Road GAN: An Image to Image Translation Model for Road Network Extraction in Remotely Sensed Images

Trevor Houchens

May 2020

Abstract

Road maps often rely on government data and user reports, leading to poor data collection in rural areas. Using generative adversarial networks, a deep learning architecture, I automatically extracted road maps from remotely sensed images. Data was sourced from Planet Labs and Open Street Map, a publicly available road database.

After curating a dataset with examples and labels, I used Google Colab to train "Road GAN" for 260 epochs. The resulting network was capable of creating accurate road geometries (linear structures) and matching them up with road-like objects in the Planet scenes. For comparison purposes, I also trained an SVM for road classification. Using precision, recall, and F1 score as performance metrics, as well as qualitative analysis of the outputs, I found that Road GAN outperformed the SVM.

1 Introduction

In the modern era, computers and cellphones have provided us with invaluable tools, and some of the most important of these tools are digital maps. It's easy to take it for granted that we can pull out a phone, open Google Maps, and instantly have directions to wherever we want to go, but the fact that we have maps that are so reliable and accurate isn't a given. There is no easy way to keep track of all of our roads, and roads are constantly being added, changed, or abandoned.

Google has stated that their map data often comes from authorities (like governments) that collect data on roads, and it can also be corrected by users that report mistakes. However, in areas with fewer people or no authority tracking roads, they generally turn to remotely sensed data and machine learning techniques. This does not come without challenges though. Google has said that leaves concealing roads and dry riverbeds that look like dirt roads are some challenges that they've encountered, but there are surely many more.[1]

Despite the potential challenges, remote sensing has a lot to offer anybody who is trying to keep track of our constantly changing road network. Roads have a clearly recognizable structure, and asphalt has a reflectance spectrum different from other common land covers like water or vegetation. While there may be some edge cases that make certain roads tricky to identify, there is a lot of information to be gained by turning to remotely sensed imagery.

Though problems like tree cover may make road classification more difficult, we can still try to mitigate their effects. One option would be to try tweaking handcrafted models to account for every edge case that we can think of. Instead, I propose to use a deep learning network, specifically, a generative adversarial network (GAN), to learn a road extraction technique given labeled training samples.

2 Principles

In a deep learning network, the actual steps taken by the network to extract and process information is somewhat of a black box. There are generally millions of learned parameters and there is not an easily understandable meaning behind the weights. We can, however, speculate as to what types of information the network might make use of. It has to use what it is given as input to produce the outputs, so it will most likely use the same information that we might use with more traditional remote sensing techniques.

A common method of separating surface type classes in remote sensing is by using the spectra of each pixel. For vegetation, we expect a strong reflectance in the infrared, a small bump in the green, and lower reflectances for red and blue. For water, we expect low reflectance across the visible and near IR wavelengths. Most roads are made of asphalt which has it's own unique spectrum. Fresh asphalt has a reflectance of $<5\%$ in blue wavelengths, which increases almost linearly to around 10% in the near IR. As asphalt ages, it's reflectance increases; the spectral shape is maintained but the actual reflectance increases until it ranges from around 10% in blue to 20% in near IR.[2]

While the shape of an asphalt spectrum is much different from the spectra of water or vegetation, there are other types of land cover (especially anthropogenic structures) that may be more easily confused with asphalt if spectrum is all that we consider. If we were to visually inspect a map, we would likely be able to recognize a road simply by it's long, winding shape, without any consideration of it's spectrum. Through the use of convolutional filters and multiple layers, our deep learning model should be able to detect relevant structural information that could help it filter out road like structures by shape.

In reality, when a human looks at a satellite image, they are using a lot of context clues to determine what each image component represents. If we wanted a program to look for long winding structures that had a spectrum similar to that of dry dirt or sand, it might reasonably come up with a beach. However, a human could easily distinguish a beach from a road by observing that the beach runs along the edge of a water body. These subtle (or not so subtle) context clues are important to our ability to extract structure from remotely sensed images, but are almost impossible to hard code into any sort of computer program. When a deep network with millions of parameters is trained on many training examples, it can learn to extract the information that will facilitate the best classification without us having to explictly code in what information it should be looking for.

3 Data

One of the main challenges of deep learning is achieving the right quantity and quality of data. In this scenario, the training data consisted of remotely sensed images and their corresponding road masks. For the images, I used data from Planet Labs since they provided imagery with 4 bands (Red, Green, Blue, Near IR) and more importantly, 3 meter resolution. The Landsat resolution of 20m is too coarse to pick up most roads, but with 3m resolution they should be visible. The data I used for training all came from western Rhode Island and was taken in early April, 2020. The motivation for this was that there were few, if any, leaves out at this time, which meant that roads were clearly visible.

