Final Report Document

A Deep Learning Approach for Weather and Terrain Prediction from Satellite Imagery

Done by:

Team 36

- Mentor: Jayachitra VP
- Bharatharaj Babu – 2018503017
- Suriyaa T – 2018503063
- Keshav - 2018503531

Contents

1. Introduction

Nearly 40% of the Earth's land cover has un-identified geographic conditions of the land due to limited accessibility which severely limits our potential, so it becomes important to identify locations and detect atmospheric conditions of various lands to determine its land use. Deforestation contributes to nearly 7% of land being lost every year, due to several activities carried out by humans and it becomes important to identify these locations so that the stakeholders can respond to the situation immediately. To study the presence of habitats in remote areas, by detecting the presence of water in surrounding regions. To identify regions where agricultural practices are being carried out.

2. Literature Survey

Deep learning approaches like CNN are being used widely for a large variety of problems like weather prediction, face detection, object detection, etc. However, in the recent years, visual image detection has gained a large amount of popularity, with a lot of research being carried out in the field of convolutional neural networks (CNNs). G.J Scott et al's paper [1] presented a deep learning approach towards land image classification using CNN mainly focusing on images that were obtained using some remote sensing techniques.

Some of the pre-processing techniques that were carried out in this paper include data augmentation, and fine-tuning of the dataset by modifying the CNN layers. Transfer learning methods were carried out for the classification task using some popular models like ResNet, GoogleNet, and CaffeNet with ResNet obtaining the highest accuracy of 98%. Rakshith, Somnath et al [2] presented an approach for using VGG-16 model for the Amazon Rainforest dataset, which yielded a significantly

high accuracy of 94%. Another statistical approach was seen in Luis et al's paper [5] which makes use of the relationships among alpha and beta diversity indices, and ANOVA distributions in order to map the spatial distribution of plant indices from satellite imagery. Further applications of CNNs include the generation of a digital terrain model which are used in visualization applications, and landscape modelling.

The paper by Bulet Ayhan et al [4], have made use of transfer learning mechanisms in order to detect the type of vegetation so that a more accurate DTM model can be generated. Out of the various mechanisms tested out, the DeepLab V3+ model has produced the highest performance. In terms of the weather applications from satellite imagery, M.Elhoseiny et al's paper[3] has made use of a CNN approach for weather classification which has achieved an accuracy of 82.2% on weather satellite image datasets, which is a significant improvement on the state of the art weather classification tasks.

3. Proposed Work

The proposed work for our project involves the implementation of a deep learning models for the identification of weather and land usage criteria from the satellite image fed into the model. The task here is a multi-label classification and the dataset consists of jpeg and tiff images which contain an additional near infrared channel. Hence, we must carry out various pre-processing methods on the dataset before being fed into the model, and understand the dataset by using visualization techniques. After the data pre-processing is complete, we can then feed this data into the model, which will then be trained. The final stage involves testing and evaluation of the model using various performance metrics. Hence, the model which gives the highest performance will be chosen as the model for further assessment and deployment.

Architecture Diagram for Proposed Work

Flow Diagram for WTSI-DL method

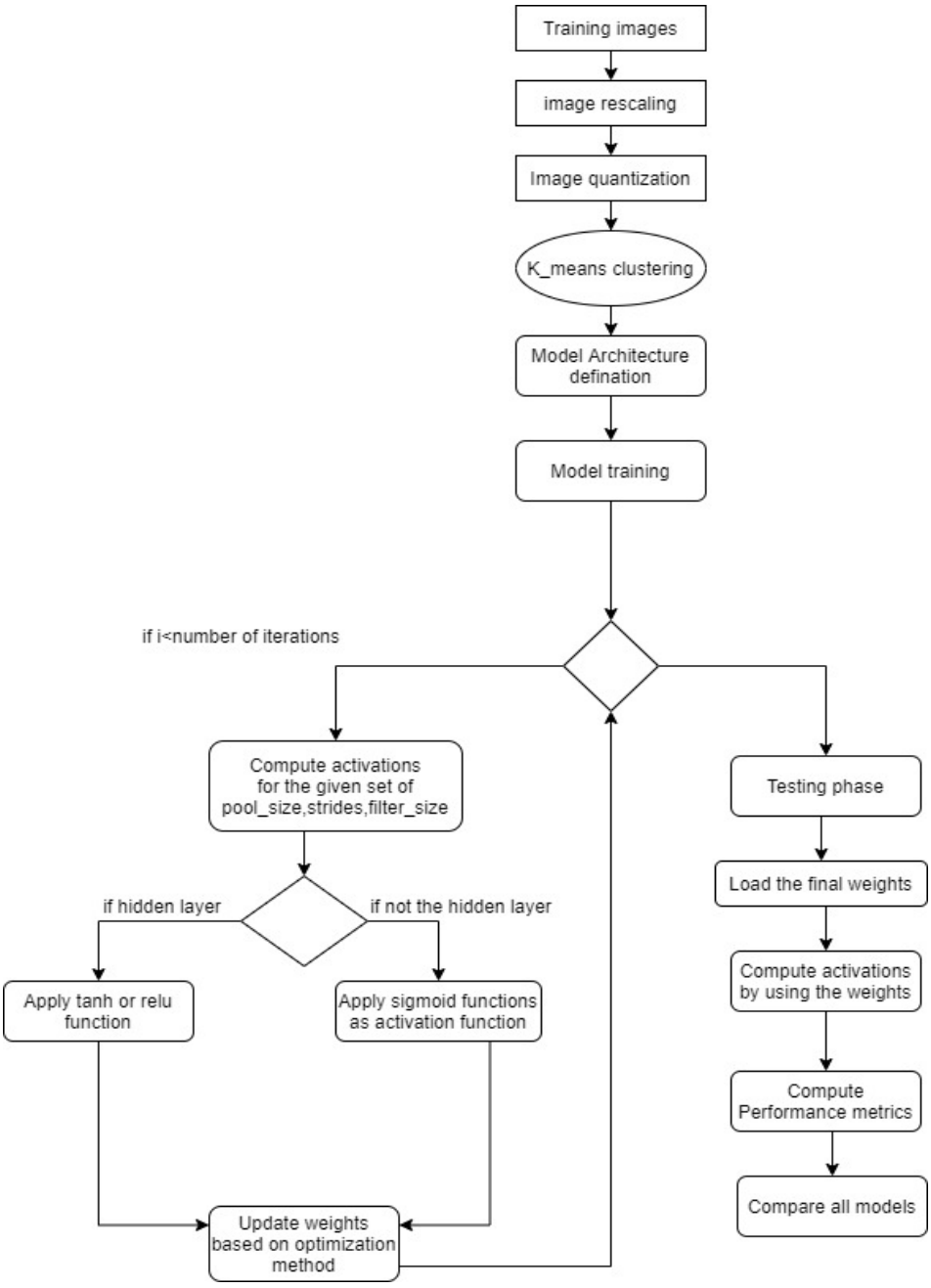


Figure 1: Flow Diagram WTSI-DL Method

Architecture Diagram for WTSI-DL method:

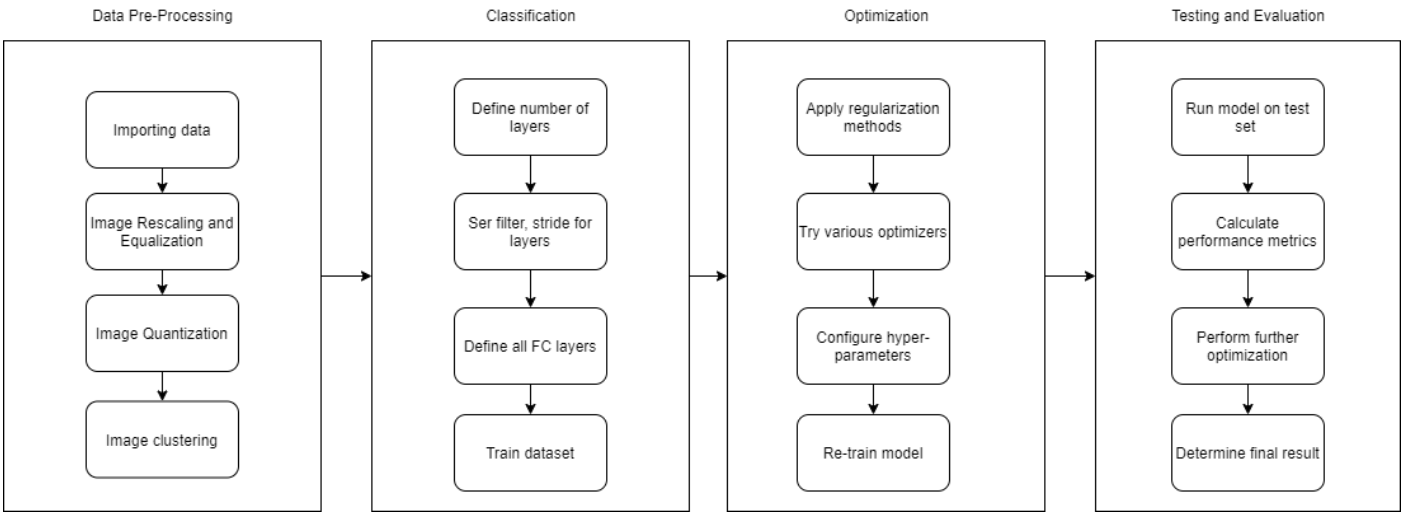


Figure 2: Architecture Diagram

The architecture diagram consists of four phases namely the data pre-processing phase, the classification phase, the optimization phase, and the final testing and evaluation phase. Each of the phases, consists of a series of steps which are highlighted in Figure 2: WTSI-DL architecture diagram method, and the flow of steps have been highlighted in Figure 1: WTSI-DL Flow Diagram

Note on Dataset:

The dataset consists of 40,479 satellite images which are available in two formats. The images are in the form of 256 X 256 pixels. The two available formats are TIF and JPEG format. The TIF images contain an additional near infrared channel which helps us capture more details from the image. The labels for the images are of 17 classes, with each image being associated with one or more labels.

