Understanding the Amazon Basin from Space

CS 231N Spring 2017, Project Report

FNU Budianto
budi71@stanford.edu

Nickolas Westman
nwestman@standord.edu

Bojiong Ni
bojiong@stanford.edu

Abstract

Deforestation, especially in the Amazon basin, contributes to biodiversity loss, climate change, and a host of other negative effects. Local stakeholders and law enforcement agents need robust software solutions to identify the cause of deforestations. This is part of an ongoing Kaggle competition, where the task is to do a multi-label classification over satellite images and the labels are various atmospheric and land conditions. A single fully retrained ResNet model achieved a F2 score of 92.25%, while the ensemble of 13 ResNet models achieved a F2 score of 92.806% (rank 38 out of 430+).

1. Introduction

Deforestation, especially illegal deforestation, contributes to global warming, habitat loss, and a host of other problems. The action to take depends on the cause of the deforestations. Given the size and scope of the Amazon rain forest, we need an automated system to quickly identify whether there is a deforestation and what causes it, so that the stakeholders can respond more quickly and effectively. The first step towards this goal is to do a multi-label classification on Amazon satellite images, where the labels are various atmospheric (e.g clear or cloudy) and land conditions (e.g agriculture or water). There are 17 labels in total and the metric to maximize is the F2 score, which is a harmonic mean of precision and recall with more emphasis on recall. This is part of an ongoing Kaggle competition hosted by Planet (https://www.kaggle.com/c/planet-understanding-the-amazon-from-space).

Our goal is to predict all of the applicable labels for the image, i.e. given an image in format of JPG [256 x 236 x 3], or GeoTiff [256 x 256 x 4], predict a binary vector of length 17, with each vector cell corresponding to one of the 17 possible tags (labels). To solve this problem, we will explore various image classification techniques and architectures. We first try a simple multi-layer perceptron and ConvNet. We also explore architectures from past winners of the ImageNet challenge [1] such as ResNet [2] and Inception [3], since this Kaggle challenge is also a image classification challenge similar to ImageNet. The results can be seen in section 5.

2. Related Work

ImageNet is the most popular image classification dataset and has been used as a benchmark of new image classification architectures and techniques. There has been a lot of work on building image classification systems for ImageNet. The first model to use Convolutional Neural Network (CNN) is AlexNet [4], which won the ImageNet challenge in 2012. Hardware limitations at that time forced AlexNet to split the model on to two devices because the full model does not fit into a single GPU. Moreover, it was not a particularly deep network (7 layers) because advanced techniques for training deep neural networks did not yet exist.

Since then, hardware advances and training techniques such as spatial batch normalization and bottleneck layers have allowed researchers to train deeper, more effective CNN architectures. ResNet [2] introduced residual network to help training a very deep neural network, which was 150 layers. Alternatively, GoogleNet/Inception [3] used a Network-in-Network model introduced the inception module, where it concatenate the outputs of multiple convolutions with different filter sizes. GoogleNet, too, was a very deep and effective network. While many network focus on deeper structure, DenseNet, [5], on the other hand, introduced an alternative approach of connecting all layers to its subsequent layers. Such structure not only alleviates the vanishing-gradient problem and strengthen feature propagation, but also reduces the number of training parameters.

Satellite image classification has been researched and experimented extensively as well, including both rule based [6] and many machine learning based. Non-neural network methods includes Markov random field model [7], which exploits both spatial class dependency between neighboring pixels in an image and temporal class dependencies between different images of the same scene. The temporal information is especially useful for detection of class changes over time. Another similar work is the detection of oil spills

