Deep Green Space: Quantifying Urban Greenery with Deep Learning

Jiuying Han, Michael Young, Andrew Campbell

Introduction

As urban areas are growing both in area and density, urban planners face a logistical challenge of identifying and categorizing the urban landscape. Even though street level images exist for most cities across the world, it is still a human-intensive activity to generate categorized city maps. Here deep learning can provide an automated approach to labeling the urban landscape.

One specific problem urban planners face is identifying the green space within an environment. Green space is associated with positive psychological benefits and is one factor in the "quality" of the urban landscape. Here, we propose using deep learning to help identify green spaces by observing street view images. Deep learning is a good tool since there is an abundance of street view data for urban environments. Once effectively trained, deep learning models require very little human hours to generate categorized urban maps. This kind of learning is feasible since identifying green spaces can be accomplished using semantic segmentation and has been demonstrated with architectures like PSPnet and DeepLabv3.

One way to measure urban green space is the Green View Index (GVI). The GVI is a measure which takes the number of pixels in an image belonging to greenery and divides that by the total number of pixels in the image. This essentially yields a green space percentage. The GVI for a given point in space can be estimated by considering the 360 degree view from that point either via a panorama or through a series of images which cover the whole space.

To estimate the urban green space for Salt Lake City, we trained a Fully Connected Convolutional Network (FCN), PSPNet and DeepLabv3 on labeled city images and used these models to calculate the GVI around the city. The methods we used to train and evaluate our models, along with the results of our efforts are detailed below.

Data

The primary dataset we used to train our networks was the Cityscapes dataset [1]. The Cityscapes dataset consists of 5,000 images of urban scenes with 30 classes annotated. The training set consists of 2975 images, the validation set 500, and the test set 1525. For our purposes, we were only interested in classifying 2 of the 30 labeled classes: vegetation (trees, bushes, etc.) and terrain (grass).

The data we used to estimate the GVI of SLC was google street view (GSV) data obtained through the google street view static API [2]. We sampled points every 100 meters and used images from 6 different angles to capture the 360 degree span of each point. This yielded a total of 57,132 images to analyze. Additionally, we hand-labeled 150 of these images as a small training set and a further 60 as a test set to use for evaluation.