In order to train a network to extract roads, it was also necessary to produce labels for these roads. The labels I produced were binary masks with 1s marking pixels in the image that were roads and 0s marking pixels that weren't roads. To produce labels, I used a public API called Open Street Map. This returned a series of coordinates for each road, as well as a road type classification (e.g. "motorway", "residential", etc.). The width of the roads was not provided by the API. Considering the large discrepancy in road width between major interstates and small alleyways, I didn't think that setting all the roads to equal width would result in a well trained model. I ended up using the remotely sensed images themselves to estimate the average road width (in pixels) for each road class, which I then used to draw the training labels. The full process is described in the methodology section.

4 Methodology

4.1 Preprocessing

The quality of training data is a significant factor in the quality of the model that it is used to train. Therefore, I attempted to match the road widths in my labeled road masks with the true road widths as accurately as possible. To do this, I relied on the gradients in the remotely sensed imagery. Since most roads will reflect radiation differently from any bordering surface type, we would expect to find a large gradient along the edge of the road, orthogonal to the center line of the road.

The gradient of the image is a measure of the first derivative of the image intensity calculated pixel wise. I calculated both the x and y derivatives by convolving the images with a Sobel filter.

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Figure 1: An x and y Sobel filter, respectively

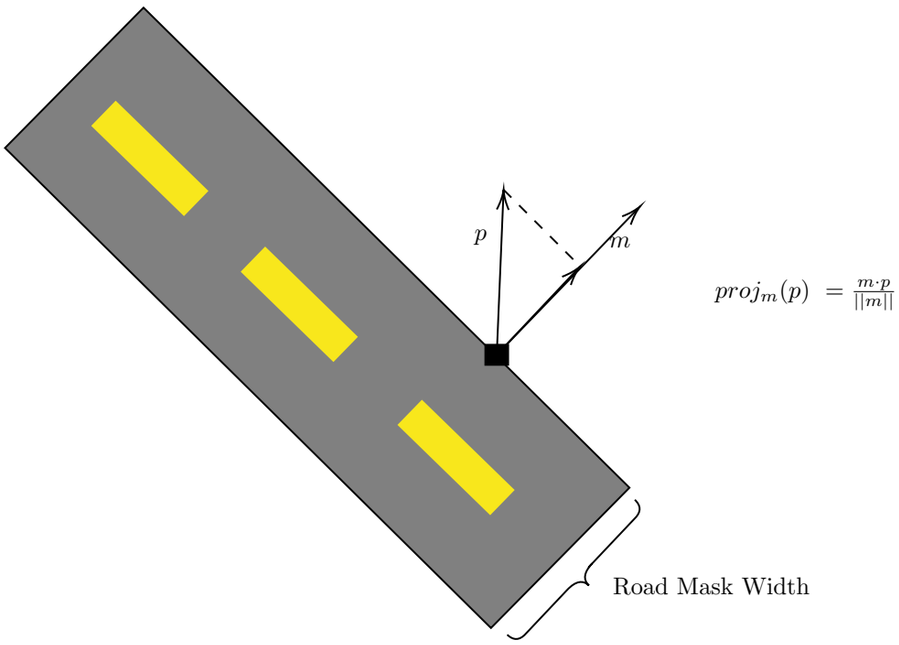

Figure 2: Diagram of the gradient component calculation performed, where p is the Planet image gradient, and m is the mask gradient

The x and y component together constitute the whole gradient. Once I had the gradients for the image, I used them to determine the optimal road width for each class of road. For each class of road, I performed the following calculations:

For each width from 1 to 15, a road mask was drawn with roads of the given width (in pixels). Gradients were then calculated for the mask. Since the masks were binary, all gradients were on the road edges, and orthogonal to the center line of the road. For every edge pixel on the road mask, I took the component of the vector from the Planet scene at that pixel onto the gradient from the road mask. In Figure 2, you can see a sample edge pixel (in black) as well as the associated mask gradient vector, Planet gradient vector, and the component of the Planet vector onto the mask vector. The gradient vector for the mask was only used as a way of calculating a vector orthogonal to the center line of the road.By storing the average component vector for each road width, I had a measurement of how well the road edges in the Planet scene lined up with the road masks for each width.

Road Type	Optimal Width (pixels)	Optimal Width (meters)
Motorway	11	33
Trunk	6	18
Primary	7	21
Secondary	4	12
Tertiary	3	9
Unclassified	3	9
Residential	3	9
Living Street	2	6
Service	3	9
Motorway Link	3	9
Trunk Link	6	18
Primary Link	4	12
Secondary Link	2	6
Tertiary Link	2	6

Figure 3: Optimal road widths by road type

Note: The optimal road widths were only calculated on one scene, since it was very computationally expensive. I may have gotten marginally better labelings if I computed the widths scene by scene instead.

After the optimal widths were calculated, I used the coordinates from Open Street Map to draw road masks for each scene. Then, I split each Planet scene (and it's corresponding road mask) into smaller 256x256 images since that is the size of the input I chose for my GAN architecture. Ultimately, I ended up with about 20,000 of these smaller scenes. However, the Planet scenes often have black pixels around the edges where no data was collected. Some training was done using the full dataset of 20,000 images, but I later threw out all 256x256 images that weren't at least 25% non-zero pixels. I ended up removing 7,006 out of 20,056 training examples when I did this. Since some images were held out for validation and testing, I ended up with 9,349 total training images after the mostly black images were discarded.

