Hindi Handwritten Character Recognition using Deep Neural Network

T. Raguntthar
Assistant Professor, Computing Technologies
SRM Institute of Science and Technology
Chennai, Tamil Nadu
raguntht@srmist.edu.in

Yug Desai
Computer Science and Engineering
SRM Institute of Science and Technology
Chennai, Tamil Nadu
yd8421@srmist.edu.in

***Abstract*—A Handwritten Hindi character identification system based on various Deep learning techniques is presented in this research. Researchers are actively focused on handwritten character recognition because it has potential applications in fields including human-robot interaction, automatic data entry for commercial documents, historical document and assistive technology for blind and visually impaired users. In this study, we suggest a method for identifying handwritten Hindi characters utilizing deep learning techniques such as Convolutional Neural Networks (CNN) and Lenet-5 Architecture with Optimizer RMSprop (Root Mean Square Propagation) and Adaptive Moment (Adam) Estimation. The suggested system was evaluated on samples of photographs from user-defined data sets after being trained on samples of a huge amount of database images. From this experiment, very high accuracy recognition result was achieved. We have compared training curve over both the architecture with given two optimizer.**

Keywords—Hindi Character Recognition, CNN, Lenet-5, Adam, RMSProp, DNN, ANN, ML, AI

I. INTRODUCTION

Nowadays, communication is a basic necessity for people to share information. Communication can be done through various mediums, some of which are internet, letter, telegram, etc. Internet communication can be performed through phone, laptop, in the form of pdf, images, text and email. These forms of communications are also used in businesses, government and school. Different languages are used for the exchange of information. Before internet everything was conducted offline, in the form of handwritten documents. Since internet started gaining popularity, it brought about change in the way such information exchange gets conducted. Everything started becoming online.

Hindi Handwriting Recognition system helps in connecting the offline handwritten documents to its online form. In this system the script which represents Hindi language, that is, Devanagari Script [15] is used to classify which character the image represents. This helps in creating a bridge between a written document and an online typed document. It can pave the way to edit the offline document directly through end devices such as laptop or phone. Hindi Character Recognition System is a system through which photos of Hindi character can be digitalized and make it virtually available.

In the past OCR has been a major technique used to identify the text in an image. OCR techniques help in directly converting handwritten documents to digital form [17]. The OCR techniques have advanced a lot in languages like English and Chinese. It has yet to be a good system for Hindi texts.

The process of converting handwritten characters into digital character is performed using A subfield of artificial intelligence (AI) and computer science called machine learning focuses on using data and algorithms to simulate how humans learn, gradually increasing the accuracy of the system. Machine Learning is further categorized into two sub-fields, namely, Supervised Machine Learning and Unsupervised Machine Learning. In supervised learning the technique used is called Deep Neural Network. Deep learning is a machine learning method that teaches computers to perform tasks that humans accomplish without thinking about it. Deep Neural Network is a configuration used to train the input to get the desired output. In this paper we have used supervised learning that is Deep Learning to classify the images of characters. Unsupervised learning called Unsupervised feature learning can also be used to classify the images [20].

Deep learning techniques have been used in a variety of domains, including face detection, speech recognition, medical image classification, and image classification. Convolutional neural networks, deep neural networks, transformers, and recurrent neural networks are only a few of the new deep learning architectures that have been developed in recent years. The entire architecture has demonstrated knowledge in a variety of fields. One such area where machine learning approaches have been intensively tested is character recognition. Character recognition was offered as a use case for the first deep learning technique, which is now one of the top machine learning algorithms, in 1998 using the MNIST database [13]. The classification has successfully been applied on languages such as Arabic [16], Chinese, Malayalam, Kannada [14], Gujarati [9], etc. Many application of character recognition involves aiding blind and visually impaired people, digitalizing business documents and historical documents [11], teaching pre-primary students to write and understand characters among many others.

The fundamental building block of deep learning techniques is a number of hidden layers, each of which is made up of several neurons that calculate the appropriate weights for the deep network. The layers in the model extracts the vital features of the input given and use it to get the desired output. This is performed using thousands if not millions of parameters. Once the output is matched, the model backtracks to the input and correct the parameters to increase the accuracy. This cycle is repeated number of times to further decrease the loss and increase the accuracy of the output.

