Classifying Amazon Rainforest cover using CNN

Multi-label classification of the rainforest cover by studying sattelite images

Divyansh Malhotra
Bharati Vidyapeeth's College of Engineering
New Delhi
divyansh.sgs@gmail.com

***Abstract*—The Amazon rainforest dataset came with a daunting problem addressing an ever-increasing issue. The dataset, which comprises of satellite images of the Amazon basin, is focused on identifying the spread of deforestation. This report proposes the use of Convolution Neural Nets to solve the problem of classifying the data procured from satellites. The complexity of this problem arises from the fact that a procured image can be classified into multiple classes. The proposed network, when trained for 20 epochs, is able to produce a validation set accuracy of 76.74% and test set accuracy of 76.70% by only training on a well-balanced subset of 6000 images from the massive dataset due to processing limitations. The solution provides the processing required to train a neural network efficiently using the dataset and to find the best threshold values for classifying the image into corresponding classes**

I. Introduction

There has been an observed increase in the balance disrupting phenomenas on the face of Earth. Most of these phenomenas can be rooted down to the intervention of humans leading to nature's disruption. Deforestation is a big issue that needs to be handled immediately. To study and tackle any problem, the starting point is collection of data and understanding the issue at hand. A good method to do that in case of deforestation, introduced by the company Planet, is to use satellite images of the forests and to classify them.

(i) Satellite images of Amazon rainforest [1]

Several methods can be implemented to solve this problem. The proposed problem here talks about using a simple Convolution Neural Network (CNN). The system is designed to train easily on a basic CPU and limited processing power to give comparable results. The report talks about all the different prototypes/ models tested and the final model that was decided upon by comparing costs and then the tweaks made to the suggested model. Finally, the accuracy achieved is discussed. This network enables us to classify a land area into various classes (17 classes). This further enables us to understand any patterns observed in the deforestation of the forest cover.

The biggest advantage of the system is that it is not limited to classifying an image into only one of the classes. Rather, it allows the system to classify into multiple classes, giving it a more human-like approach

(ii) Example of multi-labels of an image

Note:- All the results in this report have been developed using a Microsoft Surface Pro 4 with an Intel® Core(TM) i5-6300U CPU with 2.40GHz speed with 4GB RAM running Windows 10 Pro

II. Literature Review

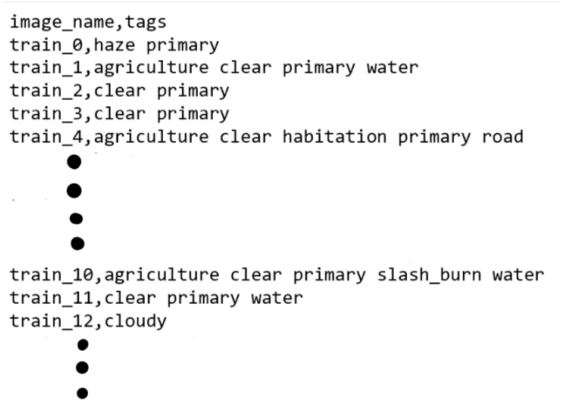
I will take this opportunity to review the dataset I am using. This dataset is available on kaggle under the problem statement https://www.kaggle.com/c/planet-understanding-the-amazon-from-space . The dataset has been provided by Planet, designer and builder of the world's largest constellation of Earth-imaging satellites, while the competition itself is hosted by Planet and SCONN at kaggle.

The dataset is a collection of satellite images (as shown in fig(i)) of the amazon rainforest basin. It is distributed in training and testing data but the labels are available only for the training data. So I only used that and will be discussing it in detail.

The training dataset has 40,479 images of 256 X 256 dimensions in RGB format. They are saved with naming of type train_x.jpg where x varies from 0 to 40,478

The labels for these images are available in a csv (Comma Separated Values) format. The labels contain two columns of data

1. The first column contains the name of the image whose labels are being provided. The labels are set in ascending order. The provision of the names ends up being handy in preprocessing of data and procuring it for processing.
2. The second column contains a string value. Each word in the string is one label of the image. Since it's a multi-label problem, the string contains the name of all the classes that the image fits in, separated by a space.



(iii) Label Format [1]

There are 17 classes in total with their representative name and distribution in the dataset tabulated below.

Class Name	**No. of images in the class**
Haze	2697
Primary	37513
Agriculture	12315
Clear	28431
Water	7411
Habitation	3660
Road	8071
Cultivation	4477
Slash_burn	209
Cloudy	2089
Partly_cloudy	7261
Conventional_mine	100
Bare_ground	862
Artisanal_mine	339
Blooming	332
Selective_logging	340
Blow_down	98

The statistics indicate that the density of data for all classes is not uniform, which proves to be a hindrance while data feeding for training, especially when working on an inferior system which cannot handle the entire data. Also, the task of getting the labels in a computable format is also a hill in the path that had to be conquered.

III. MODEL, DATASET AND METRICS

The discussed problem is being solved by implementing a Convolution Neural Network. The neural net is implemented in TensorFlow while numpy, pandas and openCV has been used for pre-processing and fetching of data.

We list out 7500 images from the entire dataset, but not randomly. We check for the classes having minimum images, and fetch them first. We continue this process (by putting up a clause to prevent the images of a used-class to be recalled, ending into an infinite loop) till the file length required (7500 in our case) has been fetched. 80% of the images will be used for training, i.e. 6000 images while the rest are equally divided into validation and testing data. We load the images only when they are to be used, saving memory.

