CS230

Road Detection Using Satellite Imagery

David Kwok
Management Science and Engineering
Stanford University
kwokd@stanford.edu

Soumya Patro
Management Science and Engineering
Stanford University
sopatro@stanford.edu

Kate Wharton
Graduate School of Business
Stanford University
kateow@stanford.edu

Abstract

Satellite images offer a wealth of low-cost, open source information that can be leveraged for various humanitarian and development challenges such as estimating population sizes. We present two models for feature detection from satellite images, focusing on road detection. The first is a binary classification model using a 14-layer ResNet to predict whether or not an image tile contains a road, achieving 80% accuracy. The second is a semantic segmentation model also using a 14-layer ResNet. We achieve much lower accuracy (0.15 precision and 0.34 recall) on the pixel-level prediction, largely attributable to limitations of the training data.

1 Introduction

Recent humanitarian crises have dramatically increased the number of refugees, and humanitarian organizations often struggle to maintain accurate, timely measures of population flows in camps. One novel idea supported by UNICEF Innovation and Bayes Impact is using open-source satellite imagery to estimate the population of informal settlements.[1] We focus our project specifically on road detection, as road miles are a relevant proxy for population mobility in remote areas.

The inputs for our models are: 1) color satellite image tiles with latitude and longitude coordinates, and 2) markers for the location of roads in these images. We train two different sets of convolutional neural network architectures (CNNs) to output: 1) a binary prediction for whether or not a road exists in an input image, and 2) semantic segmentation to predict whether each pixel in an image corresponds to a road. We discuss both sets of models in detail below.

2 Related work

We build on a growing body of literature around feature detection from aerial imagery, a problem that has occupied researchers for several decades.[2] We primarily reference more recent approaches to

[1] One demonstrated use for open-source satellite imagery is a source of data for machine learning algorithms to predict poverty.[4]

[2] The earliest learning approaches suffered from small sets of training data and small areas of context. For example, Boggess (1993) uses 5x5 pixel tiles as input images. Minh and Hinton (2010) build on this early work but use a much larger CNN with more training data and a larger context than previous algorithms.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Figure 1. Sample Input Data: Satellite images with road labels

feature detection in images, including AlexNet, VGG-16 and ResNet.[5][7][3] To our knowledge, there have not been any papers published examining the use of these modern frameworks on satellite imagery, which is the focus input of our project.

We base our work on an open-source repository authored by Ahmet Taspinar and described in his blog post, "Using Convolutional Neural Networks to detect features in satellite images."[8] The repository loads satellite images and data indicating location of roads, maps these two layers together, implements a CNN to predict the presence/absence of a road in an image tile, and evaluates the accuracy of the model. Below, we describe our significant modifications and extensions to the existing model. In particular, we offer a modern implementation for road detection that utilizes the ResNet architecture to enhance performance, and we also extend the image classification task to the much more challenging task of pixel-level road detection.

3 Dataset and Features

The dataset we use to train and evaluate our neural network consists of publicly available satellite images of the Rotterdam area.[3] We downloaded 10,000 RGB satellite image tiles of dimension 256x256x3.[4] In addition, the Dutch government maintains data on roads, contained in shapefiles that have information on the coordinates and type of each road located in the Netherlands and can be overlayed with the satellite images to act as ground-truth labels.[5] Both the satellite images and road labels are updated frequently, and we use data updated as of late 2017.

For each image, we used a custom Python script to create a boolean mask of corresponding height and width that indicated whether each pixel corresponded to a road, using the start and end coordinates of each road segment in the shapefile. Approximately 60% of the image tiles contained at least one road segment. Next, because the raw image tiles had relatively large dimensions (256x256x3), we attempted two different methods of resizing the images so that they conformed to the CNN architectures. Initially, we split each of the original images of dimension 256x256x3 into 16 images of dimension 64x64x3; however, these smaller images yielded weak performance, suggesting that larger context is helpful for road detection. Thus, we attempted an alternative method of resizing the images to 64x64x3 by using an algorithm from Python Image Library, which averages across pixel values to downsize the image. We ultimately used the downsized images in our model. We divided the preprocessed, labeled, shuffled dataset into 80% training examples, 10% validation set examples, and 10% test set examples. Figure 1[6] shows an example of the input data.

