Multiple Label Classification for Amazon Rainforest Images

Jeffrey Zhang
Stanford University
jz5003@stanford.edu

Albert Tung
Stanford University
atung3@stanford.edu

Abstract

Much work has been done in image classification tasks. We applied some of these previous approaches to the space of satellite imagery, particularly of the Amazon Rainforest. Our task was to train a multilabel classifier, which, given a satellite image of the Amazon Rainforest, attempted to predict whether or not certain tags (cloudy, habitation, water, etc.) applied to the image. We were evaluated based on our F2 score. We tried many deep convolutional neural network architectures and were able to achieve a test F2 score of .903 by training the VGG16 network from scratch.

1. Introduction

Deforestation has become a large problem globally and especially in the Amazon rainforest where large portions of the rainforest have been deforested. Not only does deforestation cause a loss of biodiversity, which could lead to undiscovered chemical and biological functions that could further knowledge in many fields, but also can contribute to climate change. Unfortunately, the vastness of the Amazon makes it difficult for government officials and advocates to understand where, when, and how the Amazon is getting deforested at such a rapid rate. Satellite images, however, are difficult to classify even for experts. With computer vision, we could potentially answer many of the questions and help reduce deforestation in crucial regions.

The problem involves classifying satellite images as one or more of 17 labels. These labels are as follows:

Atmospheric	Common Land	Rare Land
cloudy	primary	slash_and_burn
partly_cloudy	water	selective_logging
hazy	habitation	blooming
clear	agriculture	conventional_mining
	road	artisinal_mining
	cultivation	blow_down
	bare ground	

The input to our algorithm is a $256 \times 256 \times 3$ JPG satellite image that was read with standard RGB channels. The data was obtained from the Kaggle official website and was hand labeled by experts from Planet and the labels were stored in a csv file. We then feed the input and truth labels into several convolutional neural networks and receive a binary array of 17 outputs that tells us if the label is applicable to the image or not. Since the problem is a multilabel image classification problem, which means we must apply the techniques used for single-label classification but instead of using a softmax cross-entropy loss, we must use a sigmoid cross-entropy loss.

Our goal for this project is to explore the application of different convolutional architectures on this dataset as well as experimentation with pre-trained models. Our main objective is to determine which model, when trained from scratch, would best classify these images from the dataset.

2. Related Work

Much work has been done in the field of image classification, and approaches include advanced feature extraction, convolutional neural networks, and transfer learning. Even though our problem is a multilabel classification problem (on which there is less literature), we can still use methods for single label classification problems.

2.1. Advanced Feature Extraction

These approaches seek to encode images as feature vectors with a feature extractor. These are particularly useful for high resolution images with many channels ([4], [2]).

There are also unsupervised learning methods to learn feature extractors ([5]) as well as semisupervised learning methods for problems with small labeled datasets and large unlabeled datasets ([3]).

Feature vectors are then passed through classifiers such as support vector machines.

2.2. Convolutional Neural Networks

In recent years, convolutional neural networks have proven to be successful across a wide variety of image labeling tasks. Several works have applied deep convolutional

neural networks to satellite image classification image tasks in particular ([6], [12]). Many architectures used for the ImageNet Challenge (i.e. ResNet, VGG) can be applied to this problem as well.

Also relevant are fully convolutional neural network architectures, such as the UNet, which is generally used for segmentation but can be adapted to a classification problem ([8], [13])

2.3. Transfer Learning

A lot of work has been done on using transfer learning for image classification tasks ([11])

Because ImageNet is such a large dataset and several deep architectures have already been trained on it, they are a good source of pre-trained models for other classification problems, including ours.

3. Methods

We mainly tried different deep convolutional network architectures. We also experimented with pre-trained models from the ImageNet Challenge. For each network, a sigmoid cross entropy loss and a weighted sigmoid cross entropy loss was applied on the output logits instead of a softmax cross entropy loss because an image could have multiple labels. The weighted sigmoid cross entropy loss was chosen because the F2 score evaluation favors recall. Thus, by using the weighted loss we can penalize false negatives and improve our F2 score.

Our weighted loss function can be described using the following equation where w represents the weight that we put on a positive error relative to a negative error:

$$\text{targets} * -log(\sigma(\text{logits})) * w + (1 - \text{targets}) * -log(1 - \sigma(\text{logits}))$$

In additional, we also tried training separate classifiers for atmospheric labels and land labels since most if not all of the images only had one atmospheric label along with zero or more land labels. By using a softmax classifier for the atmospheric labels and a sigmoid classifier for the land labels, we attempted to ensure that each image had only a single atmospheric label.