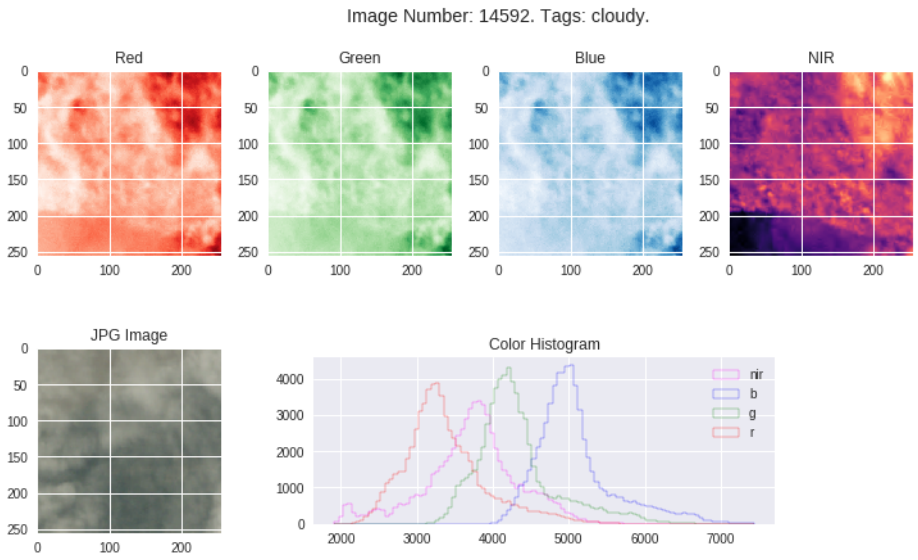

Figure 1. Example of Amazon satellite image.

in satellite radar images [8] using 1-NN and C4.5 [9], a decision tree algorithm with information entropy.

On the neural network side, Chen, *et al.* [10] used a hybrid deep neural network (HDNN) to detect vehicle in satellite images of San Francisco. HDNN works by splitting the last layer into multiple blocks to extract variable-scale features. Tang, *et al.* [11] used DNN and Extreme Learning Machine (ELM) to detect ships in airborne optical images. It works well but the limitation comes from the dataset itself, where the resolution is too coarse to be able to detect smaller ship. Jean, *et al.* [12] used CNN and transfer learning to predict poverty levels based on daytime satellite imagery. The limitation is also on the resolution of the images, which always seems to be an issue with satellite images, but soon Planet.com will be collecting daily imagery of the entire surface of earth at 3-5 resolution, which will be helpful for all image classification tasks that use satellite images. There are also previous work incorporates satellite image feature extraction. Basu, *et al.* [13] , presents two new satellite datasets called SAT-4 and SAT-6, and then proposes a classification procedure of unsupervised feature extraction followed by supervised Deep Belief Network. Other than supervised and semi-supervised learning, Baraldi, *et al.* [14] introduced unsupervised clustering artifical neural network for satellite image classification.

There have been many efforts in machine learning used to overcome skewed datasets. Some examples of note include tuning the base learning towards the minority class according to the distribution. B. Zadrozny, *et al.* [15] introduced a method of estimating the unknown distribution of classes using decision tree and naive Bayesian learning method and obtaining unbiased estimator for example dependent cost. Others also considered giving higher costs for the misclassification of examples of minority class(es) with respect to majority class(es), and therefore, trying to minimize higher cost errors [16, 17]. Another direction is to oversample minority classes. One particular technique of note is SMOTE, or Synthetic Minority Over-sampling Technique, which aims to unskew a dataset by generating fake data [18]. It's performed by taking a minority example and its nearest neighbor, and generating a linear combination of the two in any dimensional space. This technique has been shown to combat overfitting the majority examples in a classifier.

3. Dataset and Features

The training set contains 40,479 images and the test set contains 40,669 images. Each image contains 256 x 256 pixels and encoded with two methods: JPG [256 x 236 x 3], or GeoTiff [256 x 256 x 4]. While JPG only has RGB, GeoTiff includes a near-infrared (NIR) channel. An example of the image can be seen in Figure 1. The data is labeled by crowd workers so there might be some human error in the data. We have isolated 10% of the training set (the last 5479 images) as validation set. Right now, only 2/3 of the test set is graded in the public leaderboard and the rest will be graded after the competition ends (2 months from now).

This dataset is very skewed towards certain classes, as seen in Figure 2. As a result, we have very strong initial results immediately, an F2 score of 0.88 with a two-layer CNN model.

