Classifying Weather, Terrain, and Deforestation of the Amazon using Deep Multi-task Convolutional Neural Nets

Allen Zhao, Shawn Hu, and Chenyao Yu
Stanford University
353 Serra Mall, Stanford, CA 94305
arzhao@stanford.edu, shawnghu@stanford.edu, chenyaoy@stanford.edu

Abstract

We tackle the problem of multi-label classifying satellite imagery of the Amazon rainforest in domains including atmospheric conditions, terrain, and indicators of human impact in order to survey and prevent deforestation. We use a deep convolutional neural net, as well as several heuristic methods and augmentations to the dataset to achieve an ultimate F2-score above 0.92. We also discuss existing work in satellite image classification, F2 score maximization, and outlier detection for handling noisy data. We further describe various failed approaches, and the reasons for their failures; we close with discussion of future work.

1. Introduction

The Amazon Basin accounts for the largest share of human deforestation, contributing to reduced biodiversity, habitat loss, climate change, and other devastating effects. Around 17% of the Amazon rainforest has been lost in the last 50 years, mostly due to forest conversion for cattle ranching [2]. Deforestation in this region is particularly rampant near more populated areas, roads and rivers, but even remote areas have been encroached upon when valuable mahogany, gold and oil are discovered [2]. Without a method of accurately tracking these patterns of devastation, solutions are generalized at best - and completely targeting the wrong areas at worst. To fill in this knowledge gap, we focus on ways of identifying the marks of human presence via satellite imagery.

Current approaches to detect deforestation can be broadly divided into those that compare two images taken at similar periods of the year and those that monitor changes by using multiple images taken during the growing season [3]. The timespan required for these approaches could be too long and by the time deforestation is detected using these methods, it may be too late to save the forest.

Machine understanding of satellite imagery can provide a better idea than traditional methods of how and where deforestation happens as it happens and can help governments and local stakeholders respond more quickly and effectively.

We designed a CNN architecture that analyzes satellite images to detect patterns that may foreshadow future deforestation. Our goal is to detect deforestation on a global scale before it happens. The input to our algorithm is a satellite image. We then use a CNN to output predicted atmospheric conditions, terrains (such as slash-and-burn, rivers, or plain rainforest), and/or human activity.

Related Work

Similar Applications

We came across a multitude of papers that have constructed new architectures or proposed new solutions that we either implemented or considered implementing and did not have the time to do so. [5], written in 2015, lays out the groundwork for satellite image classification, proposing a large database called DeepSat that would later become the gold standard for the field.

[6] sets up a poverty metric and uses Transfer Learning to initialize their weights (along with additional nighttime data), achieving good results. Their success at transferring ResNet weights over to satellite image classification has spurred us to peruse model zoos and try several common architectures. We tried transfer learning and found that we did not have the memory to execute it successfully. The results we got from it were consequently disappointing.

[3] describes a CMFDA algorithm that implements a year-long, continuous, time-series based approach to monitoring images. However, the algorithm was developed for 30m resolution, 16-day frequency reflectance data from

the Landsat satellite, which is in a different input format than what we work with.

Examples of real world applications of satellite imagery usage for deforestation detection include The Real Time System for Detection of Deforestation (DETER), which has helped Brazils government to reduce its deforestation rate by almost 80% since 2004, by alerting the countrys environmental police to large-scale forest clearing [7]. The drawback to this system is that it takes a long time to process changes in the terrain. They use data from the Moderate Resolution Imaging Spectroradiometer (MODIS) on NASAs Terra satellite, which at its top resolution produces images with pixels covering an area 250 meters on each side [8]. This is too big to spot small changes in land cover, so it can take computer programs weeks or even months to detect that a forest is being cleared. [9] improved upon this by taking data from NASAs two active Landsat satellites, which generates images with pixels of 30 meters on each side, over 8 times higher resolution than MODIS. This enabled them to recognize a disturbance as small as a road snaking its way through a previously untouched forest, something that often appears before clear-cutting begins.

Possible Augmentations

[18] proposes using a spatial-pyramid-pooling (SPP) model network architecture to analyze images scaled to various resolutions. Instead of the computationally expensive procedure of training several deep convolutional networks on these differently scaled pictures, SPP shares parameters at initialization and trains its several networks similarly - cutting down massively on computational cost. It is still more expensive than a regular, single network system, however, and in testing did not give us satisfactory results.