3.1. Baseline

Our baseline was a simple two layer convolutional network with two fully connected layers. We applied batch normalization and pooling after the convolutional layers. A visual representation of our baseline model is shown in Figure 1.

3.2. Wide ResNet

Because ResNet was so successful on the ImageNet challenge, we wanted to apply it to our problem. However, ResNet is very slow to train from scratch, so we decided to train a wide residual network instead, which decreased the depth and increased the width. We managed to train two implementations of ResNet which were the 16×2 and the 16×4 models in which the numbers represent the number of convolutional layers in the model and the expanding factor respectively. Because each model took between 10 and 20 hours to train, we avoided higher dimension models to iterate quickly. A generalized image of our Wide ResNet model is shown in Figure 2 and for more details on the structure of the network see the original paper here: [wide resnet citation]

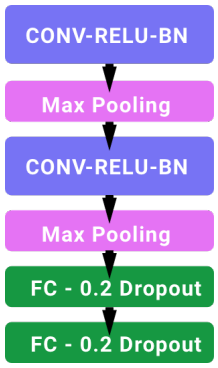


Figure 1: Architecture of baseline model

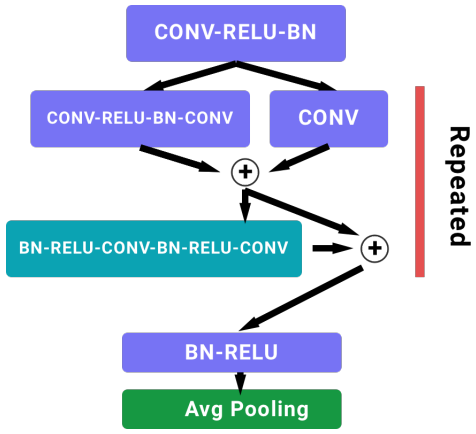


Figure 2: Architecture of Wide ResNet

3.3. VGGNet16

We tried various implementation of VGGNet16 that included the vanilla version, adding dropout layers between the fully connected layers at the end of the model, and including batch normalization in between the earlier convolutional layers of the model. Because of the depth of VGGNet16 as well as its faster training time, we felt VG-

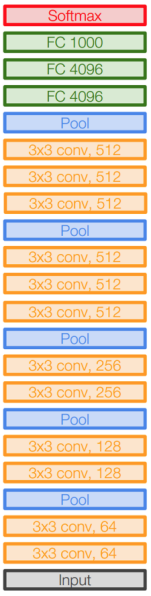


Figure 3: Architecture of VGGNet (CS 231N Lectures)

GNet16 would allow us to encompass a lot of features. Since VGGNet16 was implemented before the introduction of batch normalization, we wanted to try some implementations of the model with it to see if it would improve convergence and accuracy. In addition, we wanted to attempt combining some earlier layers with later layers would allow us to train on higher and lower level features at the same time. An image showing the architecture is displayed in Figure 3.

3.4. UNet Architectures

After exploring various resources online, we encountered previous Kaggle competitors using a series of UNet architectures. The high level idea of the UNet architectures is to incorporate higher level features pulled from the filters in the earlier convolutional and pool layers and concatenate them with the features pulled from later convolutional and pool layers in order to perform cleaner image segmentation, which is what this type of architecture has been known for. We felt that this model would allow us to better extract features that required this type of segmentation which include the habitation label and agriculture label. However, a downside of this image segmentation is that it would result in us losing accuracy on atmospheric labels. An image showing the architecture is displayed in Figure 4.

3.5. Pre-trained Models

We attempted to use some pre-trained models from Keras which included ResNet-50 and VGGNet19 that were

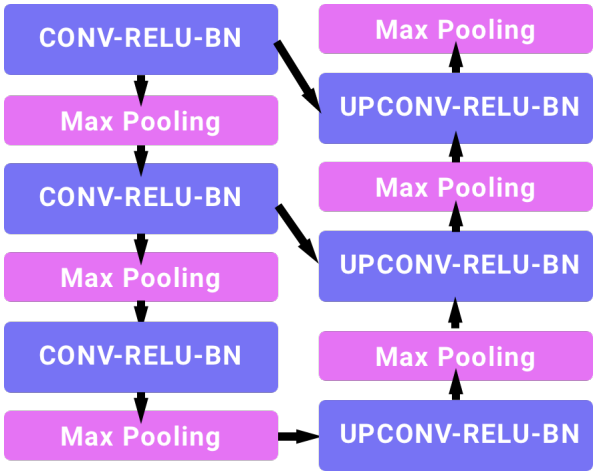


Figure 4: Generic architecture of UNet

pre-trained on ImageNet. Through this method of transfer learning, we trained the final layers with a sigmoid layer and binary cross-entropy loss function and then perform fine tuning on the earlier layers of the model.