II. STATE OF THE ART (LITERATURE SURVEY)

Vijaya Kumar Reddy R. and U. Ravi Babu [1] have presented a Handwritten Hindi Recognition System based on different Deep Learning Techniques in the paper. The different deep learning approaches are Deep Feed Forward Neural Network (DFFNN), Convolutional Neural Network with RMSProp, and Adaptive Moment (Adam) Optimizers. The dataset consisted of 35 unique Hindi characters with around 400 images per letter. The accuracy obtained on the dataset for different techniques are as follows: 1. CNN with RMSProp - 97.3% 2. CNN with Adam - 96.02% 3. DFFNN - 95.57%. From experimental results, it was observed that DFFNN, CNN-Adam, and CNN-RMSProp yield the best accuracy compared to alternate methods such as KNN, PCA, etc.

Sonika Dogra and Chandra Prakash [2] have OCR technology to recognize Hindi Handwriting. The technology uses Unsupervised Machine Learning Classification namely, Support Vector Machine (SVM). It further uses Diagonal Feature Extraction to extract features from the preprocessed image. Using SVM it achieved an average of 93.06 % accuracy. The dataset comprised handwritten characters from 20 different people.

Deepak Chaudhary and Kaushal Sharma [3] have used Convolution Neural Network, a powerful tool for solving Machine Learning (ML) problems. The architecture proposed here is “Deep Convolution Neural Network” (DCNN) for Hindi handwritten character recognition. The Model is built on LeNet-5 CNN architecture. The model was trained on 96000-character sets. Accuracy obtained using Adam Optimizer was 95.72 and RMSprop was 93.68%.

Parshuram M. Kamble and Ravinda S. Hegadi [4] have used Rectangle Histogram Oriented Gradient for feature extraction of characters. The dataset consists of 8000 samples each of 40 basic Marathi characters. The images are normalized to 20X20 pixel size. The techniques used for classification are Feed-Forward Artificial Neural Network (FFANN) and Support Vector Machine (SVM). The accuracies obtained from the classification 97.15% and 95.64% respectively.

Akash Roy [5] has proposed deep neural architectural solution for Bengali alphabet Recognition. The author has used ResNet 50 model which is 50-layer Residual Network. The model has achieved 96.8% accuracy in just 11 epochs. ImageNet dataset was used to train the model.

Raghunath Dey and Rakesh Chandra Balabantray [6] have worked on English Character dataset, which consist of 26 characters. The pre-processing includes binarization and skeletonization of the dataset, followed by segmentation of each character. A novel shape-based feature representation technique is used to label upper and lower part of a character. The character is then further recognized using sliding window technique and a concept known as LCS (Longest Common sub-sequence). The Dataset used is IAM Handwriting English Dataset. The accuracy obtained overall is 78.84%.

III. PRPOSED WORK

The proposed method has mainly 7 phases. The primary phase consists of acquiring the data from Kaggle. The dataset consists of csv file containing grayscale inverted images per row. The dataset consists of 92000 examples having 2000 data for each character for 46 various characters. The 2^{nd} phase consists of pre-processing the data. In pre-processing the data, data preparation using one hot encoding is used to alter the labels to 0 to 45. These labels map each character to an integer value for comparing the result to that of the predicted value. The next step is to split the data into training and validation set. The next phase consists of augmenting the data for giving it as input to the model selected. The data is augmented on various parameters such as horizontal and vertical shift, rotation range, zoom range and shear range. The 5th phase consists of extracting the features of the input data using different deep learning techniques. These techniques are implemented using different Deep Neural Network Architecture such as Convolutional Neural Network (CNN) and Lenet_5. These architectures are optimized further using 2 different optimization techniques namely, Adaptive Moment (Adam) and Root Mean Square Propagation (RMSProp). In the 6^{th} phase we examine the model and perform hyper-parameter tuning to get better result on the model. It is done to ensure our model doesn’t overfit or underfit the training data. The last phase consists of predicting the character image to identify which character has been drawn. The proposed steps are as shown in fig. 1.

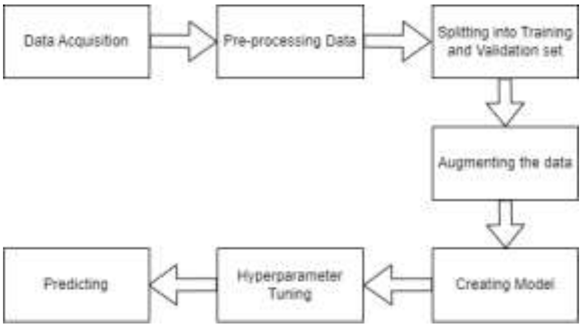


Fig. 1. Data flow cycle

The clssifier used in the Model is Sparse Catagorical Crossentropy which maps model to labels 0-45. There are many other classifier which can be used [9] one of them being Catagorical CrossEntropy which selects multiple labels instead of single label.

