Machine Learning Engineer Nanodegree

Capstone Project Report

Nicholas Condo
April 25, 2020

Understanding the Amazon from Space

Project Overview

The Amazon is the largest and most biodiverse tropical rainforest in the world, covering an area of 2.1 million square miles. It is comprised of an estimated 390 billion individual trees divided into 16,000 species. The Amazon has been referred to as the "lungs of the planet", as it helps to stabilize the earth's climate and slow global warming by fixing CO_2 and producing 20% of the world's oxygen.

Since 1978 over 289,000 square miles of Amazon rainforest have been destroyed across Brazil, Peru, Columbia, Bolivia, Venezuela, Suriname, Guyana, and French Guiana. Every minute, the world loses an area of forest the size of 48 football fields. There is concern that the destruction of the forest will result in loss of biodiversity, habitat loss, and the release of the carbon contained within the vegetation, which could accelerate global warming. Better data about the location of deforestation and human encroachment on forests can help governments and local stakeholders respond more quickly and effectively.

Problem Statement

While considerable research has been devoted to tracking changes in forests, it typically depends on coarse-resolution imagery from satellites such as Landsat (30 meter pixels) or MODIS (250 meter pixels). This limits its effectiveness in areas where small-scale deforestation or forest degradation dominate. Furthermore, these existing methods generally cannot differentiate between human causes of forest loss and natural causes.

The goal of this project is to track changes in the Amazon rainforest due to deforestation using satellite image data. The data has been provided by Planet and hosted on kaggle.com as part of a previous competition – Planet: Understanding the Amazon from Space [1]. The provided satellite chips have a ground-sample distance (GSD) of 3.7m and an orthorectified pixel size of 3m. The labels for this task were chosen in collaboration with Planet's Impact team and represent a reasonable subset of phenomena of interest in the Amazon basin. The task is a multi-label image classification problem, where each image will have one and potentially more than one atmospheric label and zero or more common and rare labels.

Over the past few years, computer vision has taken giant leaps due to advances in deep learning and convolutional neural networks. At this point, deep leaning models are able to outperform humans on many image classification tasks, such as [ImageNet](). One great benefit that has come from researchers striving to develop the best performing model on ImageNet, has been the advent of transfer learning. Transfer learning is when a model developed for one task is reused for a model on a second task. In short, a model trained on ImageNet can then be "fine-tuned" on a different dataset, achieving state-of-the-art results while requiring minimal training time. This technique has become prevalent in computer vision applications, and popular deep learning frameworks today all provide built-in support for using popular models trained on ImageNet. Harnessing the power of transfer learning, I plan to develop a deep learning model to classify the satellite images using the PyTorch deep learning framework and the fastai library.

Metrics

Performance of the model will be measured by the F2 score. The F2 score, commonly used in information retrieval, measures accuracy using the precision p and recall r. Precision is the ratio of true positives (tp) to all predicted positives (tp + fp). Recall is the ratio of true positives to all actual positives (tp + fn). The F2 score is given by:

$$(1+\beta^2)\frac{pr}{\beta^2 p + r} \quad \text{where} \quad p = \frac{tp}{tp+fp}, \quad r = \frac{tp}{tp+fn}, \ \beta = 2.$$

I will measure the success of the model based on the leaderboard of the competition: Planet: Understanding the Amazon from Space. The winner of this competition achieved a score of 0.93317 on the private leaderboard, and I will use that as my benchmark.

Data Exploration & Visualization

The dataset consists of 40,479 training images and 61,191 test images, and each image is a 256x256 sized jpeg. The file train_v2.csv is included which lists the training file names and their accompanying labels. The labels can be broken down into three categories: cloud cover labels, common labels, and less common labels. There are 17 labels in total: clear, partly cloudy, cloudy, haze, primary, water, habitation, agriculture, road, cultivation, bare ground, slash and burn, selective logging, blooming, conventional mining, artisanal mining, and blow down.

Cloud Cover Labels
Clouds are a major challenge for passive satellite imaging, and daily cloud cover and rain showers in the Amazon basin can significantly complicate monitoring in the area. Clear scenes show no evidence of clouds, and partly cloudy scenes can show opaque cloud cover over any portion of the image. Cloudy images have 90% of the chip obscured by opaque cloud cover.