4.2 Model Training

The model architecture that I chose was based off the Pix2Pix image translation model[4]. Like all GANs, it consists of two separate networks, the generator and the discriminator. In my model, the generator takes in a 256x256x4 section of a Planet scene, and outputs a 256x256 binary classification map. The discriminator takes in a 256x256x4 section of a Planet scene as well as a 256x256 binary classification image. The classification image passed into the discriminator can either be an output from the generator, or one of the "ground truth" training labels that I produced using Open Street Map data. The discriminator outputs an array of values that each correspond to a particular patch of the input image and indicate whether the discriminator believes that patch is real or generated. In reality, the entire input will be either real or generated, but this type of "PatchGAN" discriminator has been found to be effective in previous work[4].

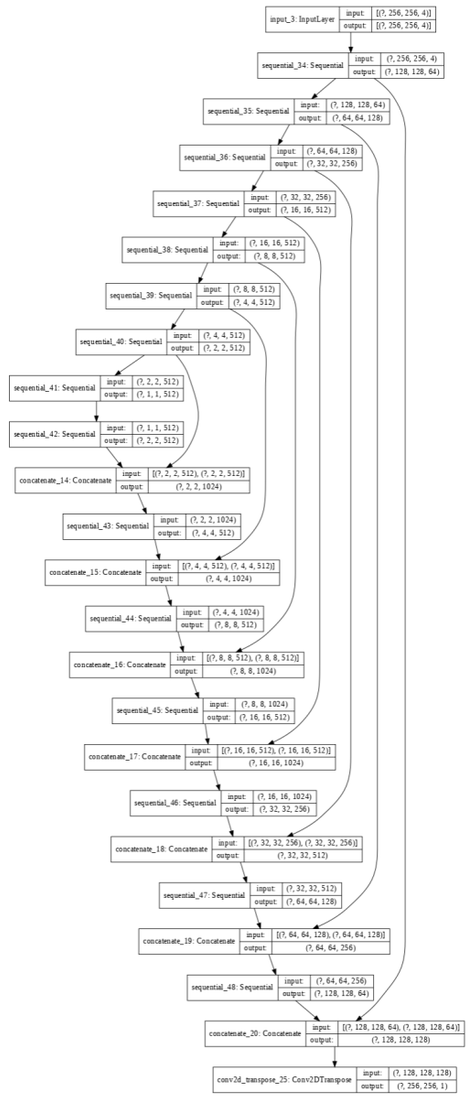

Figure 4: The generator structure, following a U-Net architecture. The input is convolved to a smaller dimension, then deconvolved with skip connectins.

The "adversarial" part of generative adversarial network comes from the fact that the generator and discriminator are competing against one another. The discriminator loss is dependent upon the discriminator being able to tell that the generated road masks are not real, and the generator's loss is dependent upon the generator being able to produce road masks that the discriminator believes are real. Thus, the discriminator and generator losses usually fluctuate around some constant value, and if they drop too low or rise too high, the training usually fails.

Initially, the training didn't go very well for my network. The discriminator loss tended to be very low

and the generator loss was very high. This indicated that it was very easy for the discriminator to tell my generated masks apart from the ground truth masks. To mitigate this imbalance, I chose to update the weights for the discriminator only during every third epoch, while the generator weights were updated during every epoch (One epoch is when the network is trained on every training example once). This resulted in more stable losses for both the generator and the discriminator (though they did fluctuate on a 3 epoch cycle).

Early in the training, it seemed that the discriminator was mostly using the structure of the mask to make it's decisions, without any regard for the Planet data. I assume that this was occurring because the generator began to produce road masks that resembled road-like structures, but these "roads" didn't line up with the roads in the Planet scene at all.

Around 50 epochs in there was a quick shift and the generated masks started to resemble the true road networks. This is very clearly shown in the generator's L1 loss (Figure 5). The L1 loss is a component of the generator loss that is defined as the absolute value of the difference between the generated mask and the ground truth mask. Therefore, if none of the generated roads line up with the true roads, the L1 loss will be very high, but if the images are similar, the L1 loss will be low. When the generator was producing road like structures that didn't line up with the true roads at all, I observed a very high L1 loss, but once the generated roads started lining up with the true roads, I saw a sharp drop in the L1 loss.

