Deep Learning for Amazon Satellite Image Analysis

Lior Bragilevsky
School of Engineering Science
Simon Fraser University
Burnaby, BC, Canada
lbragile@sfu.ca

Ivan V. Bajić
School of Engineering Science
Simon Fraser University
Burnaby, BC, Canada
ibajic@ensc.sfu.ca

***Abstract*—Machine learning can be the key to saving the world from losing football field-sized forest areas each second. As deforestation in the Amazon basin causes devastating effects both on the ecosystem and the environment, there is urgent need to better understand and manage its changing landscape. A competition was recently conducted to develop algorithms to analyze satellite images of the Amazon. Successful algorithms will be able to detect subtle features in different image scenes, giving us the crucial data needed to be able to manage deforestation and its consequences more effectively. This paper presents our entry to the competition, the results obtained, and possible improvements to the algorithm.**

***Keywords*—Convolutional Neural Network, Deep Learning, Machine Learning, Ensembling, F-Score, Keras, Tensorflow**

I. Introduction

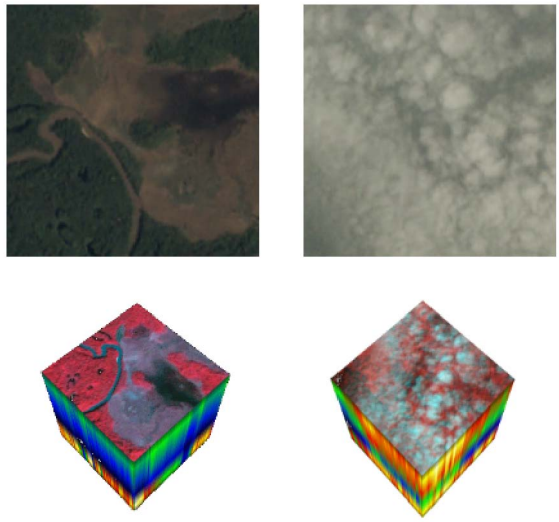
Satellite imaging has become essential in a number of areas such as monitoring floods [1] and other natural disasters [2], earthquake and tsunami prediction [3], ship tracking and navigation [4], monitoring the effects of climate change [5], and so on. Due to its importance for the Earth's climate, the Amazon rainforest has been a subject of satellite monitoring for a number of years [6], [7]. Amazon's large size and inaccessibility of some of its regions make satellite imaging the best monitoring system currently available. The ability to extract useful information from satellite images helps observe and understand the changing nature of the Amazon basin, and may help better manage deforestation and its consequences [8].

Planet Labs[1] and SCCON[2] recently organized a competition to challenge participants to analyze satellite images of the Amazon [9]. In this competition, participants were provided with various satellite images of the Amazon basin, which contain a variety of atmospheric conditions and classes of land cover and use. Some images were labeled and could be used to train various machine learning algorithms. Another set of test images was provided without labels and was used to make predictions. The predicted labels for test images were submitted to the competition for scoring. The hope was that well-trained models arising from this competition would allow for a better understanding of the deforestation process in the Amazon basin and give insight on how to better manage this process.

This work was supported in part by an NSERC USRA Award and the NSERC grant RGPIN-2016-04590.

[1]https://www.planet.com

[2]https://www.sccon.com.br/eng/



(a) agriculture, clear, primary, slash_burn, water

(b) cloudy

Fig. 1: Sample images: JPEG (top) and GeoTIFF (bottom), along with their associated labels

II. Dataset

A. Satellite Images

The provided satellite images capture scenes from the Amazon basin and come in two formats, JPEG and GeoTIFF (Geo-referenced TIFF), both with resolution of 256×256. JPEG images contain the usual red, green, and blue bands, with each pixel in each band represented by an 8-bit integer. The GeoTIFF format contains four bands: red, green, blue, and near infrared. Each pixel in each band is represented by a 16-bit integer. The near infrared band is thought to be fairly useful in analyzing atmospheric conditions, such as clouds.

