Labeling Satellite Imagery with Atmospheric Conditions and Land Cover

Shakti Sinha
Stanford University
Stanford, CA
shakti@cs.stanford.edu

Abstract

In this project, we take on the Kaggle challenge "Planet: Understanding the Amazon from Space". Our goal is to accurately label satellite images with atmospheric conditions, land use and land cover. We start with the background of the challenge and a brief overview of how we have modeled the problem. We then examine related work that is relevant to this project. In the methods section, we describe the main approach we use for getting to results. A description of the dataset we use comes next, followed by a description of our experiments. Finally, we close with conclusions and areas of future investigation.

1. Introduction

1.1. Background

Deforestation in the Amazon basin contributes to reduced biodiversity, habitat loss, climate change, and other devastating effects. Combating deforestation requires detection and understanding of markers of human activity over large regions of Earth. Utilizing automated analysis of satellite images to detect these markers can enable faster and more effective responses to activity that indicates or precedes deforestation.

The Kaggle competition "Planet: Understanding the Amazon from Space" challenges participants to develop machine learning algorithms that can accurately label satellite images. The problem is interesting for a few reasons - first is that a successful solution will have significant real-world impact. Second, the nature of images we work with in this project are fairly different from Imagenet, which is where most pre-trained models appear to operate. This gives us the opportunity to see if techniques developed for Imagenet can transfer to a different type of problem. Finally, this is a multi-label classification problem, which brings up some interesting challenges around label correlations.

1.2. Problem setup

The input to our algorithm is a satellite image of the Amazon basin. The image is represented as a 256x256 grid of pixels and 3 or 4 channels, described further in the data section. We then use a Convolutional Neural Network to output one or more predicted labels, belonging to a set of 17 possible labels, describing atmospheric conditions, land cover and land use in the input image. The primary performance metric is the average F2 score on the validation or test dataset.

2. Related work

Understanding satellite imagery has been an active area of research for applications ranging from mapping, to separating bad data from good data, to understanding social impact of world events. Much of the work in mapping has focused on detection of roads in aerial images. For example, in [12], the authors use neural networks trained on large amounts of data to detect roads. Their work demonstrates the utility of incorporating unsupervised learning as well as spatial context in computer vision tasks. The authors build upon this work by making it more robust to noisy labels, and use deep neural networks with local connectivity in [13]. Another approach to detecting roads is used in [14], where a multi-step learning approach first identifies road centers, then iteratively builds more global structures.

A more general approach is taken in [9], where satellite images are mapped to feature vectors that are then used to power a nearest neighbor search. This allows efficient visual search for images that share semantic meaning. This work uses a model pre-trained on Imagenet, indicating the potential validity of transfer search in very different contexts. Pre-trained models are also used in [4], where a comparison is made to more traditional feature based approaches. Similarly, [6] discusses transfer learning from Imagenet, and compared approaches that use features from various levels of pre-trained models.

In [7], the authors use deep neural networks trained on nighttime and daytime images, coupled with survey data, to predict household income and poverty for subjects living in the photographed regions.

Multi-label classification can be seen as a related problem to image segmentation, where the classes of detected image segments become labels of the image. [3] uses image segmentation on satellite imagery to label pictures of urban areas. Another example of using per-pixel segmentation approaches can be found in [8].

A key aspect of the problem of labeling satellite imagery involves dealing with correlated labels. Multi-label classification has been extensively studied, and a few types of approaches dominate. A good overview can be obtained in [19]. [15] explicitly addresses constraints that can exist on labels in a multi-label setting. Maximizing the likelihood of observed labels using generative models is another interesting approach described in [11]. Multi-label classification can be interpreted as a ranking problem, as demonstrated in [5], where the authors combine a ranking objective with a convolutional neural network to beat prevalent benchmarks.

Recent work has demonstrated the possibility of using Recurrent Neural Networks coupled with Convolutional Neural Networks for multi-label image annotation, using approaches that can be compared to the task of image captioning. [17] is an excellent example of this approach, and appears to be very relevant to the subject problem in this report.

Finally, given the importance of the F2 measure for this project, incorporating strategies that optimize the measure are central to the problem. [10] discusses the relationship between optimal score thresholds and various thresholding strategies, while [18] finds that simple empirical determination of thresholds can be effective.