Moreover, the following papers proposed custom thresholding as a method for optimizing F2 score performance, which inspired a version of one of our model augmentations[12] [13] [14] [15].

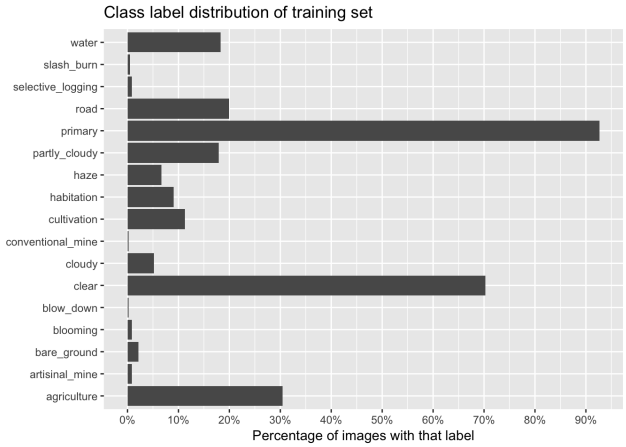
Finally, ([10] and [11]) that introduce classification improvements. [11] identifies a problem in which a model's idea of an object's spatial borders within an image do not line up to the actual borders, which can cause drops in accuracy when classifying objects that are small in a given image. Their solution is to feed a coarse ConvNet's output into a Residual Neural Network. [10] goes into various methods that can be used to mitigate the obscuring effect of cloud cover, which is a factor in many of our images.

Multitask Learning

While the restrictions on some of the labels may suggest that we just learn to predict the labels separately, [4] strongly suggests that our model can significantly benefit from learning to predict the tasks jointly. The intuition for this phenomenon is that the network can learn representations which capture information shared between the tasks, allowing it to generate a more complex model which generalizes better to the multi-task objective.

Dataset

Our data comes from Kaggle and is sourced from Planet, a company that designs and builds the worlds largest constellation of Earth-imaging satellites [20]. The dataset consists of 40,479 labeled training images (split into 35000 training, 5479 validation) given in both JPG and TIF formats, and a preliminary test set of 40,668 images, which Kaggle evaluates our performance on without revealing the labels. The images are provided in 256x256 pixel resolution, each representing a roughly 973x973 plot of land. There are 17 labels in all, and labels occur with frequencies ranging from over 90% (primary) to less than 1% (mining, slash and burn, blowdown). Four of the labels (”cloudy”, ”partly_cloudy”, ”haze”, ”clear”) correspond to weather patterns, and only one of these may be assigned. Moreover, no labels are assigned at the same time as the ”cloudy” label. Notably, the labels are slightly noisy, due to the ambiguity of some classes, as well as errors in human labeling. In the words of the Kaggle competition poster, ”part of the challenge of this competition is to figure out how to work with noisy data”.



Preprocessing

The images are scaled down to 64x64 pixels before being input into our model. We further augment our dataset by performing the following transformations before training time:

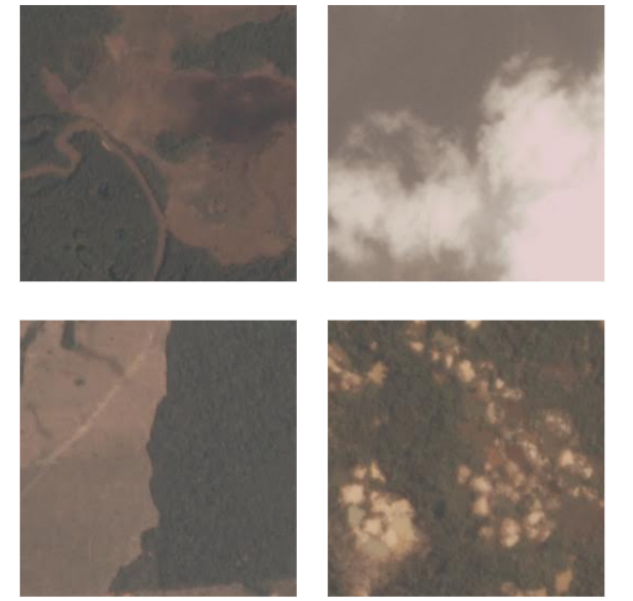


Figure 1. Four examples from our training set. Top left: agriculture, clear, primary, slash_burn, water. Top right: cloudy. Bottom left: artisinal_mine, bare_ground, clear, primary, water (likely mislabeling). Bottom right: agriculture, clear, primary, road.