3.6. Splitting Classification

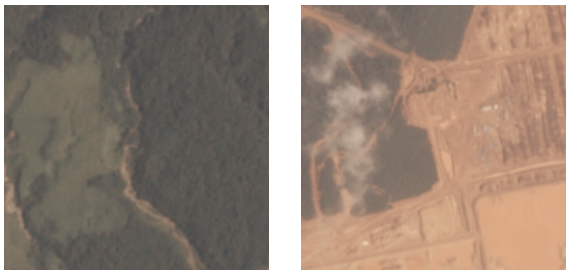
We noticed that in the training dataset, there was never more than one weather label: (clear, cloudy, haze, partly_cloudy).

We then decided to split the classification of examples into two parts, one for predicting the weather label and the second for predicting the other labels given the weather label. So we had a total of 4 models: a weather classifier, a classifier for clear examples, haze examples, and partly_cloudy examples (cloudy has no other labels).

At training we split the dataset based on true weather class, and at test time, we first determined a predicted weather class and based on that chose which classifier to use to classify the other labels.

4. Dataset and Features

Our dataset consists of 40,479 training images and 61,191 test images. Our training set consists of the first 32,000 training images and our validation set consists of the remaining training images. Some of the images in the class are shown in Figure 1, with the acknowledgement that some of the images are mislabeled as a result of noise. The class label distribution of the data set is shown in Figure 2 which shows that most of the common land labels have thousands of images that correspond to them and some of the rare labels have only a few hundred. This could potentially pose an issue with sparse data not providing enough information to classify these rarer land labels. However, since the occurrence of these rare land labels were relatively small, we did



(a) agriculture, clear, primary water

(b) conventional mine, partly cloudy, primary

Figure 5: Various images from the training set that are accompanied by their respective labels

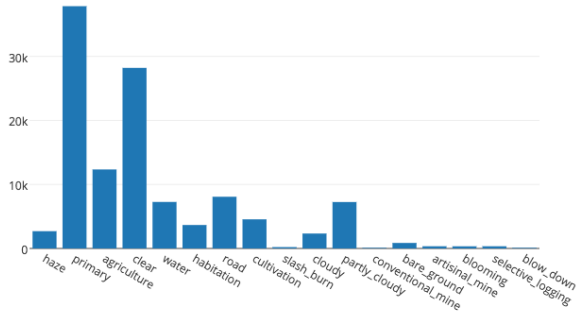


Figure 6: Distribution of labels from the training set. (Source: Kaggle)

not worry much about the misclassification of them since it would not hurt our model's results too much.

With our images, we normalized all images by computing the image mean of the training set and subtracting all images by the mean. The resolution of our images was 256 by 256 and we did not downscale the images to preserve as much of the original data as possible. No additional extracted features were added. Our data was downloaded from the Kaggle competition "Planet: Understanding the Amazon from Space" which is available here: `https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data`.

5. Experiments

Our evaluation for results on all models of this project was the F2 score which is given by:

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

Precision and recall are displayed as p and r respectively in this equation. We took the last 8,479 images of the training image dataset as our validation set. We found that evaluations on our validation set were extremely close to the final test values on the leaderboard and as a result we chose not to do cross-validation as our existing validation set was good enough and the test results shown were picked from the epoch with the highest validation score.

We also decided to stick with the Adam optimizer as brief experimentation with RMSProp and previous experience with neural networks in 224N showed that the Adam optimizer was more optimal. Furthermore, no data augmentations were explored as we wanted to test model architectures.

5.1. Early Exploration

We found that guessing the two most common labels on every image which were clear and primary generated a test value of 0.64640. Using one convolutional layer, pool layer and fully connected layer, we managed to improve the accuracy of our model to 0.73433.

