Understanding the Amazon Rainforest from Space using CNNs

Shubham Chandak
Stanford University
Stanford, CA 94305
schandak@stanford.edu

Vamsi Chitters
Stanford University
Stanford, CA 94305
vamsikc@stanford.edu

Sagar Honnungar
Stanford University
Stanford, CA 94305
sagarh@stanford.edu

Abstract

Understanding land use patterns and atmospheric conditions through satellite images may be crucial in tackling the global challenge of rapid deforestation. In this project, we propose CNN-based methods for performing automated multi-label classification of satellite images of the Amazon basin with respect to the factors described above using a dataset provided by Planet Labs, which is hosting a Kaggle competition for this task. This paper details the different models we experimented with and evaluated according to the F_2 performance metric. This includes models tailored to the specific classification task that were developed from scratch, as well as models that were built on fine-tuned ImageNet pre-trained models via transfer learning. Our best submission so far achieves an F_2 score of 0.93142 on the Kaggle test set and is ranked 11^{th} on the leaderboard.

1. Introduction

Rapid deforestation in the Amazon basin and other forests has led to reduced biodiversity, climate change and other severe effects on life on the planet. Many acres of forest land are being lost every day and many wildlife species are losing their natural habitat and facing extinction. One of the ways to tackle this global challenge is to capture high quality image data and analyze it to understand and build effective solutions.

With widespread satellite deployment, it is now feasible to capture satellite images of the region at frequent intervals and at a high resolution. High resolution imagery has been found to be very helpful in differentiating between man-made and natural causes of deforestation, but robust methods for performing this have not yet been developed. Automated classification of these images according to atmospheric conditions and land use will enable us to quickly and effectively respond to deforestation and other harmful human encroachments.

To this end, Planet Labs Inc. has proposed a Kaggle challenge- *Planet: Understanding the Amazon from Space* [4], which we are participating in as part of this project. For this competition, Planet Labs has provided a dataset of 3 - 5 meter resolution satellite images of the Amazon basin and posed the challenge of designing algorithms to label satellite image chips with respect to atmospheric conditions and various classes of land cover/land use. The problem is a multi-label classification problem for satellite images. There are a total of 17 labels: 4 labels for the weather conditions and the rest for land cover/land use patterns. Example images and their labels are shown in Figure 1. The true labels for the training images are provided to us, but not those for the test images. The submissions are ranked on the basis of average F_2 score [24] over the test set which is a function of the number of false negatives and false positives, with a larger emphasis on false negatives. The Kaggle leaderboard provides us with a benchmark to evaluate our performance against other competitors.

We approach the problem by starting with CNN models for image classification and modifying them to account for the unique challenges posed here, which include (1) multi-label classification instead of single-label classification (2) highly imbalanced classes in the training set (3) 4-channel satellite images and (4) an intricate dependence between the classes. We tried various architectures including pre-trained ImageNet [22] models and hand-made CNN models. We also experimented with various methods for dealing with multi-label classification, including a single network outputting scores for all 17 classes and separate networks for subsets of related classes. The rest of the paper details the different model architectures and the efficacy of individual experiments in maximizing the overall F_2 score.

2. Related Work

The existing literature on multi-label classification and satellite image classification proved to be very beneficial as it provided us with a baseline framework to build our model off of and some general experimentation ideas.

Figure 1. Example images from the dataset and their labels (Source: [4])

2.1. Multi-label classification

An important characteristic of the training dataset is that each image can have multiple labels. Multi-label classification has been studied in a variety of domains such as bioinformatics, music and text classification. BP-MLL [28] is a neural network algorithm that tried to tackle this problem using a novel pairwise ranking loss function to exploit the relationships among labels. However, [18] found that the cross-entropy loss was as good as the pairwise ranking loss and that neural networks trained with cross-entropy loss can encode dependencies among labels in the final layer. Equipped with this intuition and coming across successful models, such as [14], that used cross-entropy loss for multi-label image annotation, we also decided to use it.

