Multi-label Classification of Satellite Images with Deep Learning

Daniel Gardner
Stanford University
dangard@stanford.edu

David Nichols
Stanford University
davdnich@stanford.edu

Abstract

Up-to-date location information of human activity is vitally important to scientists and governments working to preserve the Amazon rainforest. We implement a Convolutional Neural Network (CNN) model to perform multi-label classification of Amazon satellite images. Our model identifies the weather conditions and natural terrain features in the images as well as man-made developments such as roads, farming, and logging. We begin by implementing a simple CNN model that achieves a 0.84 F-score. We then experiment with three deep CNN architectures that have had recent success in the ImageNet Challenge and show that a ResNet-50 model can achieve a 0.91 F-score. Our model's best performance is achieved via a number of data augmentation and ensemble techniques. Our model is particularly effective at identifying illegal mining operations. These models will enable stakeholders to pinpoint where deforestation and associated illegal activity is taking place and craft targeted policy to limit its deleterious effects.

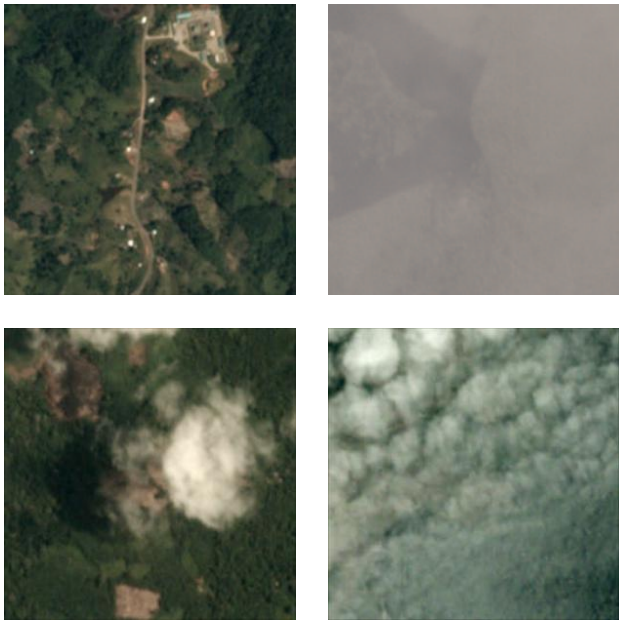


Figure 1: *Clockwise from top left: 1) clear primary agriculture habitation road; 2) haze primary agriculture water; 3) cloudy; 4) partly-cloudy primary cultivation slash-burn*

1. Introduction

Deforestation of the Amazon basin has occurred at a rapid pace over the past four decades. Governments and scientists, concerned about consequences ranging from habitat loss to climate change, need a way to monitor where and when the deforestation is occurring.

Planet, a satellite imaging company, recently released a dataset of more than 100,000 images from the Amazon basin and sponsored a Kaggle competition involving labeling the atmosphere and ground features in the images [1]. Each image is 256 x 256 pixels and has RGB and near-infrared channels. Notably, these images have at least ten times greater resolution than any earth image data used previously in tracking deforestation, with each pixel representing only three to five meters. In the past, researchers have relied on satellite images generated by Landsat, which is jointly managed by NASA and the US Geological Survey. Landsat missions often have multi-year gaps, making the data only suitable for examining overarching trends of general deforestation. The high-resolution Planet images enable identification of specific causes of deforestation and differentiation of legal and illegal human developments.

We build a model using Convolutional Neural Networks (CNN) to analyze each image and classify it with one or more of the 17 feature labels. Each image has one of four atmosphere labels and zero or more of 13 ground labels. By definition, cloudy images have zero ground labels, as none should be visible. Some ground features are human-related (habitation, slash burn, mining) while others are natural (water, blooming, blow down). More than 90 percent of images are labeled "primary", meaning that they have forest in them. Six of the other ground labels appear in fewer than one percent of images, but are often the ones we are particularly interested in identifying.

Approximately 40,000 images have labels, which we divide into our training and validation sets. The other 60,000 images are used as a test set for creating a submission to the Kaggle competition.