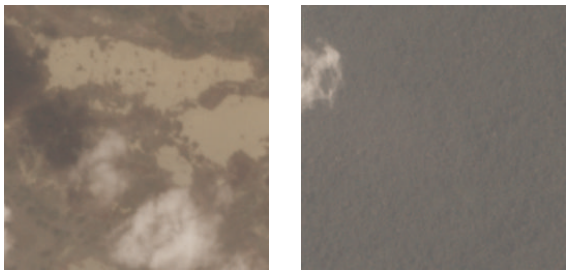
5.2. Baseline Model Results

To iterate quickly, we established a baseline model that was fast and easy to train on our dataset. A consequence of creating a fast and easily trainable model was that we could explore some basic augmentations to the model to see how this affected our F2 score. Below are some of our quantitative results from experimentations. We decided to show only the test results as most of the validation F2 scores were similar enough to the test scores that it would be redundant. All models except the last one were trained with the traditional sigmoid cross entropy loss.

Not shown in the quantitative results are our explorations for learning rates. We experimented with learning rates from 10^{-3} to 10^{-5} for the Adam optimizer and settled on a final learning rate of $5 * 10^{-5}$ with an exponential decay of 0.96 every 10,000 steps with staircase True. All models were trained for 10 epochs.

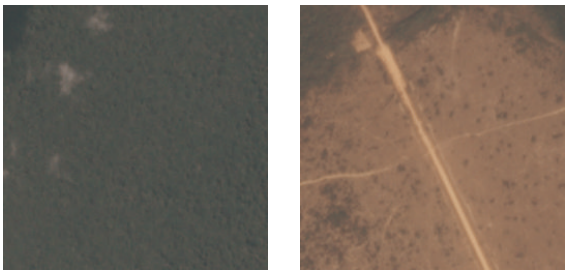
Model Description	Test F2 Score
Baseline with 0.1 dropout	0.79475
Baseline with 0.3 dropout	0.80542
Baseline w/ 0.3 dropout and data normalization	0.81339
Baseline w/ 0.2 d/o, data norm and batch norm	0.81146
Baseline w/ 0.2 d/o, data norm, batch norm, and weighted loss of 10	0.84221

In Figure 5a, our baseline classified the image as agriculture, clear, cultivation, habitation, primary, and road while the truth was agriculture, clear, primary, road.



(a) (b)

Figure 7: Example classification from our baseline model with the best validation score



(a) (b)

Figure 8: Example classification from our VGG16 model with the best validation score

In Figure 5b, our baseline classified the image as clear and primary while the truth was blooming, clear, primary.

5.2.1 Analysis of Baseline Results

Our attempts to improve the baseline results involved adding larger levels of dropout to prevent overfitting of the model. Higher dropout showed a larger increase in F2 score, but beyond 0.3 we found that this negatively affected the model. We wanted to show the impact that data normalization and batch normalization did and did not have. We see that by subtracting the mean image, we were better able to classify images but as a result it seemed as if batch normalization did little to help our model improve. Lastly, using knowledge that the F2 score favors recall, we used the weighted loss function and found that it significantly improved our F2 score.

In a brief exploration of our baseline results, we saw that our model sometimes labeled an image with multiple atmospheric labels even though it was clear that the image only had one. However, noise was a large issue in correctly labeling images. As seen in our photos in Figure 5, many times our baseline over labeled images which we believe is a result of the neural network thinking that it is better to guess labels than to miss them (a consequence of the F2 score favoring false positives over false negatives). In most examples similar to Figure 5b, we see that our model accurately classifies the most basic of images which were primary and clear images. However, the truth labels reveal that it is blooming, which is a rare label. We suspect that the network failed to classify this image completely correctly because of the low amounts of training images that had blooming or by noise in the labeling of these images.

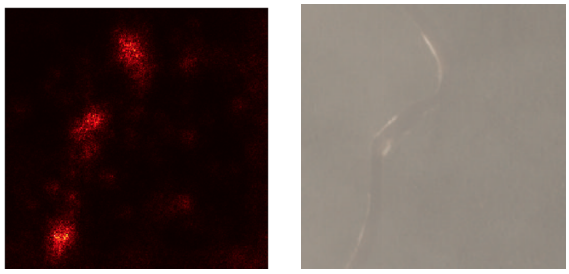
5.3. VGGNet16 Results

This model was not pre-trained on ImageNet and was trained from scratch. As similar to the results above, we only show the test F2 score evaluations. We also experimented with learning rates from 10^{-3} to 10^{-5} for the Adam Optimizer and settled on a final learning rest of $3{*}10^{-5}$ with an exponential decay of 0.96 every 10,000 steps. All models except the last, which was trained for 20 epochs, were trained for 10 epochs. If not stated, the default loss operator was the sigmoid cross entropy loss function. Lastly, all of these models were run with the pre-processing.