2.2. Satellite image classification

Several works have explored the problem of land use classification using traditional machine learning approaches. [20] used Decision Trees, [9] considered Random Forests, and [12] used Support Vector Machines (SVM) with Radial Basis Function (RBF) and polynomial kernels. Neural Networks were employed in [29, 15]. However the datasets considered in all these papers are much smaller than the one provided by Planet Labs which we use in this project. In most works on land cover classification, pixel values are directly used as features. [26] compared SIFT descriptors and Gabor filters as features instead of raw pixel values.
Popular datasets used to benchmark satellite image classification algorithms include UCMerced Land-use dataset [27] and RS19 dataset [25]. UCMerced Land-use is a publicly available dataset is composed of 2,100 aerial scene images with 256×256 pixels equally divided into 21 land-use classes. RS19 dataset contains 1,005 high-spatial resolution images with 600×600 pixels divided into 19 classes, with approximately 50 images per class.

Basu et al. in [5] introduced datasets SAT-4 and SAT-6 that consisted of satellite images in conjunction with land-use labels that were utilized by a neural network architecture called DeepSAT for classification. The group explored various architectures including CNNs but found that feature extraction followed by a deep belief network (DBN) [11] produced the best accuracy. [19] analyzed and compared different strategies for exploiting the power of existing CNNs in classifying remote sensing imagery including full training, fine tuning and using CNNs as feature descriptors. The same authors assessed the generalization of deep features of ConvNets trained on everyday objects on aerial and remote sensing image classification in [21]. Another recent Kaggle competition - *Dstl Satellite Imagery Feature Detection* [1] focused on the detection and localization of various objects in satellite images. The high level conclusions from these papers helped shape our model features and overall experimentation thought process.

3. Dataset

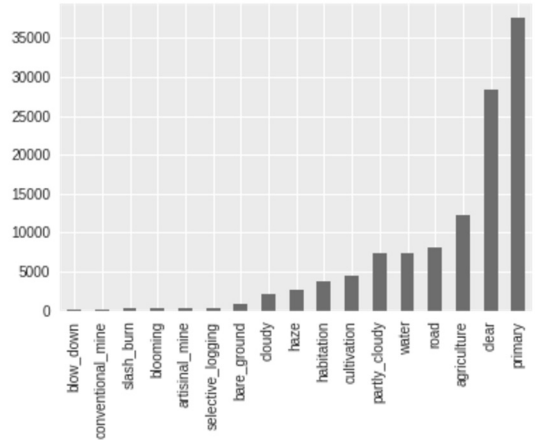


Figure 2. Number of training images per class [2]

The Kaggle dataset that we use to train our CNN model comes from satellite imagery of the Amazon Basin that was

collected over a 1-year span starting in 2016. The training set and test set consist of 40,479 and 61,191 256 $\times$ 256 images respectively. Each image is available both in a 3-channel (RGB) JPEG format and in a 4-channel (RGB+near-IR) TIFF format. There are 17 classes and each image can belong to multiple classes. The labels can broadly be broken into three groups: atmospheric conditions, common land cover/land use phenomena, and rare land cover/land use phenomena. The atmospheric condition labels are: clear, cloudy, partly cloudy, and haze. The common labels are: primary, agriculture, cultivation, habitation, water and roads. The rare labels are: slash-and-burn, selective logging, blooming, bare ground, conventional mining, artisinal mining, and blow-down. Figure 7 illustrates the distribution of training images across different class labels. It is evident that one of the more prominent challenges is dealing with the class data imbalance–that is, some of the classes have very few training images associated with it. Upon closer inspection, we see that there is a structure in the labels and the inter-relationships among labels can be exploited to design better classifiers and during post-processing to weed out some misclassifications. Figure 3 shows the co-occurrence matrix for weather labels. We see that the weather labels are mutually exclusive. Figure 4 and 5 show the co-occurrence matrices for the common labels and rare labels respectively, where it is evident that common labels have heavy overlap while rare labels have very minimal overlap.

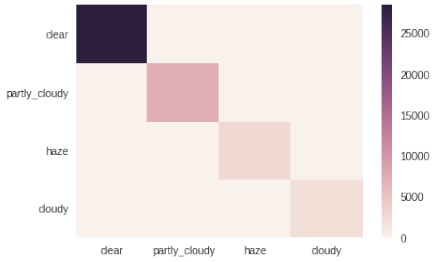


Figure 3. Co-occurrence matrix for weather labels [2]

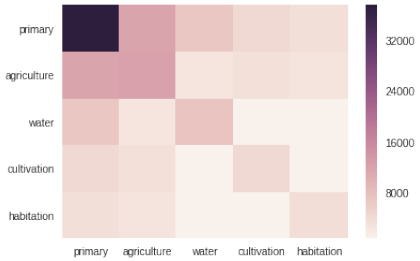


Figure 4. Co-occurrence matrix for common land cover/use labels [2]