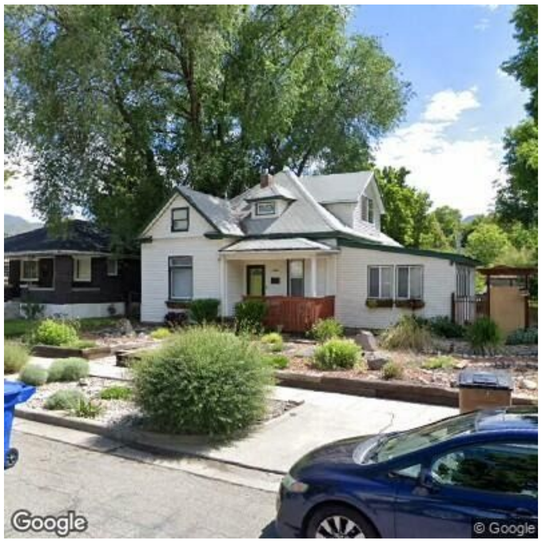


Figure 1: An example GSV image of SLC

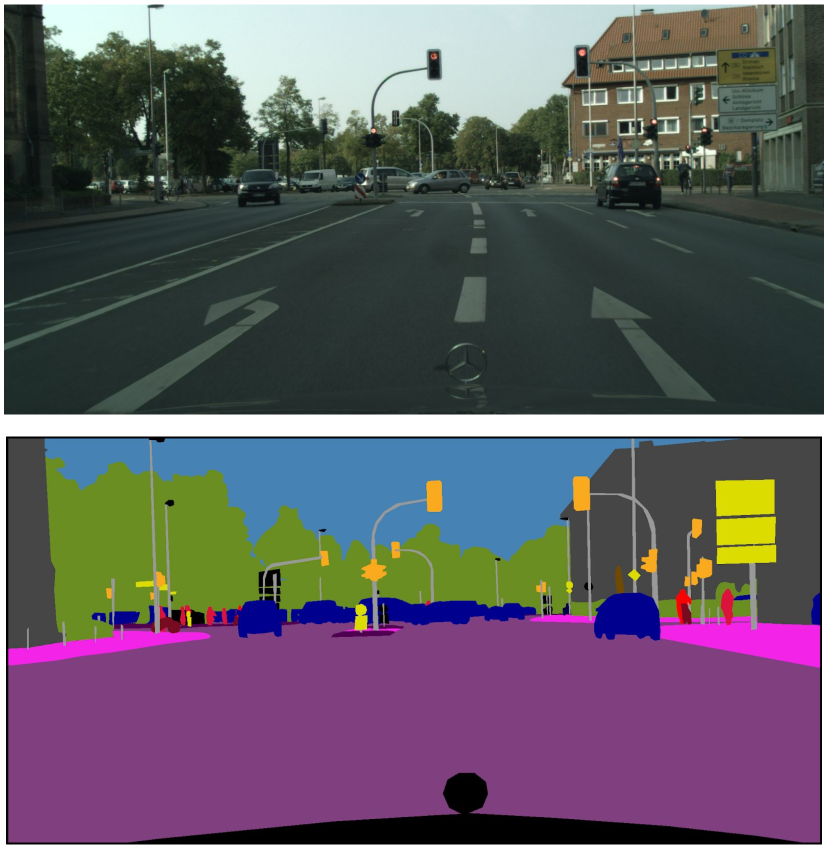


Figure 2: An example image along with its semantic labeling from the cityscapes dataset.

Network architectures

PSPNet

PSPNet was developed by Zhao et.al. in 2017. The structure of PSPNet is represented by the following graph.Given an input image (a), CNN is used at first to get the feature map of the last convolutional layer (b), then a pyramid parsing module is applied to harvest different sub-region representations, followed by upsampling and concatenation layers to form the final feature representation, which carries both local and global context information in (c). Finally, the representation is fed into a convolution layer to get the final per-pixel prediction (d).

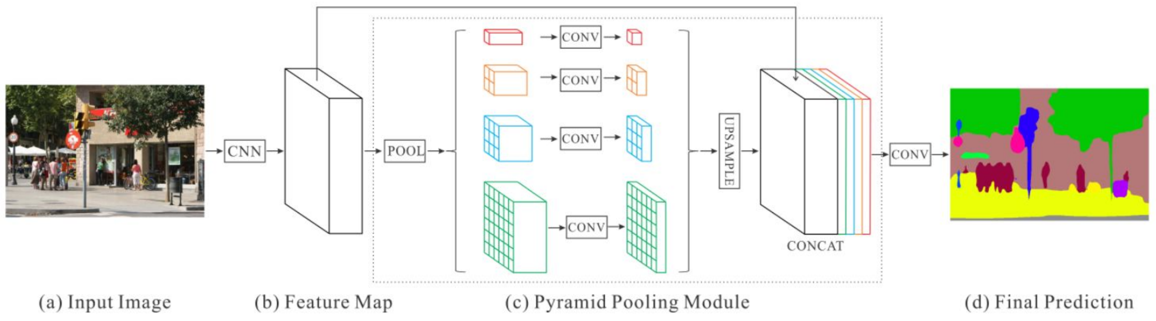


Figure 3: PSPNet architecture

DeepLabV3

DeepLabV3, developed by Chen et. al. in 2017 [4], aims at improving upon a fully connected convolutional network (FCN) for semantic segmentation tasks by introducing two innovations: 1) atrous (or dilated) convolutions and 2) atrous spatial pyramid pooling (ASPP). In a typical FCN for semantic segmentation, the input is first downsampled to improve computation and enlarge the receptive field size of filters. Following this, the features are upsampled (using transpose convolutions) in order to make predictions on each pixel in the input image.