3. Methods

3.1. Architecture

We use the Inception V3 model trained on Imagenet as the base model in our setup, using code from [1] to get started. Inception V3 is described in [16]. The architecture described in the paper allows the network to scale substantially without a large increase in number of parameters or training time, and extensively uses four design principles. This made it a good choice for our application, as we wanted a network that can be trained quickly, and which benefits from recent advances in network architectures.

The Inception V3 model has 310 basic layers. We have trained layers 172 onwards on our data using a low learning rate.

The output of the Inception model is fed to a global average pooling layer. This is followed by a dense layer with 512 nodes, ReLU activation and L2 regularization. The final layer is a 17 unit dense layer with sigmoid activation, which outputs the predictions of our algorithm.

Class label	Threshold
agriculture	0.17999999999999999
artisinal mine	0.13
bare ground	0.19
blooming	0.17000000000000001
blow down	0.050000000000000003
clear	0.14999999999999999
cloudy	0.20999999999999999
conventional mine	0.089999999999999997
cultivation	0.22
habitation	0.19
haze	0.23000000000000001
partly cloudy	0.23000000000000001
primary	0.28999999999999998
road	0.17999999999999999
selective logging	0.10000000000000001
slash burn	0.10000000000000001
water	0.14999999999999999

Table 1: Class thresholds. We see large variability in thresholds that lead to optimal F2 scores.

The final labels for the image are derived using per-label thresholds. If a class score exceeds the threshold, we assign that class label to the image. The thresholds vary significantly by class, and can be seen in Table 1. The approach used to determine the class thresholds is described in the experiments section.

3.2. Loss

As this is a multi-label classification problem, the primary loss used during training is binary cross-entropy. This loss optimizes the performance for each class independently.

To help the network work with label correlations, we add another term to the loss function. This term measures the deviation from the expected label correlations based on the training data. The term is computed using the following steps.

We first compute the predicted co-occurrence for the label scores, C_y:

$$C_y = y_p^T \times y_p$$

where y_p is the vector with the predicted class scores.

The co-occurrence loss term L_c is then given by:

$$L_c = \sum C_y * C_o$$

where C_o is the observed label co-occurrence in the training data.

The overall loss L is a weighted sum of the binary cross-entropy loss and the co-occurrence loss:

Train loss	0.1621
Train accuracy	0.9438
Validation loss	0.1657
Validation accuracy	0.9421
F2 score	0.8645

Table 2: Results after training dense layers after 20 epochs, with Inception layers frozen.

$$L = L_{ce} + \alpha L_c$$

where α controls the relative importance of the individual loss terms, and is set to 0.0001 in our project.

Based on experiments, we use the co-occurrence matrix only for atmospheric condition labels, and set non-diagonal elements to 1.

3.3. Data preprocessing

The pre-trained Inception V3 model that we use requires a minimum image size of 139x139 pixels. To keep computation tractable, we use 140x140 images in our setup. To rescale the images, we use the resize() function in openCV. The resize function tries to capture information from surrounding pixels while scaling, which helps preserve information in the smaller image. The image pixels are then normalized using the normalize() function from openCV.

The data is then split into training and validation sets. We use 70% of the data for training, and 30% for validation. This gives us 28335 training samples and 12144 validation samples. To create the training and validation splits, we use the scipy function train_test_split().

3.4. Training

We follow a 2-step training process, as described below.

3.4.1 Training the dense layers

For this step, we freeze all layers of the pre-trained Inception V3 network, and train just the dense layers that we added on top. We use the Adam optimizer, and the loss function is modified binary cross-entropy. We train for 20 epochs, and use a batch size of 128 samples. At the end of this step, we get the results listed in Table 2.

3.4.2 Jointly training Inception and dense layers

After training the dense layers, we train a subset of the Inception V3 layers. We unfreeze layers 172 onwards, and train the network for 70 epochs, using an SGD optimizer with a learning rate of 0.01 and momentum 0.9. The loss function is again modified binary cross-entropy, with a weight of 0.0001 on the co-occurrence loss. This gives us the results listed in Table 3. We see a clear improvement in the F2 score. Note that the network has overfit the training data, and there is opportunity to improve performance using regularization.

Train loss	0.0057
Train accuracy	0.9997
Validation loss	0.2160
Validation accuracy	0.9529
F2 score	0.8894

Table 3: Results after joint training after 70 epochs, with some Inception layers unfrozen.