We develop a multi-label classification CNN framework to work on this problem. A naive approach might be to train 17 different single-label networks that predict the presence of a particular atmosphere or ground feature. However, multi-label classification has been shown to robustly account for the correlative relationships that exist between multiple features. In the Planet data, this is particularly useful, as human developments almost never occur independently of one another. For example, almost all man-made structures, no matter how remote, will have a road nearby.

We build our multi-label classifier on top of CNN architectures that have performed well in the single-label ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). Namely, we experiment with VGG-16, Inception-v3, and ResNet-50. Our model takes an input image, computes a score for each of the 17 features, and then uses a cutoff threshold to decide which labels to keep.

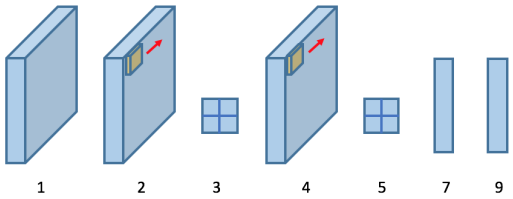
2. Background/Related Work

The field of Remote Sensing, which broadly involves collection of earth data via satellites, has long been interested in image classification. The recent explosion in the availability of massive amount of earth data has led to breakthroughs in machine learning approaches to analysis. Santos et al [2] proposes a Bag of Words algorithm that annotates images using correlation representations. Chen et al [3] improves on this result by adding spatial information into the BoW model.

Barsi et al [4] utilized neural networks to examine satellite data in 2003, and the introduction of CNNs sparked a renaissance of training to perform remote sensing tasks. Hung et al [5] used CNNs to identify weeds from a UAV camera, while Hu et al [6] used CNNs to perform remote sensing scene classification, and Chen et al [7] used CNNs to identify vehicles in satellite images.

These papers generally built task-specific models to achieve their best performance, but in 2014 Razavian et al [8] demonstrated that they could use the OverFeat [9] model trained for the ImageNet challenge to generalize onto other image tasks, including human feature recognition and bird classification. Penatti et al [10] used pretrained models from ImageNet on the UC Merced aerial image dataset and found that CNNs outperformed conventional object recognition methods but performed worse on remote sensing color recognition tasks.

More recently, Castelluccio et al [11] used pretrained the networks CaffeNet [12] and GoogLeNet [13] to classify remote sensing images for land use policy. To reduce design time, they used the pretrained weights and built a small refined network on top. In 2016 Karalas et al [14] used deep CNNs for multi-label classification of satellite images with great success.



Layer	Dimensions
1) Input Image	256x256x4
2) Convolutional Layer – 32 3x3 filters, stride = 1, ReLU activation	254x254x32
3) Max Pooling Layer – 2x2	127x127x32
4) Convolutional Layer – 64 4x4 filters, stride = 1, ReLU activation	124x124x64
5) Max Pooling Layer – 2x2	62x62x64
6) Dropout – 0.25	62x62x64
7) FC Layer – 128, ReLU activation	1x128
8) Dropout – 0.5	1x128
9) FC Layer – 17	1x17

Figure 2: Baseline Model structure

Multi-label classification has been an important problem in image recognition for many years. Early work from Barnard and Forsyth [15] focused on identifying objects in particular sub-sections of an image. More recently, Wei et al [16] demonstrated that weights from networks pre-trained on single-label classification have great success transferring to multi-label classification. Wang et al [17] proposes using recurrent neural networks to capture semantic label dependency and improve multi-label classification.

3. Methods

We used Google's TensorFlow and the open-source neural network Python package Keras for the majority of our model development. We used R and Excel for data visualization and threshold optimization.

Our design steps were as follows: First we developed a basic CNN architecture, then we built three deep CNN models with VGG-16 [18], Inception-v3 [19], and ResNet-50 [20], and finally we added on data augmentation and ensemble techniques to the deep CNN models. Each model involves feeding in batches of images and getting scores for the 17 features out. We use a sigmoid activation function on the final layer of all of our networks so that each feature has a score between 0 and 1. We label an image with features that have scores above a certain threshold value (naive and later optimized).

We experimented with softmax as well but found that it pushed the rare ground labels to near-zero probability.