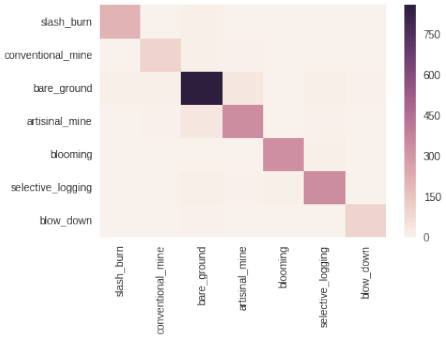


Figure 5. Co-occurrence matrix for rare land cover/use labels [2]

4. Methods

4.1. Architectures

Transfer Learning

To utilize the features from pretrained ImageNet models, we studied the benchmark results of various architectures in [6]. We experimented with several pretrained models like InceptionV3 [23], ResNet50 [10] and Xception [7] available in the Keras Applications library [3]. The output from the last convolution layer from the pretrained model was passed to a FC layer with 1024 neurons, a Batch Normalization layer [13] and ReLU activation after global average pooling and dropout layer (with 0.5 dropout probability). This was followed by a dropout layer with 0.5 dropout probability. The output layer consisted of 17 neurons with sigmoid activation. The loss employed was a sum of binary cross-entropy losses over the 17 classes. During the prediction phase, the output scores for the 17 classes are thresholded. The entire pre-trained model was finetuned while training (details in the results section - code inspired by [3]).

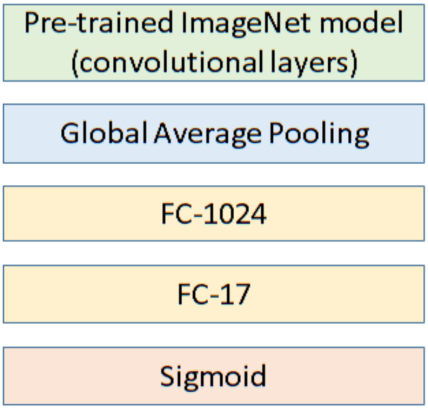


Figure 6. Architecture for transfer learning using pre-trained ImageNet models

Hierarchical Model

To explicitly account for the label dependence and class imbalance, we also designed a model using three CNNs

with the same basic architecture but different final layers. The first network is a 4-class single-label classifier for the weather labels with softmax loss (since the weather classes are mutually exclusive). The second network is a multi-label classifier with a sigmoid layer at the end. This provides labels for the six commonly-occurring land-use patterns plus an additional label for the seven rarer classes clubbed together. This helps in balancing the classes in the training set. The third network is a multi-label classifier for the seven rarer classes. This is trained only on the images that contain at least one of the rarer labels. For prediction, we use separate thresholds for the three networks determined by their F_2 score on the entire training set. We use the output of the 3rd network only if the score for the additional label in the 2nd network is above some threshold.

The architecture for the three networks consists of 10 (Conv-BN-ReLU) layers with 3 stride 2 Max Pooling layers followed by two (FC-BN-ReLU) layers with 2048 and 1024 neurons. The Conv layers use 3x3 filters with stride 1. The number of filters progressively increases as we go down the network with the first Conv layer containing 32 filters and the final one containing 128 filters. We experimented with this model using different input images with 3-channels.

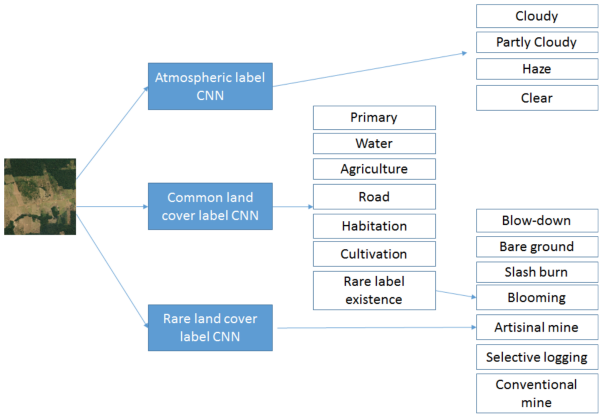


Figure 7. Hierarchical model with separate CNNs for subsets of related classes; rare label CNN is used only if the common label CNN predicts the existence of a rare label

4-channel Model

We also trained the hierarchical model above with 4-channel 32x32 TIFF images. The entire architecture was same except for the first convolution layer.

