IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Object Detection and Recognition of A Handwritten Hindi Character On a Background Image

Chitra Desai
Professor and Head
Department of Computer Science
National Defence Academy, Pune, India

Abstract: To localize the object within an image, object detection, and recognition, the subfields of computer vision are used together. When a handwritten character is to be recognized, it involves pattern recognition in the image for the given set of handwritten characters. In this paper set of Hindi, language characters is used for training the model. Convolution neural network and transferring learning is used for generating the model. The images in the training dataset are augmented. Augmented images significantly contribute to improving model performance. This paper aims at identifying the presence of a handwritten Hindi character on the background image, that is object detection, and identifying the character, that is object recognition using transfer learning and convolutional neural network.

***IndexTerms* - Object detection, Object recognition, Convolutional neural network, handwritten Hindi Character**

1. INTRODUCTION

Object detection and object recognition are important tasks involved in computer vision with different aspects. Object detection is about detecting the presence and location of the object in an image, while object recognition aims at identifying the specific category of the object. Object detection uses a combination of object classification and localization techniques.
Several techniques evolved over a period of time for object detection - like template matching [1], statistical classifier [2], sliding window [3], and deep learning[4]. Depending upon the size and complexity of the objects in the image, the required speed and accuracy with which the object in the image needs to be identified, these techniques vary. In 2001 to detect objects in the image and classify them based on edges, corners, and lines the most popular technique known as Haar cascade [5] was proposed. Haar cascade-based classifier scans images at different scales and locations and is one of the old techniques widely used for face detection. However, it is observed to be less accurate compared to deep learning models which evolved over a period of time. Histogram Oriented Gradient technique [6] proposed in 2005 for human detection is another important technique that divides the image into small cells and computed the gradient and magnitude for each cell of the image to extract features and then uses a classifier–support vector machine to detect the objects. Before the advancement of convolutional neural networks and transfer learning these algorithms and techniques for object detection proved to be effective but had their own limitations.
For processing complex and huge amounts of data to extract high-level features from images, deep learning has proven to be the most effective alternative. Deep learning makes use of convolutional neural network architectures for object detection to produce eloquent results.
To design a CNN-based model for object detection and recognition there are several factors that need to be taken into consideration. The foremost important factor is the data size and its complexity. The larger the data size and greater the complexity, the deeper and erudite the deep learning model required. When it comes to individual images, the image resolution and image size also impact the performance of the model. High-resolution images require more computational resources and deeper models to extract relevant features. Another important factor is the choice of appropriate architecture which significantly impacts the performance of the model. Other factors like regularization [7], transfer learning [8], optimization algorithm [9], and availability of hardware resources also impact in designing of the CNN model of object detection and recognition.

The paper aims at object detection and recognition using transfer learning and convolutional neural network. The object to be detected and recognized is the handwritten Hindi character on a background image. The data has more than 5000 images for training and is complex in nature for the task to be performed. The images in the training data set is augmented by using various techniques. Various factors are taken into consideration for designing and implementing the model using transfer learning and CNN.

2. METHODOLOGY

The dataset containing the images are split into train test dataset. The model is pre-trained on VGG16[10] and implemented using the Keras library [11]. The model is fine-tuned and evaluated for performance. Using the trained model prediction for the presence or absence of character is made for the unseen dataset that is the test dataset. Further, the model for handwritten Hindi character recognition is designed and implemented, and tested for accuracy.

3. DATA

The data [12] is organized in folders as shown in Figure 1. The training folder has two more folders – *background* and *hi* similar to the test folder. The training data size is 5875 with 4450 in the *background* folder and 1425 in a *hi* folder. The test data size is 98 with 52 in the background folder and 46 in hi folder. The 98 images in the test data are labeled 1 to 98. The image width is 64 and the height is 64. Figure 2 shows sample *background* training data and Figure 3 shows sample *hi* training data. As seen in Figure 3, each image has only one single handwritten Hindi character. Also from image samples shown in Figure 2 and Figure 3, it is observed that the dataset consists of augmented images in training samples. In general, various transformation techniques like scaling, flipping, rotation, cropping, and blurring can be applied to the original image to create the augmented image [13], which significantly contributes to improving model performance. Image augmentation also acts as a regularization technique that prevents model overfitting. It reduces the need for data collection and contributes to improving model performance.