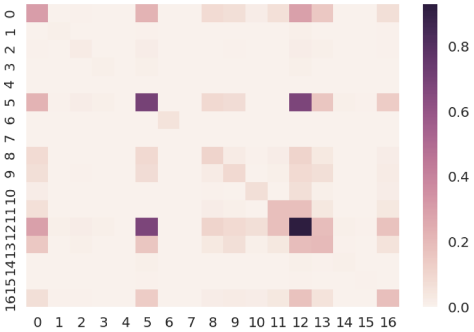


Figure 1: Co-occurrence among image labels. Numbers along axes represent label classes.

4. Dataset and features

The dataset contains 40479 labeled satellite images. The labeling has primarily been done using CrowdFlower. Two types of images have been provided, JPG and TIF. Both JPG and TIF images are 256x256 pixels. The JPG images have 3 channels - Red, Green and Blue. The TIF images have 4 channels - Red, Green, Blue and IR.

The public leaderboard on Kaggle uses F2 scores on test data whose labels are withheld. The private leaderboard uses test data that is withheld.

4.1. Correlation in labels

The labels have significant correlations. For example, every image has exactly one atmospheric condition label from among clear, haze, partly cloudy and cloudy. Labels like “habitation” tend to occur with other markers of human activity. “Cultivation” and “agriculture” don’t co-occur in images. We utilize information from co-occurrence of labels as described in the methods and experiments sections. Fig. 1 shows a heatmap for the co-occurrence matrix for the labels,

Figure 2: Labeling inconsistency: Image that is labeled as "cloudy". We can see primary rainforest below the clouds, which our classifier detects.

4.2. Noise in labeling

As the data has been labeled by non-expert human raters, we see inconsistencies with the way labels have been assigned. For example, the "cultivation" label is supposed to be a subset of "agriculture". However, there are many images that are only labeled as "cultivation", and not "agriculture". Another example of inconsistency is that some images that have haze are only labeled with "haze", while others are also labeled with land features that lie under the haze. See Figure 2 for an example of a similar issue.

5. Experiments and results

5.1. Architecture options

We experimented with two architecture options in this project. The first was a relatively simple setup with four convolutional layers followed by dense layers. The second option we considered was using a model pre-trained on Imagenet.

5.1.1 Simple CNN

In this setup, we have four convolutional layers. Every layer has 32 filters. We use dropout for regularization, and 2x2 max pooling for reducing the spatial resolution. The convolutional layers are followed by two dense layers with 256 units each. The second layer uses L2 regularization, with regularization strength of 0.01. We use dropout between the dense layers. Finally, the output layer is a dense layer with 17 units, sigmoid activation and L2 regularization with strength of 0.01.

The starter code that was used as a base for this architecture is at [2].

REGULARIZATION: We started our experiments using an image size of 64x64, and observed a good train/validation curve (Fig. 4 (a)). On experimenting with higher resolution of 128x128, we saw significant overfitting, caused by the much larger dense layer (Fig. 4 (b)). To compensate for this, we introduced L2 regularization, which had the desired effect of reducing overfitting (Fig. 4 (c)).

The best F2 score we achieved using this approach was 0.8974 on a 64x64 input image. For 140x140 input images, we achieved a score of 0.8645.

5.1.2 Pre-trained model

Imagenet images appear to be significantly different to the satellite images we encounter in this project. We therefore started with the hypothesis that pre-trained models will not show good results, and will not be useful for our application.

To test this hypothesis, we created an architecture that combined a pre-trained Inception V3 model with dense layers with sigmoid activations. Details of the architecture we used can be found in the methods section of this report.

On training the model based on Inception V3, we find that we can achieve significantly higher F2 score compared to the simple CNN described in the previous section, for the same size input image. This invalidates our hypothesis, and shows that pre-trained models can be used in very different settings. The reason this happens is probably that the features learnt in the first few layers are application independent, and represent general structures in data.

5.2. Label correlations

As described in Section 4.1, the labels in this task are correlated. An examination of failure cases shows that using label correlation can help improve performance. Fig. 5 shows two images that have been assigned multiple atmospheric condition labels, which results in mistagging.

We have attempted to address this situation using an additional term in the loss equation, as described in the methods section. This term incentivizes co-occurrence pattern in the predicted labels that match the co-occurrence pattern that is observed in the training data. We define a co-occurrence loss matrix that has high weights on cells that correspond to labels that should not co-occur (Fig. 6 (a)). After incorporating a loss term for the co-occurrence, the predicted label distribution changes to become more similar to what we see in data. A heatmap showing the change in label co-occurrence because of this loss term shows the effect in reducing co-occurrence (Fig. 6(b)). Table 4 shows the reduction in co-occurrence due to the loss.

