCR is handwritten character recognition. The research on handwritten character recognition began in the late 1960s, and at that time, only the handwritten numeric characters were addressed by the system [1]. Over the years, the technological approach for solving this problem has developed, thus improving the accuracy of the system [2]. Handwritten OCR for the English language which comes under the Latin script has almost developed into a full-fledged system. The research on handwritten OCR for Devanagari script is very limited compared to Latin script. Devanagari script includes languages like Hindi, Marathi, and Nepali. The most widely used Devanagari script language is Hindi with over 500 million people using this language [3].

In Devanagari script, there are 13 vowels and 36 consonants as shown in Fig. 1. There are 14 modifiers in Devanagari, out of which 11 are of vowels and 3 are of

M. Mhapsekar · P. Mhapsekar · A. Mhatre (✉) · V. Sawant
Department of Information Technology, Dwarkadas J. Sanghvi College of Engineering, Vile Parle, Mumbai, India
e-mail: mhatreaniket121@gmail.com

M. Mhapsekar
e-mail: mhapsekarmandar@live.com

P. Mhapsekar
e-mail: prathmesh1297@gmail.com

V. Sawant
e-mail: vinaya.sawant@djsce.ac.in

H. Vasudevan et al. (eds.), *Advanced Computing Technologies and Applications*, Algorithms for Intelligent Systems,
https://doi.org/10.1007/978-981-15-3242-9_14

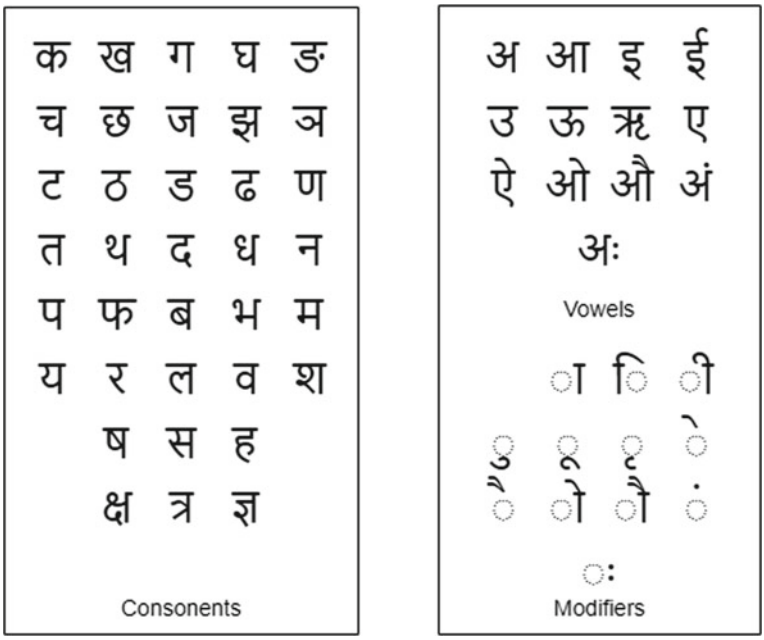

Fig. 1 Devanagari characters

'rakars.' These modifiers are combined with the consonants to form a modified character (the character with a modifier). Apart from vowels, consonants, and modified characters, we also have compound characters. Compound characters are formed by combining two or more simple characters. The compound characters are more complex in structure than the simple characters. There are 10 digits in Devanagari script. Devanagari is written from left to right, and there is no concept of uppercase/lowercase. Figure 2 shows a word in Devanagari script. Every character in Devanagari consists of a line above it which is called as the 'Shirorekha.' A character in Devanagari is divided into four sections, namely the top section, main section, side section, and the bottom section. The top section is the above the Shirorekha which consist of some modifier which is known as the upper modifier; the main section consists of the main character; the side section and the bottom section also

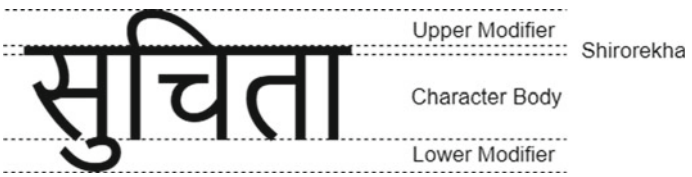

Fig. 2 Devanagari word

consist of some modifier which is known as the side modifier and the lower modifier, respectively [2].

