Multi Label Image Classification on VHR Satellite Images

Raj Shah[1], Sagar Patil[2], Anish Malhotra[3] and Prof.Ranjita Asati[4]
[1, 2, 3, 4]*Department of Computer Engineering, University of Mumbai,*
Atharva College of Engineering, Malad, Mumbai, India

Abstract

Multi label Image classification using Convolutional Neural Network (CNN) is yet very difficult when it comes to performing. However, Single Label Image Classification can be performed easily and promisingly. As there are many categories of objects in a real world image, it becomes difficult to label them under various categories and also due to less number of multi-label training image and high complexity.This paper proposes a multi label image classification done on Sequential CNN model taking 4-8 layers. We have trained our model with the UC MERCED Dataset. Our model gives 88% accuracy considering Top-3 accuracy which is significantly good for performing Multi label image classification which itself being complex.

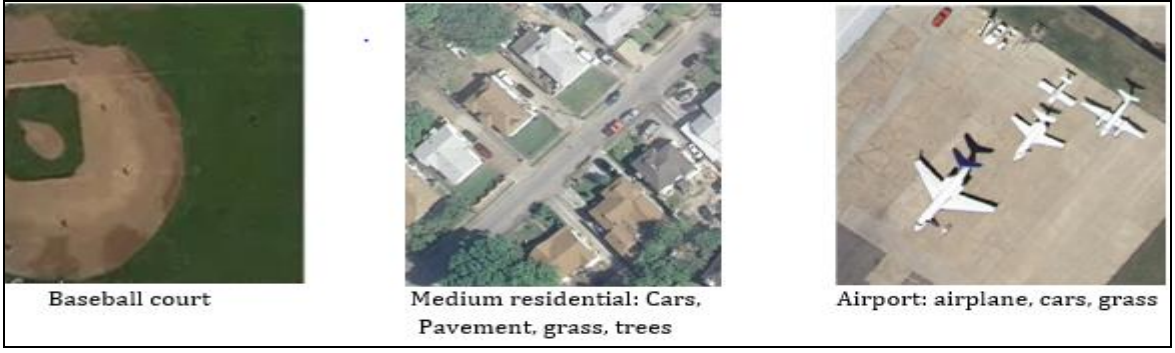
Keywords**:** *Convolutional Neural Network, image classification, Top-3 Accuracy.*

1. INTRODUCTION

In this quick moving PC age, the interest for Image Classification has been expanding rapidly. Among these undertaking, satellite Image arrangement [1] turns into a fundamental assignment. It requires a picture to be named in one or various classifications as indicated by the classes of articles present in it. We realize that in genuine world, [2] different scenes comprise of pictures for the most part of more than one class of various classifications. Along these lines, it gets hard to characterize names absolutely. In understanding to this there are different strategies that have been proposed.Their success rate on Single Label Image Classification using Convolutional neural network is been covered in recent studies. These studies have further given rise to performing Multi Label Image Classification using CNN [3]. This is still a problem to interpret since the complexity in Multi label classification is high and also due to the reason, the assumption that foreground objects are roughly aligned, which is usually true for single-label images, does not always hold for multi-label images.

Single Label Image Classification is performed on different Convolutional Neural Network using pre-trained dataset. From the total of 21 semantics of the UC MERCED Dataset VHR scene of images depends on the overall semantic theme used for training. For example, Figure 1.delineates three scenes from the UC-Merced dataset from the *Baseball court*and*Medium residential* categories,*airport* respectively[4].

In single label classification, for the last layer either the soft max or sigmoid actuation work is utilized relying upon the quantity of classes. In this work, sigmoid capacity is utilized, as the yield is as probabilities. Multi-mark classification can likewise be executed with both.Each label gives its probability of existence in the image under testing. Based on this, Top-3 labels are selected considering Top 3 high probability labels from the trained labels. All the labels are brought to a range of every other labels in terms of their probabilities.



Baseball court

Medium residential: Cars, Pavement, grass, trees

Airport: airplane, cars, grass

Fig 1: Single and Multi label images from UC MERCED Dataset

1. Related work

Multi label classification can be implemented in the following ways.

1) Problem transformation method - converts multi-label learning and inference problems onto a series of single label learning problems, and

2) Algorithm adaptation method- extends specific single label learning algorithms to handle the multi-label case.