3.1. Baseline CNN architecture

Our first implementation was a mesh of data exploration code [21] provided by the Planet Kaggle representative as well as a simple Keras starter notebook [22] created by an-

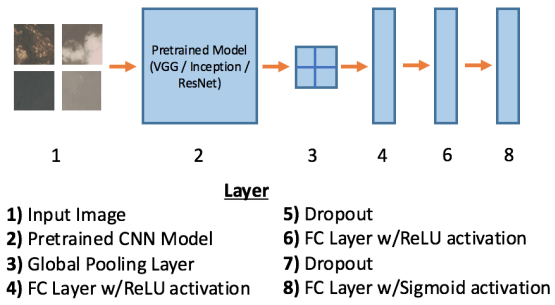


Figure 3: Deep Model Structure

other user in the competition. It uses two convolutional layers with ReLu activation and max pooling followed by two dense layers. No preprocessing or resizing of the images is required, so input images had RGB and Near-IR channels.

3.2. Deep CNN architecture

We proceeded to create a model incorporating deep networks that performed well on the ImageNet Challenge. For each named deep network, we attached a pooling layer and three dense layers to the end. Keras is invaluable for integrating the deep CNN network with our dense layers, as it allows you to import the network structure and download the pre-trained weights of each network. We then explicitly freeze the weights in the imported network and train just the weights in our added-on layers. The pre-trained models require three input channels, so we had to strip off the Near-IR information.

We built models with the following networks:

3.2.1 VGG-16

VGG was originally developed for the ImageNet dataset by the Visual Geometry Group at the University of Oxford. The highlights of this model are that it utilizes 16 layers and 3 x 3 filters in the convolutional layers. It is designed to take in 224 x 224 images as input.

3.2.2 Inception-v3

Inception-v3 is another ImageNet-optimized model. It is developed by Google and has a strong emphasis on making scaling to deep networks computationally efficient. It takes in 299 x 299 images.

3.2.3 ResNet-50

ResNet-50 is a model developed by Microsoft Research using a structure that uses residual functions to help add considerable stability to deep networks. ResNet won the ImageNet Challenge in 2015, and ResNet-50 is the 50-layer version of ResNet. It uses 224 x 224 images.

3.3. Implementation

We used the binary cross-entropy loss function and the Adam optimizer. We wrote a custom metric to track the F score during training, which was required since the standard scipy F score function cannot be used on tensors. During training, we used Keras callbacks to save model weights when the model's performance on tracked metrics improved. However, this functionality is limited to predefined metrics, so we used accuracy to determine when the model improved.

The entire 40,000 labeled images could not be loaded all at once, even with 120 GB of memory on our Google Cloud instance. To remedy this, we used Keras' model.fit_generator method, which loads batches of images from a generator and fits the model. The ImageDataGenerator class allows for easy image resizing to fit the current model. However, Keras' image generators do not support multi-label classification, as they currently label images based on the directory they are in (one label for everything in the directory). Instead we used a Keras library modification we found on Stack Overflow that passes a label dictionary into the generator with a length-17 binary array value for each image, representing its feature labeling [23]. With some tweaking, we were able to get this to work for our problem, which paid off later on when we wanted to do quick data augmentation.

This code architecture also made it difficult to perform k-folds cross validation during the training process. We had to separate our training and validation data (32000-8000 split) and put them in separate folders. Since the data from Planet was already randomized, we just took the top-numbered 8000 files to get our validation set.

4. Experiments

We ran our baseline model and the three deep network models on vanilla data. We then ran our best performing model on augmented data and ran a small ensemble model.

4.1. Evaluation

The evaluation metric for each model is average F2 score, which is defined for one sample as:

$$F_2 = (1 + 2^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(2^2 \cdot \text{precision}) + \text{recall}}$$

where Precision = $\frac{TP}{TP+FP}$ and Recall = $\frac{TP}{TP+FN}$.