The use of multilayer perceptron network is considered as a milestone in the field of handwritten character recognition but it needs a good feature extractor to extract relevant features, in which the multilayer perceptron can work to classify the character [4]. A better approach to this is to use a deep neural network. Convolutional neural network (CNN) is one of the classes of deep neural network. It does not require a feature extractor. It has an inbuilt feature extractor that works directly on the image and extracts the best feature from it for the classification [4]. In CNN, the classification accuracy increases as we increase the number of layers in the network; but at one point, above which if we increase the number of layers, the accuracy will start to saturate and eventually degrade. This is caused due to the vanishing gradient problem so it seems like the shallower network performs better than the deeper network. This problem is called the degradation problem [5]. Residual network (ResNet) was introduced to solve this problem. In ResNet, we have shortcut connections. Shortcut connections are those connections that skip one or more layers. The shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers [6].

In this paper, we have used residual network for Devanagari handwritten character recognition and showed through experiment how much the accuracy of the classification increases by using ResNet compared to the current state-of-the-art method.

The paper is organized into four sections. Section 1 gives an introduction to the paper. Section 2 deals with the work related to Devanagari handwritten character recognition. Section 3 contains the details about residual network which is the proposed approach in the paper. Experiments and results comparing the residual network with the current state-of-the-art method are shown in Sect. 4.

2 Related Work

In this section, we have summarized various techniques and methods used in Devanagari handwritten character recognition over the years which have given good results and performances.

2.1 Support Vector Machine (SVM)

Support vector machine is a supervised machine learning classification algorithm which when provided with labeled training data outputs a hyperplane that categorizes the new data into different classes. In two-dimensional space, this hyperplane is defined as a line dividing a plane into two parts, wherein each class lays on either side [7]. SVM requires an explicit feature extractor which extracts the features from

an image and produces a feature vector which is used by the SVM classifier. The paper [8] has used SVM as a classifier with Zernike moment as a feature extractor. It achieved an accuracy of 98.37% using their own database consisting of 9600 characters.

2.2 *Artificial Neural Network (ANN)*

Artificial neural network is a collection of nodes called as artificial neurons which resembles the neurons in the human brain. These neurons are connected to each other. ANN consists of three layers: the input, intermediate hidden layer, and the output layer. Each connection in the network has a weight associated with it. These connection weights are updated until the network is able to perform the task for which it is trained using a method called backpropagation [9]. ANN requires an explicit feature extractor like HOG, Zernike moment.

In HOG feature extractor, the distribution of directions of the gradient is used as a feature. The gradient is useful because the gradient value is high at the edges and corners. In HOG, the image is divided into cells where in each cell we calculate the magnitude and direction of gradients. Histogram of each cell is calculated based on the magnitude and direction of the gradient. A specific group of cells is combined into blocks in which normalization is performed. After normalization, values in the block are combined to form a single feature vector [10]. The features extracted by the extractor are applied to any classifier like ANN and SVM for the classification. The paper [10] has implemented HOG as a feature extractor along with ANN as a classifier. It achieved an accuracy of 82.66% using the ISI handwritten character database with input image of size 32×32.

2.3 *Bidirectional Long Short-Term Memory (BLSTM)-Based Recognition*

BLSTM is used in RNN. Unlike ANN and CNN, RNN is not a feedforward neural network in which the data flow in one direction from input to output one layer at a time. In RNN, the output of the layer is added to the next input and fed back to the same layer. In LSTM, the node in RNN is replaced by an LSTM cell which has the ability to remember or forget previous contexts by using several gates. Similarly, BLSTM is bidirectional LSTM in which the learning sequence is in forward as well as backward direction [11]. BLSTM is a classifier so it needs a feature extractor to provide feature vector as input to it. BLSTM can be used with CNN as well as HOG-based feature descriptor which is proposed in the paper [10]. It has achieved accuracy of 94.56% and 79.54 with CNN and HOG as a feature extractor, respectively. It used ISI handwritten character database with input image of size 32×32.