[3] <https://www.pdok.nl/en>; <https://data.overheid.nl/data/dataset>

[4] Bounding box coordinates: (51.82781, 4.44428), (52.00954, 4.73177)

[5] <https://www.rijkswaterstaat.nl/apps/geoservices/geodata/dmc/nwb-wegen/geogegevens/shapefile/Nederland_totaal/>

[6] Image source: Taspinar, A. (2017).

Figure 2. ResNet Architecture

4 Methods

4.1 ResNet for Image-Level Classification

For the task of predicting the presence of a road in a given image tile, we tested several CNN architectures on the processed data. Each algorithm takes as input a single 64x64x3 satellite image tile and generates a probability that the tile contains a road. For each algorithm, we used the binary cross entropy loss function, averaged across training examples. In this cost function, m refers to the number of examples, y to the ground-truth label, and a to the predicted probability output by the model.

$$-\frac{1}{m}\sum_{i=1}^{m}(y^{(i)}log(a^{[L](i)}) + (1-y^{(i)})log(1-a^{[L](i)]}))$$

We attempted several different model architectures and hyperparameters, including AlexNet, VGG-Net16, and ResNet-50[5][7][3][7], but found the best performance from a simplified version of the ResNet with just one convolutional block and two identity blocks (Figure 2). A key feature of ResNet architectures is the presence of "shortcut connections," allowing the network to easily learn the identity mapping, which does not add extra parameters or computational complexity, so the network can be built deeper more efficiently. After the first convolutional filter in the ResNet, we introduced a convolutional block and two identity blocks. The convolutional block and identity block have three conv2D layers each, and we perform batch normalization followed by a ReLU activation layer after each conv2D layer. Finally, the output is flattened and propagated through three fully-connected layers. All the weights in the network were initialized as Glorot-uniform.

INPUT -> CONV2D -> BATCHNORM -> RELU -> MAXPOOL -> CONVBLOCK -> IDBLOCK*2 -> AVGPOOL -> FC1 -> FC2 -> FC3 -> OUTPUT

The input to this network was a 64x64x3 image and the output was a softmax unit that assigned probabilities to the labels. We call this network ResNet-14, since we use ten convolutional layers, three fully connected layers, and one output layer.

4.2 ResNet for Pixel-Level Road Detection

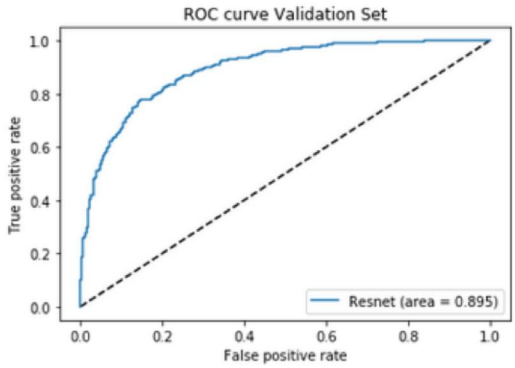
While predicting the presence of a road may be useful in certain applications, we embarked on the more challenging task of creating a model to predict roads at the pixel level. Specifically, given a satellite image with three RGB channels, our model outputs a single-channel matrix of the same height and width, where each element contains a binary prediction of whether the pixel corresponds to a road. Thus, we used the binary cross entropy cost function, averaged across each pixel in our output and averaged across training examples.

For this pixel-level road detection task, we implemented the same ResNet-14 architecture that performed well for the classification task. While the input was kept the same, we modified the output layer to contain 4,096 units with sigmoid activation to predict the probability that each pixel corresponded to a road. This vector of 4,096 was then reshaped into a 64x64 matrix, corresponding to the original height and width of the input image.