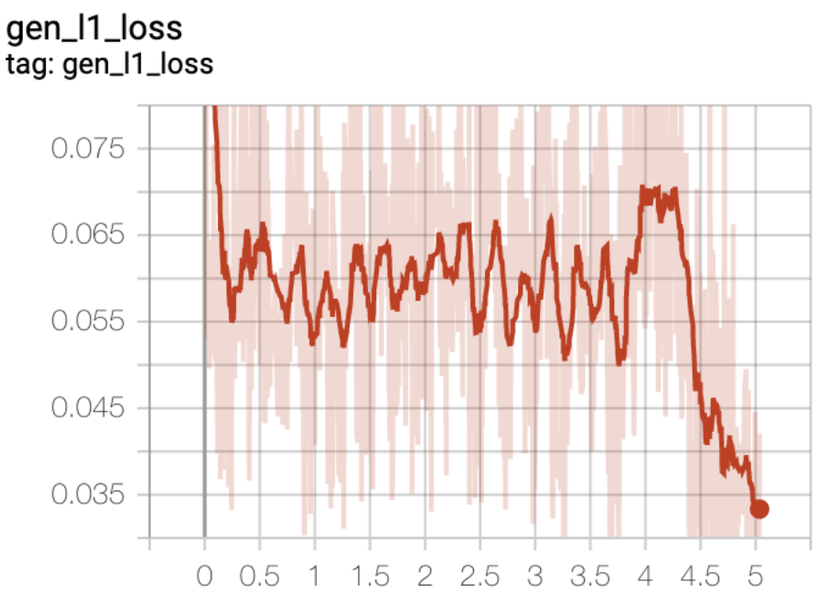

Figure 5: Smoothed L1 loss over the first 5 hours of training

Figure 6: Sample input, ground truth, and generator output around 3 hours in. The generator output appears vaguely road-like but does not match up with the real roads.

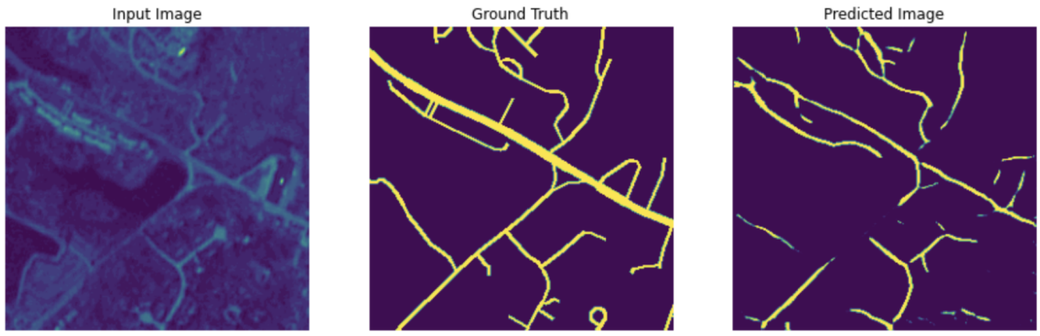

Figure 7: Sample input, ground truth, and generator output around 5 hours in. The generator output has started to follow the actual road pattern.

All of the training was done through Google Colab using at various times Nvidia P100 and K80 GPUs. I trained the network on two separate instances that I ran at the same time. On both instances, I used the full dataset (including all-black images), but on the second instance I swapped out that dataset for one with the black images removed around epoch 80 to see if it would train faster. Both models had exactly the same architecture, just different training data. The first model was trained for around 310 epochs, and the second model was trained for around 260 epochs (roughly 24 hours). I chose to use the second model in my analysis since it appeared to produce better outputs based on my observations of the training.

4.3 Other Methods

The accuracy of Road GAN is hard to interpret since the problem it is trying to solve is complicated. The best way to get a sense of what the results mean is to compare them to some other classification method. To create a comparison point, I trained a support vector machine (SVM) to classify individual pixels as either roads or not roads. Support vector machines are linear classifiers that map the input space into a binary output. I trained an SVM that took in 4 band reflectances and output either 1 for a road, or 0 for no road. My SVM utilized the kernel trick with a radial basis function, which means that instead of just being a linear classifier, it actually had access to many calculated features which allowed it to learn complicated non-linear functions of the original 4 input bands. Since indices like NDVI or NDWI are just non-linear functions of multiple bands, my intent with using an SVM was to produce a classifier that could perform as well as any contrived index.

Note: Regarding the feasibility of using SVMs to classify remote sensing scenes: it took about 5 hours for my trained SVM to fully classify one Planet scene since it had to individually predict each of the tens of millions of pixels. On the other hand band math only takes a few seconds.

5 Analysis and Results

While Road GAN was training, the results that I was observing were results from the training data. This can often be misleading, since the model could fit the training data very well but fail to generalize to other examples. Therefore, I chose 4 Planet scenes to use for testing once the model's training had finished. Three of these scenes came from western Massachusetts. They were also taken during April 2020 and they have the same resolution and bands as the training images. For these images I tried to include suburban areas with dense road networks as well as rural areas with fewer roads. The fourth image was a scene from western Rhode Island taken during July 2019. I included this scene since it has much more leaf cover than the training data. To classify each scene, I divided it up into 256x256 patches, fed the patches through the GAN, and then stitched them back together again.