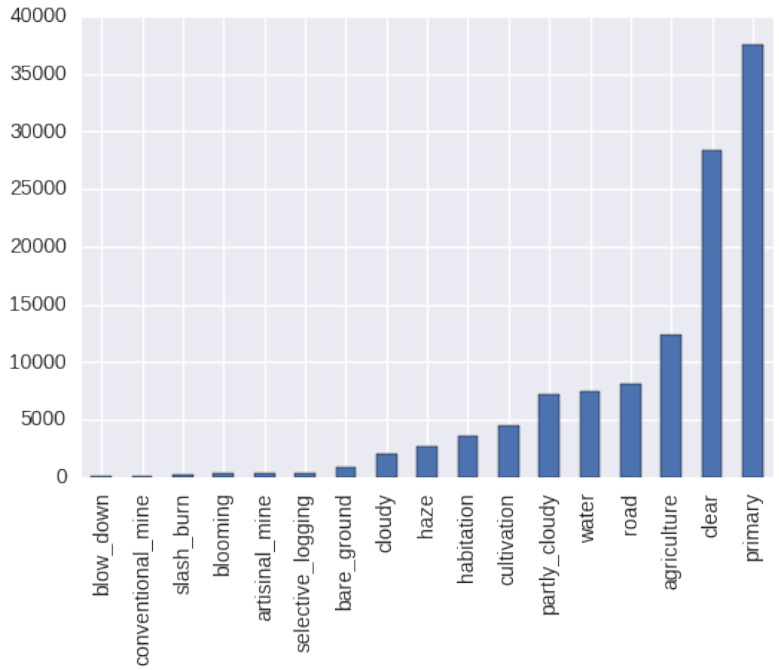

Figure 2. Train data labels histogram

3.1. Labels

Each image has at least one atmospheric label from the set ('clear', 'haze', 'partly cloudy', 'cloudy'). Additionally, a label can optionally have one or more rare labels such as 'blooming,' which signifies massive 30+ foot diameter trees, or 'artisinal mine' which represents an illegal mining operation, usually near a river. The full set of labels is listed in Figure 2.

4. Methods

4.0.1 Prediction Layer and Loss Function

Since this is a multi-label classification problem, the commonly used softmax prediction is not ideal in our case. Instead, we used sigmoid function ($\sigma(x) = \frac{1}{1+e^{-x}}$) for prediction and the binary cross entropy function for loss. Sigmoid normalizes the input data to range of $(0, 1)$, which models the probability of the class label being applicable to the image. For prediction, we use thresholds that gives the highest validation F2 score and each class label has its own threshold.

We predict a class tag as applicable to the image if the model gives a probability measure greater than the threshold for that particular class. We perform an exhaustive search over thresholds after training the model.

We used weighted binary cross entropy loss as our optimization loss function defined as

$$H(y, y') = -\sum_i w_i(y_i \log y' + (1 - y_i) \log(1 - y'))$$

where y is the truth label, y' is the output probability for each class label from the model, and w_i is the class weight. The less frequent class will have higher weight, so that there is more incentive to get the prediction right for the rarer classes. The class weight is defined as:

$$w_i = \frac{N}{C \sum_{j=1}^{N} \mathbb{1}\{y_j = i\}}$$

where N is the number of samples and C is the number of classes.

Since this is a multi-label classification problem with 17 labels, each label has two classes (True and False). Thus, we also tried a loss function with 34 weights defined as:

$$H(y, y') = -\sum_i w_{pi} y_i \log y' + w_{ni}(1 - y_i) \log(1 - y')$$

where w_{pi} is the weight for positive class for label i and w_{ni} is the weight for negative class for label i.

$$w_{pi} = \frac{N}{2 \sum_{j=1}^{N} \mathbb{1}\{y_{ji} = 1\}} \quad w_{ni} = \frac{N}{2 \sum_{j=1}^{N} \mathbb{1}\{y_{ji} = 0\}}$$

4.1. Optimizer

Our input data is high in dimension($256 \times 256 \times 3$ or $256 \times 256 \times 4$). Each dimension may have a different gradient magnitude, thus we choose to use Adam or Adadelta optimizer which adjusts the effective learning rate in each dimension adaptively.