Labels:

- Atmospheric Conditions: Cloudy, partly cloudy, hazy
- Common Land Cover/Use: Primary rainforest, water, habitation, agriculture road, cultivation, bare ground.
- Uncommon Land Cover/Use: Slash and burn, conventional mining, selective logging, artisanal mining, blooming, blow down.

Sample Images from the Dataset:

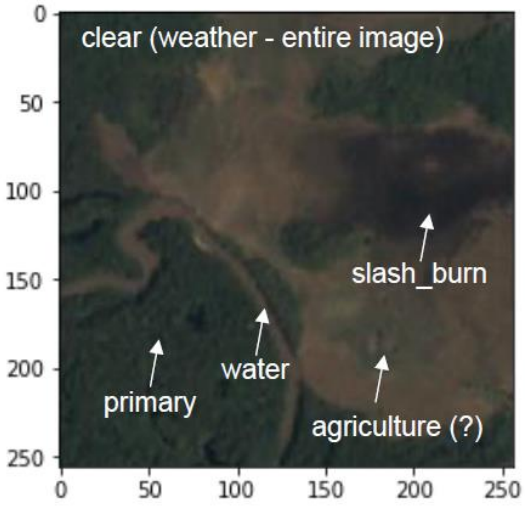


Figure 3: A sample jpeg image from the dataset with the tags marked

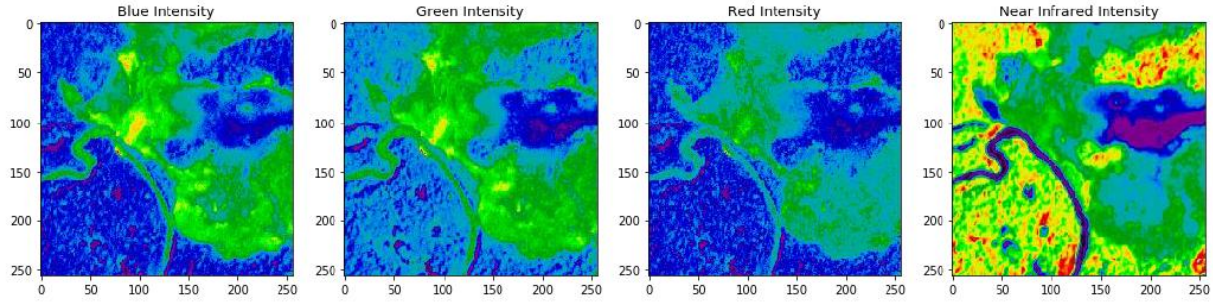


Figure 4: A sample tiff image showing the 4 channels present in it

3.1. Modules of the project

3.1.1 Data Pre-Processing:

In this phase, we will be carrying out some basic pre-processing methods on the data before feeding it into the model and also conduct some level of exploratory data analysis on the data before we begin to build models for it.

a. Importing data

In this phase, all the tif and jpeg format images are imported along with the CSV files which hold the labels. The CSV file is stored in a pandas data frame for easier processing and the images are viewed using the OpenCV package. Here the images are converted into numpy arrays and the target labels are converted into one-hot-encoded vectors.

b. Image Rescaling and Equalization:

This is an image data augmentation method, and equalizing the images can help a neural network better identify and extract features from the image. Generally an image contains unbalanced pixel intensities by having a particular pixel with high intensity.

So to resolve this issue, we rescale every image so that the pixel intensities are equally distributed.

Image Rescaling:

$$minval, maxval = minimum\_value\_pixel, maximum\_value\_pixel \quad (1)$$

$$rescaled\_image = \left(\frac{255}{maxval - minval}\right) * (image - minval)) \quad (2)$$

After image resizing the histogram for pixel intensity is equally distributed.

Comparison of Raw and Equalized Image:

The intensity values of the pixels are distributed

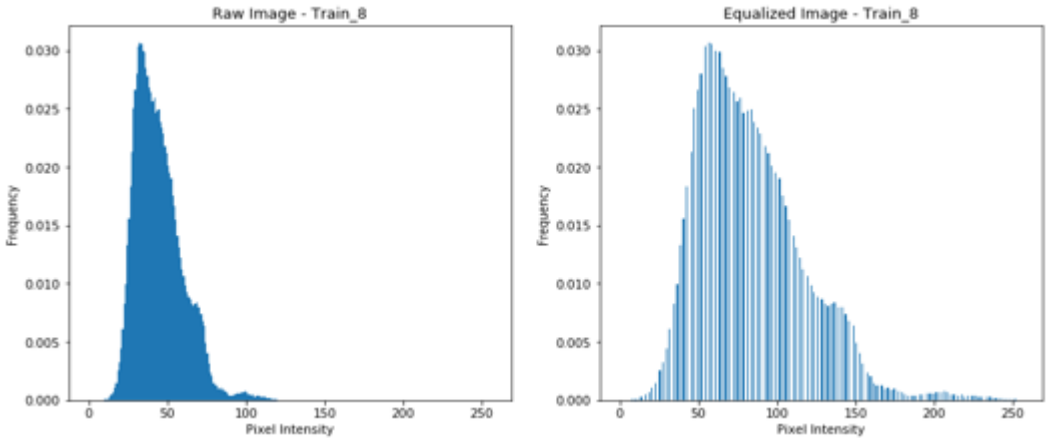


Figure 3: Histograms of Raw and Equalized Image

c. Image Quantization

K-Means clustering technique is used for extracting the dominant colors from the image. By extracting the dominant colors from the image, we can essentially replace each pixel of the image with the dominant color of the cluster each pixel belongs to.

This process is called image quantization where the number of distinct colors used is reduced in order to reduce the file size.

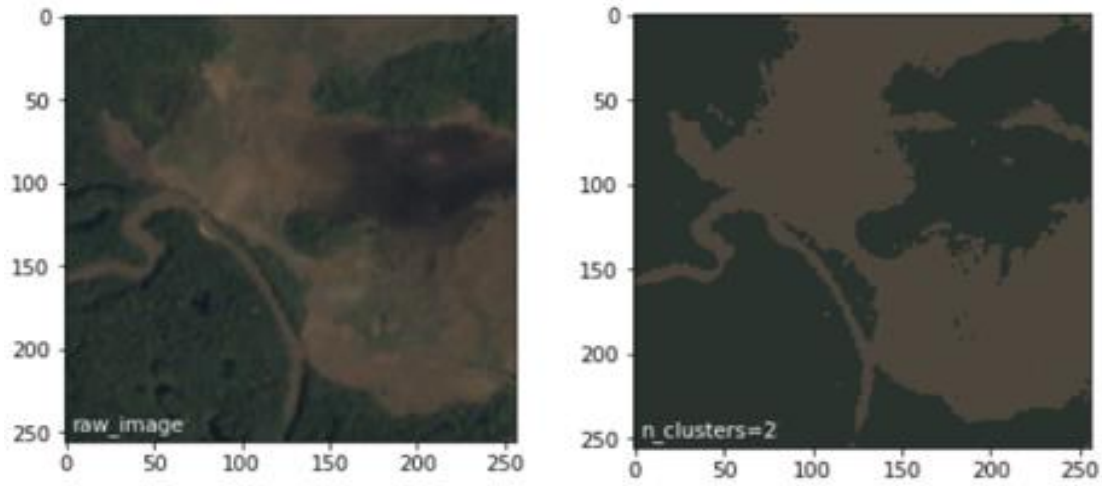


Figure 4: raw image vs quantized image

d. Image Clustering

Clustering the images together based on their respective tags helps us to better understand the relation between various tags, and helps us understand which phenomenon occurs with another. However clustering does not yield the exact classification, hence we would have to be using deep learning methods in the next sections.

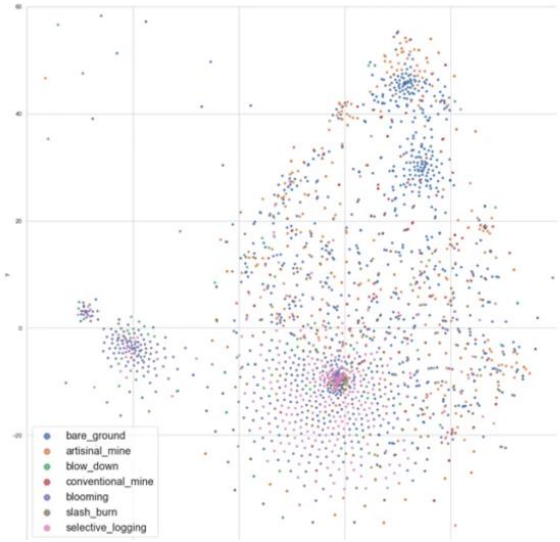


Figure 5: Images clustered based on the tags

3.1.2 Building Various Models for the Data:

A CNN model will be able to capture the temporal and spatial dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. For this project, various CNN models have been developed, and trained, evaluated and compared on the basis of the performance metrics.

Formulae

$$Input\ image\ shape: (n\ x\ n) \quad (3)$$

$$Input\ filter\ shape: (f\ x\ f) \quad (4)$$

$$Output_{image_{shape}} = (\frac{n+2*p-f}{s} + 1, \frac{n+2*p-f}{s} + 1\ ) \quad (5)$$

Here**:** n = input dimension, f = filter size, p = padding, s = stride

Models:

Various CNN models that have been implemented as part of the project include the following as listed below:

1. Base Model

The base CNN model has been implemented for augmented images of the training set, and using the Adam optimization technique. The input shape for the given model is (128, 128, 3) and the output layer contains 17 neurons.

2. Dropout Model

The dropout model has been implemented to account for the over-fitting of the training set by the base model. In dropout regularization, certain nodes are randomly chosen to be dropped out hence improving the performance. A dropout layer of 0.2 is added to each Conv-2d and max-pool block, followed by a final dropout layer of 0.5.