To get a quantitative metric of how well the network was performing, I used precision, recall, and the

F1 score. These are defined as follows:

$$\text{precision} = \frac{\text{true positives}}{\text{false positives} + \text{true positives}}$$

$$\text{recall} = \frac{\text{true positives}}{\text{false negatives} + \text{true positives}}$$

$$\text{F1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

In this case, the precision indicates what fraction of the roads labeled by the GAN were actually roads, and the recall indicates what fraction of the true roads were correctly labeled by the GAN. The F1 score is a useful way of combining precision and recall, and it has a maximum value of 1. On the testing scenes, Road GAN achieved the following results:

Scene	Precision	Recall	F_1 Score
W. Mass. 1	0.287	0.176	0.218
W. Mass. 2	0.595	0.182	0.279
W. Mass. 3	0.474	0.192	0.274
W. RI	0.347	0.254	0.293
Average	0.480	0.189	0.271

At first glance, these numbers don't seem particularly exciting. Other recent studies utilizing deep learning for road extraction have reported precision and recall of almost 90% [5]. However, the method I used to calculate the scores could be at least in part to blame. These precision and recall scores were calculated on a pixel by pixel basis. That means that each pixel classification output by the GAN was compared with the ground truth pixel value. Therefore, if the road was slightly off to the side of the ground truth, it would be marked as completely wrong. If the drawn road was only half the width of the ground truth road, then the recall for that area would only be 50%. The method I used seems reasonable, and is certainly the easiest to implement, but it could be slightly misleading since a road could still be marked even if it doesn't line up completely with the ground truth.

When we compare the Road GAN results with the precision and recall that the SVM achieved on one Planet scene, Road GAN's performance seems more significant:

Scene	Precision	Recall	F_1 Score
W. Mass. 1	0.114	0.257	0.158

The precision is significantly lower, and the recall is a bit higher, leading to a worse F1 score overall. The figures included later will help to make sense of the relative precision and recall of Road GAN versus the SVM.

An interesting result of the precision and recall table is that the Road GAN scores for western Rhode Island during the summer of 2019 were actually very high, despite the leaves present. That scene actually had the highest F1 score of any of the test scenes. There are a few potential explanations for this. I think the most likely reason for the high F1 score is that I chose a relatively rural area for this scene so that there would be plenty of leaves present. However, since it was rural, there were fewer roads, and most of the roads in the scene were relatively large. This means that the classification task was probably easier to begin with when the leaves aren't considered.

Another reason that the model may have done comparatively well on the leafy scene is that it simply wasn't that good to begin with. There are plenty of unobscured roads that Road GAN couldn't identify, which means that the leaves may not be the biggest challenge it's facing. I think that once the network becomes more advanced the leaves may become a relevant issue.