Figure 1: Example of a cloudy scene. Credit: [Planet]

Primary Rain Forest

The overwhelming majority of the data set is labeled as "primary", which is shorthand for primary rainforest, or what is known colloquially as virgin forest. Generally speaking, the "primary" label was used for any area that exhibited dense tree cover.

Figure 2: Approximately 25,000 acres of untouched primary rainforest. Credit: [Planet]

Water (Rivers & Lakes)

Rivers, reservoirs, and oxbow lakes are important features of the Amazon basin, and we used the water tag as a catch-all term for these features. Rivers in the Amazon basin often change course and serve as highways deep into the forest. The changing course of these rivers creates new habitat but can also strand endangered Amazon River Dolphins.

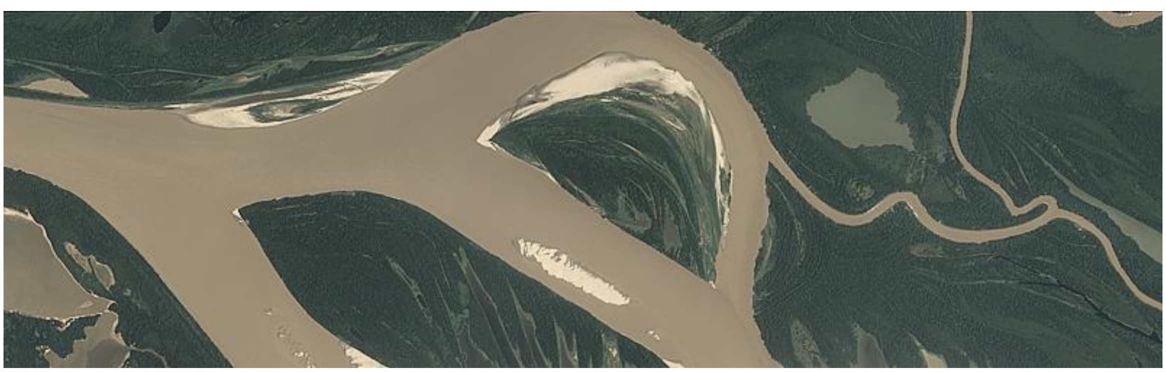


Figure 3: A large, slow river with significant sand bars. Credit: Planet

Habitation
The habitation class label was used for chips that appeared to contain human homes or buildings. This includes anything from dense urban centers to rural villages along the banks of rivers. Small, single-dwelling habitations are often difficult to spot but usually appear as clumps of a few pixels that are bright white.

Figure 4: A large city in the Amazon basin. Credit: [Planet](#)

Agriculture
Commercial agriculture, while an important industry, is also a major driver of deforestation in the Amazon. For the purposes of this dataset, agriculture is considered to be any land cleared of trees that is being used for agriculture or range land.

Figure 5: An agricultural area showing the end state of "fishbone" deforestation. Credit: [Planet](#)

Road
Roads are important for transportation in the Amazon, but they also serve as drivers of deforestation. In particular, "fishbone" deforestation often follows new road construction, while smaller logging roads drive selective logging operations. For our data, all types of roads are labeled with a single "road" label. Some rivers look very similar to smaller logging roads, and consequently there may be some noise in this label.

Figure 6: Classic "fishbone" deforestation following a road. Credit: [Planet]

Cultivation

Shifting cultivation is a subset of agriculture that is very easy to see from space and occurs in rural areas where individuals and families maintain farm plots for subsistence. This type of agriculture is often found near smaller villages along major rivers, and at the outskirts of agricultural areas. It typically relies on non-mechanized labor and covers relatively small areas.

Figure 7: Zoomed-in area showing cultivation along the right side of the river. Credit: [Planet]

Bare Ground

Bare ground is a catch-all term used for naturally occurring tree free areas that aren't the result of human activity.

Figure 8: A naturally occurring bare area in the cerrado. Credit: [Planet]

Slash and Burn
Slash-and-burn agriculture can be considered to be a subset of the shifting cultivation label and is used for areas that demonstrate recent burn events. This is to say that the shifting cultivation patches appear to have dark brown or black areas consistent with recent burning.

Figure 9: A zoomed-in view of an area with shifting cultivation and evidence of a recent fire. Credit: [Planet]

Selective Logging
The selective logging label is used to cover the practice of selectively removing high value tree species from the rainforest (such as teak and mahogany). From space this appears as winding dirt roads adjacent to bare brown patches in otherwise primary rain forest.