Profound convolutional systems have additionally been stretched out to tackle the multi-name issue through different methodologies. In such manner, few have adjusted the single mark CNN via preparing with positioning misfortune work implied for multi-name expectation [6] while some utilized a pre-prepared single name CNN for multi name order by totalling the yield results from various theories through max pooling [7]. A few works have additionally considered the higher-request mark conditions to improve execution by utilizing a CNN-RNN model [8].

In the event of multi-name characterization, writing is for the most part non-existent in RS as far as we could possibly know. One of the ongoing works incorporate [9], where a CNN is effectively used to recognize the various sorts of land covers by relegating at least one marks to watched unearthly vectors of the multispectral picture pixels.

2. Methodology

Algorithm for performing Multi label Image Classification on Tensor Flow.

i. Start
ii. Install Tensor flow GPU 2.0
iii. Import necessary libraries
iv. Import Pandas, Numpy
v. Clone Dataset
vi. Training
vii. Testing
viii. Result
ix. End

3. Review of Literature

Q.Weng et al. [10] utilized pre-prepared CNN to take in profound and hearty highlights from the pictures of the UMLU informational index. The creators adjusted the CNN design by supplanting the completely associated layers of the CNN by the extraordinary learning machine classifier. Y Zhen etal.The new measurement is joined into a DCNN model for picture grouping in UMLU informational index. C Cao et al. [11] concentrated on the assessment of eight moved CNN-put together models with respect to land-use characterization undertakings and utilization of the best performing moved CNN-based model as a classifier to order and guide the land-use. The outfit of CNN is made from ResNet and DenseNet designs pre-trained on ImageNet informational collection. Extra layers of ResNet and DenseNet designs are added to make a start to finish profound learning pipeline for picture arrangement. H Parmar [12] proposed a multi-neighbourhood LBPs joined with closest neighbour classifier can accomplish a precision of 77.76% for picture grouping on UMLU informational index. In [13], K Karalas et al. use a CNN to recognize the various kinds of land covers by relegating at least one marks to watched phantom vectors of the multi-ghostly picture pixels. J Li et al. [14] used a class enactment map (CAM) encoded CNN model prepared utilizing unique RGB patches of ImageNet informational index and consideration map based class data. The parameters of the engineering are then utilized for grouping on the (UMLU) informational index.

4. Description of Dataset

The dataset used is a UC MERCED Dataset.

There are 21 semantics of image set which are listed below. [5]

There are 100 scenes for each of the following classes each image measuring 256*256 pixels.

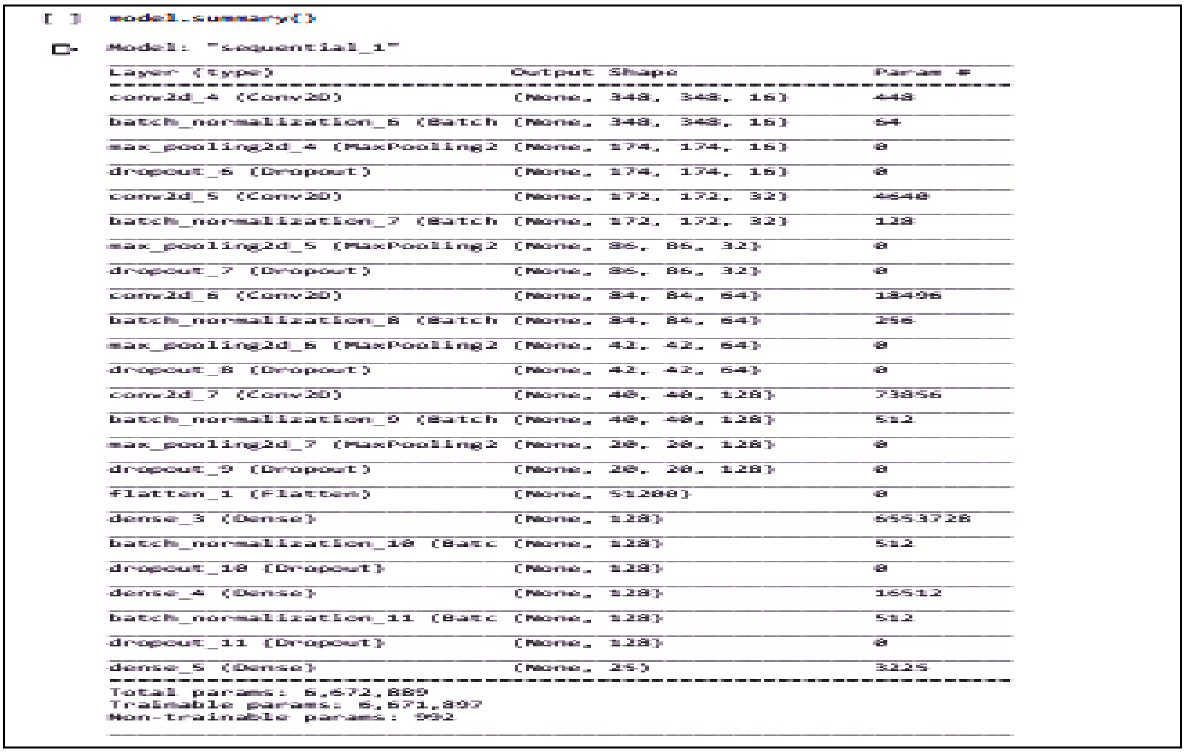
▶ *agriculture*	▶ *beach*	▶ *baseball diamond*