4.2. Measurement Metrics

We incorporate two measurement metrics, accuracy and F2 score. For this multi label classification problem, accuracy is defined as

$$\text{accuracy} = mean(\mathbb{1}\{y_{ij} = y'_{ij}\})$$

where y_{ij} is the true binary label for example i class j and y'_{ij} is the predicted binary label for example i class j.

F2 is a weighted harmonic mean of precision and recall:

$$F_2 = 5 * \frac{Precision * Recall}{4 * Precision + Recall}$$

4.3. Neural Network Structures

4.3.1 MLP Model

We first tested with a MLP model with two layers of dense (a.k.a fully connected) layers:

$$\text{FC(1024)} \rightarrow \text{Relu} \rightarrow \text{Drop(0.5)} \rightarrow \text{FC(17)}$$

4.3.2 Baseline/ConvNet Model

Our baseline model is a multi layer convolution neural network with the following structure:

$$\text{Conv3-32} \rightarrow \text{Conv3-32} \rightarrow \text{Relu} \rightarrow \text{max-pool2} \rightarrow$$

$$[\text{Conv3-32} \rightarrow \text{Relu} \rightarrow \text{max-pool2}] \times 2 \rightarrow \text{Drop(0.5)} \rightarrow$$

$$\text{FC(1024)} \rightarrow \text{Relu} \rightarrow \text{Drop(0.5)} \rightarrow \text{FC(17)}$$

Conv3-32 means a convolution layer with 32 (3×3) filters. max-pool2 means a max pooling layer with pooling size of (2×2).

4.3.3 Transfer Learning

ResNet and Inception are performing well in capturing image features for classification on ImageNet. Thus, we built a transfer learning model with the pretrained ResNet and Inception weights. Since our classification label sets are different from the ImageNet challenge. We excluded the fully connected (dense) layers from the pretrained model and instead added two non-pretrained fully connected (dense) layers:

$$\text{pretrained ResNet50/InceptionV3 excludes dense layers}$$

$$\text{FC(1024)} \rightarrow \text{Relu} \rightarrow \text{Drop(0.5)} \rightarrow \text{FC(17)}$$

4.3.4 Full Retrain

In addition to transfer learning, we also tried to train the full ResNet50 and DenseNet [5], with the dense layers replaced with the ones in transfer learning. We used [19] for the DenseNet implementation. It turns out the full retain with ResNet50 out performs all other models.

4.4. Exponential Moving Average

In addition to the traditional weights update by back-propagation, we also used the weights update with exponential moving average. I.e. the weights (hidden parameters) are updated as:

$$w'_{t+1} = \frac{\partial Loss}{\partial w}$$

$$w_{t+1} = (1 - \delta)w_t + \delta * w'_{t+1}$$

The implementation is based on [20].

4.5. Optimal Threshold Selection

Different from single label classification problem where only the label with highest score is selected as prediction, our problem is to select all labels applicable to the image. Thus we used a different strategy of sigmoid activation at the final prediction layer. At training time, we used naive probability cut of 0.5 as the activation threshold. However, the model may have different optimal threshold for each class. Thus we used the validation set to find the threshold for each class that results in best F2 score. The implementation is based on [21].

4.6. Ensembles

Another strategy to improve our F2 score is to ensemble multiple models together. We tried both averaging and majority vote ensemble strategy on 13 fully trained ResNet50.

4.6.1 Averaging

The average method is to compute the mean of predicted sigmoid values from all models in each class and use these mean values as the final prediction scores. Then we used the strategy mentioned above to find the best thresholds for the mean sigmoid scores for each label class.

4.6.2 Majority vote

The majority vote method is to have each model finds its own best threshold on each class and make prediction with its best thresholds. Then we count how many models have selected a particular label class. If more than half of models select the label, we predict the final result of this label as selected.

Table 1. Models' performance on the validation and test set.

Model	Val F2 (%)	Test F2 (%)
Naive	64.6	N/A
MLP	64.6	N/A
DenseNet	66.98	N/A
ConvNet	88	88
InceptionV3 Transfer	88	N/A
ResNet Transfer	90.4	90.6
ResNet (single model)	92.6	92.25
ResNet (ensemble)	93.03	92.806

4.7. Data Augmentation

We feared that a skewed dataset would lead to over fitting and prevent good model performance. We tried two different approaches for data augmentation: 90-degree image rotation, as well as using Keras's preprocessing library, ImageDataGenerator.