Figure 10: The brown lines on the right are logging roads. Note the small brown dots around the road. Credit: [Planet]

Blooming
Blooming is a natural phenomenon found in the Amazon where particular species of flowering trees bloom, fruit, and flower at the same time to maximize the chances of cross pollination.

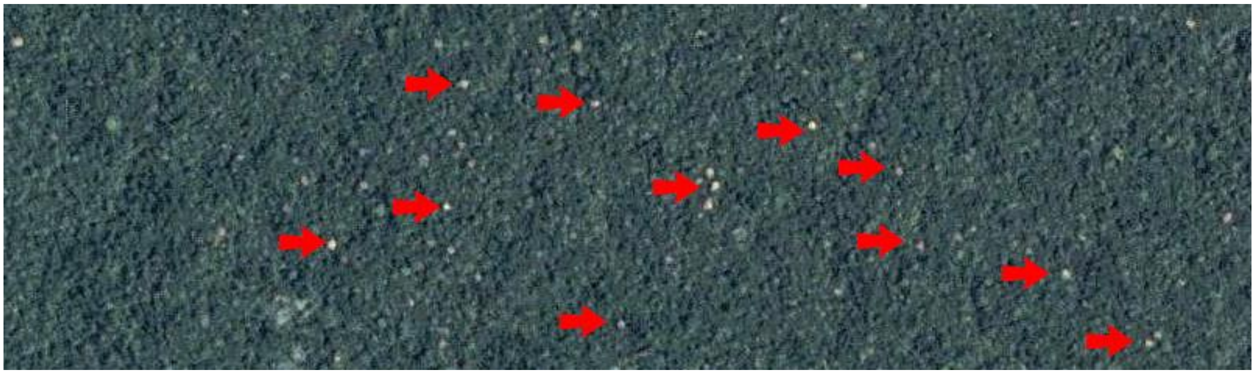


Figure 11: A zoomed-in view of a bloom event in the Amazon basin. Credit: [Planet]

Conventional Mining

There are a number of large conventional mines in the Amazon basin and the number is steadily growing. This label is used to classify large-scale legal mining operations.

Figure 12: A conventional mine in the Amazon basin. Credit: [Planet]

Artisanal Mining

Artisanal mining is a catch-all term for small scale mining operations. Throughout the Amazon, especially at the foothills of the Andes, gold deposits lace the deep, clay soils. Artisanal miners, sometimes working illegally in land designated for conservation, slash through the forest and excavate deep pits near rivers. They pump a mud-water slurry into the riverbanks, blasting them away so that they can be processed further with mercury - which is used to separate out the gold. The denuded moonscape left behind takes centuries to recover.

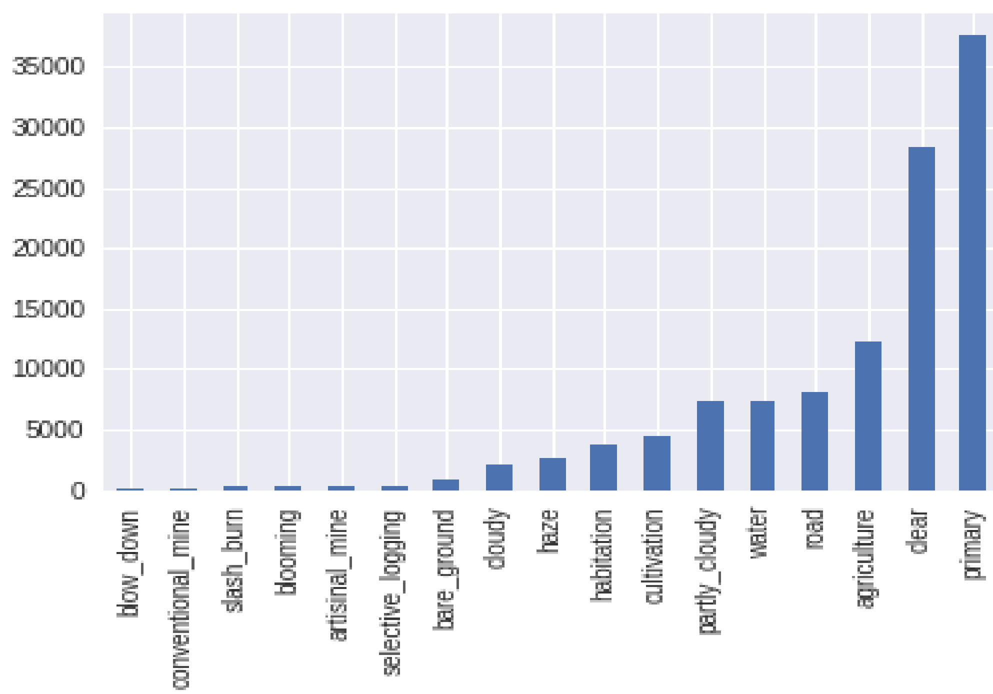
Figure 13: A zoomed-in image of an artisanal mine in Peru. Credit: [Planet]