- Horizontally and vertically flipping the image at random;
- Horizontally shearing the image by a factor uniformly distributed between 0 and 0.1;
- Randomly shifting the image in both the horizontal and vertical direction, by a number of pixels uniformly distributed between 4 leftward and 4 rightward;
- Rotating the image a random amount.

Note: In particular, we did not choose to apply random zooming to our images, since in our dataset, pixels correspond to fixed physical sizes.

Model

Evaluation

We evaluated the performance of our model using the Kaggle competition's metric, the F_2 *score*, which is calculated as follows:

$$F_2 = (1+\beta^2)\frac{pr}{\beta^2 p + r},\ \beta = 2$$

with p the precision (ratio of true positives to predicted positives) and r the recall (ratio of true positives to all positives. This score is just a weighted measure of the precision and the recall; in particular with $\beta = 2$ we have recall more important than precision. To be specific, in our calculation of recall and accuracy, we consider the prediction of each individual label to be a separate task.

Base Architecture

Following the poor (relative to the leaderboard) performance of our simple starter network, we drastically expanded the layer counts, filter counts, and overall size of the network. Given that we could only put 4 max-pooling layers when using 64x64 sized images, we used a network that progressively increased the number of filters in layer convolutional blocks. We also used pairs of convolutional layers in series to remedy our padding issues; the first layer in each pair is padded such that the output size is equal to the input size while the second layer shrinks the output. All max pool layers are 2x2 pool stride 2 layers, while all dropout layers unlink 25% of all input neurons. We progressively increase the number of filters in each convolutional block (except the third due to memory concerns) to increase the number of high-level features learned by the network; this is a progressive increase to limit the amount of parameters that would be used otherwise. This architecture served us well in breaking past 0.91 on our F2-score.

To handle the multi-class, multi-label objective, for each image we predict a score for each label and minimize the binary cross-entropy loss across all labels. We determine whether to assign a label based on whether the associated score exceeds a constant threshold.

During each of the training runs, this architecture is first trained with a binary-cross-entropy loss, discarding outlier images. Once finished, we lower the learning rate and train based on a modified loss function that adds the binary-cross-entropy loss with a custom F2 score loss that we created. This heuristic has empircally given us consistent small gains in both validation and test F2 scores.

Augmentations

This section details various modifications we make to our base neural net to improve our F2 score.

Determining the Prediction Thresholds for F2 Score

Our base model only learns to minimize the cross-entropy loss between its predicted scores and the true labels. In order for the model to actually make predictions, for each class it must determine a threshold for which a score higher than the threshold corresponds to a positive prediction and a score lower than the threshold corresponds to a negative prediction. These parameters are not part of our model, and attempting to optimize these by some analytic means is extremely difficult. Therefore, we determine these thresholds empirically: for each class we determine the threshold