A. *Convolutional Neural Network*

Convolutional Neural Network is being used in many different fields. Some of which are Image Classification, Search Engines, Recommender System, and Social Media. The most prominent application of Convolutional Neural Network is image classification [7]. In this custom architecture 7-layers have been used to classify the characters having 2-Convolutional layers with Relu Activation, 2-MaxPooling layers, 1-Flatten and 2-Dense layers. The Architecture is as shown in Fig 2.

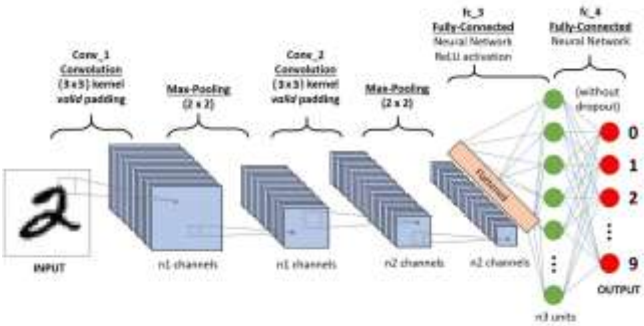


Fig. 2. Convolutional Neural Network Architecture

B. *LeNet-5*

The network is known as Lenet-5 since it contains five layers with learnable parameters. It combines average pooling with three sets of convolutional layers. With stride 1, the tanh activation occurs in the convolutional layer. We have two fully linked layers following the convolution and average pooling layers. Finally, a SoftMax classifier places the photos in the appropriate class. Fig. 3 shows the architecture of the model where each Convolutional layer extracts features from 5X5 compression of the previous image. The other model that can be applied for the following project includes multilayer perceptron and radial basis function neural networks [8], Support Vector Machines [10] and LSTMs [18]. The other ready to use architecture consist of InceptionV3-Net, VGG19-Net, and ResNet50 which can also be directly applied to classification problem [12].

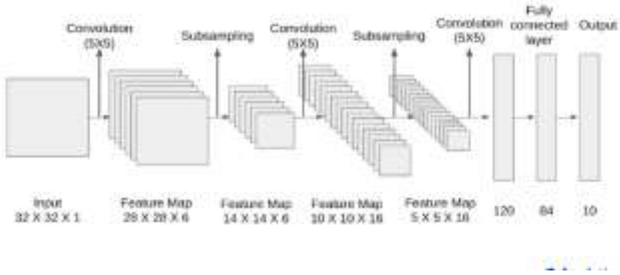


Fig. 3. Lenet-5 Architecture

C. *Adam Optimizer*

Deep learning fields like Computer Vision and Natural Language Processing heavily utilise the Adam optimizer. Stochastic gradient descent is extended by Adam, also known as the Adaptive Moment Optimizer. Adam is simple to implement, memory-light, and has good computational performance. Adam keeps a per-parameter learning rate rather than a single learning rate hence the name Adaptive Moment. The equation is as shown in fig. 4.

$$m_t = \beta m_{t-1} + (1-\beta)\left[\frac{\delta L}{\delta w_t}\right]$$
$$w_{t+1} = w_t - \alpha m_t$$

Fig. 4. Adam Equation

Wt = weights at time t
Wt+1 = weights at time t+1
αt = learning rate at time t
∂Wt = derivative of weights at time t
mt-1 = aggregate of gradients at time t-1 [previous]
∂L = derivative of Loss Function
β = Moving average parameter (const, 0.9)
mt = aggregate of gradients at time t [current] (initially, mt = 0)

D. *RMSProp Optimizer*

RMSProp is similar to gradient descent plus momentum. It restricts the vertical direction oscillations. RMSProp stands for Root Mean Square Propagation where its equation is given by fig. 5.

$$v_t = \beta v_{t-1} + (1-\beta) * \left[\frac{\delta L}{\delta w_t}\right]^2$$
$$w_{t+1} = w_t - \frac{\alpha_t}{(v_t+\varepsilon)^{1/2}} * \left[\frac{\delta L}{\delta w_t}\right]$$

Fig. 5. RMSProp Equation

Wt+1 = weights at time t+1
β = Moving average parameter (const, 0.9)
Wt = weights at time t
Vt = sum of square of past gradients. [i.e sum(∂L/∂Wt-1)] (initially, Vt = 0)
αt = learning rate at time t
∂L = derivative of Loss Function
∂Wt = derivative of weights at time t
ϵ = A small positive constant (10-8)