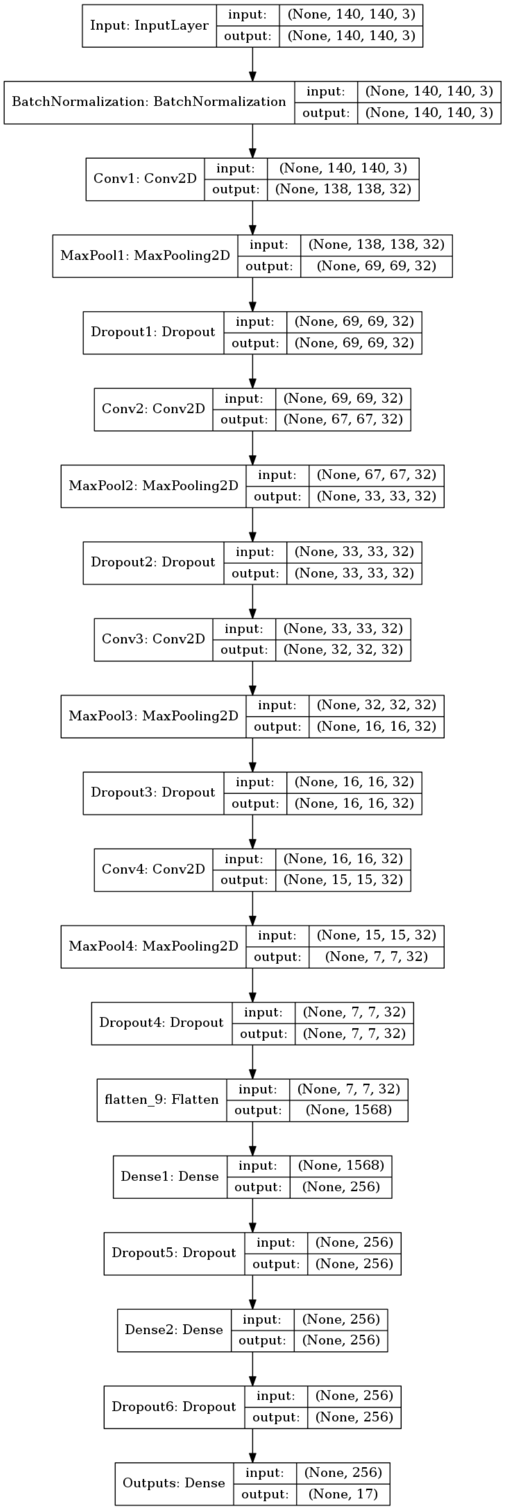

Input: InputLayer
input: (None, 140, 140, 3)
output: (None, 140, 140, 3)
BatchNormalization: BatchNormalization
input: (None, 140, 140, 3)
output: (None, 140, 140, 3)
Conv1: Conv2D
input: (None, 140, 140, 3)
output: (None, 138, 138, 32)
MaxPool1: MaxPooling2D
input: (None, 138, 138, 32)
output: (None, 69, 69, 32)
Dropout1: Dropout
input: (None, 69, 69, 32)
output: (None, 69, 69, 32)
Conv2: Conv2D
input: (None, 69, 69, 32)
output: (None, 67, 67, 32)
MaxPool2: MaxPooling2D
input: (None, 67, 67, 32)
output: (None, 33, 33, 32)
Dropout2: Dropout
input: (None, 33, 33, 32)
output: (None, 33, 33, 32)
Conv3: Conv2D
input: (None, 33, 33, 32)
output: (None, 32, 32, 32)
MaxPool3: MaxPooling2D
input: (None, 32, 32, 32)
output: (None, 16, 16, 32)
Dropout3: Dropout
input: (None, 16, 16, 32)
output: (None, 16, 16, 32)
Conv4: Conv2D
input: (None, 16, 16, 32)
output: (None, 15, 15, 32)
MaxPool4: MaxPooling2D
input: (None, 15, 15, 32)
output: (None, 7, 7, 32)
Dropout4: Dropout
input: (None, 7, 7, 32)
output: (None, 7, 7, 32)
flatten_9: Flatten
input: (None, 7, 7, 32)
output: (None, 1568)
Dense1: Dense
input: (None, 1568)
output: (None, 256)
Dropout5: Dropout
input: (None, 256)
output: (None, 256)
Dense2: Dense
input: (None, 256)
output: (None, 256)
Dropout6: Dropout
input: (None, 256)
output: (None, 256)
Outputs: Dense
input: (None, 256)
output: (None, 17)

Figure 3: Simple CNN architecture

	cloudy	haze	partly cloudy
No loss clear	0.006	0.0351	0.026
With loss clear	8.27e-05	2.88e-04	1.99e-04

Table 4: Co-occurrence of atmospheric labels before and after applying co-occurrence loss.