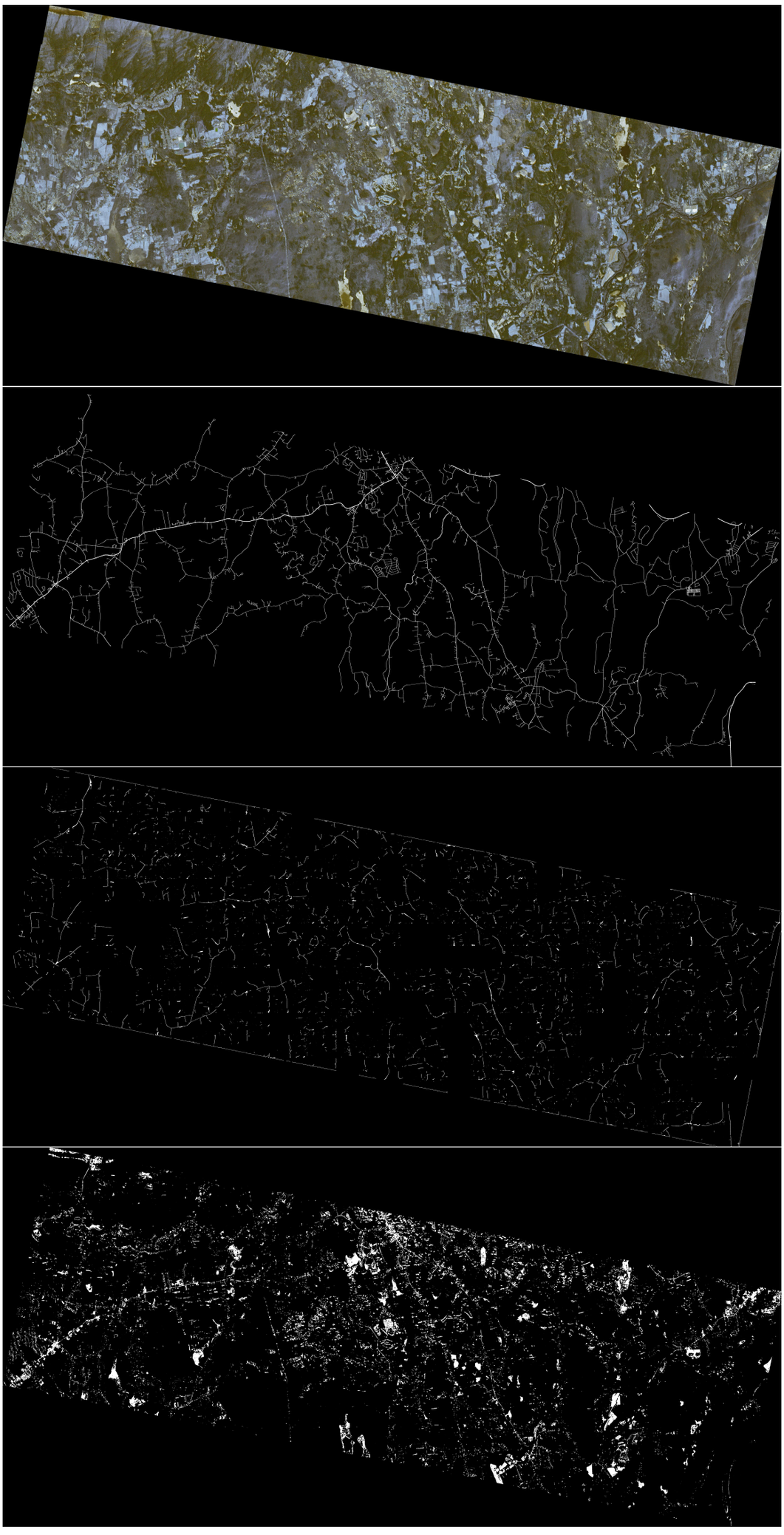

Figure 8: From top to bottom: Stretched and color adjusted Planet scene, ground truth road mask, Road GAN generated road mask, SVM generated road mask

The actual output of the GAN gives us a bit more information that the precision and recall scores do. The most noticeable difference between the ground truth and the generated output is the organization. The structure of the road network is very apparent in the ground truth image. In the Road GAN output, we can start to see a bit of this structure, but there is lots of noise in the background from spuriously detected roads.

Looking at the Planet scene itself, though, we can see that these noisy background roads might actually be more legitimate than they seem. About a third of the way across the scene, there is a large line in the North-South direction that looks like a road. It is slightly visible in the Road GAN output, but it does not appear on the ground truth. Structures like this that appear road-like but are actually railways or other objects may be responsible for a significant portion of Road GANs misidentified roads. During the training, I noticed that in many instances driveways would be identified as roads, but they wouldn't always be labeled in the ground truth.

If we consider the performance of the network in identifying road-like structures that a human might reasonably classify as a road, then it is probably significantly more accurate than the numbers imply. There is definitely room for improvement though. If railroads or driveways are things that consistently cause Road GAN to make mistakes, it is likely that in some future training cycle this would become apparent to the discriminator and the generator would be forced to improve.