IV. Implementation

The above proposed work has been implemented on Google Colab. The work has been implemented using tensorflow, keras, numpy, pandas, and matplotlib libraries. The dataset has been taken from Kaggle website. The dataset consists of 92,000 examples of 1024 dimension each. The input has been taken as csv file of 92000 x 1025 dimensions, last column being the label for each example. After getting the dataset it has been cleaned and pre-processed using pandas' data frame. One hot-encoding has been applied to the labels to convert it to a standard label from 0 to 45 numbers. The resulting pre-processed data has then been split into processing data and labels. The processing data has further been split into training and validation set where the training set is 80% of the total data and rest 20% is validation set after shuffling the whole dataset. The data has then been visualized using matplotlib library for some few examples as shown in fig. 6. Each image is 32x32 dimension black and white colour. The images are outputted on a 3x3 axis where each of the example is a random input image.

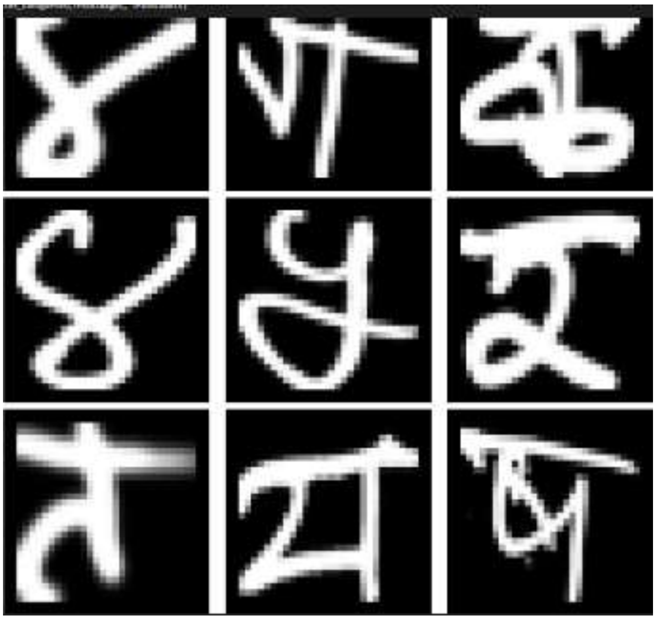
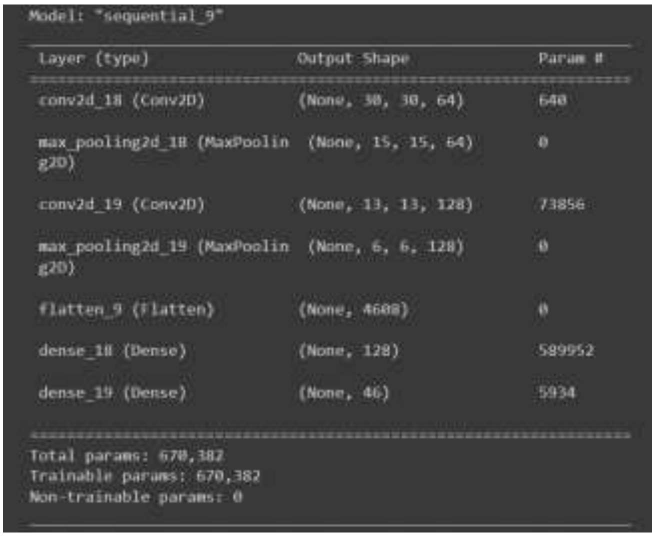


Fig. 6. Examples of the Input Image

The dataset is further rescaled and augmented with several parameters such as rotation range, width and height shift range, shear range, zoom range and fill image using ImageDataGenerator class. In next step Machine Learning model is created using keras library. There are two models that are implemented. One being Convolutional Model, other being Lenet-5 Model. The Model layers have been shown as in fig. 7 along with 670,382 total trainable parameters. The convolutional layer takes in the input data and creates 64 different filters for each input. The kernel size used in the layer is 3x3. The images have then been diminished to half the total dimension by adding max-pooling layer after each convolutional layer. ReLu activation has been set for each of the convolutional layer.