Blow Down

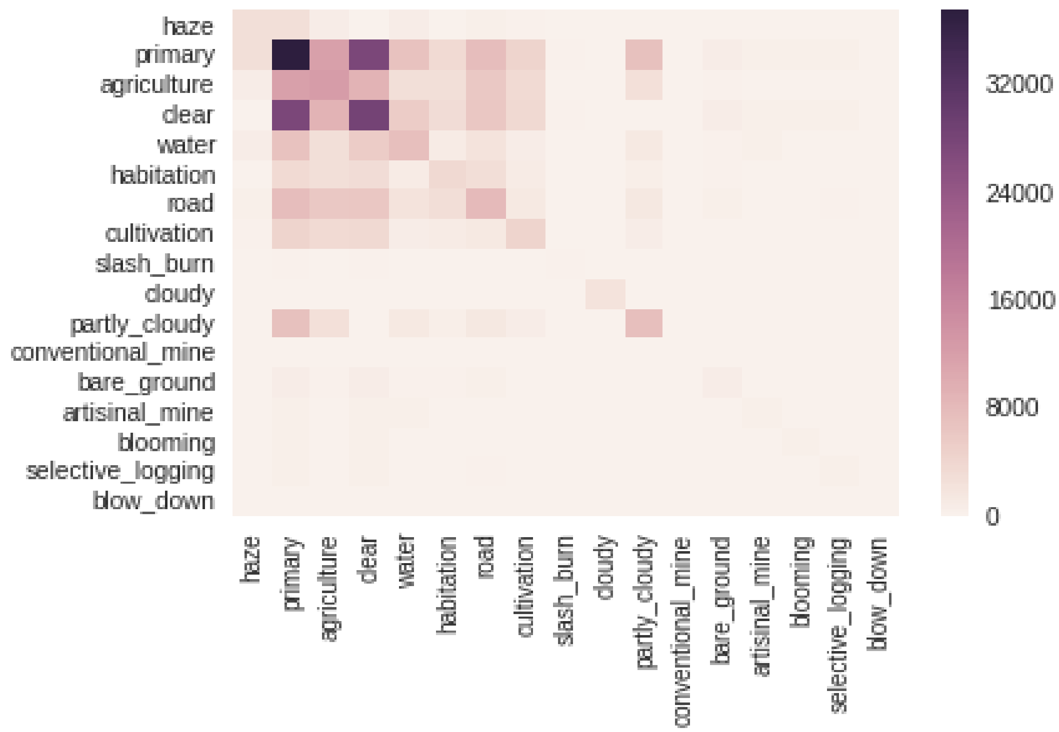
Blow down, also called windthrow, is a naturally occurring phenomenon in the Amazon. Briefly, blow down events occur during microbursts where cold dry air from the Andes settles on top of warm moist air in the rainforest. The colder air punches a hole in the moist warm layer and sinks down with incredible force and high speed (in excess of 100MPH). These high winds topple the larger rainforest trees, and the resulting open areas are visible from space.

Figure 14: A blow down event in the Amazon. Credit: Planet

A histogram of the label occurrences shows a distribution of the labels, with most frequent label being 'primary':



A heatmap of the cooccurrences of the labels gives an idea of how often the labels appear for the same image:



Data Preprocessing

Because this data was acquired through a Kaggle competition, not much data preprocessing was necessary as far as cleaning up the data. However, the images are preprocessed and augmented as they are fed into the neural network for training. Each image is normalized using the mean and standard deviation from ImageNet data, since each of the models I used had been pre-trained on ImageNet.