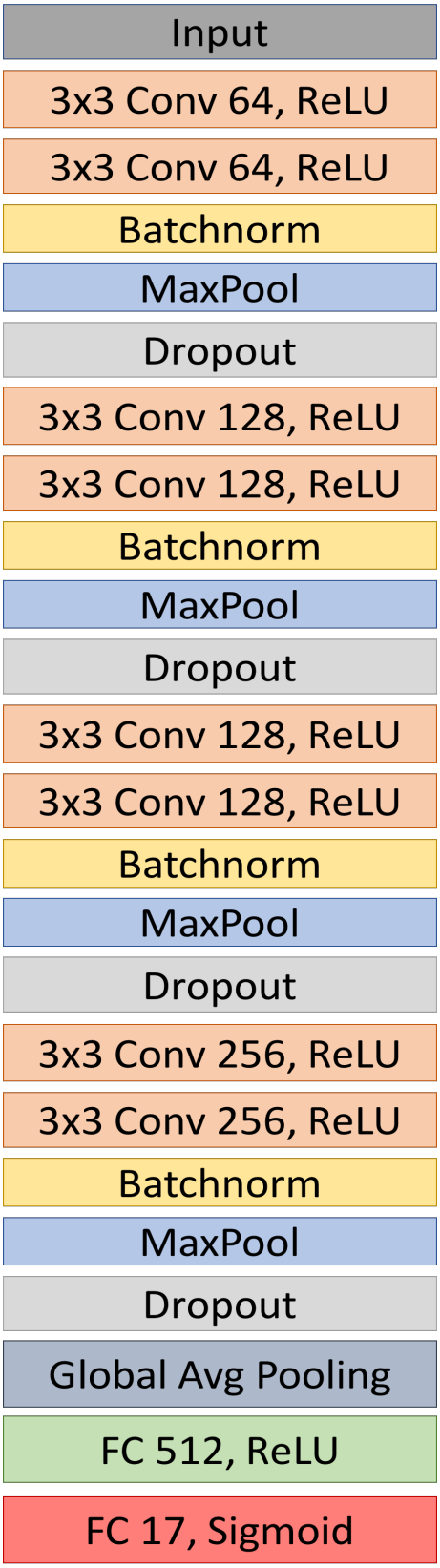


Figure 2. The architecture of our model.

which maximizes the F2 score on a small validation set by brute-force computing the score at threshold values of 0, 0.001, 0.002, etc.

Handling Noisy Data: Removing Outliers

We follow a variation of the approach in [23], which is in turn based on a simple variation of the k-nearest-neighbor algorithm, to cope with potentially mislabeled data. In particular, we performed the following process:

- Train the model for some number of epochs. Once the model is sufficiently trained, the values in its later layers may be interpreted as representations in some feature space.
- Append 17 extra dimensions corresponding to the 17 labels to the feature space, with an image's entry in each coordinate equal to 0 if it does not have the label, and λ if it does. The intuition here is that if the data are mostly separated in this feature space, they should cluster strongly in the label-augmented space. Then any mis-labeled data will be very far from other members of both its label and its true class, and λ is a parameter that describes how strongly we weigh the importance of being an outlier in the original feature space against the importance of being an outlier in the label-augmented space.
- Compute the distances between all the points in this feature space, and find the n points which maximize the sum of distances to the k nearest neighbors.
- Remove these n points from the dataset and resume training.

In particular, we execute this process with n = 500 (i.e, removing only 1.2 percent of the dataset), k = 10, and λ = 23.

Model Ensembling

We trained 3 separate instances of our model, and took the mean of the predicted scores before performing our F2 thresholding heuristic. This did not result in a consistent net increase on our strongest model, however, it consistently provided about +.015 on simpler versions of our model.

Results and Discussion

One of our final models achieved an F2 score of 0.923 on Kaggle's hidden test set, resulting in a ranking of 72nd on the Kaggle leaderboard; moreover, we achieved scores as high as 0.94 on our validation set. To provide context and scale to the results, the top-ranked entry as of 12 June 2017

has a score of 0.9333, and the difference between consecutive entries near the top is very often between 0.0001 and 0.0005. The following figure describes the rough benefit obtained by each approach:

Agent	F-score
Baseline - Starter Code	.84
Improved Architecture	.901
F2 Thresholding Heuristic	+.007
Batch Normalization, Global Average Pooling	+.004
Dataset Augmentation	+.003
Outlier Removal	+.004
Model Ensembling	+0

Table 1. Scores Achieved by Various Augmentations (in order)

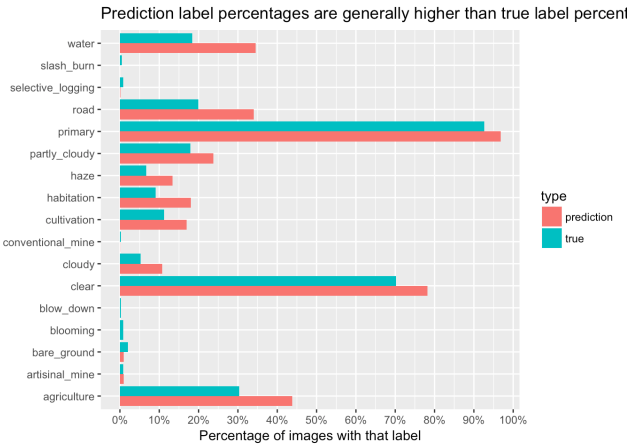


Figure 3. Distribution of our model's predictions on the training set. It differs quite notably from the base distribution.