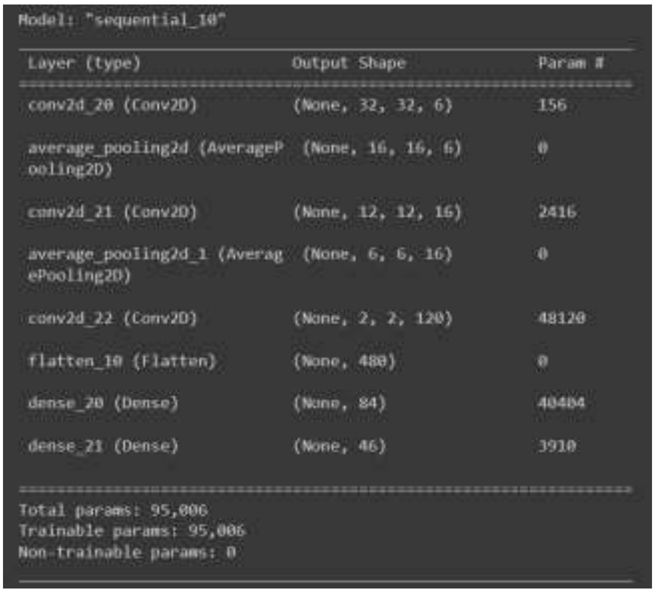
Model: "sequential_9"

Layer (type)	Output Shape	Param #
conv2d_18 (Conv2D)	(None, 30, 30, 64)	640
max_pooling2d_18 (MaxPooling2D)	(None, 15, 15, 64)	0
conv2d_19 (Conv2D)	(None, 13, 13, 128)	73856
max_pooling2d_19 (MaxPooling2D)	(None, 6, 6, 128)	0
flatten_9 (Flatten)	(None, 4608)	0
dense_18 (Dense)	(None, 128)	589952
dense_19 (Dense)	(None, 46)	5934

Total params: 670,382
Trainable params: 670,382
Non-trainable params: 0

Fig. 7. Convolutional Neural Network Model

The Lenet Model has been described as shown in the fig. 8. The Lenet Model Architecture is similar to that of the CNN model shown in fig. 7. The key difference being the use of average pooling layer instead of max pooling layer and use of 5 x 5 kernel size in covolutional layer instead of 3 x 3. The total trainable paramaters has drastically been decreased to 95,006 that is dropping almost 580,000 parameters lowering the training time exponentially.



Model: "sequential_10"

Layer (type)	Output Shape	Param #
conv2d_20 (Conv2D)	(None, 32, 32, 6)	156
average_pooling2d (AveragePooling2D)	(None, 16, 16, 6)	0
conv2d_21 (Conv2D)	(None, 12, 12, 16)	2416
average_pooling2d_1 (AveragePooling2D)	(None, 6, 6, 16)	0
conv2d_22 (Conv2D)	(None, 2, 2, 120)	48120
flatten_10 (Flatten)	(None, 480)	0
dense_20 (Dense)	(None, 84)	40404
dense_21 (Dense)	(None, 46)	3910

Total params: 95,006
Trainable params: 95,006
Non-trainable params: 0

Fig. 8. Lenet-5 Model

Both models have been compiled using Adam optimizer and RMSprop optimizer. The model has been trained using training set and examined using validation set. Accuracy and Precision has been plotted against the epochs of training. The learning curve looks as shown in fig. 8, fig. 9, fig. 10, fig.11.

A. *Convolutional Neural Network with Adam optimizer*

The given Architecture successfully predicts the data with 97% accuracy on training set and 98.5% accuracy on validation set. The learning curves for loss and accuracy show the model is not affected by bias and variance. The learning curve converges at the end of 25 epochs. The loss as shown in fig. 9 being 9% and 4% on training and validation set. The accuracy is as mentioned above shown in fig. 10 becomes 97% and 98.5% by the last epoch. The high accuracy and low loss obtained is due to hyperparameter adjustment done on Adam optimizer to 0.0005 learning rate.