One of the disadvantages to excessive downsampling for semantic segmentation is that the spatial information of the input is muddied. In order to preserve this spatial information, atrous convolutions are introduced. Atrous convolutions (picture “swiss cheese convolutions”) are a way to capture a wider field of view of features without driving up computation or parameters.

Another issue with a vanilla FCN is their inability to capture features at different scales of the input. The ASPP used in DeepLabV3 tackles this issue by first applying atrous filters of different sizes (to capture different scales) to the feature map of the CNN backbone followed by combining these outputs together and upsampling to the desired output size. They also incorporate the image-level features by applying global average pooling to the last layer of the CNN backbone and combining that with the aforementioned atrous outputs.

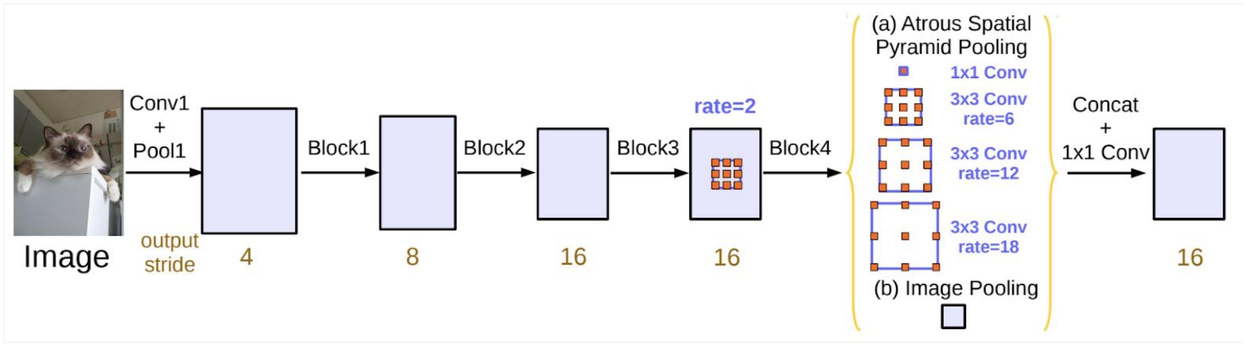


Figure 4: Depiction of the DeepLabV3 network w/ atrous convolution and the ASPP w/ image-level feature augmentation.

For this project, we used PyTorch's implementation of DeepLabV3 with a ResNet50 backbone (https://pytorch.org/docs/stable/torchvision/models.html#semantic-segmentation). We also took advantage of transfer learning by beginning our training using the pre-trained weights from the COCO 2017 training set.

FCN-ResNet101

FCN-ResNet101 is a fully convolutional network model with a ResNet101 backbone. Fully Convolutional neural networks differ from traditional convolution networks by accepting arbitrary sized input and producing the same sized output. In this project we used Pytorch's implementation of a FCN with ResNet101 as a backbone. Pytorch has a pre-trained FCN-ResNet101 on the COCO 2017 training set. However, we used the untrained model and attempted to train it on the cityscapes dataset. This architecture served as the baseline accuracy for our other more sophisticated network models.