4.2. Improving Performance

We discuss some techniques used to improve the models described above. The associated results are presented in the subsequent section.

4.2.1 Data Augmentation

There is scope for variation in the exact location of houses, roads, mines and other features in the image that would be useful for the network to learn for this classification problem. Given the limited training data, we make use of data augmentation such as introducing random rotations, horizontal and vertical flips to make our models invariant to the exact location and help in better generalization performance. While fine-tuning entire ImageNet models like ResNet, data augmentation is crucial for prevent overfitting.

4.2.2 Class-specific thresholds

For the transfer learning models, we use separate threshold for each class instead of using a common threshold for all classes. This improves the F_2 score because it allows different thresholds for classes based on their frequency and the confidence of the model for the particular class. To find the optimal threshold for each class, we loop over the classes, adjusting the threshold for each class till we reach a stationary point. While this doesn't guarantee a globally optimal choice, it works well in practice.

4.2.3 Ensembling

The simple technique of ensembling multiple independent models and averaging their results at test time proved to be very effective in improving overall accuracy. Since different models are more accurate for different classes, the aggregate decision of independent models is less noisy than the classification made by a single model. This underlying idea of diversification makes our overall model less prone to overfitting.

4.2.4 Test-time augmentation

For each test image, we generated multiple copies by randomly flipping and transposing the images. Then we took a majority vote of the predictions on all these images to obtain our final predictions on the test set.

5. Experiments and Results

5.1. Evaluation metric

We rely on the F_2 score as the primary evaluation metric. The F_2 score is a special case of the F_β score which is defined as

$$F_\beta = (1+\beta^2)\frac{pr}{\beta^2 p + r}$$

where $\beta = 2$, p and r are precision and recall, defined as

$$p = \frac{tp}{tp+fp},\ r = \frac{tp}{tp+fn}$$

where tp, fp and fn are the number of true positives, false positives and flase negatives respectively. The mean F_2 score is calculated by averaging the F_2 score for each image in the test set.

5.2. Training

The training phase involves tuning a number of hyperparameters, including the optimization algorithm, learning rate, and regularization. We used a 90-10 training-validation split for all models. For preprocessing, we subtracted the training set mean from the images and rescaled the images according to the input size of the model. We used Adam [16] with batch size of 32-128 (depending on GPU memory and model size) as the optimization algorithm for all the models.

For the hierarchical model, we started the training with the default learning rate of 0.001 and used a learning rate decay schedule wherein the rate was reduced by half every time the validation loss plateaued. We typically trained each of the three networks for 25 epochs. For the third network for classifying the rarer labels, we used the second network as a pretrained model in order to avoid overfitting to the 2000 images for the rarer land use classes.

For the transfer learning architecture, we initially kept the weights in the pretrained network frozen for 10 epochs. After that, we reduced the learning rate by a factor of 10 and trained the entire network for 10 epochs. In some cases we trained for 5 more epochs at a reduced learning rate.

5.3. Results

We used Keras [8] with Tensorflow backend for all our experiments. The experiments were run on a Google Cloud instance with 8 CPU-cores and a NVIDIA Tesla K80 GPU. We made use of starter codes provided on Kaggle [2] for loading images and submitting results.

5.3.1 Quantitative Evaluation

We present our performance on the test set for each of the architectures in Table 1. For the transfer learning models, we see that using more powerful models improves results. Interestingly, increasing image size for Xception does not improve performance. This might be due to performance saturation for large image sizes.

The hierarchical images achieve very good performance even for a relatively shallow network and small resolution images. However, we were unable to replicate this success for higher resolution images (200x200) or using pretrained ImageNet models as the architecture for the three networks. This was primarily due to overfitting, and several experiments using extensive data augmentation and other techniques like dropout failed to improve results. The 4-channel TIFF images perform much worse than the 3-channel JPG images. As discussed in several Kaggle discussions, this is due to calibration issues and mislabeled images.

Ensembling 8 transfer learning models (some architectures were trained more than once) and 2 hierarchical models achieves our current best F_2 score of 0.93142 which is placed at the 11^{th} position (out of 436) on the Kaggle leaderboard (as of 6/12/17).