3. VGG-16 Model

Transfer learning is the reuse of pre trained model on the current dataset. Here we have used the pre-trained weights and the model architecture of VGG-16 model. In the VGG-16 some of the layers have been frozen for better performance. The input shape given is (128, 128, 3) and the output layer contains 17 neurons. Optimizer used is the Adam optimizer.

4. Custom Model 1: Resnset-50 concatenated with CNN Base Model

Here we have used the pre-trained weights and the model architecture of Resnet-50 model. After incorporating the Resnet-50 model, we are freezing some layers and adding some fully connected layers for better performance. We have also added some dropout layers with a reduction value of 20% to avoid overfitting. The final layer contains 17 neurons.

5. Custom Model 2: VGG-16 merged with the CNN Base Model

Here the custom model is first created and stored and then the model with VGG-16 is created. Both the models have a common output shape with 17 neurons in the last fully connected layer. Due to the similarity of the output shape, the output layers of both the models are merged into a single model. After merging them, a bunch of fully connected layers are added to train these neurons.

6. Ensemble Model: Bagging of Base Model CNNs

Here, we have developed an ensemble model which uses the bagging method for combining various models together. The final output is decided on the basis of majority rule, and each model uses bootstrapping, to sample input images and output classes randomly from the dataset for training purposes. For our project, we have set k=5, which means that were are evaluating 5 models, and hence we will get higher performance metrics from this model.

Model Architectures:

1. Base Model

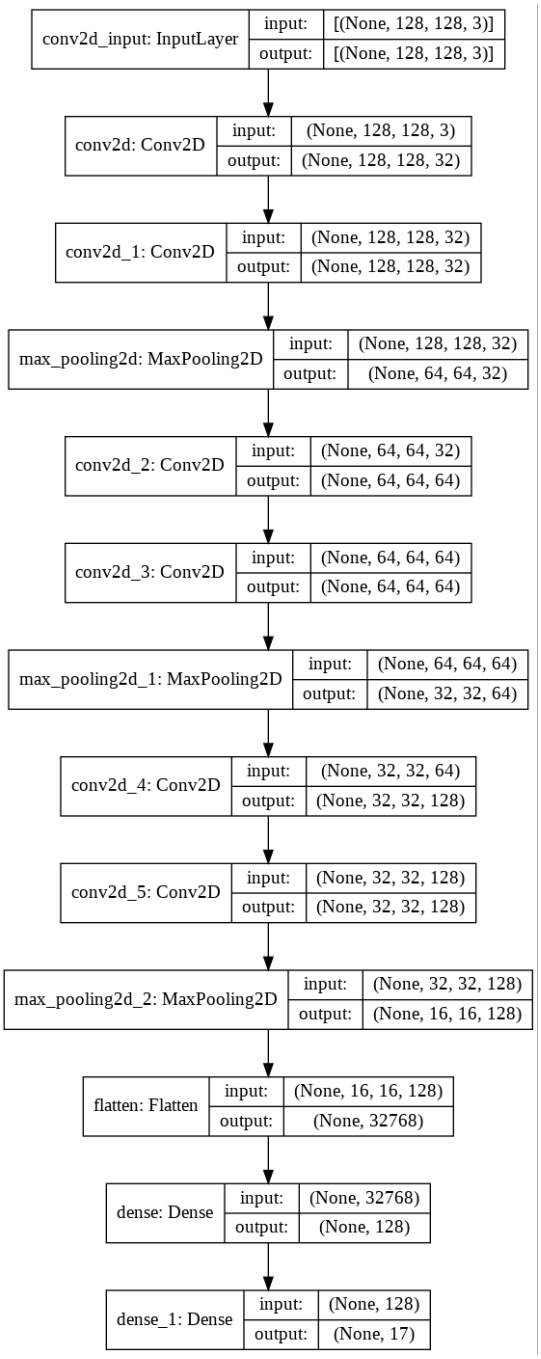


Figure 6: Base Model Architecture

2. Dropout model

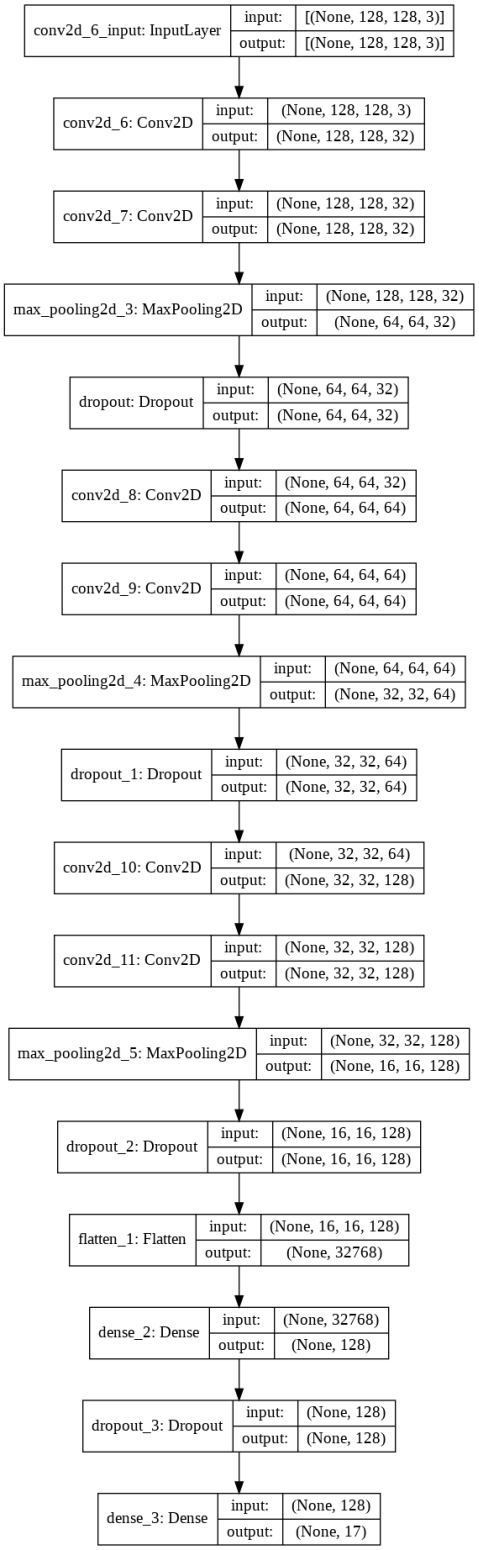


Figure 7: Dropout Model Architecture

3. VGG-16 Model

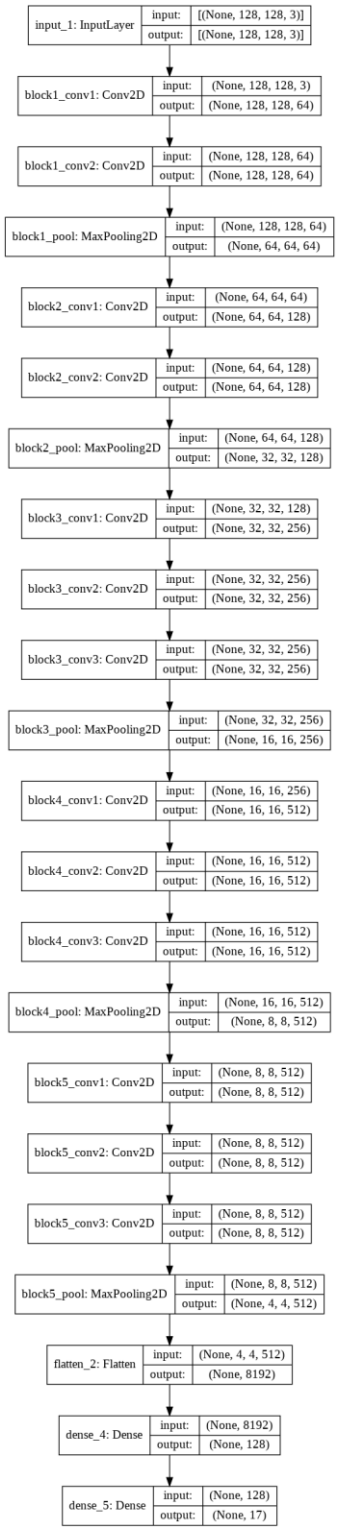


Figure 8: VGG16 Model Architecture

4. Custom Model 1:ResNet-50 Model concatenated with the base model:

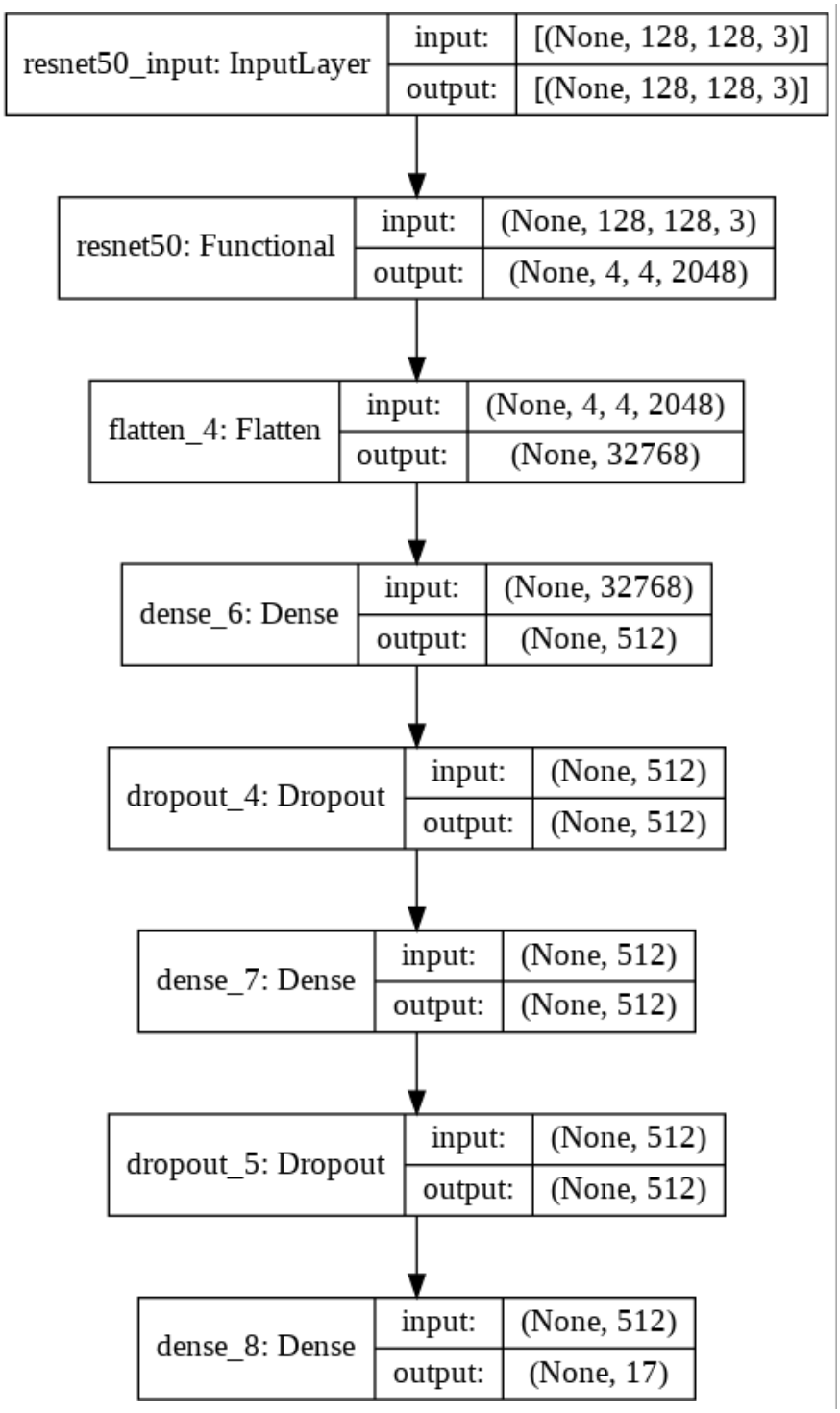


Figure 9: Custom Model 1 Architecture

5. Custom Model 2: VGG-16 merged with the base model:

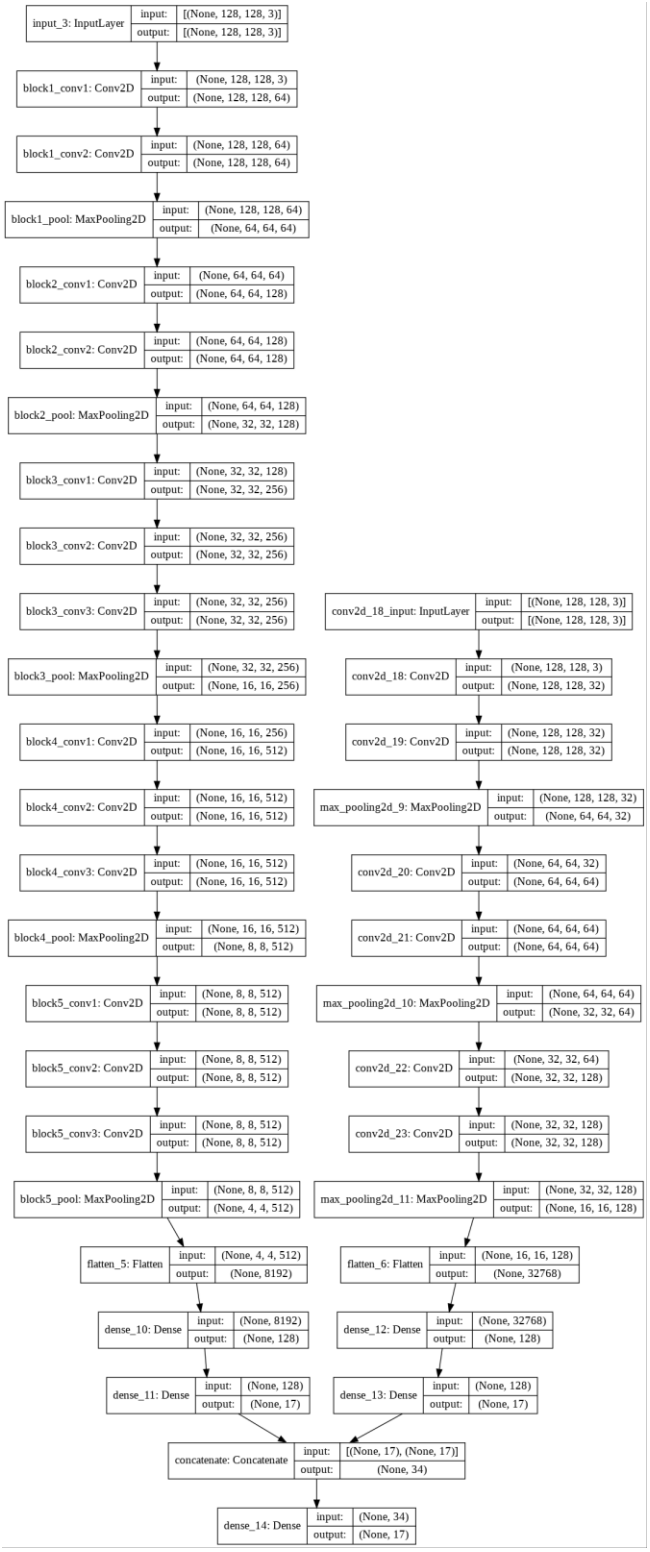


Figure 10: Custom Model 2 Architecture

These above listed models have been implemented and their performance has been measured using the below listed performance measures in Section 3.1.4.

Diagnostic curves have also been plotted in order to show how F-beta varies as training takes place, F-beta vs epochs, and also to show the variation between the cross entropy losses as training takes place.

3.1.3 Optimization Methods

a. **Dropout Regularization**

It is a computationally cheap method to regularize a deep neural network It effectively simulates a large number of networks with very different structures which makes the modes in a network generally more robust to the inputs. Generally a small amount of dropout after each block, large amount of dropout is applied to the fully connected layers.

b. **Adam Optimization**

Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models. Adam combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems.

Algorithm1: Adam Optimization

```
BEGIN:

INITIALIZE: vdW = 0, vdW = 0
INITIALIZE: sdW = 0, sdb = 0
on iteration t:
# can be mini-batch or batch gradient descent
        compute dw, db on current mini-batch
        vdW = (beta1 * vdW) + (1 - beta1) * dW                  # momentum
        vdb = (beta1 * vdb) + (1 - beta1) * db                  # momentum
        sdW = (beta2 * sdW) + (1 - beta2) * dW^2                # RMSprop
        sdb = (beta2 * sdb) + (1 - beta2) * db^2                # RMSprop
        vdW = vdW / (1 - beta1^t)                               # fixing bias
        vdb = vdb / (1 - beta1^t)                               # fixing bias
        sdW = sdW / (1 - beta2^t)                               # fixing bias
        sdb = sdb / (1 - beta2^t)                               # fixing bias
        W = W - learning_rate * vdW / (sqrt(sdW) + epsilon)
         b = B - learning_rate * vdb / (sqrt(sdb) + epsilon)

END:
```

c. Image Augmentation

Image augmentation is carried out for each image in order to produce more images by creating modified versions of the raw image. Hence, by carrying out data augmentation we get more images for each tag and hence can expect higher performance for the models. In our models, the image augmentation has been carried out by flipping images horizontally and vertically and also by rotating the images.

3.1.4 Evaluation Criteria

Some of the performance metrics which are used for evaluating the performance of the model are:

a. Precision:

Precision tells us about the fraction of images classified correctly.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (6)$$

b. Recall

Recall tells us about the percentage of total relevant results classified by the algorithm.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (7)$$

c. F-Beta Score:

F-beta score is used to evaluate the model whenever we want to give more weightage to either precision or recall, which can be varied by varying beta.

$$FBeta = (1 + \beta^2) * \frac{(precision * recall)}{(precision + recall)} \quad (8)$$

d. Mean Accuracy

Mean accuracy of an image is defined as the mean of the accuracies of all images in the dataset classified by the CNN by taking into account the number of tags correctly classified in each image.

$$Mean\ Accuracy = \sum_{i=1}^{m} \frac{TP(y_{pred}, y_{true}) + TN(y_{pred}, y_{true})}{TP(y_{pred}, y_{true}) + FP(y_{pred}, y_{true}) + TN(y_{pred}, y_{true}) + FN(y_{pred}, y_{true})} \quad (9)$$

4. Identified Tools

1. Python3 using Google Colab for GPU utilized training of model.
2. Keras and Tensorflow Backend
3. Matplotlib for Visualization

Google Colab has been chosen for the purpose of training and testing the CNN model, as it supports the Python3 language, along with extended computing capabilities such as a GPU which speeds up the training process.