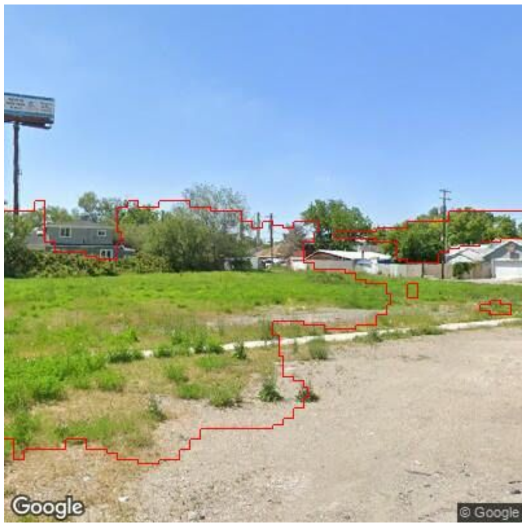
Results

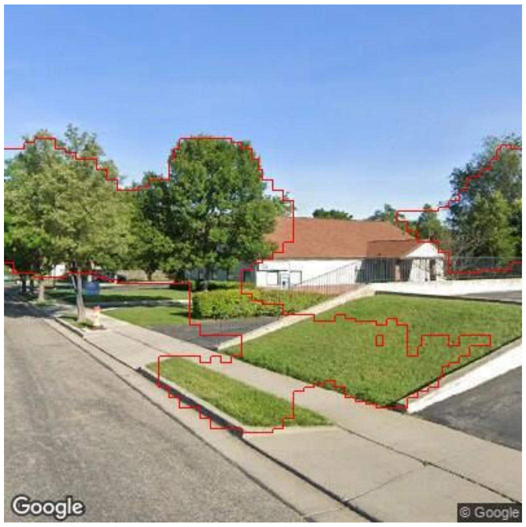
FCN-ResNet101

We trained the Pytorch implementation of FCN-ResNet101 using Adam and CrossEntropyLoss. The model was only trained on the "terrain" and "vegetation" labels for the cityscapes dataset. Additionally, the model was fed the "fine" annotated data from cityscapes. After 50 epochs the model had a validation pixel accuracy of .5497 and an mIoU of .366. Due to the long training times of the ResNet101 backbone and the poor accuracy and mIoU, we discontinued using the model for validation on the Salt Lake City dataset. For comparison the pre-trained FCN-ResNet101 has pixel accuracy of .919 and mIoU of .637.

PSPNet

We used Resnet50 as a backbone to train the PSPNet. After 100 epochs, the miou of PSPNet is 0.59. Some examples of the predicted green spaces of Google Street View images are shown as follows:





The model did a better job in predicting trees than grasses. The potential reasons for the failure of this model could be: 1. Resnet50 is not complicated enough as the backbone of this model; 2. We used batch size of 8 for the training, a larger batch size might perform better; 3. Most of the images (i.e., 2975) we used for training are from the Cityscapes dataset, only a small part of the training dataset (i.e., 150) are from Google Street View images; 4. We only trained 100 epochs, which might not be enough for the segmentation project.

We did not use PSPNet for the calculation of Green View Index due to its bad performance, and use DeeplabV3 instead.

DeepLabV3

Initially, we trained DeepLabV3 on the cityscapes dataset to classify trees. We ran training for 7 epochs (each epoch taking ~3 hours!) and the results are shown below.

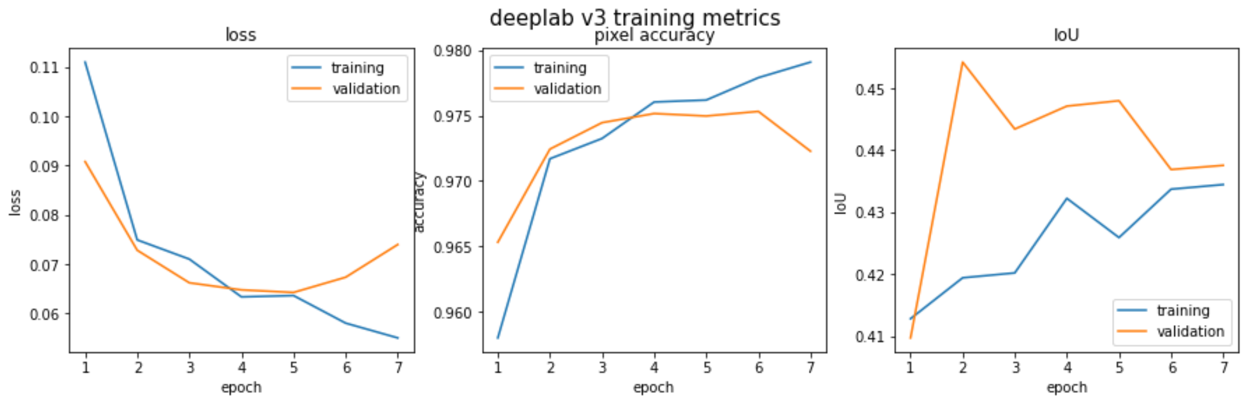


Figure 5: Training and validation metrics from DeepLabV3 trained on cityscapes to recognize trees.