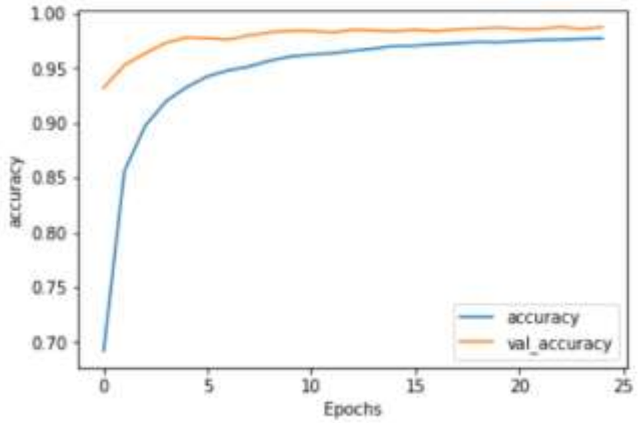


Fig. 9. Accuracy Curve for CNN with Adam

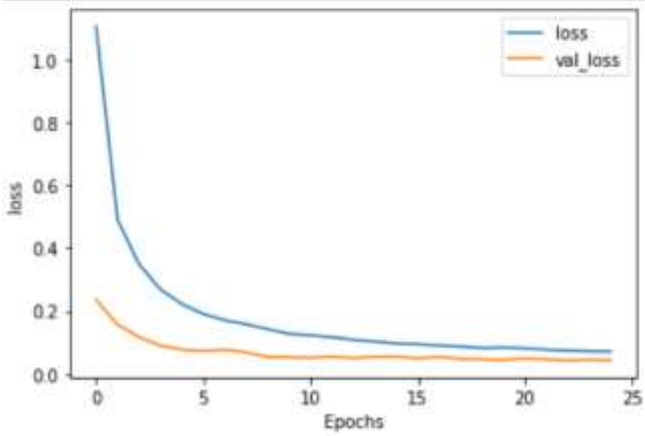


Fig. 10. Loss Curve for CNN with Adam

B. Lenet-5 Network with Adam optimizer

The Curve in fig. 11 shows loss converging to almost 17% and 8% for training and validation set respectively. The curve becomes quite stable after first 8 to 9 epochs. While fig. 12 also converges by the end of 25 epochs. The accuracies obtained by the curve are 94.6% and 97.4% on training and validation set respectively.

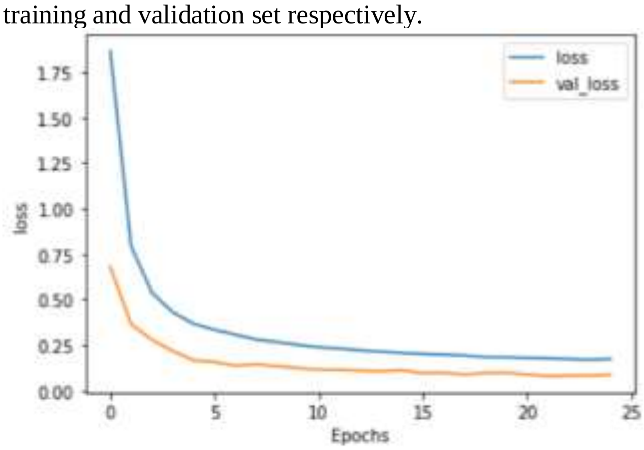


Fig. 11. Loss Curve for LeNet-5 with Adam

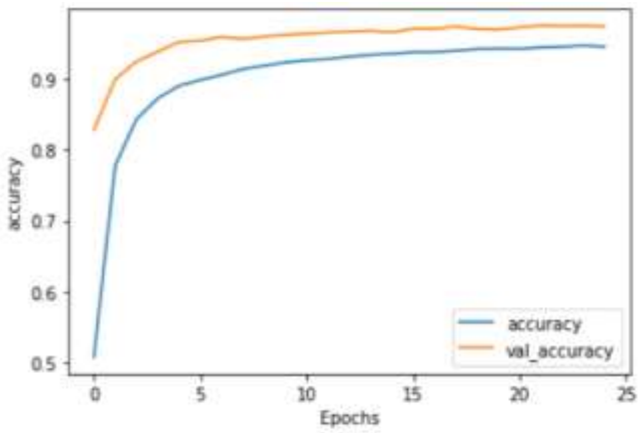


Fig. 12. Accuracy Curve for LeNet-5 with Adam

C. Convolutional Neural Network with RMSprop optimizer

The Curve in fig. 13 shows loss being almost 18% and 20% for training and validation set respectively. The curve becomes unstable after 12 to 13 epochs for validation set. The validation loss keeps fluctuating after 12 epochs which indicates poor result due to optimizer. The reason might be the learning rate of RMSprop since it is constant unlike Adam Optimizer. The average accuracy obtained during training is around 94% and 97% on training and validation set as shown in fig. 14. The validation graph also fluctuates after around 12 epochs. It is to be noted that the graph has non-converging curve. There is slight convergence initially followed by fluctuation.