We used skimage to do an in-memory rotation of image tensors. We alternated training batches on un-rotated or images randomly rotated 90-, 180-, or 270- degrees at a time. This caused a bump in f2 score, which we will discuss in later sections.

We experimented with Keras' image pre-processing package, which is a more comprehensive library for image augmentation. We used it to apply affine transformations to the images, such as rotation, scaling, shearing, and translation. However, we did not see any noticeable increase in performance relative to image rotation, and we guessed this was largely due to hyperparameter selection.

5. Experiments, Results, and tsoDiscussion

5.1. Implementation and Tuning

We implemented the models using Keras [22] with Tensorflow [23] backend. We used [24] as the base code for this project. For ResNet, we used Adadelta optimizer with a learning rate of 0.5 and decay of 0.002. For DenseNet, we used Adadelta with a learning rate of 0.5 and decay of 0.001. For the other models, we used Adam optimizer [25] with a learning rate of 0.001 and decay of 0.001. We used mini-batch size of 32. Additionally, we used dropout to regularize our fully-connected layers.

5.2. Results

The validation and test F2 scores for the models that we have tried can be seen in Table 1. As of 06/12/2017, we got rank 38 out of 430+ teams. Top of leaderboard is at 93.334% F2. Ensembled ResNet performs the best on this image classification task. Ensembling with averaging method out performs majority vote. Dataset augmentation by rotation helped to increase the F2 score by 0.2%.

Table 2. Per label F2 using ResNet (ensemble).

Label	Val F2 (%)
primary	99.15
clear	97.69
partly_cloudy	94.2
agriculture	90.4
cloudy	87.64
road	86.63
artisinal_mine	85.71
water	85.53
haze	79.07
habitation	78.49
cultivation	68.56
conventional_mine	46.51
bare_ground	42.96
selective_logging	29.41
blooming	19.89
slash_burn	3.16
blow_down	0

5.2.1 Error Analysis

In order to have better insight of the error distribution, we also checked the performance on each class. Table 2 is a breakdown of per class F2 score on validation set for the ensembled ResNet model. As we expected, the primary label (most prevalent label in the dataset) get very high (99.15) F2 score, while labels with very small population (e.g. slash_burn, blow_down) in the data set get very low or even zero F2 score.

In addition, we also found that the original labels on the data set have some noise that affected the F2 score. An example of such noise can be seen in Figure 3. On the left image we can see that it's clear, with a little bit of forest on the bottom left and the rest is water, but somehow the actual label for the image is cloudy.

The saliency maps of some validation images can be seen in Figure 4. Although the maps are not perfect, they somehow correspond to the labels they represent. The water's saliency map is focused on top left since the water location is on top left and clear's and cloudy's saliency map are all over the images since cloudy is an atmospheric label.

The confusion matrices for 3 labels (out of 17) can be seen in Figure 5, 6, and 7. The model performed well when the distribution of True and False samples are balanced (e.g. the clear label), but the model performed badly when the distribution is imbalanced. Even after using the weighted loss with 34 weights, for the blow_down label, the model just predicted False for every sample.