2.4 Convolutional Neural Network (CNN)

Convolutional neural network is a class of deep neural network, which takes an input image, assigns importance to various aspects in the image and be able to differentiate one from the other. CNN does not require an explicit feature extractor. In CNN, the first operation performed is the convolutional operation which is performed by the convolutional layer. In this layer, it has different feature detector/filter/kernel to detect features from the image by performing the convolutional operation. Thus, the output of this layer is different feature maps for each feature detector. Then it uses an activation function like ReLU to maintain the nonlinearity in the image. Next step is max pooling to make the model flexible enough to find the feature from the image even in an improper condition of the image. Then it performs flattening to make the feature in single vector form. At the last, it creates a full connection layer (ANN) for the classification of the image [12]. The paper [4] has proposed the use of CNN for Devanagari handwritten character recognition. It achieved an accuracy of 98.47% using the Devanagari handwritten character dataset with input image of size 32 × 32.

3 Proposed Approach

In this section, we have given details of residual network which is the method we have used for Devanagari handwritten character recognition.

3.1 Residual Network (ResNet)

Over the years, deep convolutional neural networks have made a series of breakthroughs in the field of image recognition and classification. Networks are going deeper to solve more complex tasks but due to the problem like vanishing gradient, if we have a sufficiently deeper network, it may not be able to learn even the simpler problems. If we keep increasing the layers of a model, at one point the accuracy will start to saturate and eventually degrade. This is called as the degradation problem [5].

He et al. [13] first demonstrated the depth problem and proposed a remarkable solution which has since allowed the training of over 2000 layers with increasing accuracy. Residual network consists of residual blocks. Figure 3 shows the structure of the residual block. In Fig. 3, we can see some layers using skip connection. Consider a neural network, which has x as input and approximates $H(x)$. Let us denote the difference between these as $R(x)$ whose equation is given below

$$R(x) = H(x) - x \tag{3.1.1}$$

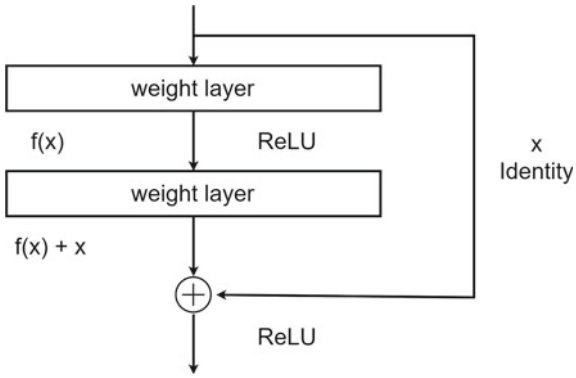

Fig. 3 ResNet block

$R(x)$ is a residual function. If one hypothesizes that multiple nonlinear layers can asymptotically approximate complicated functions, then it is equivalent to hypothesize that they can asymptotically approximate the residual functions. We can see that the layers in the residual block are trying to learn the residual function $R(x)$. From this, we get an equation.

$$F(x) = H(x) - x \tag{3.1.2}$$

So the original function becomes $F(x) + x$ which are evident in Fig. 3. Because of these skip connections, we can propagate larger gradients to initial layers and these layers also could learn as fast as the final layers, giving us the ability to train deeper networks, solving the problem of vanishing gradient [6, 9].

4 Experiments and Results

In this section, we compared the current state-of-the-art method for Devanagari handwritten character recognition which is the convolutional neural network with the proposed method of residual network. We have obtained results corresponding to the various architectures of ResNet and CNN which are described in detail below.

4.1 Dataset

The dataset used for the experiment is Devanagari handwritten character dataset (DHCD) which is the work of Acharya [4]. The dataset contains 92,000 images of handwritten Devanagari characters. The dataset comprises 46 classes out of which 36 are alphabets and 10 are numbers. There are total of 2000 sample images of each character. Each character image is of size 32 by 32 pixels where 28 by 28 pixels is

character body which is padded by 2 pixels on all four sides. The dataset is divided into 80% train images, i.e., 73,600 images and 20% test images, i.e., 18,400 images.