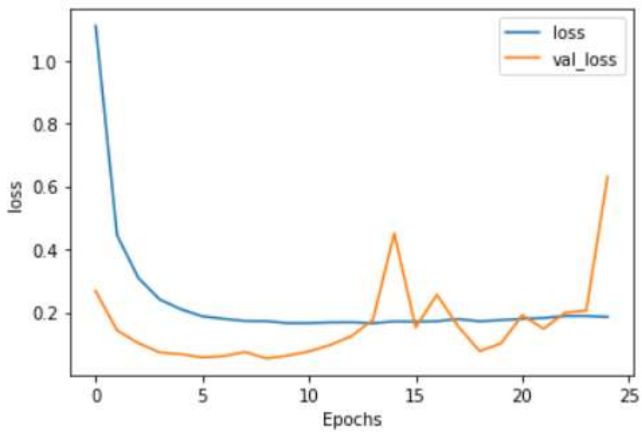


Fig. 13. Loss Curve for CNN with RMSprop

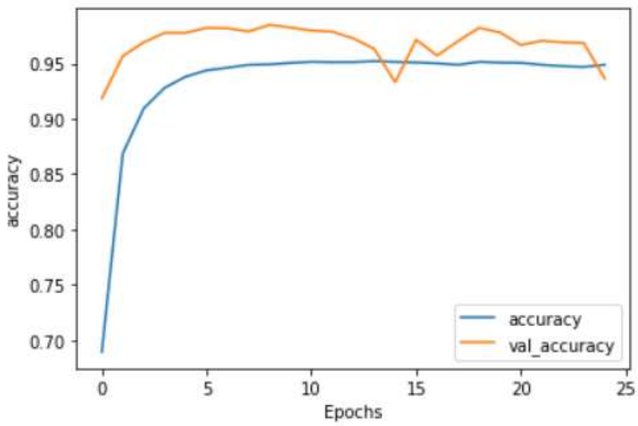


Fig. 14. Accuracy Curve for CNN with RMSprop

D. Lenet-5 Network with RMSprop optimizer

The Curve in fig. 15 shows loss converging to almost 21% and 9% for training and validation set respectively. The curve becomes quite stable after first 5 to 6 epochs. While fig. 16 also converges by the end of 25 epochs. The accuracies obtained by the curve are 93.6% and 97.2% on training and validation set respectively.

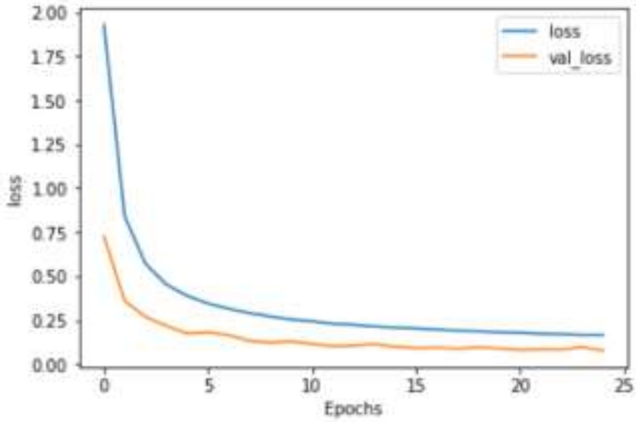


Fig. 15. Loss Curve for LeNet-5 with RMSprop

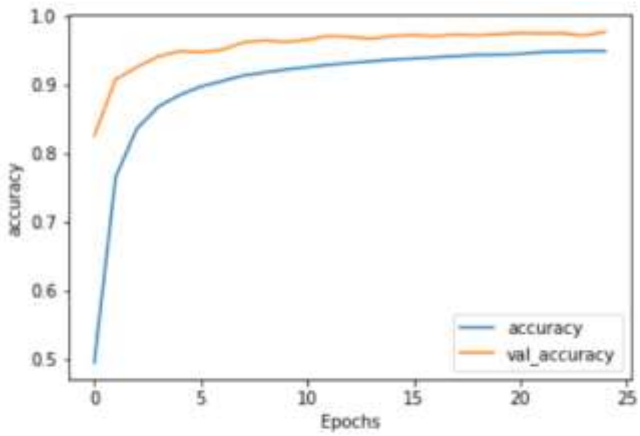


Fig. 16. Accuracy Curve for LeNet-5 with RMSprop

V. Results discussion

The model predicts 98.5% accuracy on new examples using Convolutional Neural Network with Adam Optimizer. This accuracy is quite high compared to other models. The other models, that is Lenet-5 with Adam, Convolutional with RMSprop and Lenet-5 with RMSprop attain accuracy of 96.6%, 96.1% and 97.7% respectively.

VI. Conclusion

Convolutional Neural Network and Lenet-5 Model have been applied with both Adam and RMSprop optimizer. The accuracy has been calculated concluding in CNN with Adam optimizer outperforming all the other architecture with an overwhelming 98.5% accuracy.