Overall, 101,670 images were provided for the competition [9], in both formats. Of these, 40,479 images have the associated ground truth labels and can be used for training various machine learning and analysis models. The remaining 61,191 images were provided without labels. These constitute the test set, and label predictions made on this set were submitted to the competition for evaluation by the organizers. Sample images from the training set are shown in Fig. 1. In the figure, JPEG images are shown directly above the

978-1-5386-0700-8/17/$31.00 ©2017 IEEE

TABLE I: Image Labels with Corresponding Frequency and Index Values

(a) Labels & Frequencies [9]

Label	Frequency
primary	37840
clear	28203
agriculture	12338
road	8076
water	7262
partly_cloudy	7251
cultivation	4547
habitation	3662
haze	2695
cloudy	2330
bare_ground	859
selective_logging	340
artisinal_mine	339
blooming	332
slash_burn	209
blow_down	101
conventional_mine	100

(b) Labels & Index Values

Label	Index
blow_down	0
bare_ground	1
conventional_mine	2
blooming	3
cultivation	4
artisinal_mine	5
haze	6
primary	7
slash_burn	8
habitation	9
clear	10
road	11
selective_logging	12
partly_cloudy	13
agriculture	14
water	15
cloudy	16

corresponding GeoTIFF images. The GeoTIFF images are shown as data cubes, with the top face of the cube composed of near infrared, red and green bands. The composition of each band is visible along the sides of the GeoTIFF cubes.

B. Class Labels

Each of the provided satellite images may have up to 17 labels, making this a multi-label classification challenge [10]. For the 40,479 training images, the labels and their corresponding frequencies are summarized in Table Ia. As seen in the table, 'primary' is the most frequent label, with more than 90% of training images containing this label. On the other hand, 'blow_down' and 'conventional_mine' are the least frequent, each being associated with less than 0.25% of the training images.

C. Scoring

Participants had to use their trained models to predict the labels of each of the 61,191 test images and then submit the results to obtain a score and be ranked in the contest. Submissions were evaluated based on the F_2 score, which is computed using precision (p) and recall (r) [11]:

$$F_2 = \frac{5 \cdot p \cdot r}{4 \cdot p + r}, \tag{1}$$

where

$$p = \frac{t_p}{t_p + f_p}, \; r = \frac{t_p}{t_p + f_n}, \tag{2}$$

and t_p, f_p, and f_n represent the number of true positives, the number of false positives, and the number of false negatives, respectively.

III. Convolutional Neural Network Models

We used a combination of a deep custom Convolutional Neural Network (CNN) model along with several other CNN architectures to tackle the satellite image labelling problem. The labels were abstracted into a 17-dimensional binary label vector, where each index serves as an indicator for a certain

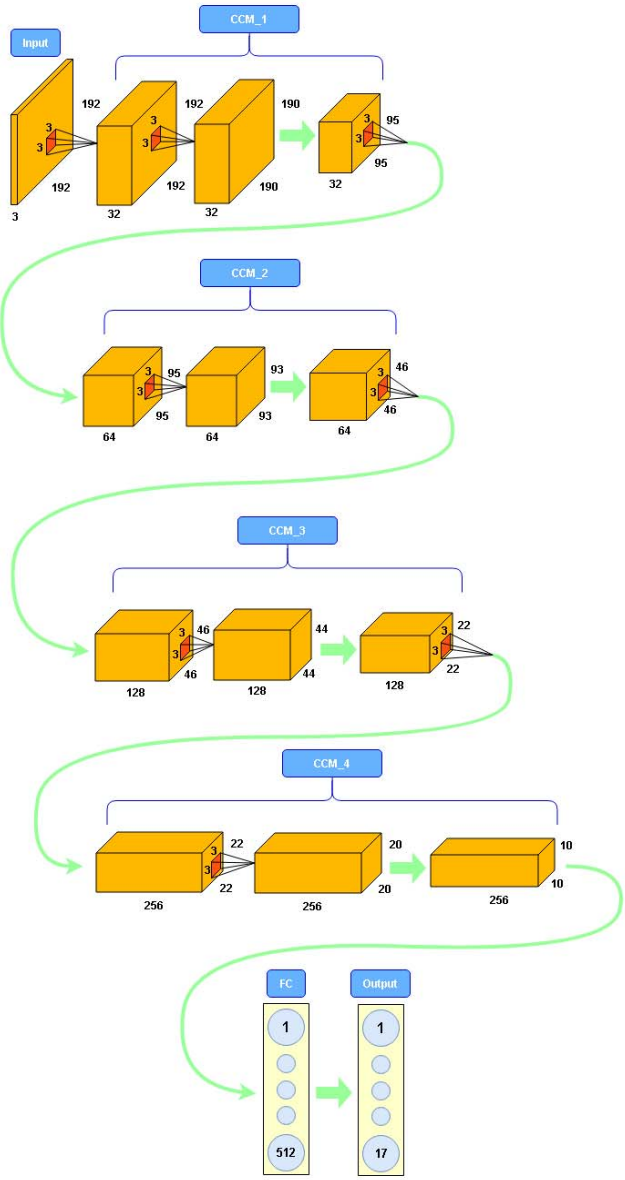


Fig. 2: Custom CNN Model

label according to Table Ib. So, for example, an image with a label 'bare_ground' would have the label vector

$$\mathbf{y} = (0, 1, 0, 0, ..., 0),$$

while the image with labels 'blow_down' and 'conventional_mine' would have the label vector

$$\mathbf{y} = (1, 0, 1, 0, ..., 0).$$

A. Custom CNN Architecture

We created a deep CNN model inspired by the well-known VGG network architecture [12]. This architecture includes a sequence of Convolution-Convolution-Maxpool (CCM) "super-layers," each of which consists of two back-to-back convolutional layers followed by max-pooling. Each convolutional layer has rectified linear unit (ReLU) activation,