Model	Input size	F_2 score
Transfer Learning		
InceptionV3	200x200x3	0.92640
Xception	200x200x3	0.92648
Xception	224x224x3	0.92646
ResNet18	256x256x3	0.92653
ResNet50	224x224x3	0.92744
Hierarchical		
-	32x32x3	0.91479
-	64x64x3	0.92323
-	96x96x3	0.92457
-	200x200x3	0.92387
4-channel	32x32x3	0.91083
InceptionV3	139x139x3	0.91495
Ensemble		
-	-	**0.93142**

Table 1. F_2 score on test set for various models. Results for transfer learning models used test-time augmentation. Ensemble was obtained by taking majority vote of 10 best submissions.

Table 2 shows the precision, recall and accuracy for each class, for predictions of ResNet50 model on the entire training set. The classes are arranged in the order of increasing frequency. The accuracy is very high for the rarer labels because most of the images do not have these labels. However this high accuracy does not translate to high precision and recall measures for these classes. Both precision and recall get better for classes with more examples. The relatively low accuracies for road, agriculture and cultivation classes are discussed in the qualitative evaluation section below. For most classes, the recall is higher than the precision because the thresholds are chosen to maximize the F_2 score which favors higher recall.

5.3.2 Qualitative Evaluation

We performed qualitative evaluation of our models to better understand their behavior. We present some of the results here for our best individual model, the ResNet50 transfer learning model. Figure 8 displays some images from the training set wherein the model got all the labels correctly. While the first three images contain relatively common labels, the last two contain one rare label. Thus, even with the

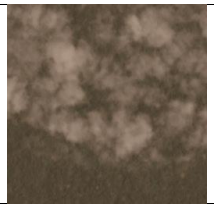
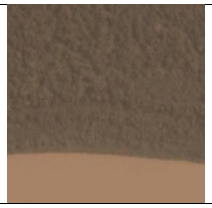
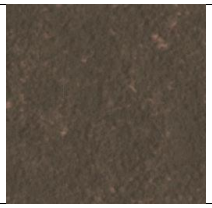
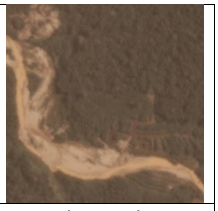
Image					
True labels	primary partly_cloudy	agriculture clear habitation primary road	clear primary water	clear primary selective_logging	artisinal_mine clear primary water

Figure 8. Some correctly classified images.

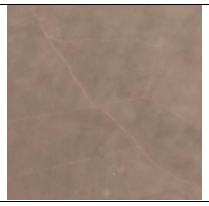
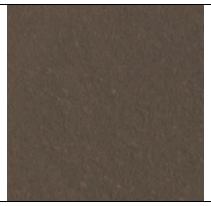
Image					
True labels	agriculture haze road primary	clear primary *blooming*	agriculture primary partly_cloudy	clear primary road *selective_logging*	partly_cloudy primary
Predicted labels	agriculture haze road primary *partly_cloudy*	clear primary	*cultivation* agriculture primary partly_cloudy	clear primary road *water*	partly_cloudy primary *clear*

Figure 9. Some misclassified images

Class	Precision	Recall	Accuracy
blow_down	0.404	0.520	0.996
conventional_mine	0.901	0.550	0.998
slash_burn	0.279	0.540	0.990
blooming	0.349	0.286	0.989
artisinal_mine	0.818	0.920	0.997
selective_logging	0.757	0.229	0.992
bare_ground	0.438	0.570	0.975
cloudy	0.682	0.976	0.975
haze	0.634	0.863	0.957
habitation	0.697	0.852	0.953
cultivation	0.463	0.825	0.875
partly_cloudy	0.845	0.979	0.964
water	0.779	0.867	0.930
road	0.731	0.944	0.919
agriculture	0.752	0.959	0.891
clear	0.927	0.992	0.940
primary	0.968	0.997	0.967

Table 2. Class-wise precision, recall and accuracy on the training set for the transfer learning ResNet50 model.

extremely small amount of data, the model is able to learn the rare classes, a fact which is further exemplified by the saliency maps discussed later.