The F2 score prefers recall to precision, as can be seen in Figure 4. This means we punish false negatives more

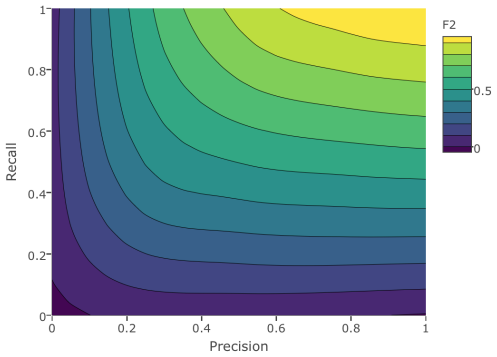


Figure 4: Contour plot of F2 score

severely than false positives. This makes sense for a problem where we are trying to detect some rare phenomena: we would prefer to identify all occurrences even if we end up making some false positive mistakes.

4.2. Baseline Model

We ran our basic model for 20 epochs and achieved an F2 score of 0.84. The top of the Kaggle leaderboard is around 0.93 F2, so this needs improvement.

4.3. Deep Models

VGG-16 was not able to break 0.90 F2 in the first 40 epochs I ran, while both Inception-v3 and ResNet-50 were. ResNet-50 was the best, achieving an F2 of 0.907 after 50-60 epochs. Full results in Table 1.

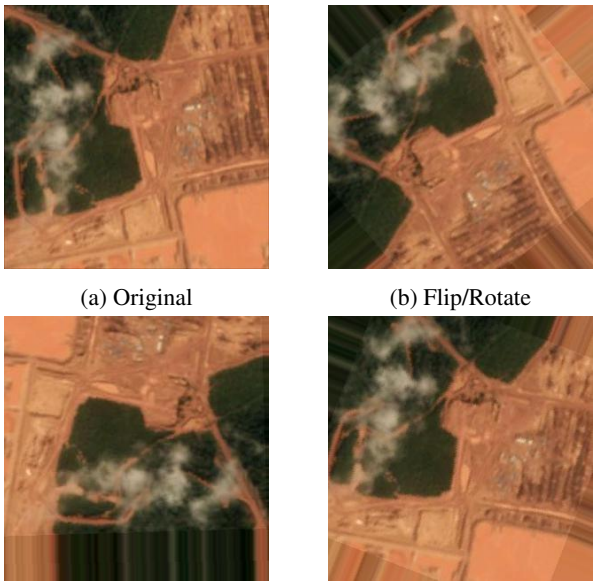
4.4. Data Augmentation and Ensembles

Our models' 0.03 performance gap as compared to the Kaggle leaders is mostly attributable to poor performance on the rarer ground features. To ameliorate this, we attempted various data augmentation techniques in the hopes of providing more rare-feature training examples. Keras' image generator allows for on-the-fly image augmentation. We specified that images could be flipped, rotated, and shifted, thereby creating additional labeled training images.

We trained three ResNet-50 models with augmented data for 40 epochs each. We then made predictions on the validation data using a voting ensemble method. This improved on our best ResNet F2 score by 0.001.

4.5. Tuning

As part of the tuning process, experimentation with classification thresholds was one of the main areas we looked at. We started with an (naive) initial setting of 0.5 across classes to get baseline model performance. After this, we performed an exhaustive search to find the best uniform threshold to apply to all classes simultaneously.



(a) Original (b) Flip/Rotate

(c) Shift (d) Rotate

Figure 5: Data Augmentation

Since the F2 of each class is independent, we were then able to run a threshold search to find the best threshold for each of the 17 classes at a cost of 17×N (as opposed to N^{17} if they had been dependent (N here is the size of the search space, in this case we used 100 increments to search every 0.01 increment).

Figure 6 shows the improvement in F2 obtained on the ResNet model by transitioning from a uniform threshold of 0.5 to a uniform threshold of 0.2, and then further to independently tuned individual class thresholds. The updated uniform threshold gives about a 0.04 boost to F2, while the class-specific thresholds add an additional 0.001 improvement to F2.

Model	Train F2	Val F2
Baseline	0.875	0.836
VGG-16	0.903	0.897
Inception-v3	0.912	0.901
ResNet-50	0.921	0.907
ResNet-50 (data aug/ensemble)	**0.922**	**0.908**

Table 1: Model classification performance

5. Discussion

Our best ResNet-50 model achieved a similar 0.907 F2 score on the Kaggle submission test set, placing us in the top third of the 450+ submissions.