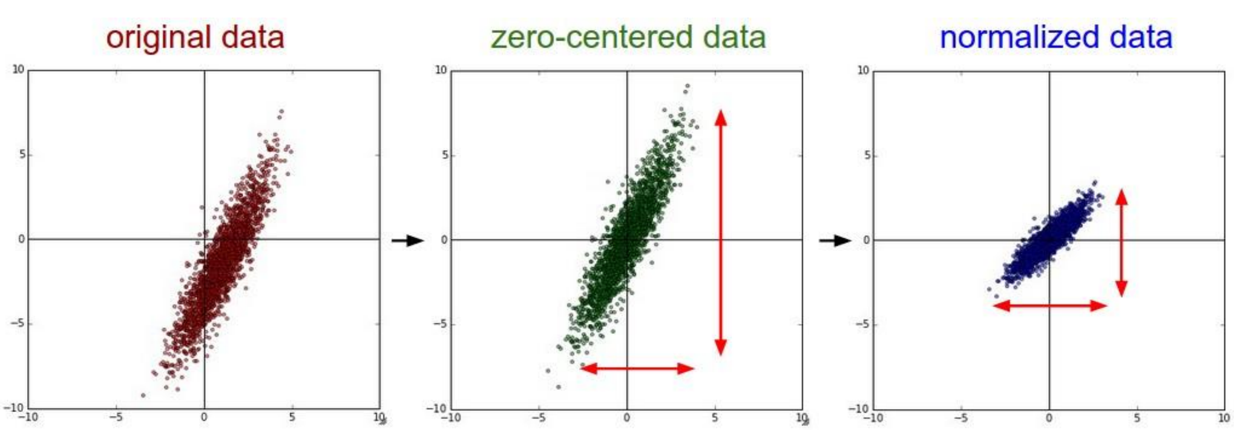


Figure 15: Common data preprocessing technique. Credit: Stanford CS231n [2]

Another technique commonly implemented when training computer vision models is data augmentation. Data augmentation is implemented in the fastai library using the get_transforms() function. From fastai docs:

get_transforms [test] [source]

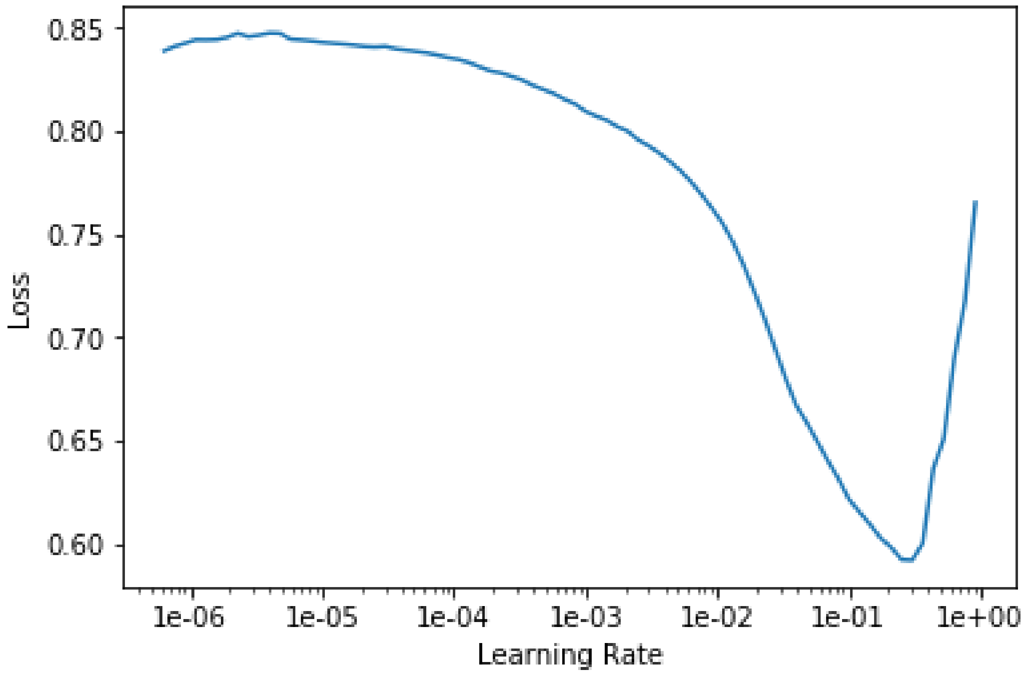
```
get_transforms ( do_flip : bool = True ,  flip_vert : bool = False ,
max_rotate : float = 10.0 ,  max_zoom : float = 1.1 ,  max_lighting : float = 0.2 ,
max_warp : float = 0.2 ,  p_affine : float = 0.75 ,  p_lighting : float = 0.75 ,
xtra_tfms : Optional [ Collection [ Transform ] ] = None )  →  Collection [ Transform ]
```