Figure 9 displays some misclassified images with their true and predicted labels. In the first and the last image, our model predicts two weather labels due to confusing image structure. Even though each image has a single weather label, allowing multiple weather predictions reduces false negatives and hence improves the F_2 score. The model misses some rare labels in the second and the fourth image, which is expected because of the extremely small number of available images. Another common mistake, seen in the fourth image, is the confusion between road and water in the satellite images. Near-infrared band can help distinguishing the two classes; however, we didn't pursue the 4-channel images due to reasons mentioned before. The third image highlights an inconsistency in the training data where the distinction between 'agriculture' and 'cultivation' classes seems to be arbitrary and a lot of images have both labels. Thus, we see that a lot of misclassifications can be attributed to the rarer labels and inconsistencies in the data. This suggests that there might be a hard-to-breach limit due to these factors.

To understand if the model was really looking for the right features of the image while making predictions, we looked at the saliency maps for some images and some selected classes. These were obtained using the library keras-vis [17]. Figure 10 shows four saliency maps. In the first two images, the saliency map highlights the parts of the images close to the road and the water. The third image shows that the model correctly identifies the location of a rare label, enhancing our confidence that the model is able to capture the important features for the rarer labels. In the fourth figure corresponding to the same image but with the label of 'partly_cloudy', the saliency map covers the cloudy area

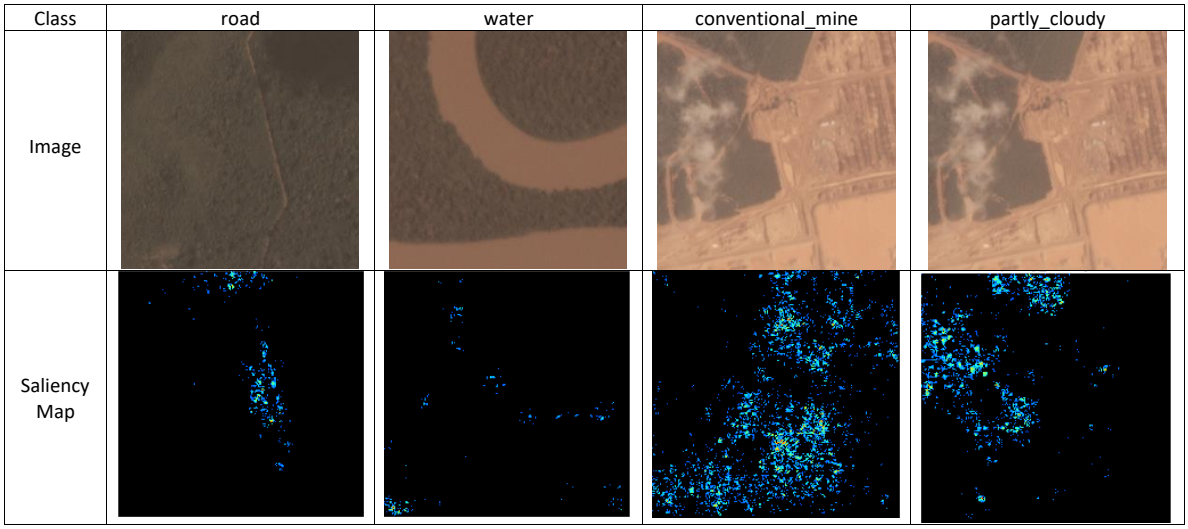


Figure 10. Saliency maps for different classes

but seems to extend to the top portion of the image as well.

5.4. Discussions and Analysis

In this section we perform ablative studies and try to understand the results and the impact of various hyperparameters.

5.4.1 Training

Figure 11 shows the loss profile for three instances of training the transfer learning models. Subfigure (a) shows the loss profile for InceptionV3 network when only horizontal and vertical flips are used for data augmentation. In the first 10 epochs, only the FC layers are trained and then the entire network is finetuned at a lower learning rate. Due to the small data size, we quickly see overfitting on the training set. Subfigure (b) shows the same model but with usage of random rotations, zooms and shifts. Here overfitting doesn't occur and the final validation loss is much better. The test set F_2 scores corresponding to (a) and (b) are 0.91481 and 0.92398, respectively (without test-time augmentation). Several resources suggest that the CONV layers should be frozen in the first few epochs to prevent destruction of the pretrained weights. However as subfigure (c) shows, the ResNet50 model trains fine even when the entire model is trained directly (starting with a low learning rate). Subfigure (c) also shows that reducing learning rate can help when the loss profile starts plateauing.