Figure 6 details the F2 score, Precision, and Recall for

	clear	cloudy	partly_cloudy	haze	primary	agriculture	road	water	cultivation	habitation	bare_ground	artisinal_mine	selective_logging	blooming	slash_burn	blow_down	conventional_mine	TOTAL
Threshold	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	
F2	0.953	0.730	0.756	0.682	0.986	0.757	0.660	0.526	0.367	0.392	0.000	0.014	0.000	0.000	0.000	0.000	0.000	**0.836**
Precision	0.882	0.546	0.627	0.399	0.947	0.633	0.546	0.342	0.310	0.415	1.000	1.000	1.000	0.000	1.000	1.000	1.000	
Recall	0.972	0.797	0.798	0.829	0.996	0.796	0.697	0.608	0.385	0.387	0.000	0.011	0.000	0.000	0.000	0.000	0.000	
	Baseline Model with Uniform Threshold																	
Threshold	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	
F2	0.970	0.787	0.813	0.581	0.984	0.805	0.699	0.615	0.275	0.355	0.083	0.582	0.035	0.000	0.000	0.000	0.135	**0.865**
Precision	0.925	0.955	0.795	0.822	0.977	0.865	0.875	0.883	0.763	0.892	0.667	0.907	0.400	1.000	1.000	1.000	1.000	
Recall	0.982	0.754	0.818	0.541	0.986	0.791	0.666	0.572	0.237	0.308	0.068	0.534	0.028	0.000	0.000	0.000	0.111	
	ResNet Model with Naïve Uniform Threshold																	
Threshold	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	**0.2**	
F2	0.972	0.853	0.863	0.749	0.990	0.885	0.820	0.769	0.631	0.670	0.259	0.764	0.235	0.068	0.000	0.000	0.471	**0.907**
Precision	0.890	0.871	0.629	0.583	0.968	0.691	0.629	0.631	0.387	0.513	0.457	0.809	0.429	0.286	1.000	1.000	0.615	
Recall	0.995	0.848	0.951	0.806	0.996	0.951	0.888	0.813	0.748	0.725	0.234	0.753	0.211	0.057	0.000	0.000	0.444	
	ResNet Model with Optimal Uniform Threshold																	
Threshold	**0.32**	**0.11**	**0.24**	**0.17**	**0.26**	**0.23**	**0.2**	**0.21**	**0.21**	**0.18**	**0.15**	**0.23**	**0.14**	**0.16**	**0.21**	**0.09**	**0.15**	
F2	0.973	0.875	0.863	0.747	0.990	0.883	0.820	0.765	0.626	0.683	0.356	0.770	0.310	0.111	0.000	0.000	0.479	**0.908**
Precision	0.905	0.754	0.644	0.535	0.969	0.708	0.629	0.643	0.405	0.470	0.356	0.846	0.310	0.200	1.000	1.000	0.409	
Recall	0.991	0.912	0.943	0.829	0.995	0.942	0.888	0.803	0.724	0.770	0.356	0.753	0.310	0.100	0.000	0.000	0.500	
	ResNet Model with Optimal Variable Threshold																	
Frequency	0.701	0.185	0.048	0.066	0.931	0.306	0.198	0.179	0.112	0.089	0.024	0.009	0.008	0.008	0.006	0.002	0.002	
	Atmoshpere				**Ground**													

Figure 6: Performance of ResNet model at different threshold levels

each feature, sorted by frequency. Comparing the results from the baseline model to the best ResNet model shows an across-the-board improvement.

The baseline model is quite good at predicting primary and clear labels, which makes sense since the data has the most training examples of these types. The baseline performs moderately well on most of the uncommon atmosphere and ground features, but gets 0.0 F2 for nearly all of the rare features. This is the key deficiency of the baseline model - by failing to identify the rare labels, it is unable to pinpoint rare but critical human encroachment that we are most interested in. A multi-label classifier attempting to curb deforestation should be able to identify mining, logging, and slash-and-burn activity to be truly effective.