▶	*airplane*	▶	*chaparral*	▶	*dense residential*
▶	*forest*	▶	*freeway*	▶	*golf course*
▶	*harbour*	▶	*intersection*	▶	*medium residential*
▶	*mobile home park*	▶	*parking lot*	▶	*overpass*
▶	*river*	▶	*runway*	▶	*sparse residential*
▶	*storage tanks*	▶	*tennis court*	▶	*building*

- ReLu: Rectified Unit (ReLu) is the activation function defined as: $f(x)$ = $max(0; x)$ d)
- Max Pooling: Applies 2D max-pooling operation. Max pooling means the selection of the max value in the 2x2 filter matrix i.e input area. For example, a filter with max pooling (2,3,4,5) means '5' will be selected.
- Dropout: Dropout is used to prevent the neural network from over fitting. Dropout is established by keeping a solitary neuron in dynamic mode with some likelihood p, or setting it to dormant mode in any case. The info neuron is scaled by 1/p in the event that it isn't de-activated.
- Loss Function: Cross-entropy loss function is used which has the form: $Li = -\log(efu$ $\sum efjj)$
- Normalization: Normalization is required so that all the inputs are at a comparable range. This can be done to force the input values to a certain range.

5. Implementation and Training.

As shown in figure.2, Multi label image classification is performed on Sequential Model of Convolutional Neural Network (CNN). There are 6,672,889 number of "Total Parameters", 6,671,897 no of "Trainable Parameters" and 992 number of "Non- Trainable Parameters". Total '4' CONV2D is used. The model uses batch normalization and max-type pooling. In order to convert multi-dimensional matrix of image into simple vectors 'flatten' is used. Parameters are generated from Conv2D, Dense and Batch normalization layers. As max-pooling is used for calculation of the matrix generated by an image, it gives no parameters for training.

