Proposal for Modeling Approach

Munawar Ali, Çağatay Ayhan, Ece Karaçam, Abdullah Malik, and Mao Nishino

April 6, 2024

1 Preliminary

In this section, we will discuss the pix2pix model, which is a type of Generative Adversarial Network (GAN) used for image-to-image translation tasks. The pix2pix model works by training on pairs of images (input and output) and learning to generate the output image from the input image. In our use case, we will use the pix2pix model to generate a crime heatmap given a geographical map. This approach will allow us to even generate hypothetical crime heatmaps based on future urban planning designs.

2 Methodology

2.1 Collection of geographical maps

We will collect geographical maps of Tallahassee, FL, which will serve as the input to our pix2pix model. These maps will include information such as roads, buildings, parks, and other geographical features. We will use `osmnx`, a Python library, to download OpenStreetMap data for the area and use `matplotlib` to visualize the maps. In particular, we first plot the entire map of Tallahassee as a 16384x16384 image, and then we divide it into 256x256 patches to train the pix2pix model.

2.2 Collection of crime data

We will collect crime data from the Tallahassee Police Statistics, which includes information such as the time and date of the crime, the location of the crime, the type of the report, and the geographical coordinates of the crime. We will then plot the crime data, by using the `scatter` feature in `matplotlib`, on the 16384x16384 geographical map from the previous section, but with all components made invisible. This allows us to align the crime data with the geographical map of Tallahassee to create the output images for the pix2pix model. Similar to the geographical maps, we will divide the crime heatmap into 256x256 patches.

2.3 Training the pix2pix model

We will train the pix2pix model on the pairs of input (geographical maps) and output (crime heatmaps) images. The model will learn to generate crime heatmaps from geographical maps. We will use an available library implemented in `PyTorch` and train it on a GPU for faster convergence.

3 Evaluation

To evaluate the performance of our model, we will use mean squared error (MSE) as the loss function. We will calculate the MSE between the predicted crime heatmap and the actual crime heatmap. As a baseline, we will also calculate the MSE between the actual crime heatmap and a random noise heatmap. The baseline map will be generated by the same way as the crime heatmap, but with random points instead of actual crime data. The number of points will set to be equal to the actual data. We will compare the MSE of our model with the baseline to assess the model's performance.