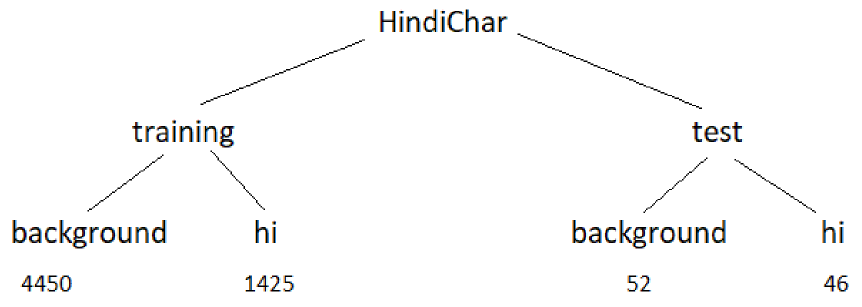


Figure 1 Training and Test Data Folders with Background and Hindi Character

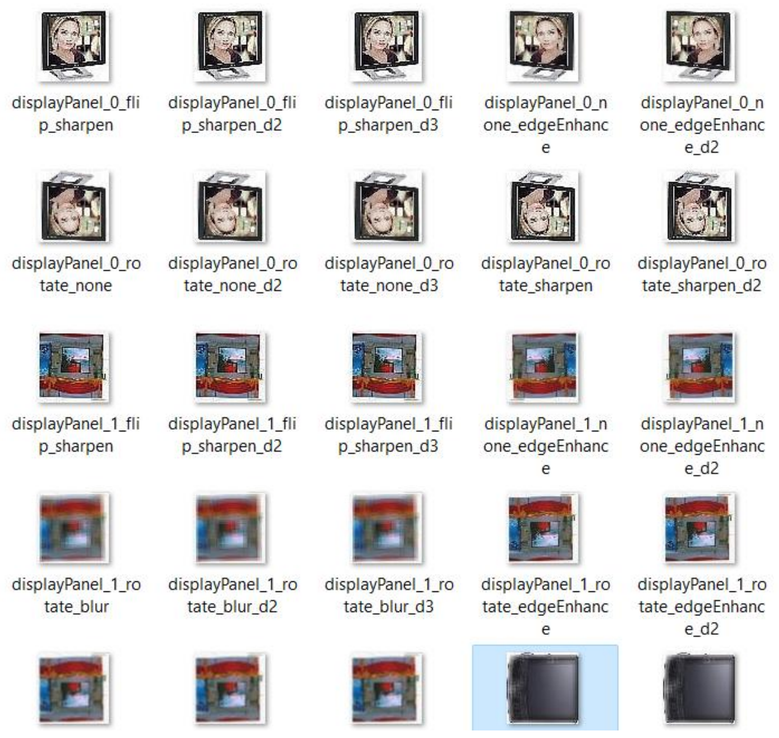

displayPanel_0_flip_sharpen
displayPanel_0_flip_sharpen_d2
displayPanel_0_flip_sharpen_d3
displayPanel_0_none_edgeEnhance
displayPanel_0_none_edgeEnhance_d2
displayPanel_0_rotate_none
displayPanel_0_rotate_none_d2
displayPanel_0_rotate_none_d3
displayPanel_0_rotate_sharpen
displayPanel_0_rotate_sharpen_d2
displayPanel_1_flip_sharpen
displayPanel_1_flip_sharpen_d2
displayPanel_1_flip_sharpen_d3
displayPanel_1_none_edgeEnhance
displayPanel_1_none_edgeEnhance_d2
displayPanel_1_rotate_blur
displayPanel_1_rotate_blur_d2
displayPanel_1_rotate_blur_d3
displayPanel_1_rotate_edgeEnhance
displayPanel_1_rotate_edgeEnhance_d2

Figure 2 Sample images (training: background)