Model Description	Test F2 Score
VGG w/ no dropout	0.85963
VGG w/ two 0.2 dropout layers and batch norm	0.87346
VGG w/ two 0.2 dropout layers	0.87462
VGG w/ three 0.3 dropout layers	0.87469
VGG w/ two 0.1 dropout layers	0.88145
VGG w/ two 0.2 dropout layers, batch norm and weighted loss of 20	0.88974
VGG w/ two 0.2 dropout layers, batch norm and weighted loss of 10	0.89365
VGG w/ two 0.2 dropout layers, batch norm and weighted loss of 4	0.89966
VGG w/ two 0.25 dropout layers, batch norm and weighted loss of 5	0.90321

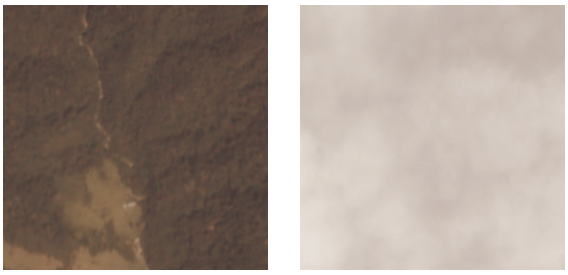
In Figure 6a, our baseline classified the image as agriculture, cultivation, partly cloudy, primary, and road while the truth was agriculture, cultivation, habitation, partly cloudy, primary, and road.

In Figure 6b, our baseline classified the image as agriculture, artisinal mine, bare ground, clear, primary, road, and water while the truth was clear, primary, and water.



(a) (b)

Figure 9: Saliency map for water



(a) (b)

Figure 10: Error in labeling

5.4. Saliency Map of VGGNet16 on Random Example

In this randomly selected example, the true image is an image of a river, but quite hazy, and the heat map is especially strong at the bends of the river.

5.4.1 Analysis of VGGNet16 Results

Similar to our baseline model, we implemented many of the same strategies that worked in our baseline model on our VGG model. We found that adding dropout significantly reduced overfitting on our image, increasing our F2 score by 0.14. More layers of dropout and higher dropout values we found to be around the 0.88 F2 score with little movement. Furthermore, we found that as a result of the pre-processing, it seemed as if batch normalization continued to not benefit the model in any meaningful way. Experiments showed that adding the weighted loss operation and optimizing that hyperparameter lead to a significant increase in our accuracy and tuning these hyperparameters ultimately led us to a 0.90321 F2 score.

Our qualitative results show a slightly different accuracy compared to our baseline. Although we continue to misclassify rare land labels, we can see that our model in the first image is less prone to guessing. However, in the second image, it seems that the truth labels were completely off as the image shows that the road exists but there is no existing water labels.

5.4.2 Attempts at post processing

We made some attempts at post processing results which took votes from the atmospheric classifier described in the split classifier section and eliminated duplicate atmospheric labels. However, this often resulted in a worse score. An analysis showed that the two images shown below one was classified as cloudy and the other partly cloudy even though they were siilar.

5.5. UNet Results

We tried three variations of UNet architectures which essentially are similar to the architectures shown in methods but differed slightly. Our first UNet, dubbed Baby UNet, had half the layers shown in Figure 4 with only one concatenation. Our second UNet, dubbed Big UNet, resembled the model shown in Figure 4. Our third UNet, dubbed Thick UNet, had more filters in the middle layers than the other UNet architectures. This was our attempt at exploring some new architectures and experiment with non-standard models in order to obtain better information.

All experiments were ran with 10^{-5} learning rate unless specified for the Adam Optimizer and trained for ten epochs.

Model Description	Test F2 Score
Baby UNet with weighted loss op of 10	0.85963
Big UNet with weighted loss op of 10	0.85734
Big UNet with weighted loss op of 10, two FC layers with 0.1 dropout, and 10^{-4} lr	0.87820
Thick UNet with weighted loss op of 10, two FC layers with 0.1 dropout, and 10^{-4} lr	0.88161

The UNet results were slightly worse than the other models, but we see an increase in test F2 as the model became larger and more complex.

5.6. Wide ResNet Results

Because the models for Wide ResNet took nearly half a day to an entire day to train, we could not iterate through the hyperparameters as quickly and only experimented with 10^{-4} and 10^{-5}, ultimately choosing our learning rate to be 10^{-4}. We used a weighted loss op with a weight of 5, which

was chosen based on the VGG results. Each model was trained for 15 epochs.