Initial F2	0.8559
Final F2	0.8894
Improvement	0.033

Table 5: Improvement in F2 score from optimizing score thresholds.

5.3. Choosing the classification thresholds

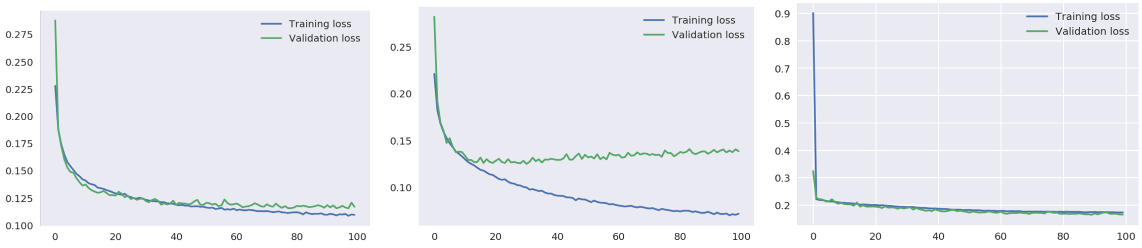
As this is a multi-label classification problem, we have to independently choose the thresholds that separate presence and absence of a label for an input image. Our hypothesis is that a threshold different from 0.5 is likely to work better, for two reasons. First, the data has significant class imbalance, and score thresholds help compensate for it. Second, the evaluation metric F2 score penalizes recall errors more than precision errors, and score thresholds can help us use this information.

There are two options for optimizing the score thresholds - we can either bake this into the model learning using approaches like weighted training data. Or we can empirically choose the best threshold after training is complete. Empirically choosing the best threshold after the model is learnt has been found to be an effective and efficient approach in [18]. We try various thresholds between 0 and 1 for every label, and find the combination that works the best. This gives us a significant improvement in F2 score, as seen in Table 5.

6. Conclusions and future work

In this project, we explored three areas relating to labeling satellite imagery - label correlation, validity of transfer learning and optimization of score thresholds. We determined that addressing label correlation is important to produce sensible output in a multi-label classification task. On transfer learning, we found that models trained on very different data can still show promising results by using fundamental structures in images, Finally, we saw that choosing the right classification threshold is critical for good performance of multi-label, recall oriented algorithms.

Future work in this area can explore the utility of segmentation algorithms for such tasks. We have observed errors where a river might be tagged both as a river and as a road. Segmentation can help ensure that specific parts of the



(a) 64x64

(b) 128x128

(c) 128x128 regularized

Figure 4: Train and validation loss for various input sizes.

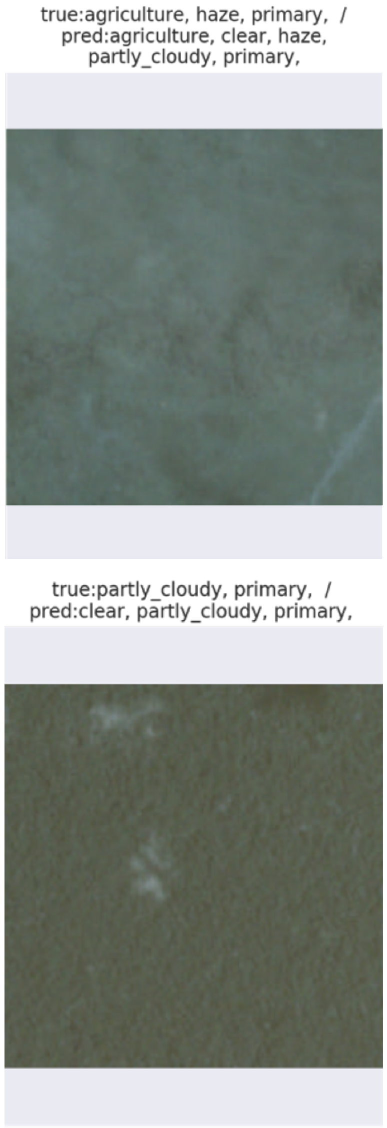

true:agriculture, haze, primary, /
pred:agriculture, clear, haze,
partly_cloudy, primary,
true:partly_cloudy, primary, /
pred:clear, partly_cloudy, primary,

Figure 5: Algorithm mistakes for correlated labels.

image do not get assigned multiple, incompatible labels.