- *do_flip:* if True, a random flip is applied with probability 0.5.
- *flip_vert:* requires do_flip=True. If True, the image can be flipped vertically or rotated by 90 degrees, otherwise only a horizontal flip is applied.
- *max_rotate:* if not None, a random rotation between -max_rotate and max_rotate degrees is applied with probability p_affine.
- *max_zoom:* if not 1.0 or less, a random zoom between 1.0 and max_zoom is applied with probability p_affine.
- *max_lighting:* if not None, a random lightning and contrast change controlled by max_lighting is applied with probability p_lighting.
- *max_warp:* if not None, a random symmetric warp of magnitude between -max_warp and maw_warp is applied with probability p_affine.
- *p_affine*: the probability that each affine transform and symmetric warp is applied.
- *p_lighting:* the probability that each lighting transform is applied.
- *xtra_tfms:* a list of additional transforms you would like to be applied.

I kept most of the default values for the transforms, except I set flip_vert=True, max_lighting=0.1, max_zoom=1.05, and max_warp=0.

Implementation

For this task, I chose to implement deep learning models using the fastai library, which sits on top of PyTorch. The fastai library simplifies training neural nets using modern best practices. One of the staples of the fastai library is its default implementation of Cyclical Learning Rates for Training Neural Networks – Leslie N. Smith [3]. This is a relatively unknown technique of setting the learning rate, which improves classification accuracy while needing fewer iterations vs fixed learning rate values. Instead of monotonically decreasing the learning rate, this method lets the learning rate cyclically vary between reasonable boundary values. Another benefit of fastai is that it makes transfer learning incredibly easy.



My final solution consists of an ensemble of five models all pre-trained on ImageNet: resnet50, resnet101, resnet152 – He, et al. [4], densenet121, and densenet169 – Huang, et al. [5]. I fine-tuned each model separately, splitting the training data into 80% training and 20% validation. I used progressive resizing of the images to train the models, a technique I learned about from the fast.ai course [6], taught by Jeremy Howard. Progressive resizing is a method of training computer vision models where you first train the model on a smaller size than the original, then continue to increase the size of the image two or three times until you finally train on the original image size. I first trained the models on 64x64 pixel images for five epochs with only the last layer of the network trainable, then another five epochs with the entire network trainable. I do the same for 128x128 pixel images, and then the original 256x256 pixel images. I tried using sizes of 128 -> 224 -> 256, but found better performance with 64 -> 128 -> 256. Finally, I trained for five more epochs on all of the available training data, combining the training and validation sets. I also tried semi-supervised learning, using the labels from my best submission to train on the test set, but found my performance to slightly decrease. I use batch sizes of 64 during the entire training process, while the learning rates are chosen prior to each block of five epochs using fastai's lr_find() function.

To ensemble the five models, I get the predictions for each model, and combine them by taking the mean of the probabilities for each label. Since this is a multi-label classification problem, I need to set a threshold on which to include a label. I implemented a function to find an optimal threshold for each label separately, but I actually achieved better performance setting the threshold to .17 for all labels. One other method I leveraged, which significantly boosted performance, was test-time augmentation. This means that the data augmentation used to train the model is also performed when making predictions. The fastai library provides a function for this which takes the average of the regular predictions with weight beta=0.4 with the average of predictions obtained through augmented versions with weight 1-beta.

Refinement

In order to get a baseline score with which to measure future experiments' impact, I implemented a resnet50 without progressive resizing, using an image size of 256x256 pixels. I did not use test-time augmentation, and set the threshold to 0.20. This model achieved an F2 score of 0.92746 on the competition's private leaderboard. When experimenting, I always tested and looked to improve the F2 score on my validation set, and only after I was satisfied with a feature I would test it on the competition leaderboard. This was to prevent overfitting to the test set. Overall, I found my validation scores to be in line with leaderboard results. To improve upon the baseline score, I first implemented progressive refinement. I experimented with a couple different sizes, and settled on 64 -> 128 -> 256, which bumped up my F2 score to 0.92881. After utilizing test-time augmentation my score increased further to 0.92979. I then began experimenting with thresholding the predictions to see if I could squeeze out a couple tenths of a percent. I was able to increase my score to 0.93024 by setting the threshold to 0.17. Finally, I combined the training and validation sets and trained for five more epochs to achieve a score of 0.93041 with the single resnet50 model.