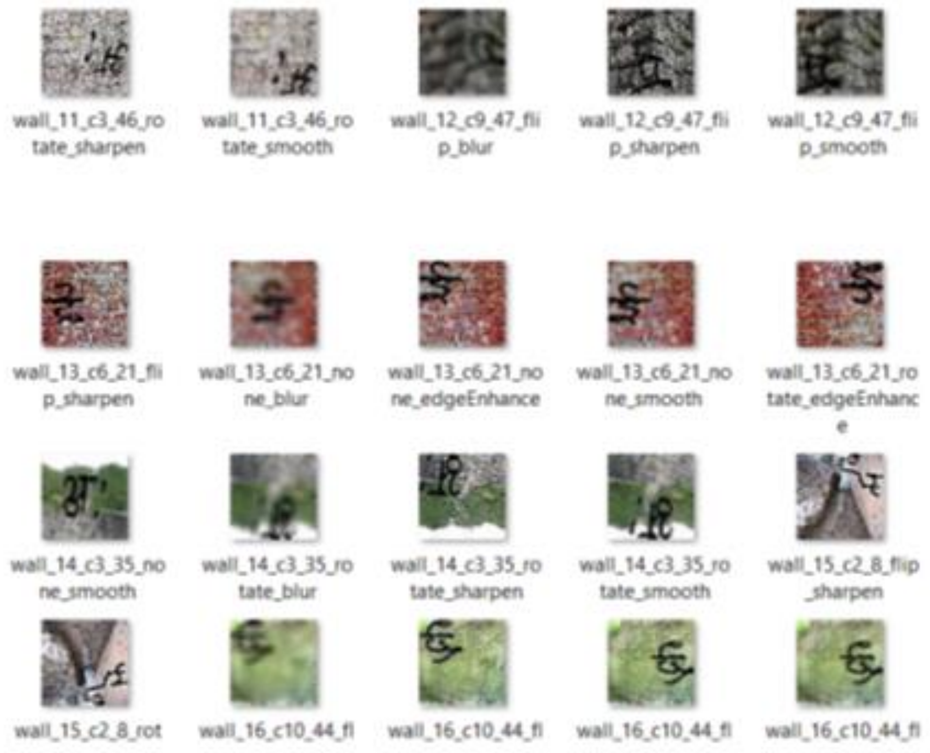

wall_11_c3_46_rotate_sharpen
wall_11_c3_46_rotate_smooth
wall_12_c9_47_flip_blur
wall_12_c9_47_flip_sharpen
wall_12_c9_47_flip_smooth
wall_13_c6_21_flip_sharpen
wall_13_c6_21_none_blur
wall_13_c6_21_none_edgeEnhance
wall_13_c6_21_none_smooth
wall_13_c6_21_rotate_edgeEnhance
wall_14_c3_35_none_smooth
wall_14_c3_35_rotate_blur
wall_14_c3_35_rotate_sharpen
wall_14_c3_35_rotate_smooth
wall_15_c2_8_flip_sharpen
wall_15_c2_8_rot
wall_16_c10_44_fl
wall_16_c10_44_fl
wall_16_c10_44_fl
wall_16_c10_44_fl

Figure 3 Sample images (training: hi)