References

[1] .R, Vijaya Kumar & Babu, U.. (2019). Handwritten Hindi Character Recognition using Deep Learning Techniques. International Journal of Computer Sciences and Engineering. 7. 1-7. 10.26438/ijcse/v7i2.17..

[2] DOGRA, SONIKA & Prakash, Chandra. (2012). PEHCHAAN: HINDI HANDWRITTEN CHARACTER RECOGNITION SYSTEM BASED ON SVM. International Journal on Computer Science and Engineering. Vol. 4. 718-722.

[3] D. Chaudhary and K. Sharma, "Hindi Handwritten Character Recognition using Deep Convolution Neural Network," *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2019, pp. 961-965.

[4] Parshuram M. Kamble, Ravinda S. Hegadi, Handwritten Marathi Character Recognition Using R-HOG Feature, Procedia Computer Science, Volume 45, 2015, Pages 266-274, ISSN 1877-0509.

[5] Himani Kohli, Jyoti Agarwal, Manoj Kumar, An improved method for text detection using Adam optimization algorithm, Global Transitions Proceedings, Volume 3, Issue 1, 2022, Pages 230-234, ISSN 2666-285X.

[6] Dey, Raghunath & Rakesh, Chandra. (2019). A Novel Sliding Window Approach for Offline Handwritten Character Recognition. 178-183. 10.1109/ICIT48102.2019.00038.

[7] Ciregan, D.; Meier, U.; Schmidhuber, J, "Multi-column deep neural networks for image classification", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI,USA, 16–21 June 2012.

[8] B. K. Verma, "Handwritten Hindi character recognition using multilayer perceptron and radial basis function neural networks in Neural Networks", 1995. Proceedings, IEEE International Conference on. vol. 4, pp. 2111-2115, 1995.

[9] Baheti M. J., Kale K.V., Jadhav M.E., "Comparison of Classifiers for Gujarati Numeral Recognition", International Journal of Machine Intelligence, Vol. 3, Issue 3, pp. 160-163, 2011.

[10] Abdul Rahim Ahmad, Christian Viard-Gaudin, Marzuki Khalid, Emilie Poisson (2004) "Online Handwriting Recognition using Support Vector Machine" TENCON 2004 IEEE Region 10 Conference, Vol. No. 1 , pp 311-314.

[11] R. Saabni, A. Asi, J. El-Sana "Text line extraction for historical document images" Pattern Recognition Letters, 35 (2014), pp. 23-33

[12] Rajpal, Danveer, Akhil Ranjan Garg, Om Prakash Mahela, Hassan Haes Alhelou, and Pierluigi Siano. 2021. "A Fusion-Based Hybrid-Feature Approach for Recognition of Unconstrained Offline Handwritten Hindi Characters" *Future Internet* 13, no. 9: 239. https://doi.org/10.3390/fi13090239Lecun, Y.; Bottou, L.; Bengio,

[13] Y.; Haffner P, "Gradient-based learning applied to document recognition", Proc. IEEE 1998, 86, 2278–2324.

[14] R.S. Hegadi "Recognition of printed Kannada numerals based on zoning method" International Journal of Computer Application (2012), pp. 4-8

[15] Raman Nitin Kali, Gandhi Subham, Khurana Jitender "Study and analysis of Devanagari handwritten character recognition techniques" International Journal of Science and Research, 2 (6) (2013)

[16] MA Radwan, M Khalil, H. Abbas "Neural networks pipeline for offline machine printed Arabic OCR" Neural Processing Letters, 48 (2) (2018), p. 2018

[17] CC Tappert, CY Suen, T. Wakahara "The state of the art in online handwriting recognition" IEEE Trans. Pattern Anal. Mach. Intell., 12 (8) (1990), pp. 787-808

[18] T Breuel, A Ul-Hasan, M Al-Azawi, F. Shafait "High-performance OCR for printed English and Fraktur using LSTM networks" 12th international conference on document analysis and recognition (2013)

[19] D Wu Wang, A Coates, A Ng "End-to-end text recognition with convolutional neural networks" Proceedings of the 21st international conference on pattern recognition (ICPR2012), IEEE (2012), pp. 3304-3308

[20] A Coates, B Carpenter, C Case, S Satheesh, B Suresh, TT Wang, AY. Ng "Text detection and character recognition in scene images with unsupervised feature learning" 2011 International Conference on Document Analysis (2011), pp. 440-445