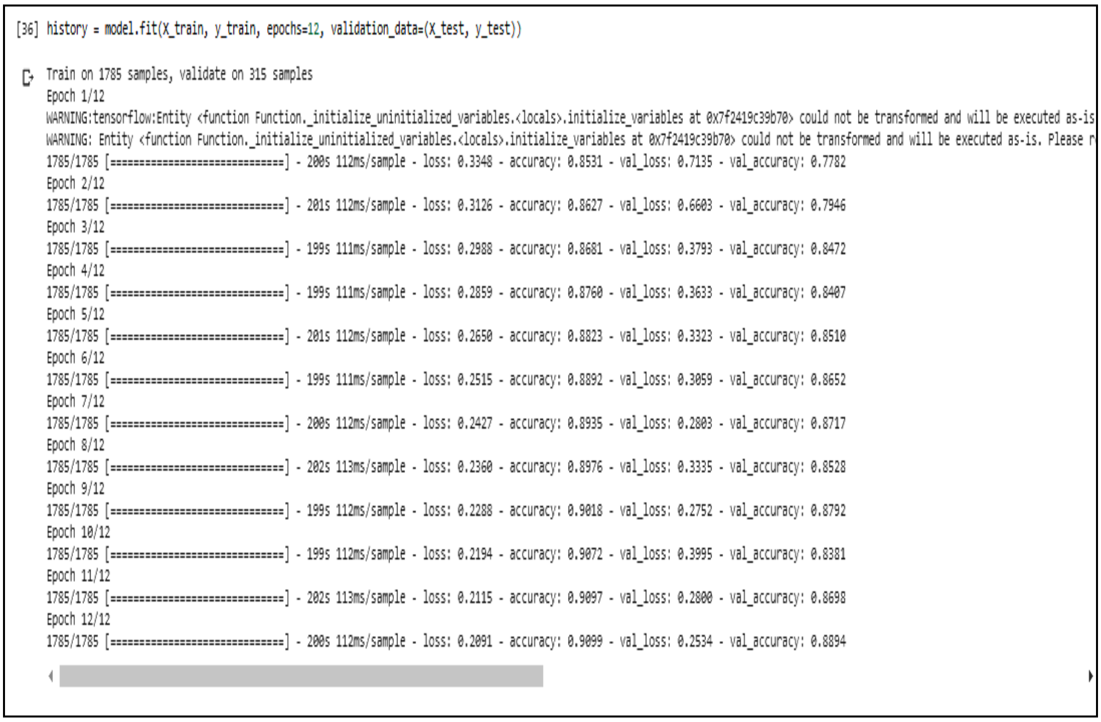


```
[ ]  model.summary()

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 348, 348, 16)      448
_________________________________________________________________
batch_normalization_6 (Batch (None, 348, 348, 16)      64
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 174, 174, 16)      0
_________________________________________________________________
dropout_6 (Dropout)          (None, 174, 174, 16)      0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 172, 172, 32)      4640
_________________________________________________________________
batch_normalization_7 (Batch (None, 172, 172, 32)      128
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 86, 86, 32)        0
_________________________________________________________________
dropout_7 (Dropout)          (None, 86, 86, 32)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 84, 84, 64)        18496
_________________________________________________________________
batch_normalization_8 (Batch (None, 84, 84, 64)        256
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 42, 42, 64)        0
_________________________________________________________________
dropout_8 (Dropout)          (None, 42, 42, 64)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 40, 40, 128)       73856
_________________________________________________________________
batch_normalization_9 (Batch (None, 40, 40, 128)       512
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 20, 20, 128)       0
_________________________________________________________________
dropout_9 (Dropout)          (None, 20, 20, 128)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 51200)             0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               6553728
_________________________________________________________________
batch_normalization_10 (Batc (None, 128)               512
_________________________________________________________________
dropout_10 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               16512
_________________________________________________________________
batch_normalization_11 (Batc (None, 128)               512
_________________________________________________________________
dropout_11 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_5 (Dense)              (None, 25)                3225
=================================================================
Total params: 6,672,889
Trainable params: 6,671,897
Non-trainable params: 992
_________________________________________________________________
```

Fig.2Sequential model with description.

Training of the dataset is done using 1785 samples and 315 validated samples. A Sequential CNN model used here is trained using 12 epochs i.e training of data from start to end is done 12 times. After training of these many number of samples testing of random images. Considering each sample, there is a certain amount of loss and accuracy as shown in figure.3