The distribution of the classes in the derived mini dataset of 7500 images is tabulated below.

Class Name	**No. of images in the class**
Haze	2697
Primary	5192
Agriculture	1372
Clear	1911
Water	1272
Habitation	433
Road	1154
Cultivation	575
Slash_burn	209
Cloudy	2089
Partly_cloudy	803
Conventional_mine	100
Bare_ground	862
Artisanal_mine	339
Blooming	332
Selective_logging	340
Blow_down	98

The images were resized to 32 X32, and were implemented in a batch size of 64. This was common to all the tested prototypes.

Several prototypes were designed and tested. It being a multi-label problem, cost was used to be the deciding factor for the choice of basic model. All the prototypes are loosely discussed below with their corresponding cost after 20 epochs.

Note: The final layer was kept for the beginning evaluation to be default (softmax) since the cost change isn't much visible using sigmoid activation. All have sigmoid cross entropy used to evaluate cross entropy

1. Conv-Relu-MaxPool-Conv-Relu-MaxPool-Ann-Sigmoid-Drop-Ann(Gradient Descent Optimizer(1e-4)) →0.345806
2. Conv-Relu-MaxPool-Conv-Relu-MaxPool-Conv-Relu-Ann-Sigmoid-Drop-Ann(Gradient Descent Optimizer(1e-4)) →0.323698
3. Conv-Relu-MaxPool-Conv-Relu-MaxPool-Conv-Relu-Ann-Sigmoid-Drop-Ann(Adam Optimizer (1e-3)) →0.813556
4. Conv-Relu-MaxPool-Conv-Relu-MaxPool-Conv-Relu-Ann-Sigmoid-Drop-Ann(Momentum Optimizer (1e-4,1e-5)) →0.317303
5. Conv-Relu-MaxPool-Conv-Relu-MaxPool-Conv-Relu-Ann-Sigmoid-Drop-Ann(Momentum Optimizer (1e-4,1e-6)) →0.327407
6. Conv-Relu-MaxPool-Conv-tanh-MaxPool-Conv-tanh-Ann-Sigmoid-Drop-Ann(Gradient Descent Optimizer(1e-4)) →0.313783
7. Conv-tanh-MaxPool-Conv-tanh-MaxPool-Conv-tanh-Ann-Sigmoid-Drop-Ann(Momentum Optimizer (1e-4,1e-5)) →0.313754

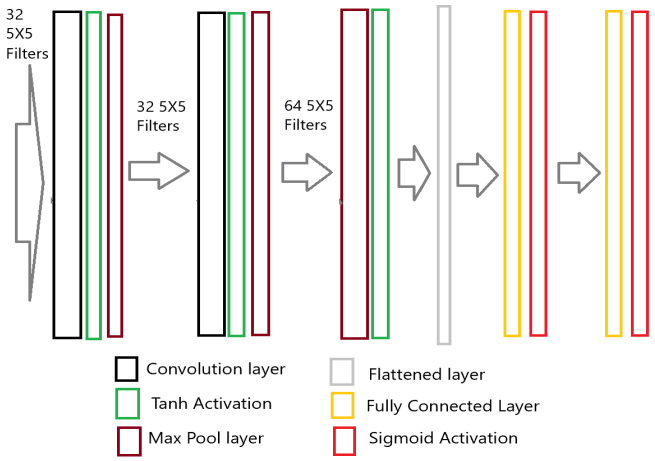
Key:

Conv-Convolution Layer, Relu-Relu activation, tanh- Tanh activation, Sigmoid-Sigmoid activation, MaxPool- Max pooling layer, Ann- Hidden layer, Drop- Dropout layer, Momentum Optimizer (Learning Rate, Momentum rate),Gradient Descent Optimizer (Learning Rate)

The 7th prototype was chosen as the base model due to the least cost of all the prototypes tested. The final model has sigmoid activation for the final layer.

Once 20 epochs were run, threshold was set of what was to be considered 1 and 0 for a class on the basis of obtaining best accuracy in training and validation data. Then validation and testing accuracy were calculated using these thresholds.

IV. Results



(iv) Model Architecture

The system, after 20 epochs of training on 6000 images, still had cost of 0.835335. Yet, the system gave an accuracy of 76.74% on validation set and 76.70% on testing set, which are good values as compared to the cost (which is high due to use of sigmoid in the end).

V. CONCLUSION

The system surely has a large cost, but the increase in accuracy indicates that the system designed, if provided more computational power and time, can exceed and even overcome the cost shortcoming. A larger (a system capable of processing it) or a more balanced dataset (originally) will surely help improve the accuracy.

On the other hand, the network, given the drawback of a not-so powerful system, gives a good accuracy. Also, one phenomenon that was noticed is, that initialization of data has a huge role to play in accuracy. A random run of the same system had produced ~83% accuracy, on the very same system with the very same parameters.

The issue of deforestation is ever-rising, and we need proper, systematic and quick data analyzing to prevent this doom. This system may prove to propel us further in that direction.

Acknowledgment

I would like to acknowledge Dr. Brejesh Lall, Mr. Abhishek Gagneja, and my teachers Mayank Bansal and Aditya Arora for their continuous efforts. It is said that a teacher is a guidepost to his students. I was grateful to have them to guide me.

References

[1] https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data -Dataset source