The mean jaccard index (mIoU) values are not correct (there was a bug in the code) but the general trends are accurate. As seen in both metrics (Figure 5) and the model prediction capabilities (Figure 6), training on the cityscapes data proved relatively effective for our task.

Upon further thought, however, we decided that we wanted to incorporate grass into our definition of "green space", so we took our best performing tree model and retrained on both the "vegetation" class (trees, shrubbery) and the "terrain" class (grass). Due to computational limitations, we only ran for 4 epochs.

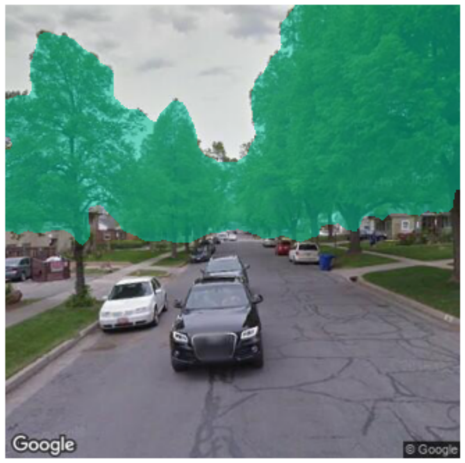


Figure 6: A prediction on GSV SLC data with the DeepLabV3 model trained on trees.

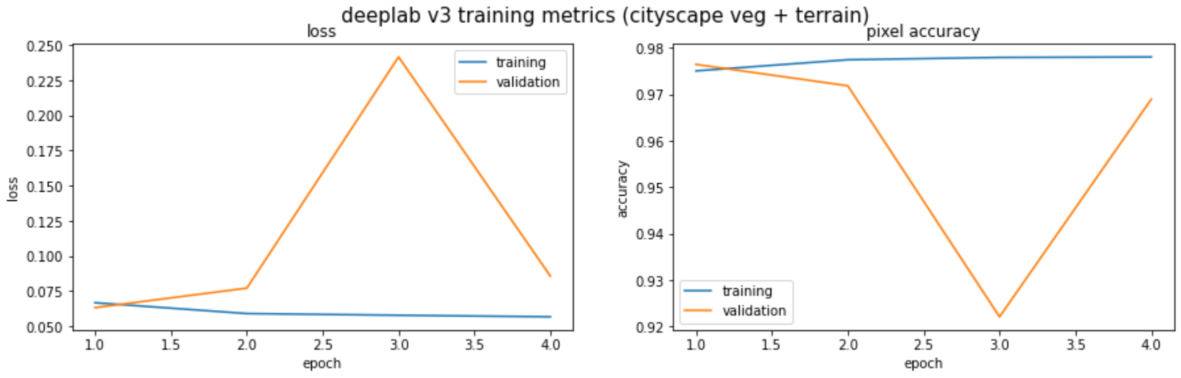


Figure 7: Training and validation metrics using a model trained on both vegetation and terrain.

Through testing this model on some images from our GSV SLC data, we noticed that it wasn't performing too well. It had a hard time classifying greenery through chain link fences and classifying grass in general. This could be due to a lack of abundant grass in the cityscapes set. We determined that in order to get the best performance possible with our model, we would need to train on some hand labeled images of our GSV SLC data.

After hand labeling 150 GSV SLC images for training and another 60 for testing, we took our model previously trained on the cityscapes set and trained on the SLC data for 25 epochs. Our results are shown below.