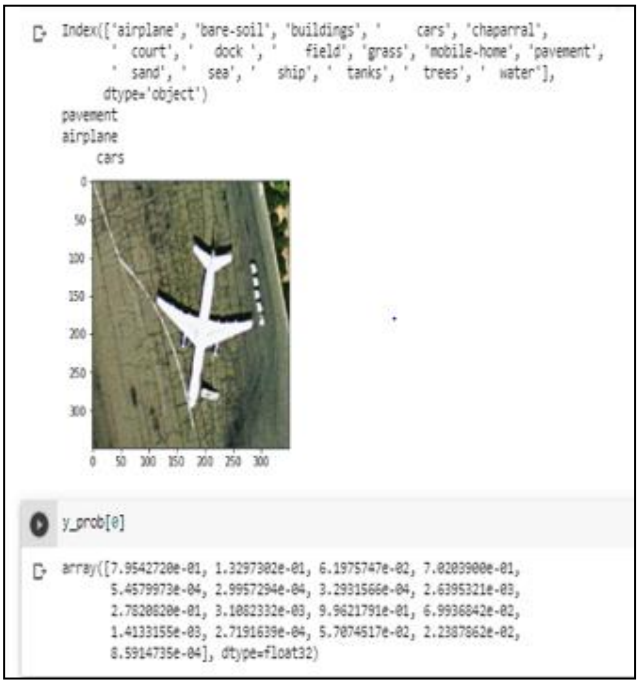


```
[36] history = model.fit(X_train, y_train, epochs=12, validation_data=(X_test, y_test))

 Train on 1785 samples, validate on 315 samples
 Epoch 1/12
 WARNING:tensorflow:Entity <function Function._initialize_uninitialized_variables.<locals>.initialize_variables at 0x7f2419c39b70> could not be transformed and will be executed as-is
 WARNING: Entity <function Function._initialize_uninitialized_variables.<locals>.initialize_variables at 0x7f2419c39b70> could not be transformed and will be executed as-is. Please r
 1785/1785 [==============================] - 200s 112ms/sample - loss: 0.3348 - accuracy: 0.8531 - val_loss: 0.7135 - val_accuracy: 0.7782
 Epoch 2/12
 1785/1785 [==============================] - 201s 112ms/sample - loss: 0.3126 - accuracy: 0.8627 - val_loss: 0.6603 - val_accuracy: 0.7946
 Epoch 3/12
 1785/1785 [==============================] - 199s 111ms/sample - loss: 0.2988 - accuracy: 0.8681 - val_loss: 0.3793 - val_accuracy: 0.8472
 Epoch 4/12
 1785/1785 [==============================] - 199s 111ms/sample - loss: 0.2859 - accuracy: 0.8760 - val_loss: 0.3633 - val_accuracy: 0.8407
 Epoch 5/12
 1785/1785 [==============================] - 201s 112ms/sample - loss: 0.2650 - accuracy: 0.8823 - val_loss: 0.3323 - val_accuracy: 0.8510
 Epoch 6/12
 1785/1785 [==============================] - 199s 111ms/sample - loss: 0.2515 - accuracy: 0.8892 - val_loss: 0.3059 - val_accuracy: 0.8652
 Epoch 7/12
 1785/1785 [==============================] - 200s 112ms/sample - loss: 0.2427 - accuracy: 0.8935 - val_loss: 0.2803 - val_accuracy: 0.8717
 Epoch 8/12
 1785/1785 [==============================] - 202s 113ms/sample - loss: 0.2360 - accuracy: 0.8976 - val_loss: 0.3335 - val_accuracy: 0.8528
 Epoch 9/12
 1785/1785 [==============================] - 199s 112ms/sample - loss: 0.2288 - accuracy: 0.9018 - val_loss: 0.2752 - val_accuracy: 0.8792
 Epoch 10/12
 1785/1785 [==============================] - 199s 112ms/sample - loss: 0.2194 - accuracy: 0.9072 - val_loss: 0.3995 - val_accuracy: 0.8381
 Epoch 11/12
 1785/1785 [==============================] - 202s 113ms/sample - loss: 0.2115 - accuracy: 0.9097 - val_loss: 0.2800 - val_accuracy: 0.8698
 Epoch 12/12
 1785/1785 [==============================] - 200s 112ms/sample - loss: 0.2091 - accuracy: 0.9099 - val_loss: 0.2534 - val_accuracy: 0.8894
```

Fig.3 Model training.