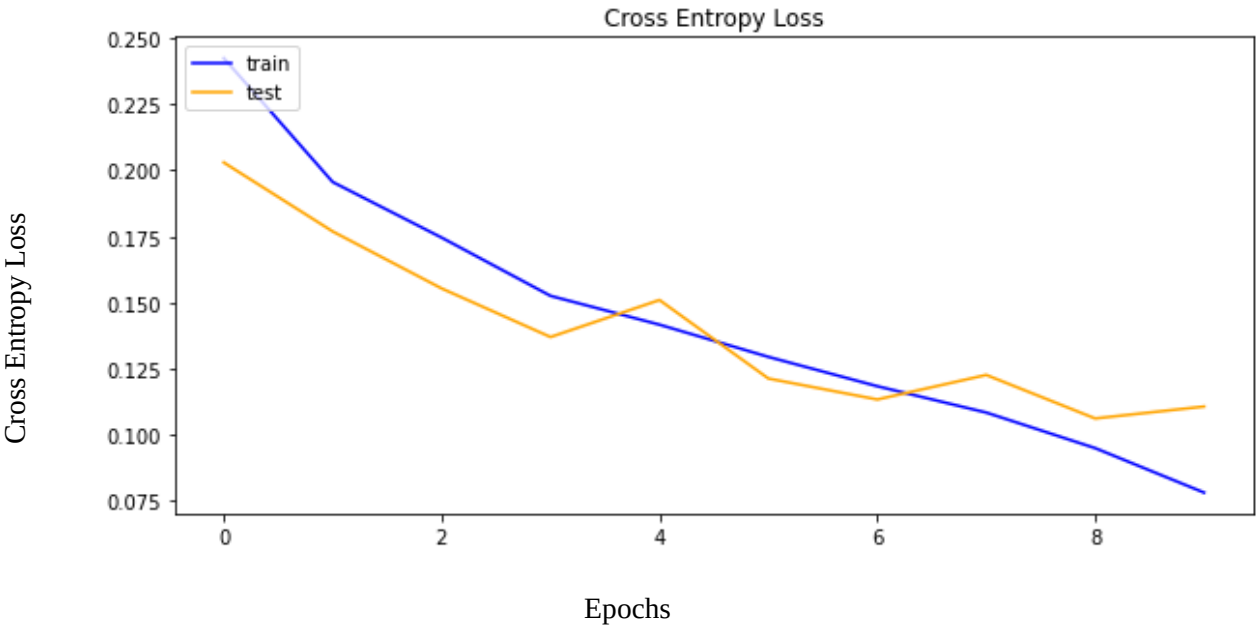
Python also supports visualization with the help of the MatplotLib package that enables us to interactively view our data.

Tensorflow and Keras are some of the most popular deep learning frameworks that make the process of defining and training the model simpler.

5. Implementation and Result Analysis

The results for the various models that have been evaluated has been visualized in this section through line graphs carried out through a series of epochs depicting the variation of cross entropy loss with epochs increasing, and the variation of F-beta which is used as the metric for evaluation vs the no of epochs.

1. Base CNN Model



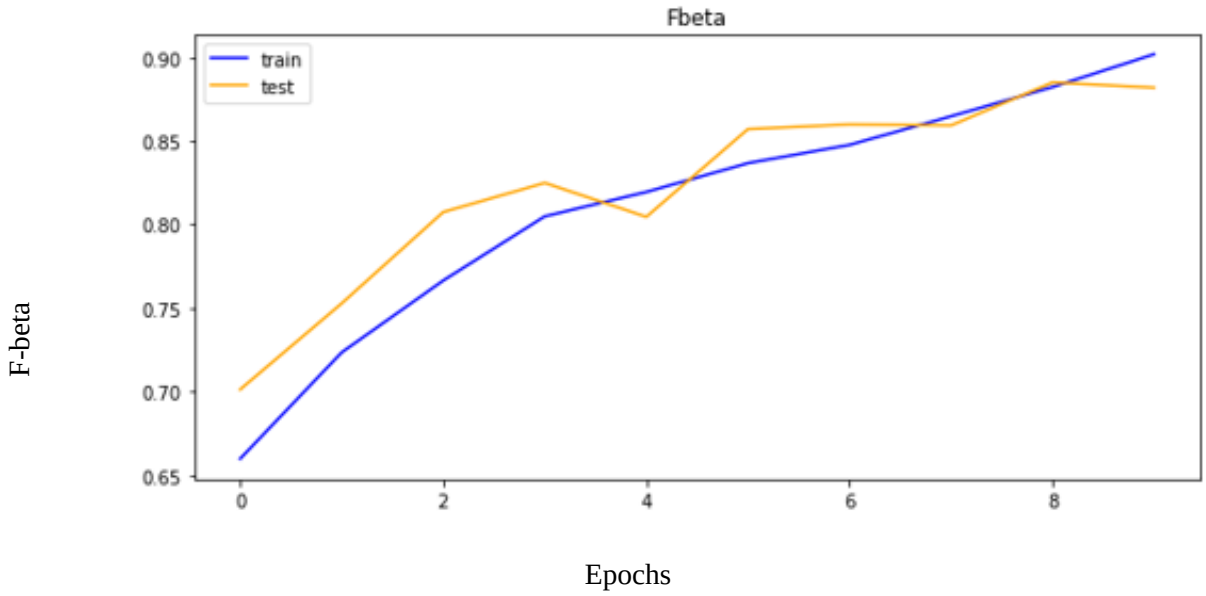
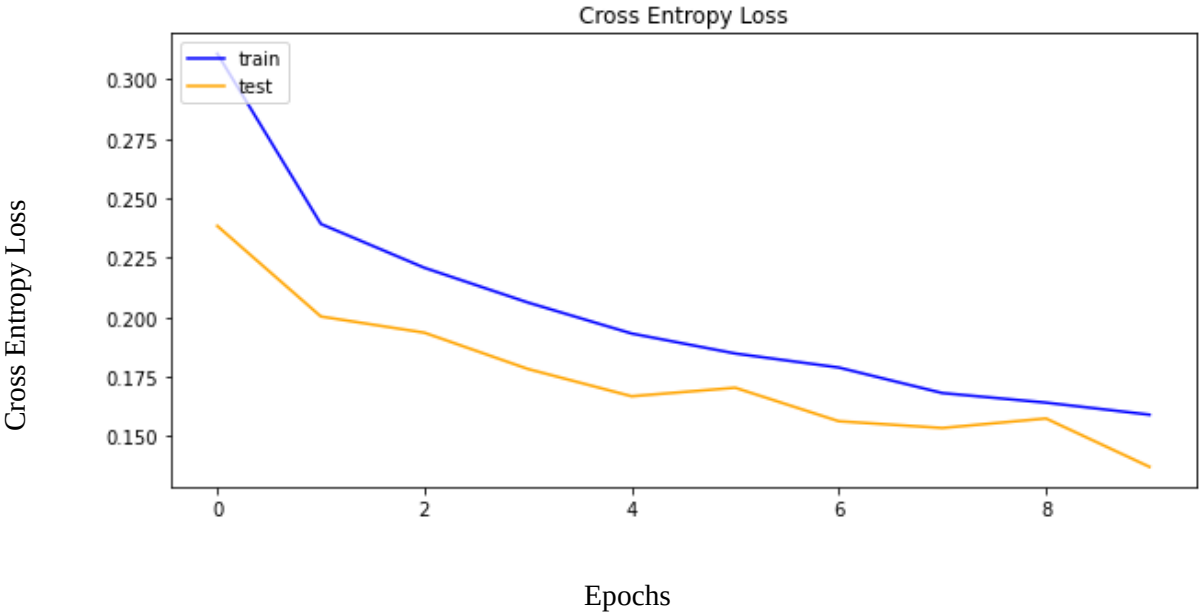