Failed Approaches

Given that the weather labels were mutually exclusive, we attempted to restrict the weather labels to a single prediction using a separate CNN whose scores were converted to predictions with a different scheme. We hypothesized that this might work in particular because the weather features were likely to be a function of fundamentally different patterns from the other features- e.g, hazy weather is a function of the whole image being brown, whereas the presence of a river is a local feature. However, this approach did not succeed in practice. One straightforward reason for this is the fact that the F2 metric penalizes false negatives more heavily than false positives. Therefore, there are many situations where it may be better for the expected F2 score to predict two labels, guaranteeing a false positive to hedge against the risk of a false negative. Moreover, in separating the weather labels, we lose out on the benefits of multitask learning, as described in [4]- our model for the non-weather labels now does not implicitly learn features specific to each weather type.

We also quickly experimented with removing labels on the instances marked "cloudy", as well as with normalizing the weather class scores so that they would sum to 1, before computing the F2 thresholds. These approaches failed for similar reasons- presumably, the unnormalized weather class scores were actually distributed more appropriately for maximizing the F2 score.

Weaknesses of the Model

The distribution of the predictions show two key differences between what our model predicts and the true labels. First, a few labels are almost never predicted: "slash_burn", "conventional_mine", "blow_down", and "blooming". This is most likely because there are very few training images with these labels (209, 100, 101, 332 respectively) , and so the model cannot properly learn the features that correspond to those labels. Second, each of the other labels are predicted much more often than they should be. This can be explained by the fact that the F2 score penalizes false negatives much more heavily than false positives and so the model compensates for that by simply predicted more labels to attempt to reduce the number of false negatives. For example, images with the true label "haze" almost always have a "cloudy" prediction as well, presumably because the two atmospheric conditions tend to look fairly similar and so our model maximizes the expected F2 score by simply predicting both.

More broadly, since our model is not an excellent approximator of the original distribution of labels, even though it achieves a high F2 score, in practice it may not be very useful in a variety of situations. Obvious examples from our analysis include situations where it is relatively important to predict rarer labels, or situations that more heavily penalize false positives.

Future Work

Exploiting the Label Correlations

Although it is theoretically possible for a multitask network to learn relationships between the classes, when working with a dataset of limited size it should be helpful to more explicitly encourage the network to make use of our prior information- e.g, the empirical co-occurrence of classes, as well as the fact that the weather classes are mutually exclusive and the cloudy label is exclusive.

Broadly, we can think of two simple classes of methods for doing this:

- Add components to the loss which slightly penalize having high scores in two weather classes, as suggested by [15]. As a concrete example, one could add to the loss some constant times the product of all

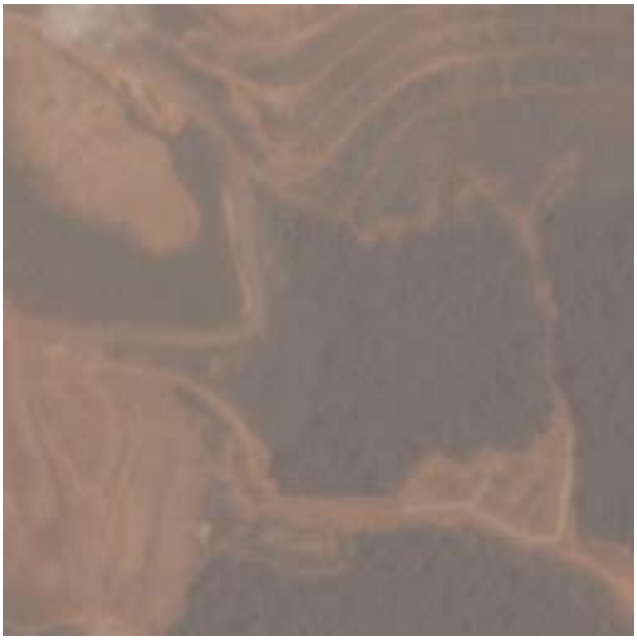


Figure 4. The model does not have enough training images with the "conventional_mine" label to learn the corresponding features and so predicts "water agriculture habitation" instead of "conventional_mine"