Results

I followed the same process as above with the other four models, and by creating an ensemble of the five models I was able to achieve an F2 score of 0.93173 on the private leaderboard. This score would've been good for 19th place out of 938 in the competition, only 0.00144 behind 1st place.

ensemble5-thresh17.csv 4 hours ago by ncondo Ensemble of resnet50, resnet101, resnet152, densenet121, and densenet169	0.93173	0.93296

6	▾5	SYSU CISLab	+3	0.93265	203
7	▴1	ods.ai	+4	0.93224	163
8	▴7	dhammack		0.93221	46
9	▴10	CCT		0.93220	444
10	—	deepsense.io		0.93216	208
11	▾4	Kyle		0.93211	262
12	▴11	young		0.93206	49
13	▾10	Urucu		0.93205	228
14	▴13	To Train Them Is My Cause		0.93201	20
15	▾10	Team samogon		0.93192	192
16	▴2	bang liu		0.93187	43
17	▴11	Heidelberg Uni & MIPT		0.93184	72
18	▴7	null		0.93183	10
19	▾6	🌴 team-amazon-forest🌴		0.93167	351
20	▾6	CVFakers		0.93159	173

Reflection

The process used for this project can be summarized as follows:

1. Identify a problem which can be solved using machine learning.
2. Find a relevant, publicly available dataset.
3. Identify a benchmark on which to compare the final result.
4. Create a baseline model to try experiments on and iteratively improve upon.
5. Achieve good performance using a single model, and train other popular architectures in a similar manner.
6. Combine the models' predictions to form an ensemble, and postprocess the results with different label thresholds.

I found step 6 to be the most challenging, yet also the most interesting. I had never created an ensemble with deep learning models previously, so learning about the theory [7] and how to successfully implement it in code was quite satisfying. Finding different thresholds for each label was a bit tedious and I encountered quite a few errors while trying to implement a solution. In the end, I created a working function to optimize the thresholds using the training set, though I found better performance by using just a single threshold for all labels. Others in the competition reported better results with varying thresholds, so this could be an area I revisit in the future.

Improvement

One of the techniques I would like to have tried but did not have the time/resources is Single Image Haze Removal Using Dark Channel Prior – He, et al. [8]. I believe this would've been quite beneficial as a many of the satellite images included haze and/or clouds which made it more difficult for the model to classify the underlying features of the forest. In fact, the winner of the competition mentioned that he used this technique in his brief description of his strategy [9]. I actually did implement the algorithm to process the entire dataset, but it took my local box about 30 hours just to process the training data, and I did not get to process the test data. This is something I plan to do in the future.

Another strategy I could try is k-fold cross validation, instead of splitting the data into train and validation sets. This way the model can see all of the training data throughout the process, instead of training on the validation data separately. The 3rd place finisher mentioned using 5-fold cross validation in their solution [10].

I could also try training even more models or different architectures to experiment with larger ensembles. The winner of the competition mentioned having a model for each different label, and the 3rd place finisher used 30 different models and after 5-fold cross-validation resulted in 150 weights files.

Overall, I am extremely happy with my results – achieving top 2% on the leaderboard with an ensemble of only five models. I hope to achieve top 10, or even best the 1st place finisher, by implementing some of the improvements above.

References

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

[2] CS231n: Convolutional Neural Networks for Visual Recognition.
https://cs231n.github.io/

[3] Leslie N. Smith, Cyclical Learning Rates for Training Neural Networks.
https://arxiv.org/abs/1506.01186

[4] He, et al., Deep Residual Learning for Image Recognition.
https://arxiv.org/abs/1512.03385

[5] Huang, et al., Densely Connected Convolutional Networks.
https://arxiv.org/abs/1608.06993

[6] Jeremy Howard, fast.ai: Practical Deep Learning for Coders, v3.
https://course.fast.ai/

[7] Ju, et al., The Relative Performance of Ensemble Methods with Deep Convolutional Neural Networks for Image Classification.
https://arxiv.org/abs/1704.01664

[8] He, et al., Single Image Haze Removal Using Dark Channel Prior.
http://kaiminghe.com/publications/cvpr09.pdf

[9] @bestfitting, My brief overview of my solution.
https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/discussion/36809

[10] @ZFTurbo, 3rd place Solution.
https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/discussion/38831