4. MODEL

4.1 MODEL FOR OBJECT DETECTION

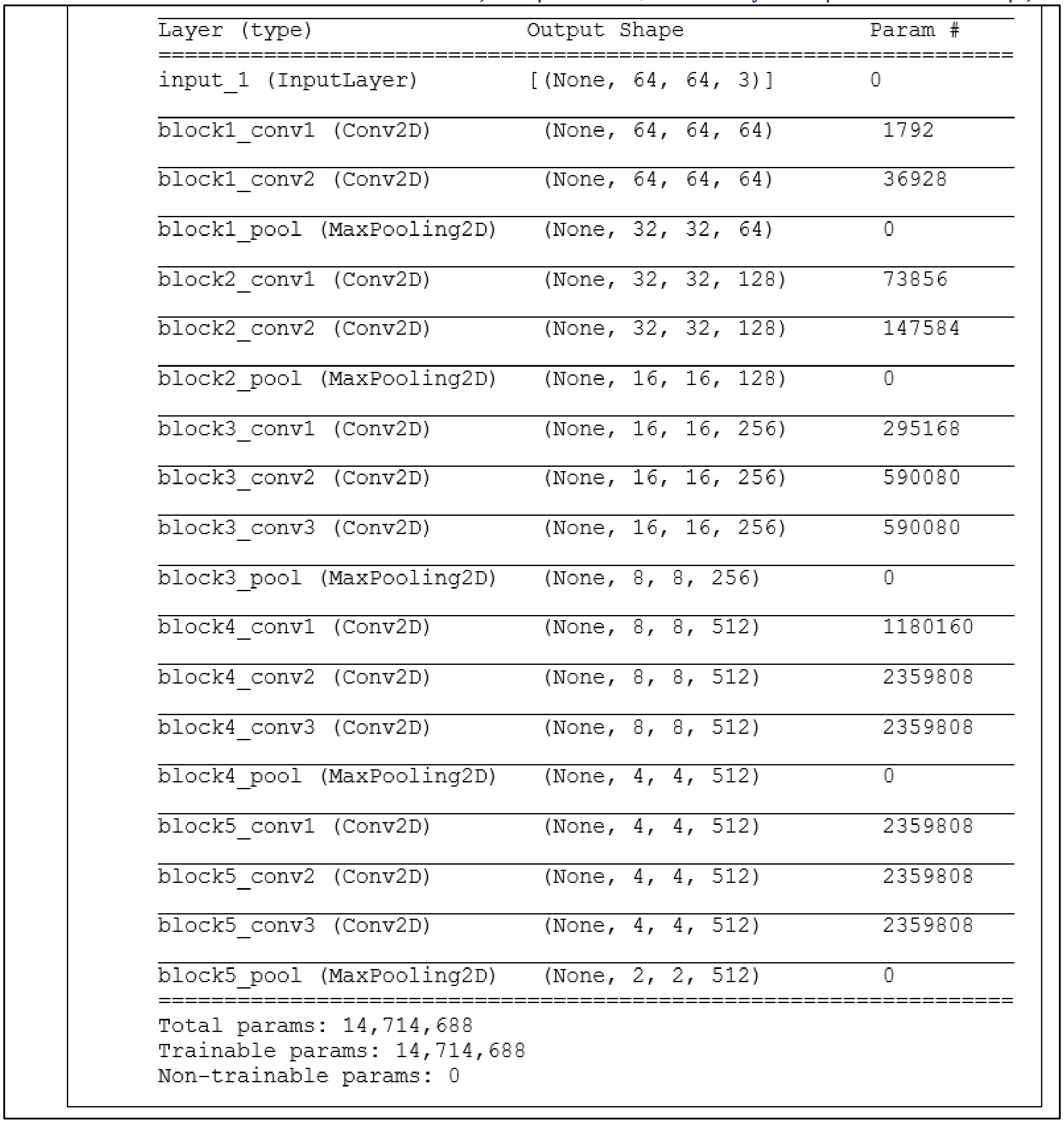
The weights of the model are pre-trained using VGG16 on the ImageNet [14] data set. The input layer has the shape of 64X64X3 because the height of the image is 64, the width is 64, and 3 represents channels in the RGB picture. The pre-trained vgg16-based model is summarized in Figure 4, let the model label be *vgg16covbase*.

On obtaining *vgg16covbase*, the next task is to perform image pre-processing. Before processing, the images are rescaled to 1/255. The original images are RGB images with coefficients in the range of 0-255, which are high to process. In order to target the values between 0 and 1, they are rescaled with 1/255. Using ImageDataGenerator class from Keras library for deep learning, the pre-processing and augmentation of image data for training deep learning models can be done. The ImageDataGenerator class provides an easy-to-use interface for generating augmented image data in real time during model training. It can read images from a directory and apply a set of transformations to each image on-the-fly. The augmented images are then fed into the model for training. This can save time and disk space compared to generating all the augmented images ahead of time and storing them on disk. The train features, train labels, and test features and test labels are extracted and a sequential model is trained using train features and train labels. The sequential model is initialized to fit the train features and train labels as summarized in Figure 5.

The output of the *vgg16covbase* model as seen in Figure 4 goes as an input to the sequential model as shown in Figure 5. In Figure 5, the flattening shows an output shape of (None, 2048), because, in Figure 4, the last block, that is block5_pool (MaxPooling_2D) has the output shape of (None,2,2,512), which after flattening becomes (2X2X512 = 2048). After flattening a dense layer with 'relu' activation is added, followed by a dropout of 0.5, this fraction of neurons is randomly selected and dropped out. Dropout is a regularization technique to avoid overfitting [15]. It prevents the neural network from relying too heavily on any one feature and encourages it to learn more robust representations of the data.