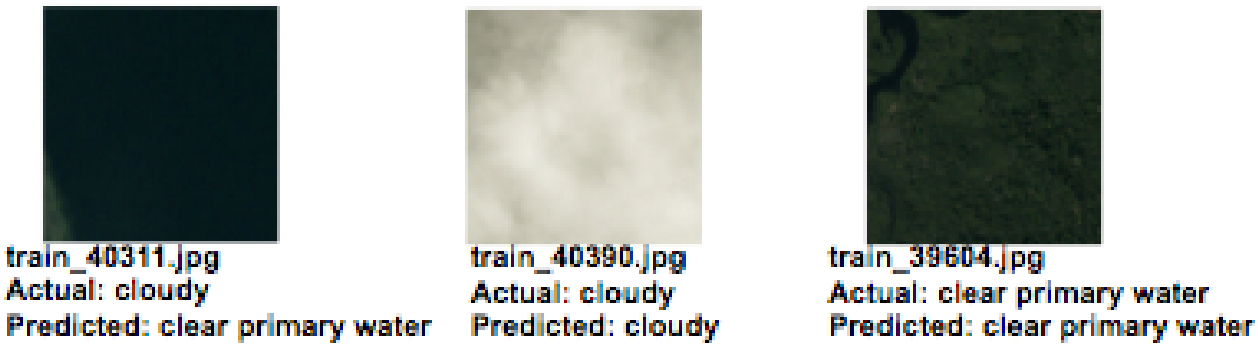

Figure 3. Example of noise in the dataset.

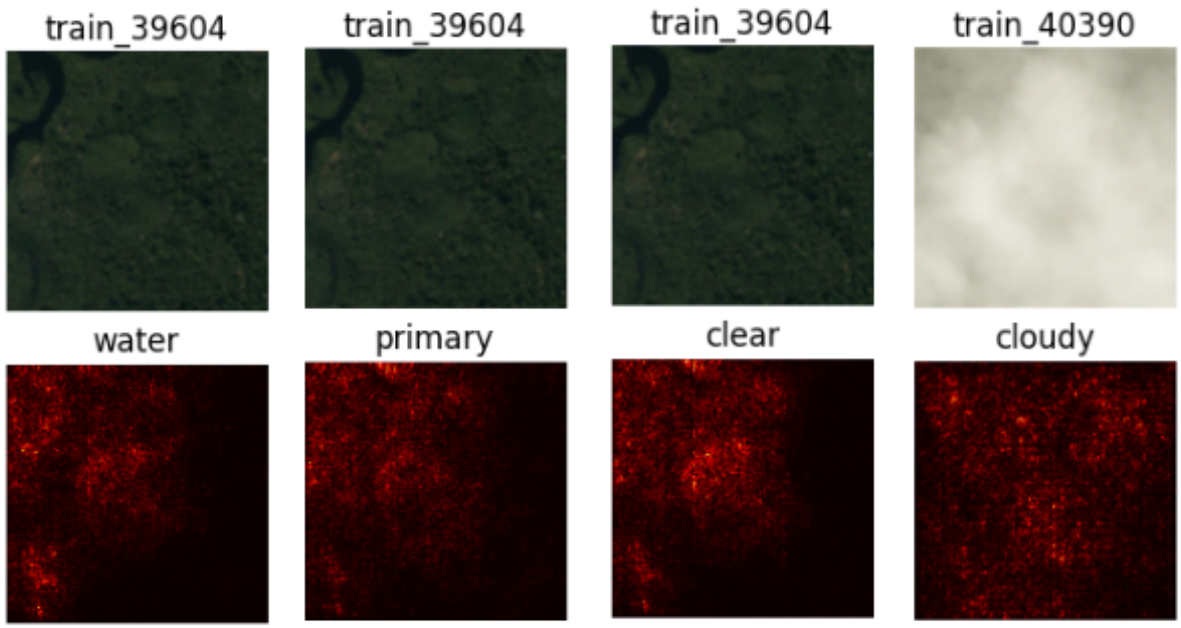

Figure 4. Saliency maps of some validation images using ResNet single model.

5.3. Other Experiments

In addition to the experiments and models mentioned in Methods section, we also tried various experiments below, but they do not help much:

- Models were trained using JPG images, which does not have the near infrared channel. An initial trial to train a ConvNet model using GeoTiff images without any preprocessing resulted in worse F2 score (0.676).
- Making the data to have zero-mean also did not help, so we might need different ways of preprocessing the GeoTiff images. We have also tried to compute RGB values and normalized difference vegetative index (NVDI) value using spectral library, but the score is still lower than the score using JPG images. The problem with the GeoTiff images is that they are not corrected for sun angle and distance.
- We have also tried using exponential moving average of ResNet's parameters during prediction but that also did not to increase the F2 score.