The ResNet model improves in most of these critical areas. It gets somewhat better (0.310 F2) at identifying selective logging, which is where expensive wood is removed rather than general clear cutting. It makes similar improvements (0.479 F2) to conventional mining, and dramatic improvements to its identification of artisanal mining (0.770 F2). The model does not show improvements at finding slash-and-burn.

6. Conclusion

Using deep CNN models designed for the ImageNet Challenge combined with task-specific refining layers produces good results on multi-label classification of satellite images. Experimenting with more exotic refining layers would be an interesting improvement. One idea might be to train new ResNet models, each time focusing on one of the rare ground features. Just as our model performed especially well on artisanal mining, it should be possible to achieve similar results with other rare features with proper data augmentation and loss weighting. We could then combine the models into an ensemble where if any of them pre-

Figure 7: Identifying Illegal Mining

dict a rare feature we would accept that.

It is unlikely that the ResNet model can make dramatic improvement from this point without a re-thinking of its optimization metric. The accuracy metric used to pick the best weights simply compares 1s and 0s and determines how many are correct. Since it uses a simple rounding to label any given feature, all or most images will never have rare features for the accuracy calculation. As such, our optimizer has a difficult time finding weights to make the accuracy go over 0.96. We can use our variable thresholds to account for this when we make our predictions, but if the model hits a ceiling on optimization improvement we will not be able to identify the remaining four percent of accuracy. Some digging into Keras callback saving could remedy this.

The ResNet's ability to identify artisanal mines makes it extremely useful to solving the overarching problem of illegal human activity in the Amazon rainforest. Conventional mining operations are defined as those run by a mining company and, while certainly destructive, are generally sanctioned by the local or federal government in which they operate. Artisanal mines are small scale and, by-in-large, illegal. Because there is no knowledge or oversight about where they are occurring, there is little to no ability for the local government to manage their impact on deforestation in the area. Having a model that can find these mines is a definite win for those concerned about the destruction of the rainforest habitat.

References

[1] Kaggle competition. planet: Understanding the amazon from space. [Online]. Available: https://www.kaggle.com/c/planet-understanding-the-amazon-from-space

[2] J. A. dos Santos, O. A. Penatti, P.-H. Gosselin, A. X. Falcaão, S. Philipp-Foliguet, and R. d. S. Torres, "Efficient and effective hierarchical feature propagation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 12, pp. 4632–4643, 2014.

[3] S. Chen and Y. Tian, "Pyramid of spatial relatons for scene-level land use classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 4, pp. 1947–1957, 2015.

[4] A. Barsi and C. Heipke, "Artificial neural networks for the detection of road junctions in aerial images," *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, vol. 34, no. 3/W8, pp. 113–118, 2003.

[5] C. Hung, Z. Xu, and S. Sukkarieh, "Feature learning based approach for weed classification using high resolution aerial images from a digital camera mounted on a uav," *Remote Sensing*, vol. 6, no. 12, pp. 12 037–12 054, 2014.

[6] F. Hu, G.-S. Xia, J. Hu, and L. Zhang, "Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery," *Remote Sensing*, vol. 7, no. 11, pp. 14 680–14 707, 2015.

[7] 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.

[8] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 806–813.

[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

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

[11] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," *CoRR*, vol. abs/1508.00092, 2015. [Online]. Available: http://arxiv.org/abs/1508.00092

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe:

Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[14] K. Karalas, G. Tsagkatakis, M. Zervakis, and P. Tsakalides, "Land classification using remotely sensed data: Going multilabel," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 6, pp. 3548–3563, June 2016.

[15] P. Duygulu, K. Barnard, J. F. de Freitas, and D. A. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *European conference on computer vision*. Springer, 2002, pp. 97–112.

[16] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan, "Cnn: Single-label to multi-label," *arXiv preprint arXiv:1406.5726*, 2014.

[17] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2285–2294.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

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

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[21] Kaggle competition: Getting started. [Online]. Available: https://www.kaggle.com/robinkraft/getting-started-with-the-data-now-with-docs

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

[23] M. Ganssauge. Keras git hub repository. [Online]. Available: https://github.com/tholor/keras/commit/29ceafca3c4792cb480829c5768510e4bdb489c5