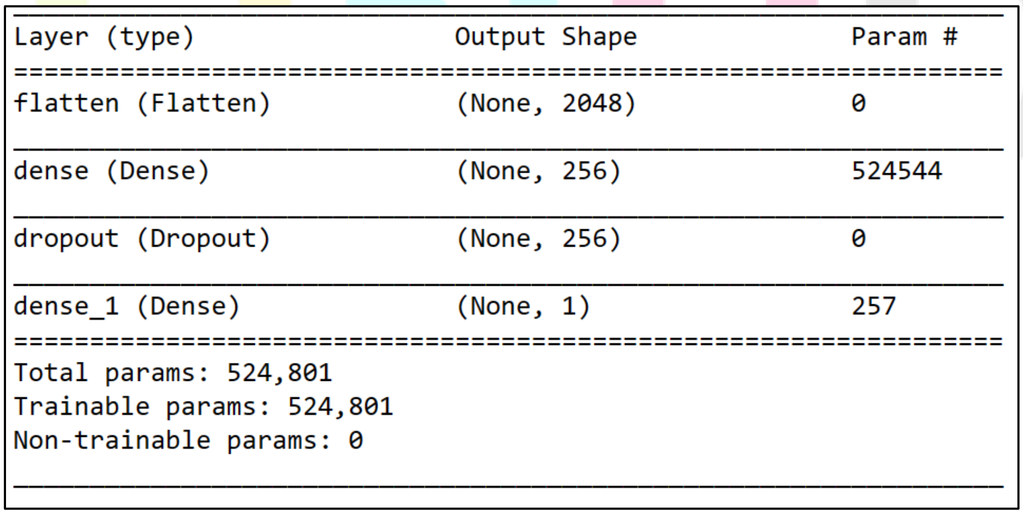
The output shape of (none, 256) typically indicates a batch of input data, where the first dimension "none" refers to the batch size, and the second dimension "256" refers to the number of neurons in the output layer. So, in the case of "dropout with output shape (none, 256)", dropout is being applied to a layer of the neural network with 256 neurons, and the output of the layer will have the same shape as the input batch, with the values of some of the neurons randomly set to zero during training.

The last layer has the output shape of (none,1). Here the task is to detect if the background image has a handwritten character or not, the output will be therefore either 0 or 1. 1 refers to the presence of a character and 0 refers that the handwritten character is not present in the image. Thus, the last output layer uses a sigmoid activation function due to the binary nature of classification.



```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)        [(None, 64, 64, 3)]       0

block1_conv1 (Conv2D)        (None, 64, 64, 64)        1792

block1_conv2 (Conv2D)        (None, 64, 64, 64)        36928

block1_pool (MaxPooling2D)   (None, 32, 32, 64)        0

block2_conv1 (Conv2D)        (None, 32, 32, 128)       73856

block2_conv2 (Conv2D)        (None, 32, 32, 128)       147584

block2_pool (MaxPooling2D)   (None, 16, 16, 128)       0

block3_conv1 (Conv2D)        (None, 16, 16, 256)       295168

block3_conv2 (Conv2D)        (None, 16, 16, 256)       590080

block3_conv3 (Conv2D)        (None, 16, 16, 256)       590080

block3_pool (MaxPooling2D)   (None, 8, 8, 256)         0

block4_conv1 (Conv2D)        (None, 8, 8, 512)         1180160

block4_conv2 (Conv2D)        (None, 8, 8, 512)         2359808

block4_conv3 (Conv2D)        (None, 8, 8, 512)         2359808

block4_pool (MaxPooling2D)   (None, 4, 4, 512)         0

block5_conv1 (Conv2D)        (None, 4, 4, 512)         2359808

block5_conv2 (Conv2D)        (None, 4, 4, 512)         2359808

block5_conv3 (Conv2D)        (None, 4, 4, 512)         2359808

block5_pool (MaxPooling2D)   (None, 2, 2, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
```