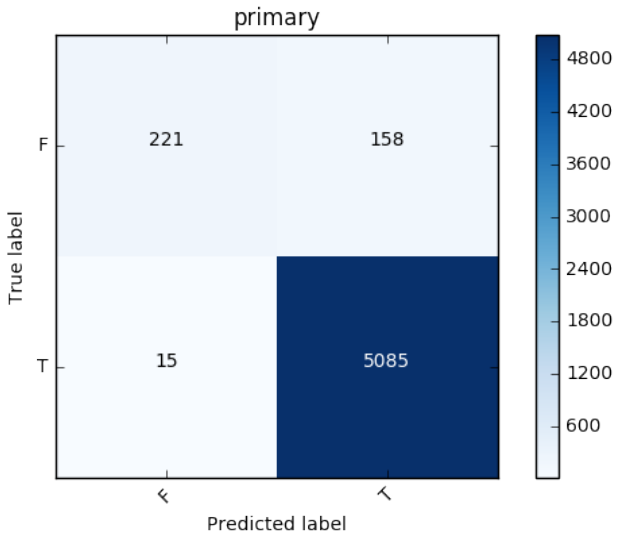

Figure 5. Confusion matrix for the "primary" label using ResNet ensemble.

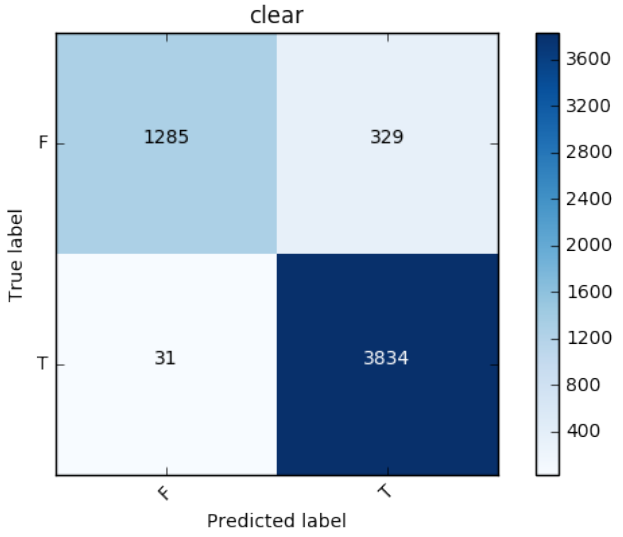

Figure 6. Confusion matrix for the "clear" label using ResNet ensemble.

6. Conclusion and Future Work

Model architectures that were successful for ImageNet task performed very well on this multi-label classification task on Amazon satellite images. A single fully retrained ResNet model achieved a F2 score of 92.25%, while the ensemble of 13 ResNet models achieved a F2 score of 92.806% (rank 38 out of 430+). We are very happy with this outcome!

Because of the imbalance labels distribution, class with high prevalence in the data set can easily get high F2 score

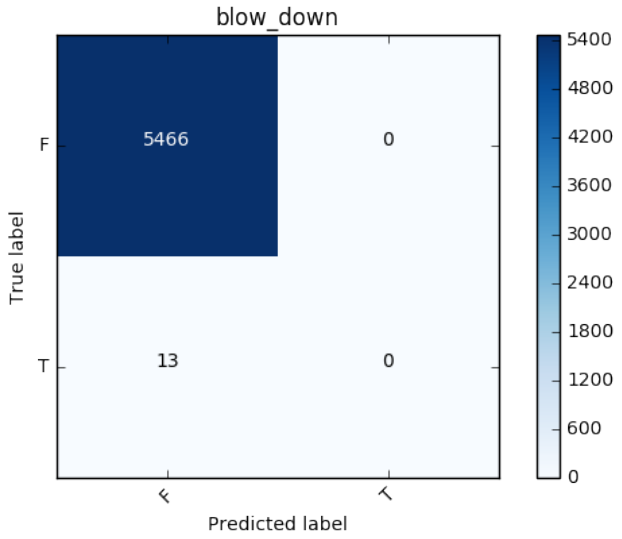

Figure 7. Confusion matrix for the "blow_down" label using ResNet ensemble.

but rarer class suffers from getting the correct prediction. A possible future work is to experiment with different loss and weight functions.