Further utilizing label correlations is another area for further exploration. In particular, approaches that incorporate label correlations in the learning process can be effective. Combinations of loss functions, for example combining binary cross-entropy over a subset of labels, with softmax over atmospheric condition labels is another approach worth attempting.

Finally, we often noticed overfitting in the experiments we conducted. Using image augmentation, through approaches like rotation, translation, flipping and cropping can enable our network to train on more varied data and be more robust to overfitting.

References

[1] https://keras.io/applications/#inceptionv3. [Online; accessed 12-June-2017].

[2] https://github.com/EKami/planet-amazon-deforestation. [Online; accessed 12-June-2017].

[3] N. Audebert, B. Le Saux, and S. Lefvre. Semantic Segmentation of Earth Observation Data Using Multimodal and Multi-scale Deep Networks. In *Computer Vision ? ACCV 2016*, pages 180–196. Springer, Cham, Nov. 2016.

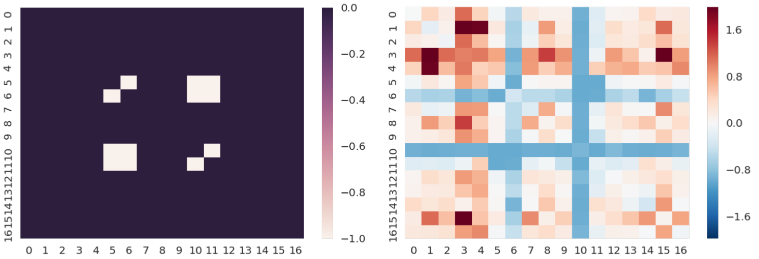
[4] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva. Land use classification in remote sensing images by convolutional neural networks. *arXiv preprint arXiv:1508.00092*, 2015.

[5] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. *CoRR*, abs/1312.4894, 2013.

[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*, 7(11):14680–14707, 2015.

[7] N. Jean, M. Burke, M. Xie, W. M. Davis, D. B. Lobell, and S. Ermon. Combining satellite imagery and machine learning to predict poverty. *Science*, 353(6301):790–794, 2016.

[8] M. Längkvist, A. Kiselev, M. Alirezaie, and A. Loutfi. Classification and segmentation of satellite orthoimagery using convolutional neural networks. *Remote Sensing*, 8(4):329, 2016.



(a)

(b)

Figure 6: Figure on left shows co-occurrence loss matrix that we have created. Figure on the right shows the change in co-occurrence because of the co-occurrence loss

[9] G. Levin, D. Newbury, K. McDonald, I. Alvarado, A. Tiwari, and M. Zaheer. Terrapattern: Open-ended, visual query-by-example for satellite imagery using deep learning. 2016.

[10] Z. C. Lipton, C. Elkan, and B. Naryanaswamy. Optimal thresholding of classifiers to maximize f1 measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–239. Springer, 2014.

[11] H. Ma, E. Chen, L. Xu, and H. Xiong. Capturing correlations of multiple labels: A generative probabilistic model for multi-label learning. *Neurocomputing*, 92:116–123, 2012.

[12] V. Mnih and G. E. Hinton. *Learning to Detect Roads in High-Resolution Aerial Images*, pages 210–223. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[13] V. Mnih and G. E. Hinton. Learning to label aerial images from noisy data. In J. Langford and J. Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 567–574, New York, NY, USA, 2012. ACM.

[14] J. A. Montoya-Zegarra, J. D. Wegner, Ľ. Ladický, and K. Schindler. *Mind the Gap: Modeling Local and Global Context in (Road) Networks*, pages 212–223. Springer International Publishing, Cham, 2014.

[15] S.-H. Park and J. Fürnkranz. Multi-label classification with label constraints. In *Proceedings of the Proceedings of the ECML/PKDD-08 Workshop on Preference Learning (PL-08)*, pages 157–171, 2008.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[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*, pages 2285–2294, 2016.

[18] N. Ye, K. M. A. Chai, W. S. Lee, and H. L. Chieu. Optimizing f-measure: A tale of two approaches. *CoRR*, abs/1206.4625, 2012.

[19] M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2014.