4.2 Convolutional Neural Network (CNN)

For CNN, we have experimented two architectures having a depth of four layers and eight layers whose details are given below:

CNN with Four Layers
The CNN architecture consists of four layers consisting of two convolutional layers, two fully connected layers. The architecture is shown in Fig. 4. The input to convolutional layer is a 32 × 32 grayscale image. The convolutional layer uses a 5 × 5 overlapping kernel producing 16 feature maps of size 28 × 28. The activation function used in this layer is 'ReLU.' Each feature map has a different set of weights. All the units in a feature map share the same set of weights and so they are activated by the same features at different locations. The convolutional layer is followed by the subsampling layer. Subsampling layer reduces the resolution of the feature map from convolutional layer by max pooling the features covered by a 2 × 2 filter. This step is important because the position of the feature may vary from image to image; therefore, the model must learn the relative position of the feature instead of the absolute position. The feature map produced by the subsampling layer is of size 14 × 14 which is then applied to the second convolutional layer. The second convolutional layer also uses a 5 × 5 kernel which gives 32 feature maps of size 10 × 10 which is then applied to the second subsampling layer which is the same as the first one. The output of the second sampling layer consists of feature maps of size 5 × 5 which are flattened to 800 neurons and applied to the 1000-way fully connected layer using 'ReLU' activation function with a dropout of 0.4. The last output layer consists of 46 nodes which represent the 46 output classes. It uses the 'Softmax' activation function. The fully connected layer is the traditional feedforward network. The model is trained using 'Adadelta' optimizer with default learning rate.

CNN with Eight Layers
We have extended the above model up to eight layers consisting of five convolutional layers and three fully connected layers to increase the depth of the CNN architecture.

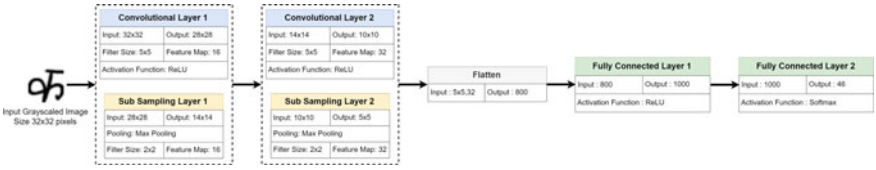

Fig. 4 4-layer CNN architecture

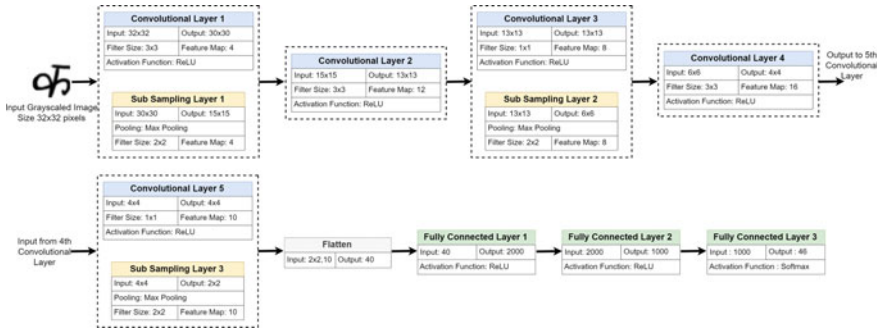

Fig. 5 8-layer CNN architecture

The architecture is shown in Fig. 5. The input to the first convolutional layer is a 32 × 32 grayscale image. This first layer uses a 3 × 3 overlapping kernel that outputs 4 feature maps each of size 30 × 30. The activation function used in this layer is 'ReLU.' The convolutional layer is followed by the subsampling layer of filter size 2 × 2 that reduces the resolution of the feature map from convolutional layer by max pooling the features. The subsampling layer outputs feature maps of size 15 × 15 which are applied to the second convolutional layer. The second convolutional layer also uses the 3 × 3 kernel which gives 12 feature maps of size 13 × 13.