6. Result and Testing

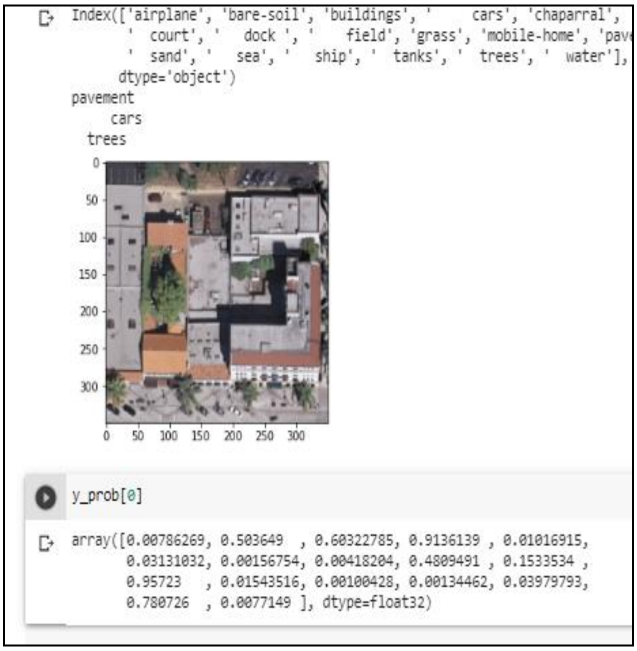


```
Index(['airplane', 'bare-soil', 'buildings', '     cars', 'chaparral',
       '  court', '   dock ', '    field', 'grass', 'mobile-home', 'pavement',
       '  sand', '   sea', '   ship', '  tanks', '  trees', '  water'],
      dtype='object')
pavement
airplane
    cars
```

```
y_prob[0]
```

```
array([7.9542720e-01, 1.3297302e-01, 6.1975747e-02, 7.0203900e-01,
       5.4579973e-04, 2.9957294e-04, 3.2931566e-04, 2.6395321e-03,
       2.7820820e-01, 3.1082332e-03, 9.9621791e-01, 6.9936842e-02,
       1.4133155e-03, 2.7191639e-04, 5.7074517e-02, 2.2387862e-02,
       8.5914735e-04], dtype=float32)
```

Fig.4 Image test(example 1)



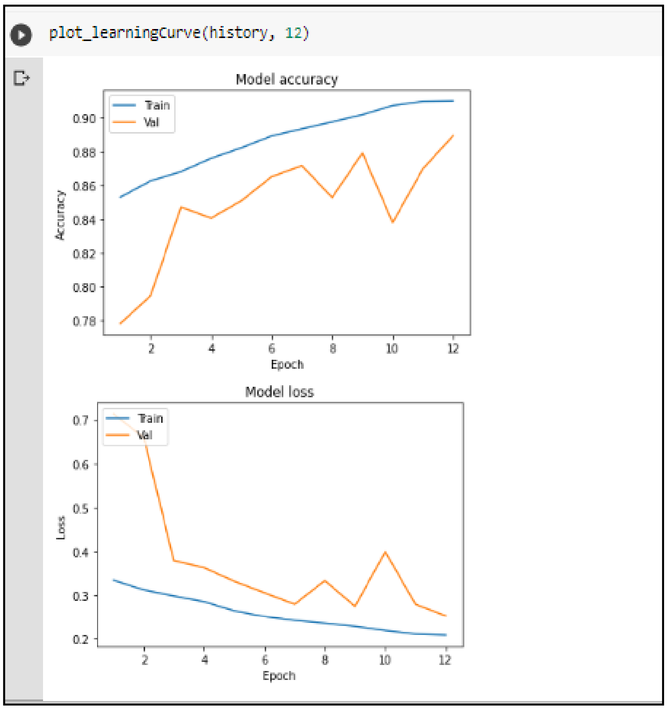
```
Index(['airplane', 'bare-soil', 'buildings', '     cars', 'chaparral',
       '  court', '   dock ', '    field', 'grass', 'mobile-home', 'pave
       '  sand', '   sea', '   ship', '  tanks', '  trees', '  water'],
      dtype='object')
pavement
     cars
  trees
```

```
y_prob[0]
```

```
array([0.00786269, 0.503649  , 0.60322785, 0.9136139 , 0.01016915,
       0.03131032, 0.00156754, 0.00418204, 0.4809491 , 0.1533534 ,
       0.95723   , 0.01543516, 0.00100428, 0.00134462, 0.03979793,
       0.780726  , 0.0077149 ], dtype=float32)
```

Fig.5 Image test(example 2)

Random image is selected and tested in the training environment. Index shows the label which are been considered while training. After testing of an image, probability of each label in that particular image is generated and Top-3 accurate labels are taken as output. Fig.4 shows Testing example 1, from the image itself it is clear that there is airplane, cars and pavement and the output result also shows the same result. Similarly Fig.5 also shows that there are trees, cars and pavement in the image.



```
plot_learningCurve(history, 12)
```

Fig.6 Graph showing accuracy and loss curve of the tested image.

we infer accuracy and loss of our selected model on the dataset. Accuracy and loss are both compared after each epoch and both the curves show variation with change in number of epoch.

Model Accuracy: Accuracy of the model is 88% in comparison to its training. As the validation curve is below training curve we say that our training is not over fitted.