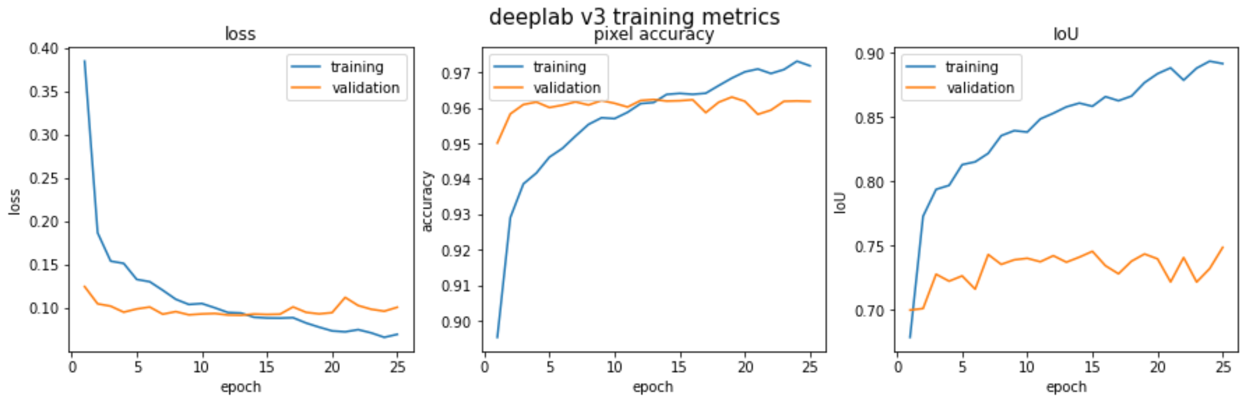


Figure 8: DeepLabV3 training and test metrics on GSV SLC data.

Our model was able to achieve a pixel accuracy of ***96.1%*** and an mIoU of ***74.8%*** on the test set (note that we didn't actually perform any hyper-parameter tuning, so we didn't have a validation set despite the labels in the figure).

In assessing the relative accuracy of the GVI calculations, we found that the mean difference between the GVI calculated between our model's predictions and the ground truth from the test set was ***1.42 +/- 1.45%***, which we deemed a success.

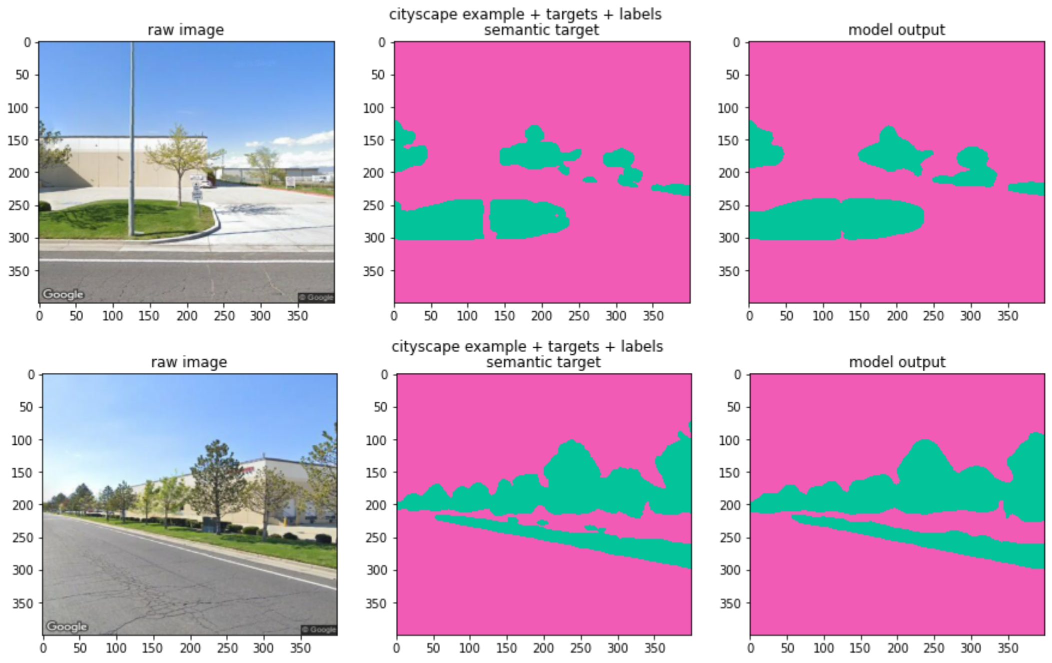


Figure 9: 2 examples from the SLC test set juxtaposing ground truth labels with the model predictions.