Another direction of improvement is to generate more augmented data, especially for rarer classes. For example, we can generate some down sampled image, blur the image, concatenate and resize two images (and their labels) into one training example. Moreover, we could follow the SMOTE pattern and linearly combine different minority examples with nearby neighbors. That way, we will make the training set more balanced on each class label.

Finally, we believe that the desired evaluation metric is not properly modeled by a cross-entropy loss, and think that using a REINFORCE [26] algorithm could properly trade off the loss between accuracy (immediate reinforcement) and f2 score (delayed reinforcement). By using this style of learning, we think that we could improve our outcomes.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[3] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks,"

in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[5] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely Connected Convolutional Networks," *ArXiv e-prints*, Aug. 2016.

[6] B. A. K. S. B. P. Lucas R, Rowlands A, "Rule-based classification of multi-temporal satellite imagery for habitat and agricultural land cover mapping," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 62, pp. 165 – 185, 2007. [Online]. Available: ftp://ftp.ccrs.nrcan.gc.ca/ad/Phenology/PhenologyPapers/Lucas_2007_Multi_Temporal_Land_Mapping.pdf

[7] T. T. Solberg, A. and A. Jain, "Markov random field model for classification of multisource satellite imagery," *IEEE Trans. Geoscience and Remote Sensing*, vol. 34, pp. 100 – 113, 1996. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=481897&tag=1

[8] H. R. M. S. Kubat, M., "Machine learning for the detection of oil spills in satellite radar images." *Machine Learning*, vol. 30, pp. 195 – 215, 1998.

[9] J. Quinlan, "C4.5: Programs for machine learning." Morgan Kaufmann, 1993.

[10] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *IEEE Geoscience and remote sensing letters*, vol. 11, no. 10, pp. 1797–1801, 2014.

[11] J. Tang, C. Deng, G.-B. Huang, and B. Zhao, "Compressed-domain ship detection on spaceborne optical image using deep neural network and extreme learning machine," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 3, pp. 1174–1185, 2015.

[12] N. Jean, M. Burke, M. Xie, W. M. Davis, D. B. Lobell, and S. Ermon, "Combining satellite imagery and machine learning to predict poverty," *Science*, vol. 353, no. 6301, pp. 790–794, 2016.

[13] S. M. R. D. M. K. S. Basu, S. Ganguly and R. Nemani, "Deepsat: A learning framework for satellite imagery," *23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2015. [Online]. Available: http://bit.csc.lsu.edu/~saikat/publications/sigproc-sp.pdf

[14] A. Baraldi and E. Alpaydin, "Constructive feedforward art clustering networkspart i and ii," *IEEE Trans. Neural Netw.*, vol. 13, 2002. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=377930

[15] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in *the 7th International Conference on Knowledge Discovery and Data Mining (KDD01)*, 2001.

[16] P. Domingos, "Metacost: a general method for making classifiers costsensitive," in *the 5th International Conference on Knowledge Discovery and Data Mining (KDD99)*, 1999, pp. 155 – 164.

[17] J. L. B. Zadrozny and N. Abe, "Costsensitive learning by costproportionate example weighting," in *the 3rd IEEE International Conference on Data Mining (ICDM03)*, 2003, pp. 435 – 442.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622407.1622416

[19] T. de Boissiere, "Densenet in keras," 2016. [Online]. Available: https://github.com/tdeboissiere/DeepLearningImplementations/blob/master/DenseNet/densenet.py

[20] S. Bahrampour, "Exponential moving average in keras," 2016. [Online]. Available: https://gist.github.com/soheilb/c5bf0ba7197caa095acfcb69744df756

[21] M. Bober-Irizar, "Better optimisation of f2 threshold," 2017. [Online]. Available: https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/discussion/32475

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

[23] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[24] M. Bober-Irizar, "Simple keras starter," 2017. [Online]. Available: https://www.kaggle.com/anokas/simple-keras-starter

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[26] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3-4, pp. 229–256, May 1992. [Online]. Available: https://doi.org/10.1007/BF00992696