Figure 11: Cross entropy loss and F-beta for base CNN Model

2. Dropout Model with Augmentation



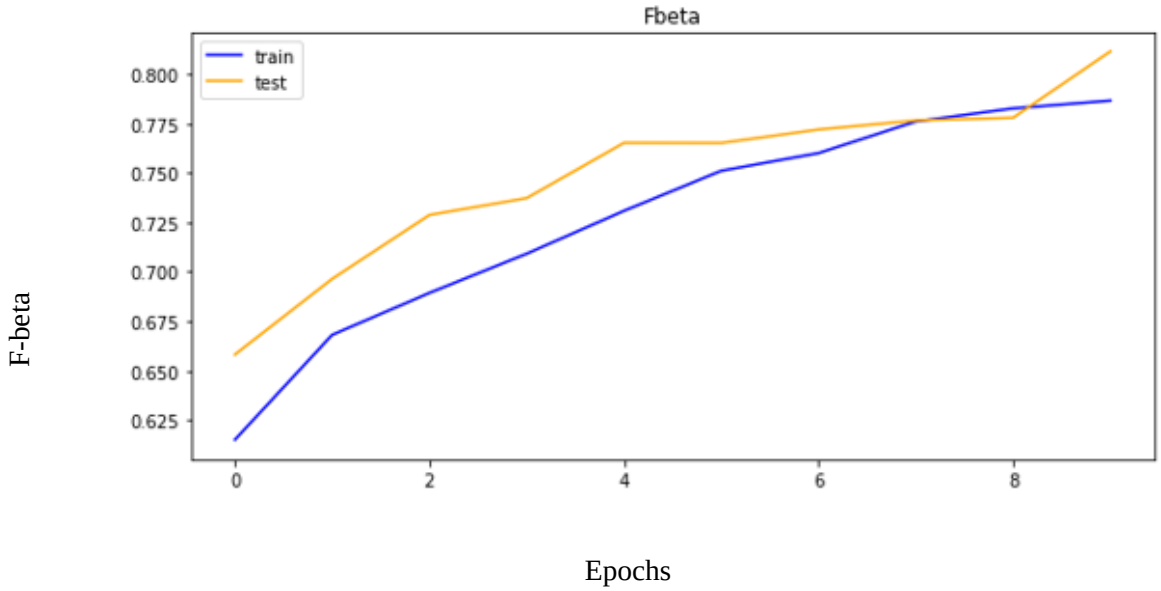


Figure 12: Cross entropy loss and F-beta for Dropout CNN Model

3.VGG-16 Model with Freezed Layers

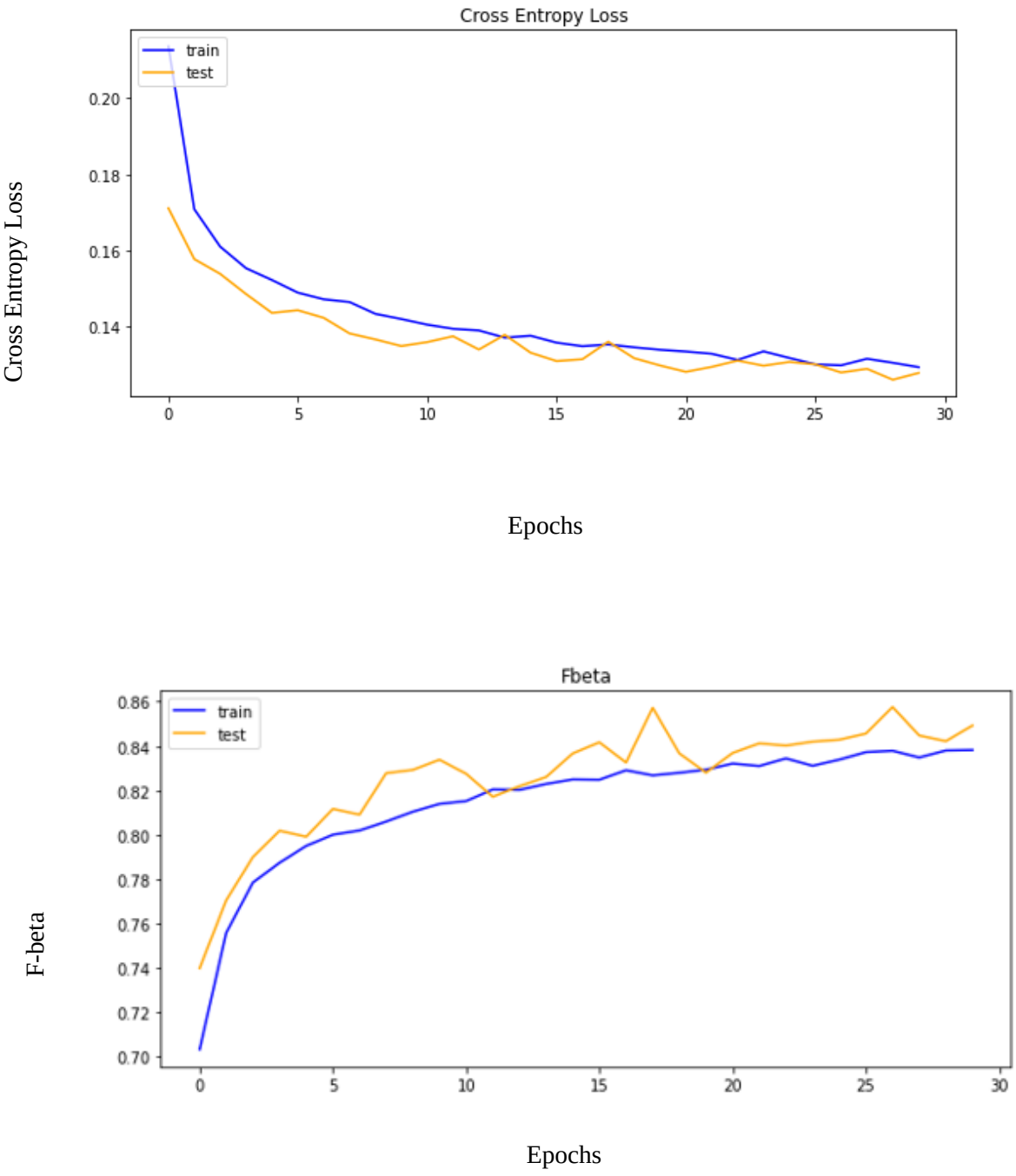


Figure 13: Cross entropy loss and F-beta for VGG-16 Model

4. Custom Model 1: ResNet Concatenated with the Base Model

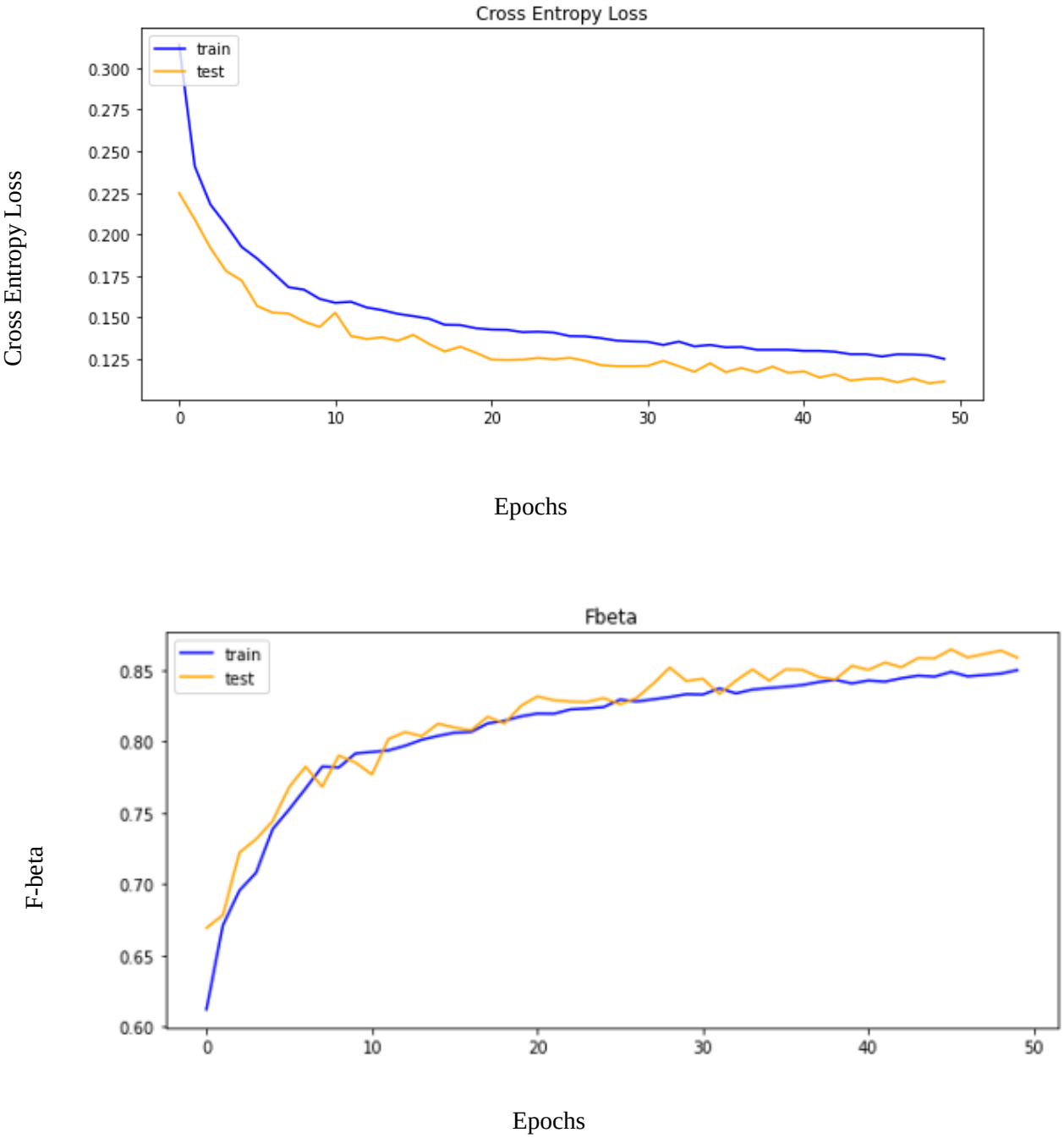
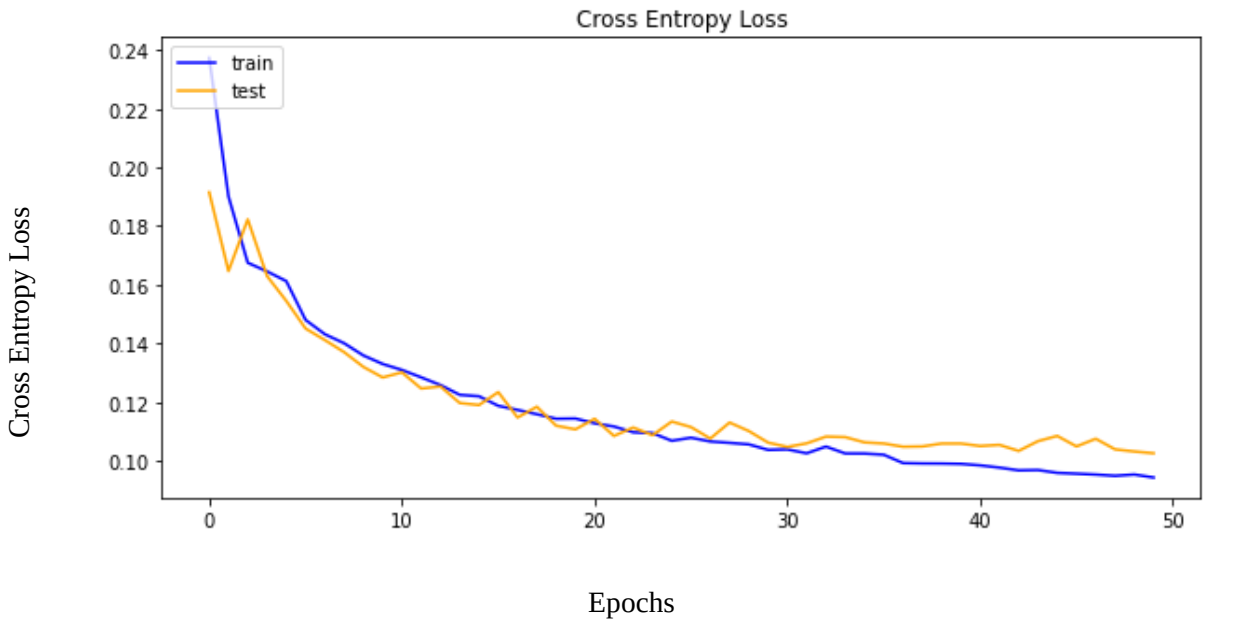