Model Description	Test F2 Score
Wide ResNet 16x2	0.895
Wide ResNet 16x4	0.89272

Small experiments were carried out to add dropout to the model, but are not shown because within the first five epochs, the model struggled to converge and the experiments were stopped.

5.7. Split Classifier Results

We used VGG16 with a softmax cross entropy loss to classify weather labels and used baseline models for each secondary classifier. We found that the secondary classifiers were able to achieve above .86 on validation on the split datasets (split by weather label), which is a bit better than the baseline model on the entire dataset, and the weather label classifier had around 92.3 percent accuracy on predicting the correct weather label. Unfortunately, when combining the classifiers back together on test data, we only got .732 F2 score on the test set.

We realized that perhaps the reason this approach failed was that if the weather label was wrong, then the other labels were most certainly wrong, so the errors of these 2 models in a sense multiply together. Because of this, we instead tried treating the weather classifier as more of a post-processing step.

5.8. Pretrained Results

Using transfer learning on Keras was a larger challenge than we had anticipated, and as a result we believe that the F2 scores that we obtained from the pre-trained models are a result of inappropriate setup. Results from others on Kaggle showed that these models could potentially reach F2 scores of 0.92 or higher, but our models barely reached validation scores 0.85 on VGG and 0.83 on ResNet-50.

6. Conclusion/Future Work

Training deep convolutional networks from scratch was the approach that had the most success on this problem, breaking .90 test F2. VGG16 performed slightly better than the wide residual network, and both significantly outperformed the baseline model. We also found that by weighting the loss function we were able to achieve a higher validation and test F2, an increase of slightly more than 0.01.

In the future, we want to experiment with an ensemble of classifiers. Additionally, we believe that brute forcing over possible thresholds for each class label instead of just .5 for all classes could potentially increase our F2 scores by around 0.01. With more computational resources, we also would like to train separate classifiers for each class in parallel. However, the labels in this problem seem to be quite correlated with one another, so it could possibly be better to explore label correlation techniques ([9], [10]). Finally, we want to look at several data augmentation techniques (rotating, translating images, etc.).

6.1. References

[1] Panchal, V., et. al (2009). Biogeography based Satellite Image Classification

[2] Melgani, F., Bruzzone L. (2004) Classification of Hyperspectral Remote Sensing Images With Support Vector Machines

[3] Yang, W., et. al (2015) Learning high-level features for satellite image classification with limited labeled samples

[4] Yu, H., et. al (2016) A Color-Texture-Structure Descriptor for High-Resolution Satellite Image Classification

[5] Li, Y., et. al (2016) Unsupervised Multilayer Feature Learning for Satellite Image Scene Classification

[6] Maggiori, E. (2017) Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification

[7] Zagoruyko, Sergey, and Nikos Komodakis. "Wide Residual Networks." [1605.07146] Wide Residual Networks. N.p., 17 Jan. 2017. Web. 13 June 2017.

[8] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation." SpringerLink. Springer, Cham, 05 Oct. 2015. Web. 13 June 2017.

[9] Bi, W., and Kwok, J. T. 2014. "Multilabel classification with label correlations and missing labels." In AAAI.

[10] Garg, Amit, Jonathan Noyola, Romil Verma, Ashutosh Saxena, and Aditya Jami. "Exploring Correlation between Labels to Improve Multi-Label Classification." [1511.07953v1] Exploring Correlation between Labels to Improve Multi-Label Classification. N.p., 25 Nov. 2015.

[11] Xie, Michael, Neal Jean, Marshall Burke, David Lobell, and Stefano Ermon. "Transfer Learning from Deep Features for Remote Sensing and Poverty Mapping." ACM Digital Library. AAAI Press, 27 Feb. 2016.

[12] Basu, Saikat. "A Learning Framework for Satellite Imagery." DeepSat. ACM, 3 Nov. 2016. Web. 13 June 2017.

[13] Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. CoRR, abs/1411.4038, 2014.

[14] M. Espnola, J.A. Piedra-Fernndez, R. Ayala, L. Iribarne, J.Z. Wang, Contextual and hierarchical classification of satellite images based on cellular automata, IEEE Trans. Geosci. Remote Sens. 53 (2) (2015) 795809.

[15] Yanfei Zhong, Feng Fei, and Liangpei Zhang. Large patch convolutional neural networks for the scene classification of high spatial resolution imagery. Journal of Applied Remote Sensing, 10(2):025006025006, 2016.