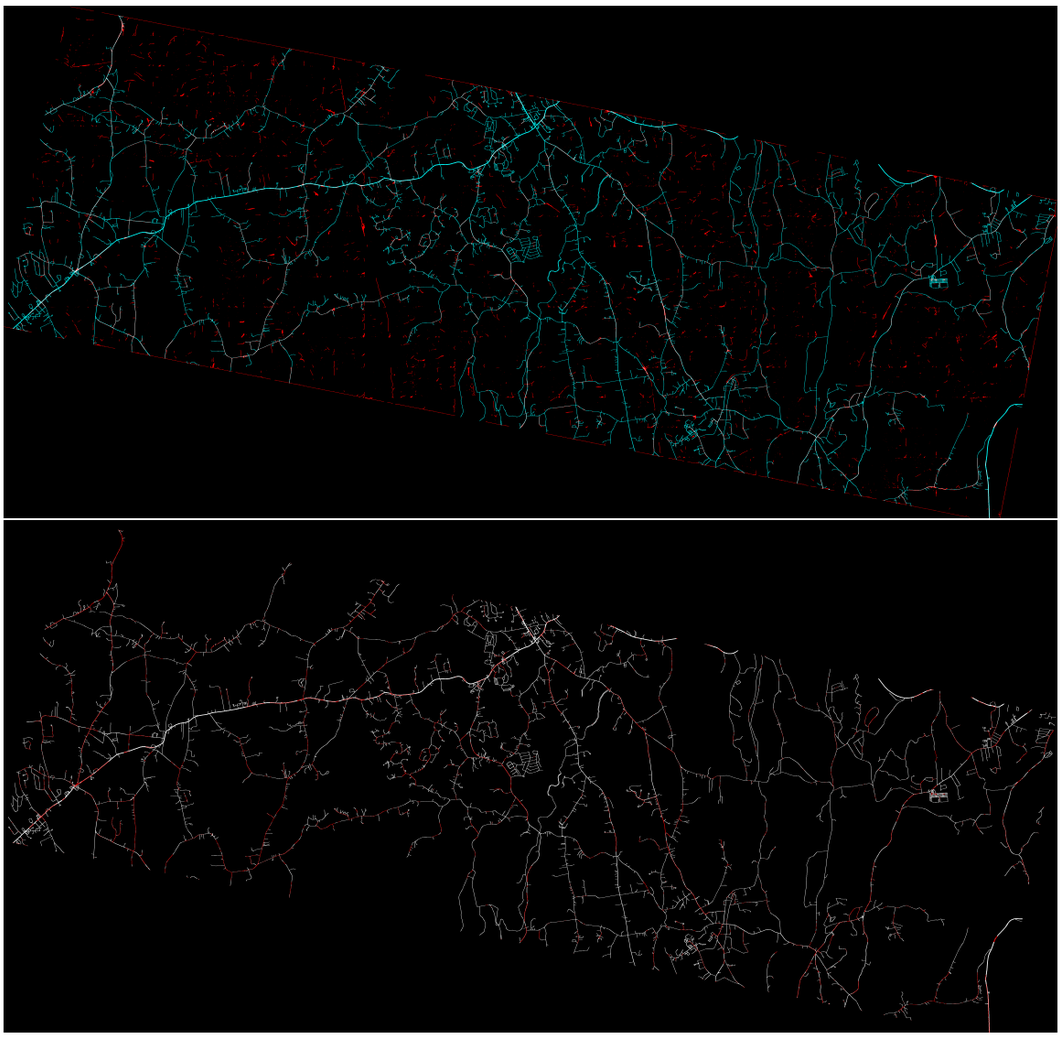

Figure 9: Above- The complete ground truth (teal) and Road GAN output (red) Below- The ground truth roads with the sections correctly labeled by Road GAN marked in red (a visualization of the recall)

The visualizations in Figure 8 highlight the difficulties in interpreting the Road GAN result. In the upper image, we clearly notice the differences between Road GAN and the ground truth, yet in the bottom we can see that there is also a lot of intersection between the two. Tying back to the precision and recall that the network achieved, it seems that Road GAN is quite good at finding roads, but about half the time these

roads turn out to be other structures (like railroads) that are similar in appearance to roads. According to the numbers, Road GAN does much worse with recall, and about 80% of real roads go undetected. Based on the recall map in Figure 8, it looks like the recall might be slightly higher if it was computed based on road segments instead of a pixel-by-pixel basis.

Visualizing the results of the SVM classification reinforces the notion that Road GAN's strength lies in its precision. The SVM should operate about as well as we can expect any pixel-wise classifier to, yet the ground truth with the SVM output overlain shows that it detects many more spurious structures than Road GAN does. It does achieve a slightly higher recall score than Road GAN, but the F1 scores show that this increase in recall is not as significant as the drop in precision. Over the entire scene the Road GAN model was 2.51x as precise as the SVM. This discrepancy in precision speaks to a more general deficit of pixel-wise classifiers, and highlights an import advantage of using convolutional models. While pixel-wise classifications such as thresholding an index may be able to achieve a similar recall to convolutional models, their weakness lies in not being able to discriminate between other objects with similar spectral properties, thus leading to a lower precision. In the case of the SVM trained here, it appears that parking lots or buildings may have been mistaken for roads.