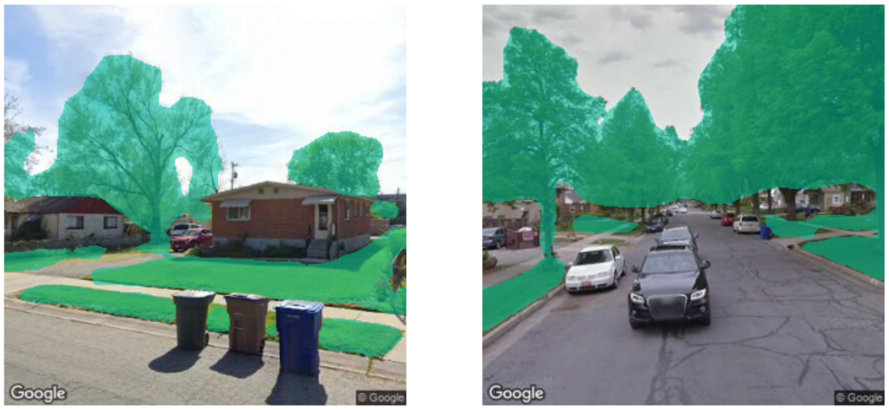


Figure 10: Successful prediction examples.

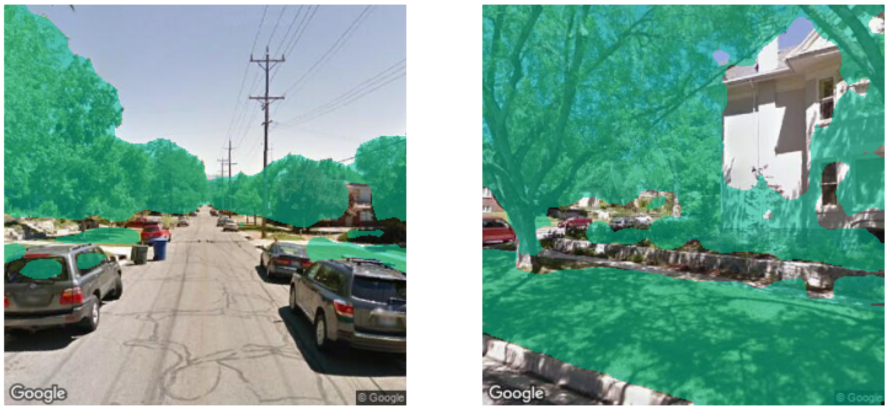


Figure 11: Unsuccessful prediction examples. Note the incorrect predictions of car windows (left) and the shadows of leaves (right).

We used our model to compute the GVI for each point in our SLC sampling. The following map shows the GVI distribution among Census Block Groups in Salt Lake City.