The main issue that arose when training this model was the imbalance of class labels in the dataset. Across all of our images, only about 10% of the pixels corresponded to road labels, while the other 90% where labeled as "other". This imbalance caused our model to achieve high accuracy by consistently predicting the absence of roads. We addressed this issue in several ways. First, we subsetted our

[7] See milestone report for details on other architectures and hyperparameters that we attempted. In this report, we only detail the highest-performing results.

Figure 3. ROC Curve for Binary Classification



training examples to only those images that contained at least one road pixel[8]; after this resampling, about 17% of all pixels corresponded to roads. Second, we updated our cost function by applying a weighting factor set to 0.83[9] that penalized false negative predictions more heavily than false positive predictions. The weighted binary cross-entropy cost function is shown below. In this equation, j is the index corresponding to a pixel, p is the total number of pixels per image, and λ is the weight parameter.

$$-\frac{1}{p*m}\sum_{i=1}^{m}\sum_{j=1}^{p}(\lambda y_j^{(i)} log(a_j^{[L](i)}) + (1-\lambda)(1-y_j^{(i)}) log(1-a_j^{[L](i)}))$$

5 Results and Discussion

5.1 ResNet for Image-Level Classification

We trained our model predicting the presence of a road for a given satellite image on the ResNet-14 architecture over 8,000 training examples and 1,000 validation set examples. We found that the Adam optimizer did not work well because of vanishing gradients. Instead, we used Stochastic Gradient Descent as our optimizer with gradient clipping set to 0.5. After performing a grid search, we settled on a learning rate of 0.01 and mini-batch size of 32, which yielded the best balance between training time and convergence of the cost function. Training on more than three epochs led to overfitting, as the accuracy on the training set continued to improve, while the accuracy on the validation set decreased. Hence, we implemented dropout regularization on the fully-connected layers (with a parameter tuned to 0.5), yielding train/validation set accuracies of roughly 80%.[10] Figure 3 plots the ROC curve for the validation set, which indicates a high Area Under the Curve (AUC) of 0.895.

We then analyzed examples of misclassified images in order to understand the deficiencies of our model. In Figure 4, Image 1 shows a field with no actual roads, but with several vertical and horizontal bands that the model could have mistaken for roads. Similarly, image 2 shows an actual wide road with a sharp bend, but the image was classified as containing no road. These examples point to the possibility that the model has been optimized to recognize narrow, straight edges as roads and may require additional training to recognize wider, bent roads as well.

5.2 ResNet for Pixel-Level Road Detection

For the task of pixel-level road detection, we trained the ResNet-14 model using the Adam optimizer with a learning rate of 0.01 and mini-batch size of 32, which yielded the best balance between training time and convergence of the cost function. As mentioned above, we originally observed biased results due to the class label imbalance (only about 10% of all pixels in the full training set corresponded to roads), so we subsetted our training set to examples containing at least one road pixel, and we used a

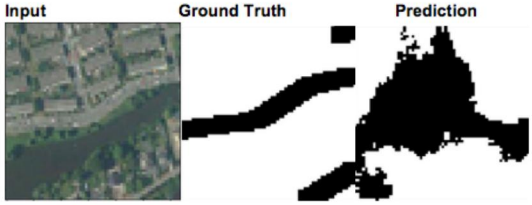
[8]The validation set and test set were not modified, so that they continued to reflect our target of the original distribution of images.

[9]This weight reflected the empirical frequency of non-road pixels across our training set.

[10]This was slightly higher than the author's original reported results, produced on the VGG-Net16 model using a much smaller subsample of just 1,200 training images and 400 validation set images.[8]

Figure 4. Error Analysis for Binary Classification

Figure 5. Error Analysis for Pixel-Level Road Detection



weighted loss function that penalized false negatives more than false positives. After training on 500 epochs, our model achieved a relatively weak precision score of 0.15 and recall score of 0.34 across all pixels on the validation set. Here, we report the precision and recall, rather than accuracy or AUC, since these measures are more robust to label imbalance.