Figure 14: Cross entropy loss and F-beta for custom model 1

5. Custom Model 2: VGG-16 Model merged with the base CNN Model



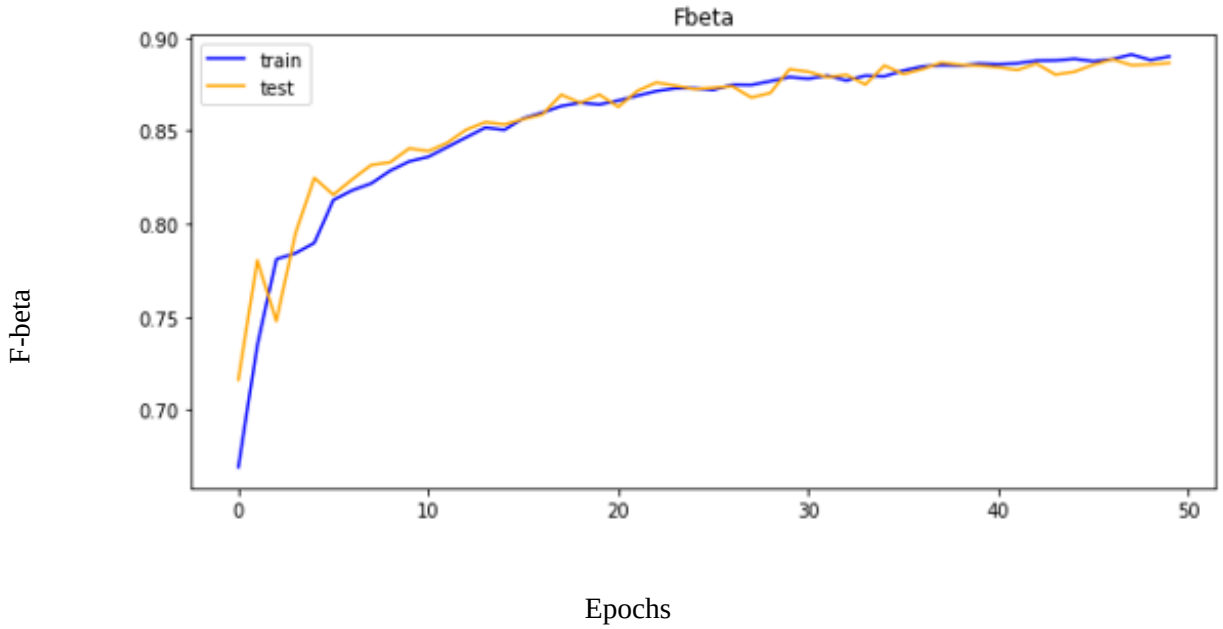


Figure 15: Cross entropy loss and F-beta for custom model 2

6. Output of the Ensemble Model: Bagging of Base CNN models

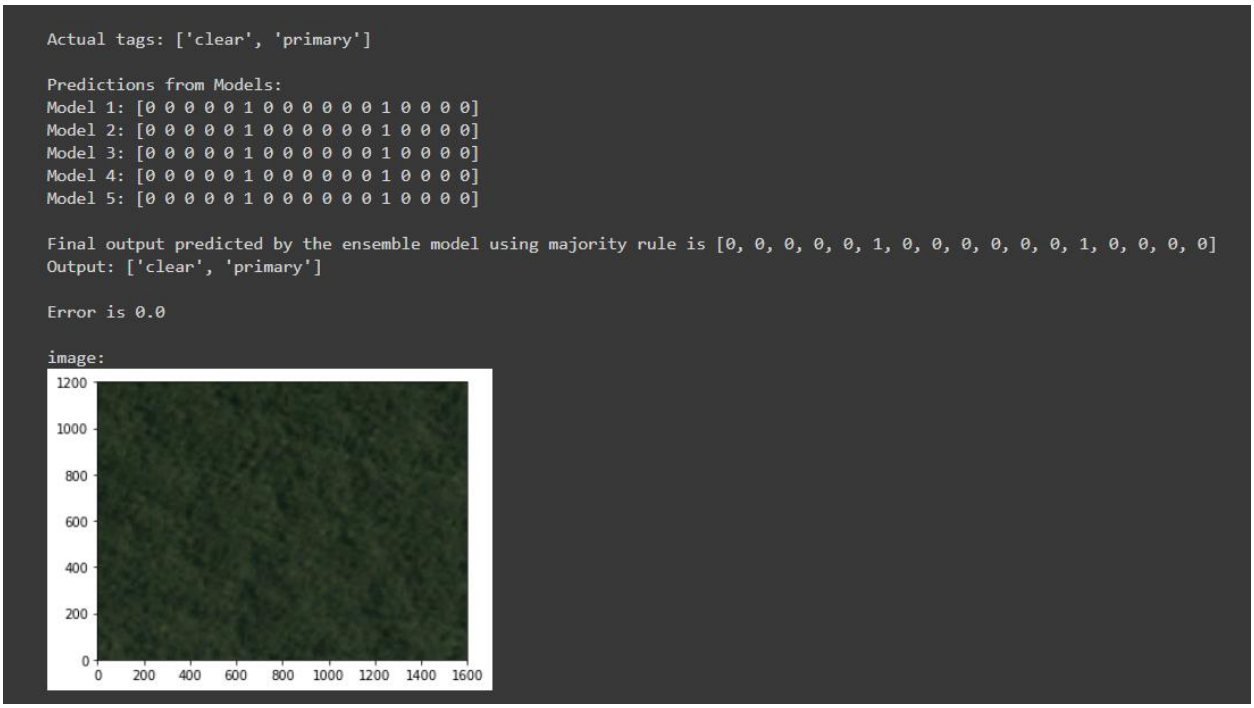


Figure 16: Output of the ensemble model

In the above example, 5 out of 5 models predicted the result correctly. Hence predictions obtained by the majority rule don't have any error for this example, and result is displayed below.

Comparison of the Various Performance Metrics

Model/ Performance Metric	Precision	Recall	F-beta	Mean Accuracy
Base CNN Model	86.39	82.66	82.62	93.94
Dropout CNN Model	89.61	79.79	80.88	94.31
VGG 16 Transfer Learning Model	89.74	84.65	84.93	95.05
Custom Model 1: ResNet + Base CNN Model	92.31	85.09	85.85	95.63
Custom Model 2: VGG-16 + Base CNN Model	91.66	88.53	88.62	96.12
Ensemble Model: Bagging of CNN Models (k=5)	90.25	95.67	93.74	97.41

Table 1: Comparison of various models

6. Conclusion

From table 1, we can see that the ensemble model gives the highest accuracy, followed by custom model 2. The highest precision is offered by Custom Model 1, for recall the highest recall value if offered by the Ensemble model, with the highest F-beta score and mean accuracy also offered by the Ensemble model.

Visualization of comparison of the various performance metrics:

Comparison of Mean Accuracies:

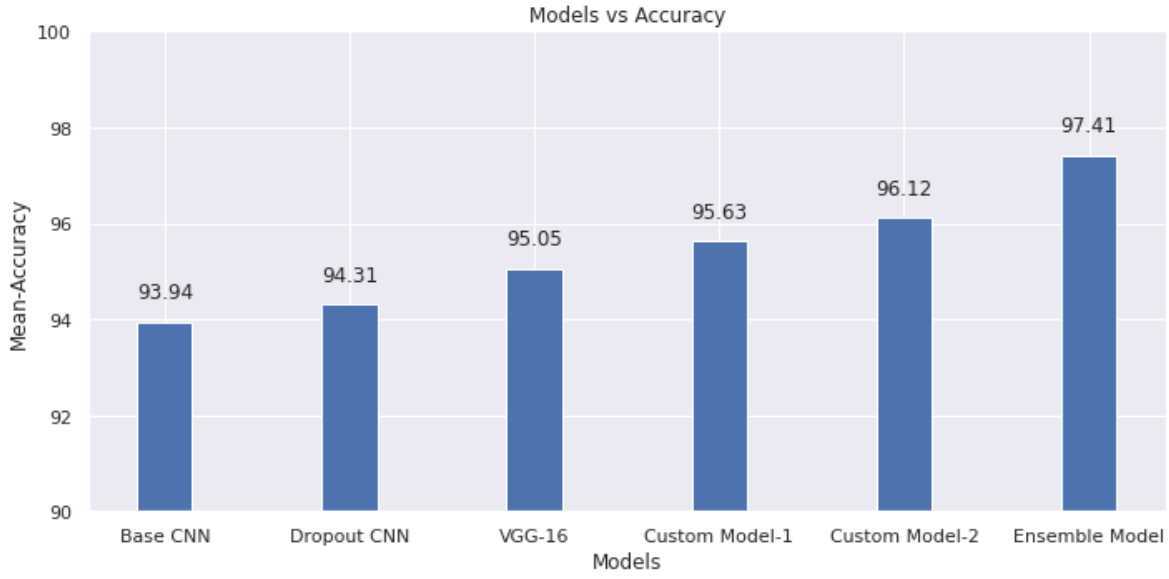


Figure 17: Models vs Mean Accuracies

Comparison of F-beta Scores:

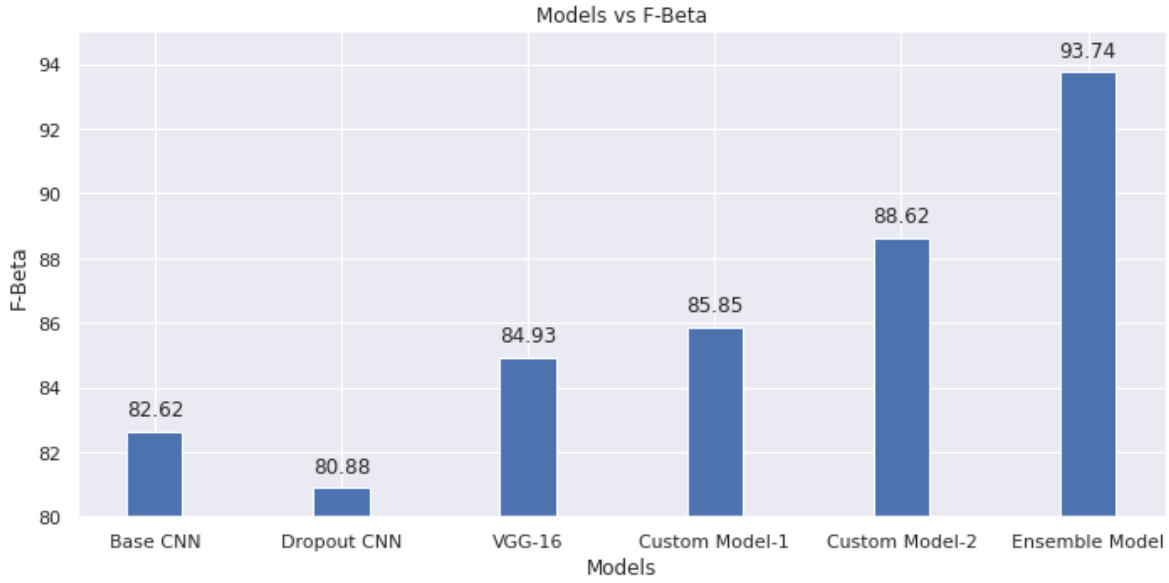


Figure 18: Models vs F-beta scores

Comparison of Precision

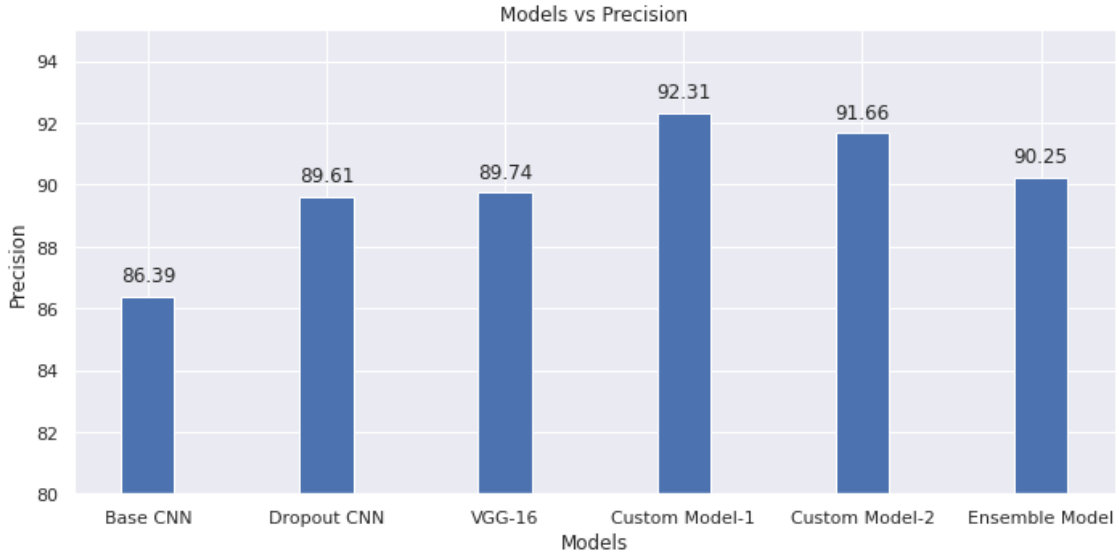


Figure 19: Models vs Precision

Comparison of recall

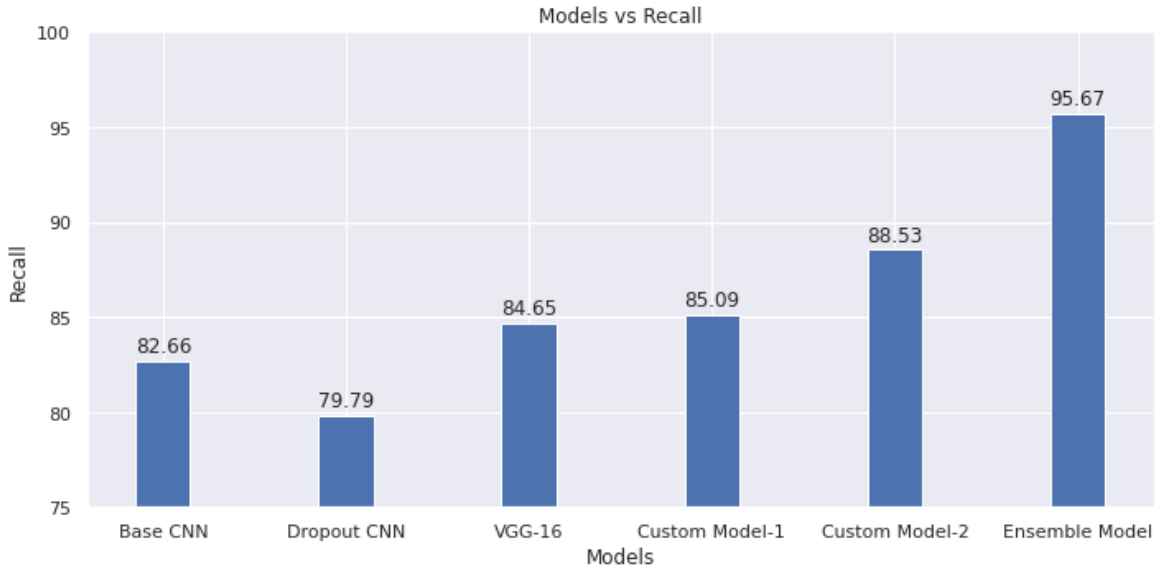


Figure 20: Models vs Recall

7. References

[1] Scott, G., England, M., Starms, W., Marcum, R., and Davis, C. 2017. Training Deep Convolutional Neural Networks for Land-Cover Classification of High-Resolution Imagery. IEEE Geoscience and Remote Sensing Letters, pp.1-5.

[2] Somnath Rakshit, Soumyadeep Debnath, and Dhiman Mondal 2018. Identifying Land Patterns from Satellite Imagery in Amazon Rainforest using Deep Learning. ArXiv, abs/1809.00340.

[3] Mohamed Elhoseiny, Sheng Huang, and A. Elgammal 2015. Weather classification with deep convolutional neural networks. 2015 IEEE International Conference on Image Processing (ICIP), pp.3349-3353.

[4] Ayhan, B., Kwan, C., Budavari, B., Kwan, L., Lu, Y., Perez, D., Li, J., Skarlatos, D., and Vlachos, M. 2020. Vegetation Detection Using Deep Learning and Conventional Methods. Remote Sensing, vol.12, no. 15,pp.1-23

[5] Hernandez-Stefanoni, J., and Ponce-Hernandez, R. 2004. Mapping the spatial distribution of plant diversity indices using multi-spectral satellite image classification and field measurements. Biodiversity and Conservation, Springer, vol.13, pp.2599-2621.

[6] W. Xiao, X. Huang, F. He, J. Silva, S. Emrani, and A. Chaudhuri 2020. Online Robust Principal Component Analysis With Change Point Detection. IEEE Transactions on Multimedia, vol 22,no 1, pp.59-68.

[7] Y. Wan, T. Li, P. Wang, S. Duan, C. Zhang, and N. Li 2021. Robust and Efficient Classification for Underground Metal Target Using Dimensionality Reduction and Machine Learning. IEEE Access, vol 9

[8] Moorthi, S.M., Misra, I., Kaur, R., Darji, N.P. and Ramakrishnan, R., 2011, September. Kernel based learning approach for satellite image classification using support vector machine. In *2011 IEEE*

Recent Advances in Intelligent Computational Systems (pp. 107-110). IEEE.

[9] Zhang, C., Li, W. and Travis, D., 2007. Gaps-fill of SLC-off Landsat ETM+ satellite image using a geostatistical approach. *International Journal of Remote Sensing*, *28*(22), pp.5103-5122.

[10] do Nascimento Bendini, H., Fonseca, L.M.G., Schwieder, M., Körting, T.S., Rufin, P., Sanches, I.D.A., Leitao, P.J. and Hostert, P., 2019. Detailed agricultural land classification in the Brazilian cerrado based on phenological information from dense satellite image time series. *International Journal of Applied Earth Observation and Geoinformation*, *82*, p.101872.

[11] Singh, A. and Singh, K.K., 2017. Satellite image classification using Genetic Algorithm trained radial basis function neural network, application to the detection of flooded areas. *Journal of Visual Communication and Image Representation*, *42*, pp.173-182.

[12] Khryashchev, V., Ivanovsky, L., Pavlov, V., Ostrovskaya, A. and Rubtsov, A., 2018, November. Comparison of different convolutional neural network architectures for satellite image segmentation. In *2018 23rd Conference of Open Innovations Association (FRUCT)* (pp. 172-179). IEEE.

[13] Rapinel, S., Clément, B., Magnanon, S., Sellin, V. and Hubert-Moy, L., 2014. Identification and mapping of natural vegetation on a coastal site using a Worldview-2 satellite image. *Journal of environmental management*, *144*, pp.236-246.

[14] Roy, A. and Inamdar, A.B., 2019. Multi-temporal land use land cover (LULC) change analysis of a dry semi-arid river basin in western India following a robust multi-sensor satellite image calibration strategy. *Heliyon*, *5*(4), p.e01478.

[15] Samson, C., Blanc-Féraud, L., Aubert, G. and Zerubia, J., 2000. A variational model for image classification and restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(5), pp.460-472.

[16] Jayanthi, S. and Vennila, C., 2020. Performance improvement in satellite image classification using adaptive supervised multi-resolution approach. *Computer Communications*, *150*, pp.200-208.

[17] Park, D.C., Jeong, T., Lee, Y. and Min, S.Y., 2011, December. Satellite image classification using a classifier integration model. In *2011 9th IEEE/ACS international conference on computer systems and applications (AICCSA)* (pp. 90-94). IEEE.

[18] Praveena, S. and Singh, S.P., 2014. Hybrid clustering algorithm and feed-forward neural network for satellite image classification. *International Journal of Engineering Science Invention*, *3*(1), pp.39-47.

[19] Nguyen, T., Han, J. and Park, D.C., 2013, October. Satellite image classification using convolutional learning. In *AIP Conference Proceedings* (Vol. 1558, No. 1, pp. 2237-2240). American Institute of Physics.

[20] Mahmon, N.A. and Ya'acob, N., 2014, August. A review on classification of satellite image using Artificial Neural Network (ANN). In *2014 IEEE 5th Control and System Graduate Research Colloquium* (pp. 153-157). IEEE.