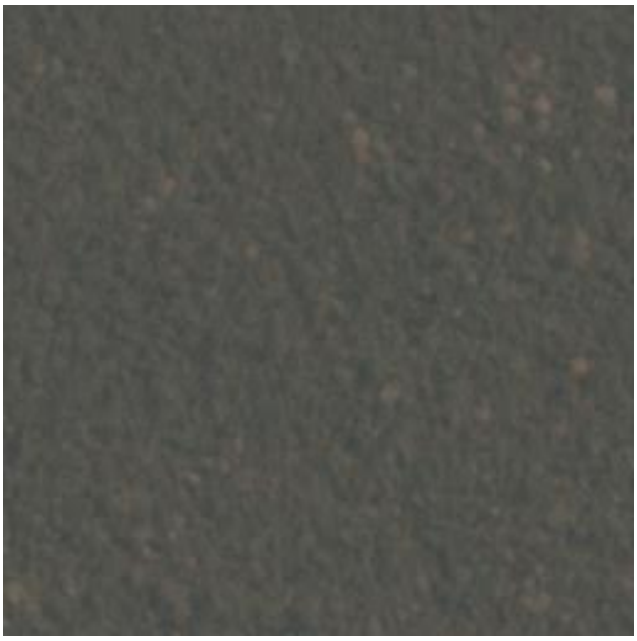


Figure 5. The model correctly predicts "clear" and "primary", but does not recognize "blow_down", the rarest label of which the training set only contains 101 examples.

weather class scores for each training sample. This quantity would be low when all weather class scores are high, and high when multiple weather class scores were medium.

- Train for the labels in a hierarchical structure, for example, by first training a network that predicts the weather label scores (not necessarily exclusively), and then either using the weather label scores as inputs to a second network, or using a different network for each weather label. (We experimented with this briefly, but abandoned the approach due to the limitations of training time and memory.)

Different Architectures

We mostly only explored approaches involving sequential neural nets, but some other competitors such as [26] showed good results with non-sequential structures. In particular, [26] uses a structure which passes both shallower and deeper convolutional layers as inputs into a higher sequence of convolutional layers, and uses many of these as inputs to an affine layer. The intuition for this kind of approach is that the final classifier may be able to more explicitly make decisions based on lower-scale features such as textures and visually small features, such as the texture of trees in the "primary" or "cloudy" label or the presence of tiny blooms for the "blooming" label.

Transfer Learning

Since our training set is not massive, it might help to adopt some of the early-middle layers of a known image classification model to improve initialization.

Conclusion

From our results, we note that standard multi-task sequential convolutional neural network architectures perform quite well at detecting atmospheric conditions and the human impact on nature from satellite data. The models we built took mere hours to train, compared to previous examples of deforestation detection methods that relied on processing MODIS or Landsat images from days to months; this opens up a tremendous wealth of possiblities for data analysis that we are all excited to see. Additionally, while the potential to greatly reduce the amount of human effort needed to locate areas of human impact anywhere around the world is fantastic, our techniques are generalizable to many image classification problems that may have been neglected thus far. We look forward to using the experience gained from building and tuning our model to explore the field further and refine the field of analytical satellite imagery.

Acknowledgements

Parts of our base model architecture are modeled after [22], who shared his code on the Kaggle competition's dis-

cussion forum. [25] provided the original basis of input image reading and processing code, as well as the basis of our F2 thresholding function; we also gained some insight from some of his dataset visualizations on the Kaggle kernsls page. We also gained significant insight from [26]'s discussion on the most important factors in performing well on this competition.

References

[1] "Deep Learning Could Predict Deforestation Before It Happens." Orbital Insight. N.p., 07 Apr. 2016. Web. 12 June 2017. ¡https://orbitalinsight.com/deep-learning-predict-deforestation-happens-3/¿.

[2] "Deforestation." WWF. World Wildlife Fund, n.d. Web. 12 June 2017. ¡https://www.worldwildlife.org/threats/deforestation¿.

[3] Diaz, Emiliano. "Online Deforestation Detection." N.p., 03 Apr. 2017. Web. 12 June 2017. ¡https://arxiv.org/abs/1704.00829¿.

[4] Rich Caruana. 1997. Multitask Learning. Mach. Learn. 28, 1 (July 1997), 41-75. DOI=http://dx.doi.org/10.1023/A:1007379606734

[5] Saikat Basu, Sangram Ganguly, Supratik Mukhopadhyay, Robert DiBiano, Manohar Karki, and Ramakrishna Nemani. 2015. DeepSat: a learning framework for satellite imagery. In Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '15). ACM, New York, NY, USA, , Article 37 , 10 pages. DOI: https://doi.org/10.1145/2820783.2820816

[6] Jean, Neal, Marshall Burke, Michael Xie, W. Matthew Davis, David B. Lobell, and Stefano Ermon. "Combining Satellite Imagery and Machine Learning to Predict Poverty." Science. American Association for the Advancement of Science, 19 Aug. 2016. Web. 05 June 2017. ¡http://science.sciencemag.org/content/353/6301/790¿.