Figure 4 Model VGG16 Summary



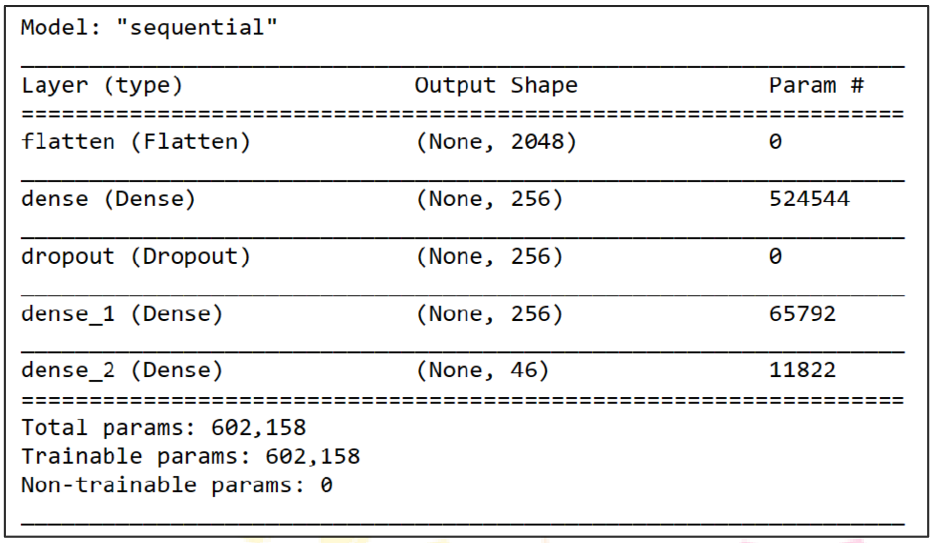
```
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 2048)              0

dense (Dense)                (None, 256)               524544

dropout (Dropout)            (None, 256)               0

dense_1 (Dense)              (None, 1)                 257
=================================================================
Total params: 524,801
Trainable params: 524,801
Non-trainable params: 0
```

Figure 5 Sequential Model Summary for Hand-Written Hindi Character Detection

Next, the model is compiled using Adam [16] optimizer and the loss function used is binary cross-entropy to compute the loss between actual and predicted values. The model is trained using train features and train labels. The epoch is set to 100 and the batch size to 32. After training the model it is used for prediction by passing random images from the test directory.