TABLE II: F_2 Scores of Various Models

Model	F_2 score (%)
Xception	92.422
VGG16	92.785
VGG19	92.747
ResNet50	92.886
InceptionV3	91.888
Custom	92.677

and consists of filters whose kernel size is 3×3. The number of filters increases towards the output.

The model architecture, shown in Fig. 2, was implemented in Keras[3] using the Tensorflow backend. Four CCM super-layers are employed, and their parameters are indicated in the figure. The output of the last layer is fed to a fully connected (FC) layer with 512 units and ReLU activation, followed by the output layer with 17 units and sigmoid activation, whose outputs serve as indicators for the 17 labels. Each CCM super-layer has a dropout rate of 0.25, while the FC layer has a dropout rate of 0.5. Additionally, the input layer and the FC layer are normalized using Batch Normalization [13]. Binary crossentropy loss function is used to measure the error during training. We experimented with different numbers of CCM super-layers and FC layers, however, the above architecture performed best in our tests, with a F_2 score of 92.677%.

B. Other CNN Architectures

In addition to the custom model, we used several other well-known CNN architectures such as Xception [14], VGG16/19 [15], ResNet50 [16], and InceptionV3 [17]. To use these models, we removed their top layer and created a FC layer with 17 outputs (same as Fig. 2), to be used in our application. The training was initialized with ImageNet weights, which are available for each of these models. Once trained on our data, these models produced the F_2 scores listed in Table II.

C. Training

From the set of 40,479 labeled JPEG images, 20% (8,096) are randomly selected for validation and the rest is used for training. Each model is trained using the Adam optimization algorithm [18] in batches of 16 to 128 (deeper networks are trained using a smaller batch size), with learning rate initially set to $5 \cdot 10^{-4}$ and reduced by a factor of 10 if the validation loss does not improve for more than two epochs. When the validation loss plateaus after two reductions of the learning rate, the training is stopped. The weights from the epoch that led to the lowest validation loss value are recorded.

To increase the diversity of the training data and improve generalization, various modifications are applied to the training images in a process called *augmentation* [19]. Specifically, horizontal and vertical flipping and rotations up to 90° are applied randomly to the training images in each batch. For rotated images, the gaps created by rotation are filled in through reflection.

For each architecture, up to five models (i.e., up to five sets of weights) are trained using the above procedure. For shallower architectures such as the custom one, where training takes less time, we trained five models, while for the deeper architectures such as Inception V3, we only trained one model. Predictions on the test images are made using the average outputs of all the models of a given architecture. Since each model outputs a real value between 0 and 1 on each of its 17 outputs, the average predictions for each label will also be real values between 0 and 1. In order to generate a binary prediction for each label, thresholds were applied to these outputs. We find optimal threshold for each output by sweeping the range between 0 and 1 in steps of 0.02 and observing the F_2 score on the training images, since their ground-truth labels are known. Once all 17 thresholds are determined this way, the average predictions are binarized using these thresholds.