Figure 12 shows the impact of learning rate (with optimizer Adam) on the weather classifier, a component of the hierarchical model. Recall that the model is a CONV-BN-RELU network which is trained from scratch. Among the fixed learning rates, 0.0005 gives the lowest validation loss after 10 epochs. However, using a learning rate schedule, with regular decrease in the learning rate, significantly outperforms fixed learning rate models. We also trained our weather classification model using SGD Nesterov momentum update rule (default learning rate = 0.01, momentum = 0.9) and observed that it resulted in a higher final loss (train loss: 0.2401, validation loss: 0.248) than the model that relied on the Adam update rule. Ultimately, even further hyperparamter tuning did not yield better results and therefore, we decided to use the Adam update rule for all of our subsequent experiments.

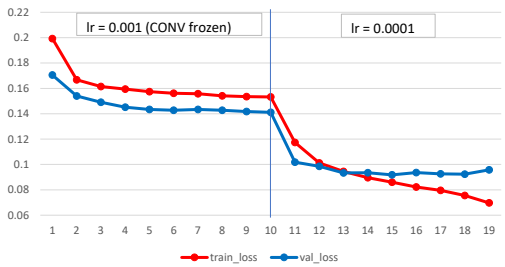
5.4.2 Class-wise threshold selection

Figure 13 shows the best separate and common thresholds for the ResNet50 model. Thresholds for more common classes seem to be lower on average. This might be due to the F_2 metric which penalizes false negatives heavily. On the training set, the F_2 scores with common and separate thresholds are 0.93441 and 0.93626 respectively. On the test set, these figures are 0.92530 and 0.92744. Thus, selecting separate thresholds is very important for improving the predictions.

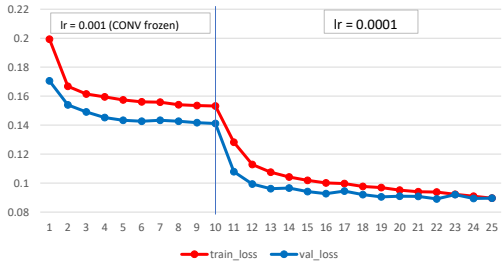
5.4.3 Test-time augmentation and Ensembling

As shown in Table 3, taking the majority of predictions of transformed versions of test images improves the test F_2 score significantly. We also tried averaging the scores first and then applying threshold on the averaged score. However, that performed slightly worse, e.g., averaging scores gives test F_2 score of 0.92635 for the Xception model, as compared to 0.92648 obtained by taking the majority.

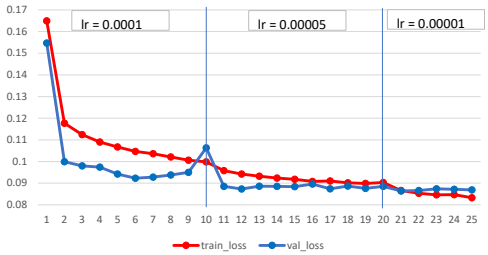
As seen in Table 1, ensembling several independently trained models provides around 0.004 improvement in the



(a) InceptionV3 without data augmentation



(b) InceptionV3 with data augmentation



(c) ResNet50 with data augmentation, directly finetuning whole network

Figure 11. Training profiles for three transfer learning instances.

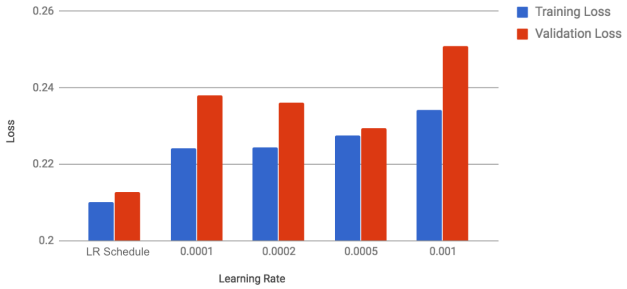


Figure 12. Loss vs. Learning Rate for Weather Network

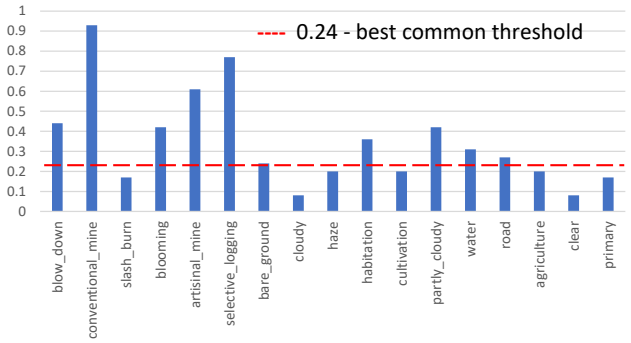


Figure 13. Class-wise and common thresholds for ResNet50 model

test F_2 score. The best results was obtained by ensembling 10 models - 8 transfer learning and 2 hierarchical. Adding more models didn't improve the F_2 score further.