The output of the second convolutional layer is directly applied as an input to the third convolutional layer having a kernel of size 1 × 1 where the number of feature map is 8 of size equal to the input, i.e., 13 × 13. Sometimes referred as one-by-one convolutional layer or network in network, this layer increases the depth of the architecture to generate deeper network without simply stacking layers. The second subsampling layer which is similar to the first one takes input from the third convolutional layer and produces feature maps of size 6 × 6. The output of the second sampling layer is passed to the fourth convolutional layer that uses a 3 × 3 kernel and outputs 16 feature maps each of size 5 × 5. Similar to the third convolutional layer, the fifth convolutional layer uses a kernel of size 1 × 1 and has 10 feature maps. The output of the fifth convolutional layer is applied to the third subsampling layer with a filter size of 2 × 2 which is flattened to 40 neurons. The flattened neurons are applied to the first 2000-way fully connected layer using 'ReLU' activation function with a dropout of 0.5. The second 1000-way fully connected layer also uses 'ReLU' activation function with a dropout of 0.5. The last output layer consists of 46 nodes which represent the 46 output classes. It uses the 'Softmax' activation function. The model is trained using 'Adadelta' optimizer with default learning rate.

The final accuracy of the model consisting of four layers obtained after training for 5 epochs is 98.79%, and for the model consisting of eight layers, we obtained a final accuracy of 96.42%. Figure 6 shows the results. So increasing the layers of the CNN model leads to a decline in accuracy which is caused due to the problem of vanishing gradient.

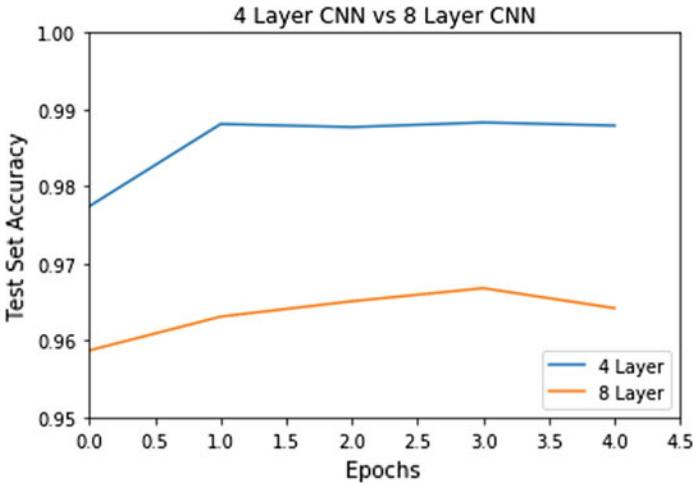

Fig. 6 4-layer CNN versus 8-layer CNN

4.3 Residual Network (ResNet)

For ResNet, we have experimented two architectures, namely ResNet 34 and ResNet 50, which consist of 34 and 50 layers, respectively. The architecture detail is given below:

ResNet 34
ResNet 34 has a depth of 34 layers. The architecture is described in Table 1. Like every ResNet, ResNet 34 also consists of a common convolutional layer and a pooling step which are followed by four convolutional layer groups having similar behavior. Each convolutional layer group uses a kernel of size 3×3 and has fixed number of feature maps which are 64, 128, 256, 512, respectively. The first group consists of 3 pairs of convolution. The second group consists of 4 pairs of convolution. The third group consists of 6 pairs of convolution, and the final group consists of 3 pairs of convolution. At the last, we have a 1000-way fully connected layer using 'Softmax' activation function before which we have an average pooling layer. The first convolution of each group uses a stride of 2 because of which the size of the feature map reduces by half. Various other parameters are set to default parameters of standard ResNet 34 [6].

ResNet 50
ResNet 50 has a depth of 50 layers. The architecture is described in Table 1. Like every ResNet, ResNet 34 also consists of a common convolutional layer and a pooling step which are followed by four convolutional layer groups having similar behavior. Each convolutional layer group consists of some triples having kernel size as 1.3 and 1, respectively. The first group consists of 3 triples of convolution. The second

Table 1 ResNet 34 and ResNet 50 architecture

Layer name	Output size	34-layer ResNet	50-layer ResNet
Convolutional layer 1	112 × 112	7 × 7, 64, stride 2	
Convolutional layer 2	56 × 56	3 × 3, max pool, stride 2	
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 2$
Convolutional layer 3	28 × 28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 2$
Convolutional layer 4	14 × 14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 2$
Convolutional layer 5	7 × 7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 2$
	1 × 1	Average pool, 1000-d fully connected Activation function: Softmax	

*Here convolutional layers are represented as *filter size, number of feature maps, stride (optional)*

group consists of 4 triples of convolution. The third group consists of 6 triples of convolution, and the final group consists of 3 triples of convolution. Number of feature maps in each convolution is shown in Table 1. The average pooling and the fully connected layer are the same as ResNet 34. Various other parameters are set to default parameters of standard ResNet 50 [6].