[3]https://keras.io/

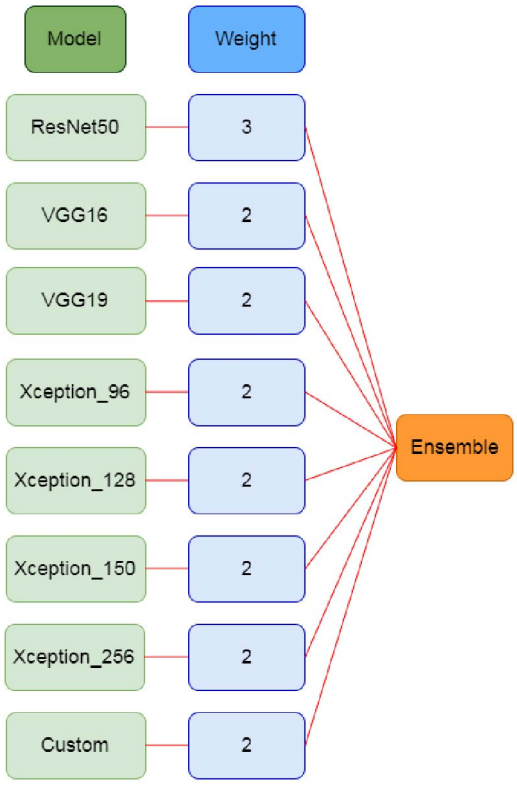


Fig. 3: Model Ensembling

D. Ensembling

Once each model generated predictions, we ensembled these predictions using weighted majority voting [20]. It became apparent that uncorrelated model predictions produced the best ensemble results. In addition, we noticed that ensembling models with the same architecture trained on varying input image dimensions marginally improves our results. This is due to the fact that larger images allow for the detection of fine features, whereas smaller images predict general features more accurately. Following these guidelines, we created a weighted majority voting ensemble as shown in Fig. 3. All models in this ensemble were trained on 256×256 JPEG images, except Xception [14], which was trained on four different resolutions: 96×96, 128×128, 150×150, and 256×256. In Fig. 3, Xception model trained on $N \times N$

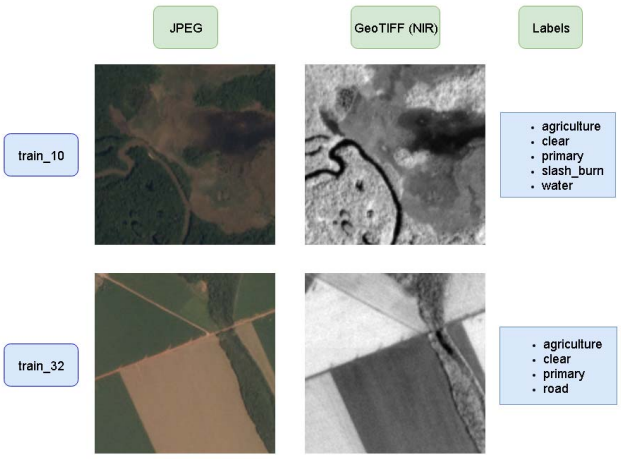


Fig. 4: JPEG vs. GeoTIFF NIR Band

images is indicated as Xception_N. The weights given to various models are indicated in Fig. 3. We decided to give ResNet50 [16] a higher weight than the rest of the models as it produced the best F_2 score (Table II).

The ensembling procedure is as follows. Let $\mathbf{y}^j$ and w^j be, respectively, the binarized predicted label vector for a given input image and the weight of the j-th model in Fig. 3. We form the vector of label votes $\mathbf{v} = (v_1, v_2, ..., v_{17})$ as

$$\mathbf{v} = \sum_{j=1}^{8} w^j \cdot \mathbf{y}^j, \tag{3}$$

where the j-th model contributes either 0 or w^j votes to the total number of votes for each label. Given the weights in Fig. 3, the maximum number of votes is $M = 17$. The final predictions $\mathbf{y} = (y_1, y_2, ..., y_{17})$ are made by applying a threshold to each label against $M/2 = 8.5$. Specifically,

$$y_i = \begin{cases} 1, & \text{if } v_i > M/2 \\ 0, & \text{else} \end{cases} \tag{4}$$

for label indices $i = 1, 2, ..., 17$. Other weight assignments are possible, so long as M is an odd number. This ensures that weighted votes v_i cannot equal the threshold $M/2$.

E. Results

Our final submission to the contest, created by the weighted majority voting ensemble described above, achieved a F_2 score of 93.008% on the test set. By comparison, the top F_2 score in the competition was 93.318%, which is only 0.310% higher than ours. Possible improvements to our method are discussed in the next section.

It is interesting to note that the F_2 score of our ensemble (93.008%) is higher than the F_2 score of the best model (ResNet50 [16], 92.886%). This shows that adding lower-performing models to the ensemble can improve the overall performance. While seemingly couter-intuitive, this phenomenon has been observed by other contestants as well [9].

F. Possible Improvements

After extensive testing, we believe we have reached the limits of the ResNet50 [16] architecture which represents our current best model. However, there are a number of other CNN architectures that may hold the potential to further improve the performance on this particular classification task. For example, DenseNet [21], which uses a feed-forward layer connection structure, is known to produce very accurate results in image classification. We plan to test this and related architectures on our satellite image analysis problem and include them in our weighted majority voting ensemble.

Another possible improvement is to increase (by additional augmentation) the number of training images that contain less-frequently observed labels, such as 'conventional_mine' (Table Ia). In the present training regime, the model sees very few such images compared to those with frequently occurring labels, so it is possible that it does not learn to generalize related patterns to test images. Increasing the frequency of rare labels in the dataset and giving the model more exposure to those labels should improve the situation.