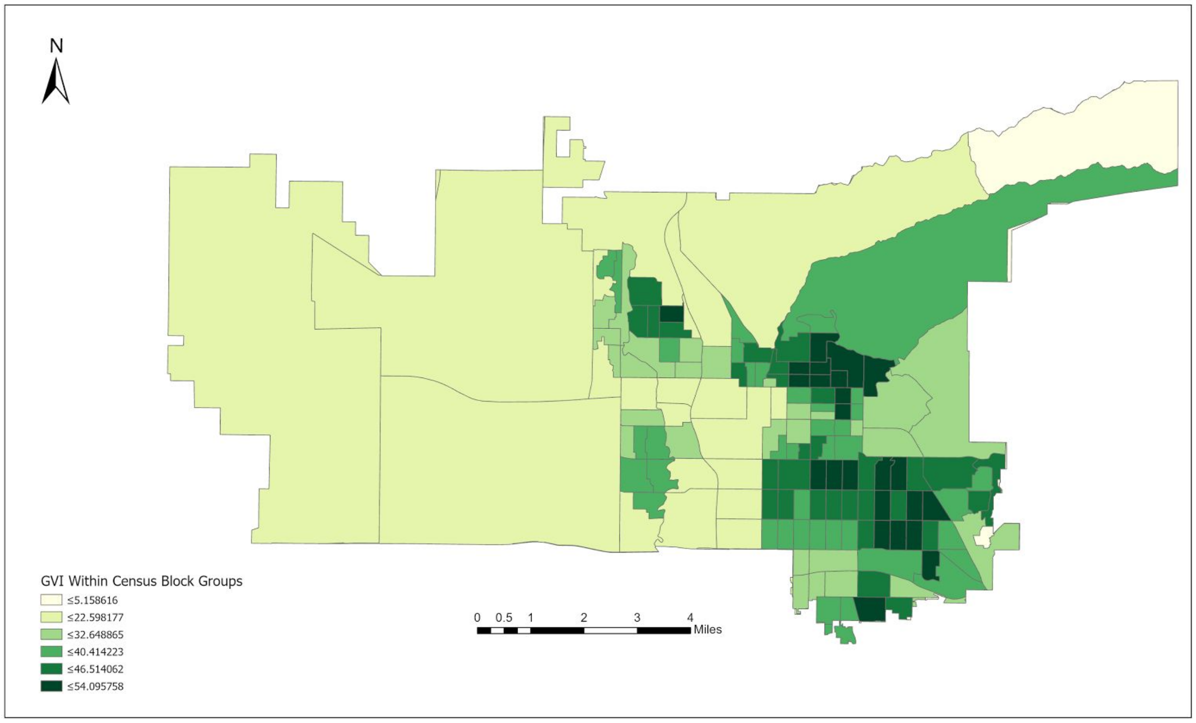


Figure 12: The average GVI for each census block in SLC.

Discussion

As discussed briefly in the results section, one of the biggest challenges we faced with this project was adapting our models trained on the cityscapes data to accurately segment our GSV data. For semantic segmentation projects (and indeed any machine learning task), a good training dataset is essential for successful learning. Diversity in the training set will improve the generalizability of the model. In this project, almost all of the training data from cityscapes focused on streets of urban places with no large areas of grass and dirt. In our GSV training and test images, however, we have many images which have non-urban scenes with large areas of lawn and barren land. It was encouraging to see that by using a very small training set (150 images) of our actual GSV data, we were able to boost the model performance to an acceptable level. However, our model was by no means perfect (as demonstrated in figure 11), and in order to improve performance further, more labeled GSV data from SLC would almost certainly help.

Choosing a suitable pretrained backbone model is important for getting a good model performance. For both of our models, we used ResNet50 as our backbone. We could have potentially achieved better performance by using ResNet101, or even ResNext, which is the current state of the art as a backbone in segmentation tasks. The main issue with using deeper backbones is of course computational expense, so any use of these models would require access to more compute than what Google Colab can usually provide. Additionally, the resolution differences between the cityscapes dataset and our GSV data might also have affected training, although it's difficult to say to what degree.

As far as calculating the GVI properly, in this project, we used 6 images heading 0, 60, 120, 180, 240, 300 degrees per sample point to represent the feeling of the green view for individuals. This method might not be perfect enough to reflect personal feelings, as it only captured the surrounding area at one vertical angle. To further capture the green view of a given point, we could sample additional images with angles titled up and down. Moreover, the importance of tree canopy and grasses could be different in the perception of green space. It's possible that just using tree canopy as an indicator for green space may align more with people's perceptions of green space and the corresponding psychological benefits obtained from regularly having access to such green spaces.

Future Directions

This project could be extended in various ways. As discussed, using deeper CNN backbones as well as using more labeled training data from GSV images would undoubtedly benefit the model performance. Additionally, we could perhaps capture a better measure of the GVI by including more vertical angles to capture a wider field of view.

More ways to extend this project involve comparing our calculated GVI with other demographic measures such as income, health, and crime. It would be interesting to see if the GVI is strongly correlated with any of these measures.

Finally, we could create a web-based, interactive map displaying our GVI data along with other demographic information to provide a way for visual engagement with and exploration of this data.

References

(1) https://www.cityscapes-dataset.com/
(2) https://developers.google.com/maps/documentation/streetview/overview
(3) Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2881-2890).
(4) Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv:1706.05587 (2017)