To better understand the model's performance and perform error analysis, we plotted input images, ground truth labels, and predictions for a number of images. Figure 5 presents an example image containing road segments that our model is unable to predict with high accuracy. This example indicates that our model fails to recognize roads as contiguous, smooth paths, and after extensive error analysis and diagnosis, we narrowed down two key reasons for the poor performance. First, because the pixel-level labels were generated only from road coordinates, they tended to be uniform in width and did not necessarily reflect the actual width of the roads as they appeared in the input images. Second, a number of the wider roads in the dataset (including freeways) were not documented as official roads in the shapefiles. These sources of inconsistency in the ground-truth labeling likely played a large role in biasing the predictions and hurting the overall performance of the model.

6 Conclusion/Future Work

In our project, we presented two Deep Learning methods for detecting roads in satellite images: one that classifies whether a given tile contains a road, and the other that predicts the exact pixels corresponding to roads. The most successful model we attempted for both tasks was a 14-layer modified ResNet. The results achieved were strong for the image-level classification task but were much weaker for the pixel-level prediction.

Future work would focus on improving the accuracy of road labels. For instance, modifying the automatically generated labels by hand, such that they better reflect actual road widths and include roads not in the "official" database, would help increase performance significantly. Further, with more computational resources, we would apply data augmentation (e.g., rotations, cropping) of training images to help the model recognize a wider distribution of angles and orientations of roads.

One surprising finding that we encountered was that the simplified 14-layer ResNet performed significantly better on both tasks than a much deeper 50-layer ResNet. One issue that deeper networks have been known to face is degradation, where training accuracy is quickly saturated and then decreases with network depth.[3] Though residual networks were designed to mitigate this problem, we still encountered issues with the implementation of a deep ResNet, and in future work, we would be interested in investigating more abstractly the reasons for the differences in performance between deeper and shallower models.

7 Contributions

David Kwok contributed significantly to the coding for data cleaning and model development. He acted as the link between the high-level strategy of the team and the details of the code development, and he was also responsible for communicating with the project mentors to raise issues the team was facing. David also drafted large portions of the milestone and final reports.

Soumya Patro primarily contributed to making things work. She built the AlexNet model in Keras. She extensively debugged and iterated multiple architectures for ResNet. She came up with innovative solutions, tuned hyperparameters and iterated quickly to tackle problems faced during training and data preprocessing.

Kate Wharton supported the project strategy and brought in relevant experience from the humanitarian and development sectors during the problem definition. She identified the Github repository used as the baseline model, designed the poster, wrote portions of the reports, and made sure the final deliverables came together.

References

[1] Boggess, J. (1993). *Identification of roads in satellite imagery using artificial neural networks: A contextual approach.* In Technical report, Mississippi State University.

[2] Chollet, F. et. al. (2015). *Keras*. https://keras.io.

[3] He, K., Zhang, X., Ren, S., and Sun, J. (2016). *Deep Residual Learning for Image Recognition.* Proceedings of CVPR, pp. 770-778, 2016.

[4] Jean, N., Burke, M., Xie, M., Davis, W., Lobell, D., and Ermon, S. (2016). *Combining Satellite Imagery and Machine Learning to Predict Poverty.* In Science 353(6301): 790–94.

[5] Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). *ImageNet classification with deep convolutional neural networks.* In NIPS, pp. 1106–1114.

[6] Minh, V., and Hinton, G. (2010). *Learning to Detect Roads in High-Resolution Aerial Images.* In Proceedings of the 11th European Conference on Computer Vision (ECCV): 210-223.

[7] Simonyan, K. and Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition.* In ICLR 2015.

[8] Taspinar, A. (2017). *Using Convolutional Neural Networks to detect features in satellite images.* Blog post and Github repository. <http://ataspinar.com/2017/12/04/using-convolutional- neural-networks-to-detect-features-in-sattelite-images/>. <https://github.com/taspinar/sidl>.