Model Loss: Loss of the model is inbetween 0.2 & 0.3. Since the validation curve for loss function is above training curve we say that it not over fitting.

7. Conclusion

This work aims to perform analysis of high resolution satellite images on various Convolutional neural network (CNN). The training time for CNN is very fast. The main problem with multi label image classification is that it requires a lot of training examples, this makes the network even more complex thereby increasing the time complexity for Image classification.Thus exploring easy and significant functionalities will also require a lot of time. And this comparison is needed to make multi label image classification even simpler with lesser errors and more accuracy. And we have successfully achieved both, less error (loss) and more accuracy. By making various changes in our model, we can obtain wide range of results but the complexity may increase. As the training time is faster it will get trained within less time, thereby providing quick results. We have got 88% accuracy which is quite good compared to other complex models which give lesser percentage in terms of accuracy and it is expected that further tuning may result in higher accuracy.

Acknowledgements

We express our gratefulness to honourable Principal Mr. S. P. Kallurkar Sir for encouragement and facilities provided to us and we would like to express our deep sense of gratitude to Prof. Suvarna Pansambal, Head of Department of Computer Engineering, Atharva College of Engineering, Mumbai, Prof. Ranjita Asati for her generous support and suggestions.

References

[1] Grant J Scott, Matthew R England, William A Starms, Richard A Marcum, and Curt H Davis,“Training deep convolutional neural networks for land–cover classification of high-resolution imagery”, IEEE Geoscience and Remote Sensing Letters, 14(4):549–553, 2017.

[2] RadamanthysStivaktakis, GrigoriosTsagkatakis, and PanagiotisTsakalides,“Deep learning for multilabel land cover scene categorization using data augmentation”, IEEE Geoscience and Remote Sensing Letters, 2019

[3] Lionel Gueguen, “Classifying compound structures in satellite images: A compressed representation for fast queries”, IEEE Transactions on Geoscience and Remote Sensing, 53(4):1803–1818, 2014.

[4] Liangpei Zhang, Lefei Zhang, and Bo Du,“Deep learning for remote sensing data: A technical tutorial on the state of the art”, IEEE Geoscience and Remote Sensing Magazine, 4(2):22–40, 2016.

[5] DimitriosMarmanis, MihaiDatcu, Thomas Esch, and UweStilla, “Deep learning earth observation classification using imagenetpretrained networks”, IEEE Geoscience and Remote Sensing Letters, 13(1):105–109, 2015.

[6]Yunchao Gong, YangqingJia, Thomas Leung, Alexander Toshev, and Sergey Ioffe,“Deep convolutional ranking for multilabel image annotation”,CoRR, abs/1312.4894, 2013.

[7] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan. Hcp,“A flexible cnn framework for multi-label image classification”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(9):1901–1907, Sept 2016.

[8] J. Wang et al,“Cnn-rnn: A unified framework for multi-label image classification”, In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2285–2294, June 2016.

[9]K. Karalas, G. Tsagkatakis, M. Zervakis, and P. Tsakalides, ”Land classification using remotely sensed data: Going multilabe”, IEEE Transactions on Geoscience and Remote Sensing, 54(6):3548–3563, June 2016.

[10]QianWeng, Zhengyuan Mao, Jiawen Lin, and WenzhongGuo,“Land-use grouping by means of outrageous learning classifier dependent on profound convolutional highlights”, IEEE Geoscience and Remote Sensing Letters, 14(5):704–708, 2017.

[11]Cong Cao, SuzanaDragi'cevi'c, and Songnian Li, “Land-use change recognition with convolutional neural system strategies”, Conditions, 6(2):25, 2019.

[12]Harjot Singh Parmar, “Land use grouping utilizing multi-neighborhoodlbps”,arXiv preprint arXiv:1902.03240, 2019.

[13]KonstantinosKaralas, GrigoriosTsagkatakis, Michael Zervakis, and PanagiotisTsakalides,“Land order utilizing remotely detected information: Going multilabel”, IEEE Transactions on Geoscience and Remote Sensing, 54(6):3548–3563, 2016.

[14]Jun Li, Daoyu Lin, Yang Wang,GuangluanXu, and ChibiaoDing,“Profound discriminative portrayal learning with consideration map for scene arrangement”,arXiv preprint arXiv:1902.07967, 2019.