[7] "Satellite Alerts Track Deforestation in Real Time." Nature News. Nature Publishing Group, n.d. Web. 13 June 2017. ¡http://www.nature.com/news/satellite-alerts-track-deforestation-in-real-time-1.19427¿.

[8] C. G. Diniz et al., "DETER-B: The New Amazon Near Real-Time Deforestation Detection System," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 8, no. 7, pp. 3619-3628, July 2015. doi: 10.1109/JSTARS.2015.2437075

[9] Hansen, M. C., P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R. G. Townshend. "High-Resolution Global Maps of 21st-Century Forest Cover Change." Science. American Association for the Advancement of Science, 15 Nov. 2013. Web. 13 June 2017. ¡http://science.sciencemag.org/content/342/6160/850¿.

[10] Jialei Wang, Peder A. Olsen, Andrew R. Conn, and Aurelie C. Lozano. 2016. Removing Clouds and Recovering Ground Observations in ... (April 2016). Retrieved June 5, 2017 from https://arxiv.org/pdf/1604.03915v1.pdf

[11] Emmanuel Maggiori, Guillaume Charpiat, Yuliya Tarabalka, and Pierre Alliez. 2016. Recurrent Neural Networks to Correct Satellite Image Classification Maps. IEEE Transactions on Geoscience and Remote Sensing (August 2016), 110. DOI:http://dx.doi.org/10.1109/tgrs.2017.2697453

[12] Ye, Nan, Kian Chai, Wee Sun Lee, and Hai Leong Chieu. "Optimizing F-Measures: A Tale of Two Approaches." Proceedings of the 29 Th International Conference on Machine Learning (2012)

[13] Lipton, Zachary C., Charles Elkan, and Balakrishnan Naryanaswamy. "Thresholding Classifiers to Maximize F1 Score." Thresholding Classifiers to Maximize F1 Score - Open Access Library.

[14] Fan, Rong-en, and Chih-Jen Lin. "A Study on Threshold Selection for Multi-label Classification." Diss. Nanyang Technological U, n.d. Web. 11 June 2017. ¡https://www.csie.ntu.edu.tw/ cjlin/papers/threshold.pdf¿.

[15] Dembczynski, Krzysztof, Arkadiusz Jachnik, Wojciech Kotowski, Willem Waegeman, and Eyke Hllermeier. "Optimizing the F-Measure in Multi-Label Classification: Plug-in Rule Approach versus Structured Loss Minimization." (n.d.): n. pag. Web. 12 June 2017. ¡http://proceedings.mlr.press/v28/dembczynski13.pdf¿.

[16] O. Gharroudi, H. Elghazel and A. Aussem, "Ensemble Multi-label Classification: A Comparative Study on Threshold Selection and Voting Methods," 2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI), Vietri sul Mare, 2015, pp. 377-384. doi: 10.1109/ICTAI.2015.64

[17] Lin, Tsung-Yi, Piotr Dollr, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie.

”Feature Pyramid Networks for Object Detection.” N.p., 19 Apr. 2017. Web. 11 June 2017. ¡https://arxiv.org/abs/1612.03144¿.

[18] Qingshan Liu, Li Zhi, Renlong Hang, and Huihui Song. ”Learning Multi-Scale Deep Features for High-Resolution Satellite Image Classification.” (n.d.): n. pag. 11 Nov. 2016. Web. 11 June 2017. ¡https://arxiv.org/pdf/1611.03591v1.pdf¿.

[19] Martn Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Vigas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

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

[21] Franois Chollet, Keras: Deep Learning for Python, (2017), GitHub repository, https://github.com/fchollet/keras

[22] Tuatini Godard, Kaggle Amazon deforestation challenge, (2017), GitHub repository, https://github.com/EKami/planet-amazon-deforestation

[23] Fabrizio Angiulli, Clara Pizzuti, Fast Outlier Detection in High Dimensional Spaces (2002), Conference: Principles of Data Mining and Knowledge Discovery,

[24] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang, Learning from Massive Noisy Labeled Data for Image Classification, Baidu Research.

[25] https://www.kaggle.com/anokas/simple-keras-starter/

[26] https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/discussion/32402