Lastly, we aim to re-train our models on 4-band GeoTIFF images rather than 3-band JPEG images. The JPEG images are attractive for training because one can reuse the architectures that were previously developed for computer vision on natural images, such as the models that came out of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [22]. However, these images do not contain the near infrared (NIR) band, so they are missing out on some potentially useful information. Fig. 4 shows two JPEG images along with the NIR band (shown grayscale) from the corresponding GeoTIFF images. One can see that some of the important features, such as rivers, are more clearly visible in the NIR band than in the JPEG images. Hence, we believe that developing CNN architectures to process 4-band GeoTIFF images would be very useful for our task and would lead to further performance improvements.

IV. Conclusion

This paper presents our entry to the multi-label image classification contest created by Planet Labs and SCCON. The goal of the contest was to create algorithms capable of automatically classifying satellite images of the Amazon basin. Our approach involves using various Convolutional Neural Network models to learn relevant features from training images and applying these models to make initial predictions regarding image labels. We then use an ensemble of such predictions to arrive at the final prediction. While our best model (ResNet50 [16]) achieved a F_2 score of 92.886%, the ensemble of models achieved a F_2 score of 93.008%. This was only 0.310% below the result of the contest winner. To further enhance our system, we have identified several areas for improvement. Our goal is to create a model that will help better understand and prevent deforestation in the Amazon basin and around the world.

References

[1] A. Refice, D. Capolongo, A. Lepera, G. Pasquariello, L. Pietranera, F. Volpec, A. D'Addabbo, and F. Bovenga, "Sar and insar for flood monitoring: Examples with cosmo/skymed data," in *Proc. IEEE Int.*

Geoscience Remote Sensing Symp. (IGARSS'13), July 2013, pp. 703–706.

[2] K. S. Chen, S. B. Serpico, and J. A. Smith, "Remote sensing of natural disasters," *Proceedings of the IEEE*, vol. 100, no. 10, pp. 2794–2797, Oct 2012.

[3] L. Losik, "Using satellites to predict earthquakes, volcano eruptions, identify and track tsunamis from space," in *Proc. IEEE Aerospace Conference*, March 2012, pp. 1–16.

[4] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, February 2017.

[5] A. Gilbert, G. E. Flowers, G. H. Miller, B. T. Rabus, W. Van Wychen, A. S. Gardner, and L. Copland, "Sensitivity of barnes ice cap, baffin island, canada, to climate state and internal dynamics," *Journal of Geophysical Research: Earth Surface*, vol. 121, no. 8, pp. 1516–1539, 2016.

[6] A. Huete, S. Running, and R. Myneni, "Monitoring rainforest dynamics in the amazon with modis land products," in *Proc. IEEE Int. Symp. Geoscience and Remote Sensing*, July 2006, pp. 263–265.

[7] G. Popkin, "Satellite alerts track deforestation in real time," *Nature*, vol. 530, p. 392393, Feb 2016.

[8] R. Huguet, "How satellite maps can halt Amazon deforestation," https://www.csmonitor.com/World/Making-a-difference/Change-Agent/2014/0618/How-satellite-maps-can-halt-Amazon-deforestation, 2017, [Online; accessed 14-June-2017].

[9] W. Kan, "Planet: Understanding the Amazon from Space," https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/, 2017, [Online; accessed 10-June-2017].

[10] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[11] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation," in *Australian conference on artificial intelligence*, vol. 4304, 2006, pp. 1015–1021.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR'15*, May 2015.

[13] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[14] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *CoRR*, vol. abs/1610.02357, 2016.

[15] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, "Object detection networks on convolutional feature maps," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1476–1481, 2017.

[16] S. T. Hang and M. Aono, "Residual network with delayed max pooling for very large scale plant identification," in *Working notes of CLEF 2017 conference*, 2017.

[17] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR'14*, Apr 2014.

[19] S. Frühwirth-Schnatter, "Data augmentation and dynamic linear models," *Journal of time series analysis*, vol. 15, no. 2, pp. 183–202, 1994.

[20] L. I. Kuncheva and J. J. Rodríguez, "A weighted voting framework for classifiers ensembles," *Knowledge and Information Systems*, vol. 38, no. 2, pp. 259–275, 2014.

[21] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "Densenet: Implementing efficient convnet descriptor pyramids," *arXiv preprint arXiv:1404.1869*, 2014.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vision*, vol. 115, no. 3, pp. 211–252, Dec 2015.