4.1. MODEL FOR OBJECT RECOGNITION

In section 4.1 experiment was performed to obtain a model which will detect the presence of a handwritten character in an image. The model was further enhanced to recognize the detected character.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 256)               524544
_________________________________________________________________
dropout (Dropout)            (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               65792
_________________________________________________________________
dense_2 (Dense)              (None, 46)                11822
=================================================================
Total params: 602,158
Trainable params: 602,158
Non-trainable params: 0
_________________________________________________________________
```

Figure 6 Sequential Model Summary for Hand-Written Hindi Character Recognition

The model is compiled using the Adam optimizer and the loss function used is sparse categorical cross-entropy to compute the loss between actual and predicted values. The model is trained using train features and train labels. The epoch is set to 100 and the batch size to 32. After training the model it is used for prediction by passing random images from the test directory.

5. RESULTS

5.1 The experimental results to test the accuracy of the Object detection model are as follows:

a. The train features and train labels and test features and test labels are extracted as shown below:
Found 5875 images belonging to 2 classes.
Found 98 images belonging to 2 classes.

b. The two classes here are labelled 0 and 1 for images without handwritten Hindi character and images with handwritten Hindi character. The test labels are as follows:

[1. 0. 0. 0. 1. 0. 0. 0. 1. 0. 1. 1. 1. 0. 0. 1. 0. 0. 0. 1. 1. 0. 0. 1.1. 0. 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 1. 1. 1.0. 0. 0. 0. 0
. 0. 0. 1. 1. 1. 0. 1. 0. 0. 1. 1. 1. 0. 0. 1. 1. 0. 0. 1.0. 1. 1. 1. 1. 1. 0. 0. 0. 1. 1. 1. 1. 0. 1. 0. 0. 1. 1. 1. 0. 1. 1. 1. 1. 0.]
(98,)

c. For feature extraction the 98 test features were selected randomly from both background and hi folder. It was observed that from background folder 43 instances with 11 redundant and 37 instances from hi with 7 redundant were chosen.

d. The model accuracy plot is useful to comment on the performance of the model. It provides insight into overfitting, underfitting or data leakage issues of the model. The model accuracy obtained is as shown in the figure 4. For 100 epochs it is observed that both the training accuracy and validation accuracy are both high and close to each other. This shows that the model is performing well and is able to generalize well to new data and is making accurate predictions.

e. Out of 98 features 50 features were predicted as background and 48 were predicted as images with handwritten Hindi characters. That is 50 were labeled 0 and 48 were labeled 1. Considering the redundant features, it is observed that out of 98, 12 are wrongly classified and the model accuracy is 87.8%. Ignoring the redundant features, it was observed that out of 80 only 7 were wrongly classified. Thus, predicting the model accuracy is 91.25%.

5.2 For the object recognition model the accuracy obtained is 89.9%.

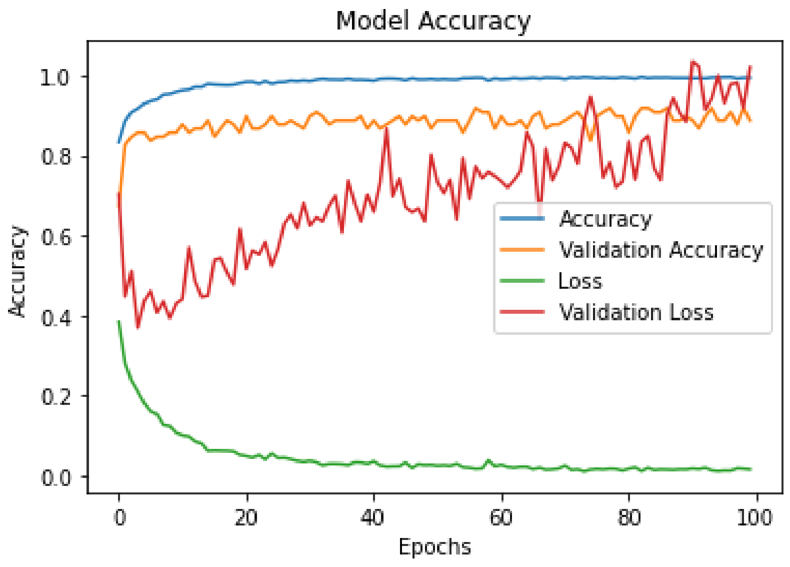


Figure 7 Model Accuracy

6. CONCLUSION

Object detection and recognition are the subsets of computer vision that are widely used in different applications. Handwritten character recognition involves pattern recognition in an image using object detection and recognition. Object detection detects the presence of the character, and object recognition identifies its category. In this paper, 46 characters of the Hindi Language are used to train the model for 46 different categories. The data set used for experimentation was a complex data set with a background image and a handwritten Hindi character on it. Factors size, regularization, transfer learning, optimization algorithm, and availability of hardware resources were taken into account for designing the CNN model of object detection and recognition. The model accuracy obtained for object detection is 91.25% and for object recognition, it is 89.9%.

REFERENCES

[1] M. A. Fischler and R. A. Elschlager, "The Representation and Matching of Pictorial Structures," in IEEE Transactions on Computers, vol. C-22, no. 1, pp. 67-92, Jan. 1973, doi: 10.1109/T-C.1973.223602.

[2] Papageorgiou, C., Poggio, T. A Trainable System for Object Detection. *International Journal of Computer Vision* **38**, 15–33 (2000). https://doi.org/10.1023/A:1008162616689

[3] G. Gualdi, A. Prati and R. Cucchiara, "Multistage Particle Windows for Fast and Accurate Object Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 8, pp. 1589-1604, Aug. 2012, doi: 10.1109/TPAMI.2011.247.

[4] W. Ouyang and X. Wang, "Joint Deep Learning for Pedestrian Detection," 2013 IEEE International Conference on Computer Vision, 2013, pp. 2056-2063, doi: 10.1109/ICCV.2013.257.

[5] Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1, pp. I-I. Ieee, 2001.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

[7] Kukačka, Jan, Vladimir Golkov, and Daniel Cremers. "Regularization for deep learning: A taxonomy." *arXiv preprint arXiv:1710.10686* (2017).

[8] Desai, Chitra. "Image classification using transfer learning and deep learning." *International Journal of Engineering and Computer Science* 10, no. 9 (2021): 25394-25398.

[9] Desai, Chitra. "Comparative analysis of optimizers in deep neural networks." *International Journal of Innovative Science and Research Technology* 5, no. 10 (2020): 959-962.

[10] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[11] Chollet, Francois et al. "Keras". In: GitHub. 2015. Available online at: https://github.com/keras-team/keras

[12] Data: https://gpuhackathons.org/event/india-academia-connect-ai-hackathon (Last Access: 21 Oct 2022)

[13] Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." *Journal of big data* 6, no. 1 (2019): 1-48.

[14] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248-255. Ieee, 2009.

[15] Hara, Kazuyuki, Daisuke Saitoh, and Hayaru Shouno. "Analysis of dropout learning regarded as ensemble learning." In *Artificial Neural Networks and Machine Learning–ICANN 2016: 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6-9, 2016, Proceedings, Part II 25*, pp. 72-79. Springer International Publishing, 2016.

[16] Reddy, R. Vijaya Kumar, and U. Ravi Babu. "Handwritten Hindi character recognition using deep learning techniques." *Int J Comput Sci Eng* (2019).