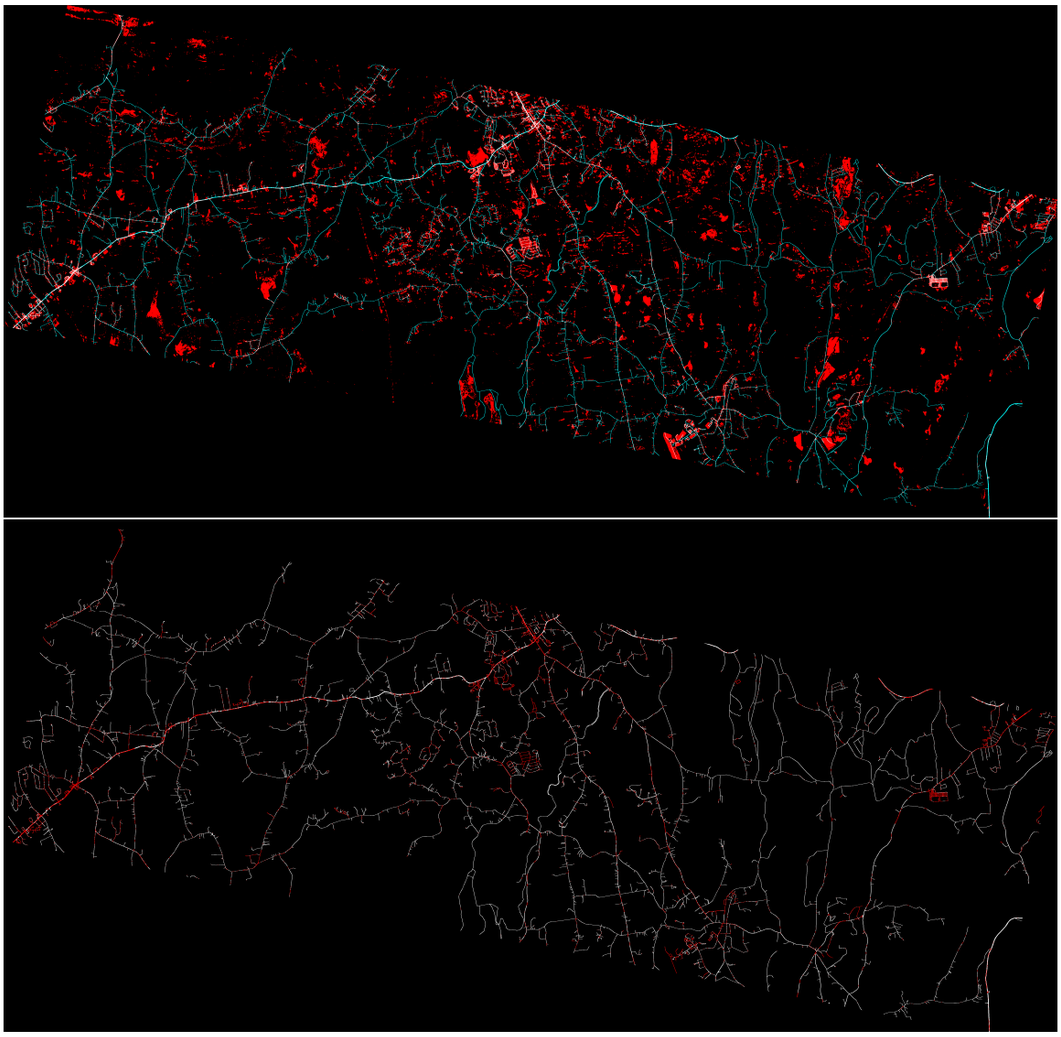

Figure 10: Above- The complete ground truth (teal) and SVM output (red) Below- The ground truth roads with the sections correctly labeled by the SVM marked in red (a visualization of the recall)

6 Conclusion

Despite it's shortcomings, Road GAN was a success. It showed steady improvement throughout the training, which gives me hope that it could become more successful with additional training. At the same time, it's current state is still impressive. Using labeled training examples, it learned how to extract the relevant

features that identify roads in remotely sensed images to a level that exceeds the capabilites of pixel-wise classification techniques. The SVM classifier did an excellent job of highlighting the strengths of Road GAN. By comparing the precision and recall of these two models, it was apparent that Road GAN's capacity to interpret structural features, rather than just spectral ones, allowed it to achieve greater precision.

Though the network proved to be successful, it posed a lot of challenges along the way, mostly in terms of data management. I had not attempted to train a network with remote sensing data prior to this, and the sheer volume of data caused significant problems. Running scripts to download roads, perform convolutions, calculate gradients or make classifications often ended up taking hours or days which is significantly longer than what I've experienced using other datasets. Google Colab proved to be an invaluable tool since it allowed me to save large image and training weight files in Google Drive, and it gave me free access to a GPU.

With more time, I would like to add additional classification techniques and compare the results with those of Road GAN and the SVM. I did try to use logistic regression on the built-up area index (BAI) to learn an optimal classification threshold for the index, but I think that my bands got ordered the wrong way in the training data, so the regression didn't work well[3]. Ideally, classification via BAI with a learned threshold would yield similar results to the SVM, but with a slightly poorer precision. This would support the notion that a pixelwise SVM classifier functions similar to an optimal index and threshold.

I also think that there is a lot of room for improvement for Road GAN with continued training and more diverse training examples. It would be interesting to further explore how the precision and recall are affected by different climates or seasons. Road GAN did pretty well when applied to a leafy scene, but I suspect that it would still benefit from incorporating leafy images into the training set.

Road GAN remains a work in progress, but it has demonstrated that convolutional networks have something to bring to the table when it comes to road network extraction, and, more generally, any landcover classification task.

7 Bibliography

1. Russell, Ethan. "9 Things to Know about Google's Maps Data: Beyond the Map — Google Cloud Blog." Google, Google, cloud.google.com/blog/products/maps-platform/9-things-know-about-googles-maps-data-beyond-map.

2. Herold, Martin. Understanding Spectral Characteristics of Asphalt Roads. National Center for Remote Sensing in Transportation, May 2004, pdfs.semanticscholar.org/0170/7b3069c50bfa0852b8b8da09c1deef52e911.pdf.

3. Shahi, Kaveh, et al. A Novel Spectral Index to Automatically Extract Road Networks from WorldView-2 Satellite Imagery. The Egyptian Journal of Remote Sensing and Space Science, 2015, A Novel Spectral Index to Automatically Extract Road Networks from WorldView-2 Satellite Imagery, www.sciencedirect.com/science/article/pii/S111098231400043X.

4. Isola, Phillip, et al. "Image-to-Image Translation with Conditional Adversarial Networks." ArXiv.org, 26 Nov. 2018, arxiv.org/abs/1611.07004.

5. Kearney, Sean P., et al. "Maintaining Accurate, Current, Rural Road Network Data: An Extraction and Updating Routine Using RapidEye, Participatory GIS and Deep Learning." International Journal of Applied Earth Observation and Geoinformation, Elsevier, 17 Dec. 2019, www.sciencedirect.com/science/article/pii/S0303243419309882.