The final accuracy for ResNet 34 obtained after training for 5 epochs is 98.73%, and for ResNet, we obtained a final accuracy of 99.35%. Figure 7 shows the results. So increasing the layers of the ResNet model increases the accuracy, thus solving the problem of vanishing gradient.

4.4 Result Summary

Table 2 gives the summary of experiments performed on different network architectures.

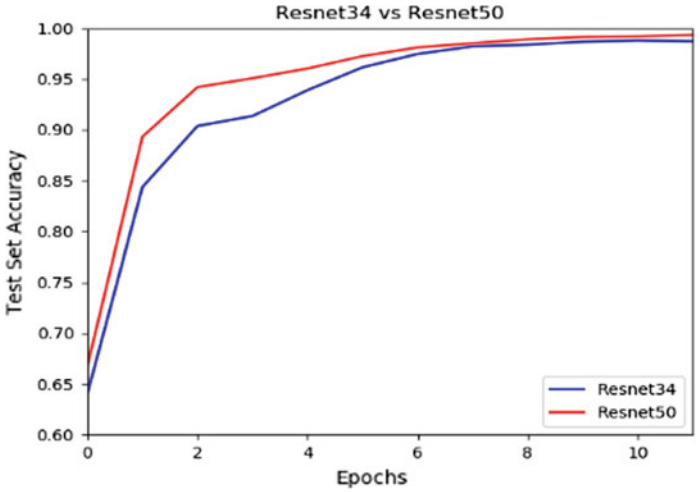

Fig. 7 ResNet34 versus ResNet50

Table 2 Experimental summary

Architecture	4-layer CNN	8-layer CNN	ResNet 34	ResNet 50
Accuracy (in %)	98.79	96.42	98.73	99.35

5 Conclusion

The comprehensive study of residual network for Devanagari handwritten character recognition and its comparison with the current state-of-the-art architecture of convolutional neural network (CNN) is the first of its kind. The proposed implementation using ResNet architecture obtained the highest accuracy of 99.35% which is significantly higher than the current state-of-the-art architecture of CNN. The highest result was obtained using ResNet 50 architecture. The proposed approach of using residual network can be the beginning of the use of the deeper network in Devanagari handwritten character recognition problem.

References

1. IBM. https://www.ibm.com/ibm/history/exhibits/rochester/rochester_chronology2.html
2. Jayadevan R, Kolhe SR, Patil PM, Pal U (2011) Offline recognition of Devanagari script: a survey. IEEE Trans Syst, Man, Cybern Part C (Appl Rev) 41(6):782–796
3. Wikipedia List of Languages by native speakers in India. https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers_in_India
4. Acharya S, Pant AK, Gyawali PK (2015) Deep learning based large scale handwritten Devanagari character recognition. In: 2015 9th international conference on software, knowledge,

information management and applications (SKIMA), Kathmandu, pp 1–6
5. Residual blocks—Building blocks of ResNet. https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec
6. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, pp 770–778
7. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20:273. https://doi.org/10.1023/A:1022627411411
8. Kale KV, Deshmukh PD, Chavan SV, Kazi MM, Rode YS (2013) Zernike moment feature extraction for handwritten Devanagari compound character recognition. In: 2013 science and information conference, London, pp 459–466
9. Understanding Residual Networks. https://towardsdatascience.com/understanding-residual-networks-9add4b664b03
10. Chakraborty B, Shaw B, Aich J, Bhattacharya U, Parui SK (2018) Does deeper network lead to better accuracy: a case study on handwritten Devanagari characters. In: 2018 13th IAPR international workshop on document analysis systems (DAS), Vienna, pp 411–416
11. Salehinejad H, Baarbe J, Sankar S, Barfett J, Colak E, Valaee S (2017) Recent advances in recurrent neural networks. arXiv preprint arXiv:1801.01078
12. Le Cun Y, Boser B, Denker JS, Howard RE, Habbard W, Jackel LD, Henderson D (1990) Handwritten digit recognition with a back-propagation network. In: Advances in neural information processing systems, pp 396–404
13. He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. arXiv:1512.03385