Model	F_2 without test-set augmentation	F_2 with test-set augmentation
ResNet18	0.92276	0.92653
InceptionV3	0.92398	0.92640
Xception	0.92471	0.92648
ResNet50	0.92682	0.92744

Table 3. Effect of test-time augmentation on test set F_2 score

6. Conclusion and Future Work

We were able to design several well-performing models for classification of satellite imagery of the Amazon Basin by fine-tuning CNNs pre-trained on ImageNet and by exploiting the inherent structure of the problem. Many tricks such as data augmentation and ensembling were employed to boost F_2 score and their impact on performance was gauged using thorough ablative studies. Our top-performing model (an ensemble of 8 fine tuned single networks and 2 hierarchical models) achieves an F_2 score of 0.93142 and places us at the 11^{th} position out of 436 teams in the Kaggle competition.

As future work, we believe the following ideas are worth exploring: 1) finding ways to incorporate salient information from the Near-IR channel, 2) trying out alternative image preprocessing strategies, 3) designing separate models for each class which use suitable resolutions and channels, and 4) exploring other ensembling techniques (multiple snapshots of the same network) to boost performance.

References

[1] *Dstl Satellite Imagery Feature Detection*. https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection.

[2] *Kaggle Kernels*. https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/kernels.

[3] *Keras Applications*. https://keras.io/applications/.

[4] *Planet: Understanding the Amazon from Space*. https://www.kaggle.com/c/planet-understanding-the-amazon-from-space.

[5] S. Basu, S. Ganguly, S. Mukhopadhyay, R. DiBiano, M. Karki, and R. Nemani. Deepsat: a learning framework for satellite imagery. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 37. ACM, 2015.

[6] A. Canziani, A. Paszke, and E. Culurciello. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.

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

[8] F. Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

[9] P. O. Gislason, J. A. Benediktsson, and J. R. Sveinsson. Random forests for land cover classification. *Pattern Recognition Letters*, 27(4):294–300, 2006.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[11] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[12] C. Huang, L. Davis, and J. Townshend. An assessment of support vector machines for land cover classification. *International Journal of remote sensing*, 23(4):725–749, 2002.

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[14] J. Johnson, L. Ballan, and L. Fei-Fei. Love thy neighbors: Image annotation by exploiting image metadata. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4624–4632, 2015.

[15] T. Kavzoglu and P. M. Mather. The use of backpropagating artificial neural networks in land cover classification. *International journal of remote sensing*, 24(23):4907–4938, 2003.

[16] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] R. Kotikalapudi. keras-vis. https://github.com/raghakot/keras-vis, 2016.

[18] J. Nam, J. Kim, I. Gurevych, and J. Fürnkranz. Large-scale multi-label text classification - revisiting neural networks. *CoRR*, abs/1312.5419, 2013.

[19] K. Nogueira, O. A. Penatti, and J. A. dos Santos. Towards better exploiting convolutional neural networks for remote sensing scene classification. *Pattern Recognition*, 61:539–556, 2017.

[20] M. Pal and P. M. Mather. An assessment of the effectiveness of decision tree methods for land cover classification. *Remote sensing of environment*, 86(4):554–565, 2003.

[21] O. A. Penatti, K. Nogueira, and J. A. dos Santos. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 44–51, 2015.

[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. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[24] Wikipedia. F1 score — Wikipedia, the free encyclopedia, 2017. [Online; accessed 16-May-2017].

[25] G.-S. Xia, W. Yang, J. Delon, Y. Gousseau, H. Sun, and H. Maître. Structural high-resolution satellite image indexing. In *ISPRS TC VII Symposium-100 Years ISPRS*, volume 38, pages 298–303, 2010.

[26] Y. Yang and S. Newsam. Comparing sift descriptors and gabor texture features for classification of remote sensed imagery. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1852–1855. IEEE, 2008.

[27] Y. Yang and S. Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 270–279. ACM, 2010.

[28] M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

[29] L. Zhou and X. Yang. Use of neural networks for land cover classification from remotely sensed imagery.