Lab1 Report

Shiyu Feng, Ep Pravitra, and Marcus Pereira

Due: Mar 13th 3017

1 Gradient Descent on Squared Loss

1.1 Algorithm

We define the loss function to be

$$l_i(t) = ||W_t f_i - y_i||^2 \tag{1}$$

where $f_i \in \mathbb{R}^{10}$ is a feature vector given by the problem. $W \in \mathbb{R}^{5x10}$ is the weight matrix that we seek to learn. $y_i \in \mathbb{R}^5$ is an indicator vector corresponds to each label. We remap the labels to be.

- Veg : $y = [\ 1\ 0\ 0\ 0\ 0\ ]^T$
- Wire :$y = [\ 0\ 1\ 0\ 0\ 0\ ]^T$
- Pole : $y = [\ 0\ 0\ 1\ 0\ 0\ ]^T$
- Ground : $y = [\ 0\ 0\ 0\ 1\ 0\ ]^T$
- Facade : $y = [\ 0\ 0\ 0\ 0\ 1\ ]^T$

We use the update law.

$$W_{t+1} = W_t - \alpha \frac{dl}{dW} \tag{2}$$

where

$$\frac{dl}{dW} = 2(W_t f_i - y_i) f_i \tag{3}$$

We choose the learning rate $\alpha = \frac{1}{\sqrt{T}}$. Our algorithm reshuffles the given data and pass them through the learning algorithm several times.

1.2 Results

We implement the result using C++ and Qt framework. We use OpenGL for visualization. The result of classification after learning both datasets are shown in Fig. **??** and Fig. **??**. Here we show classification after 10 passes of both datasets. However, we discovered that passing datasets twice is already sufficient to obtain a consistent W. There is very little difference in outcome between 2 and 10 passes.

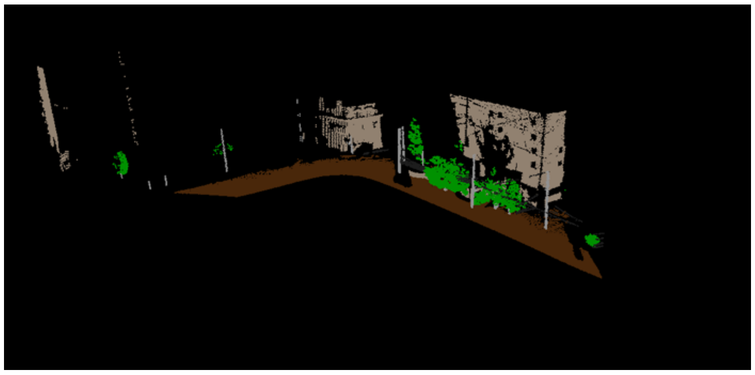
Figure 1: given Dataset 1

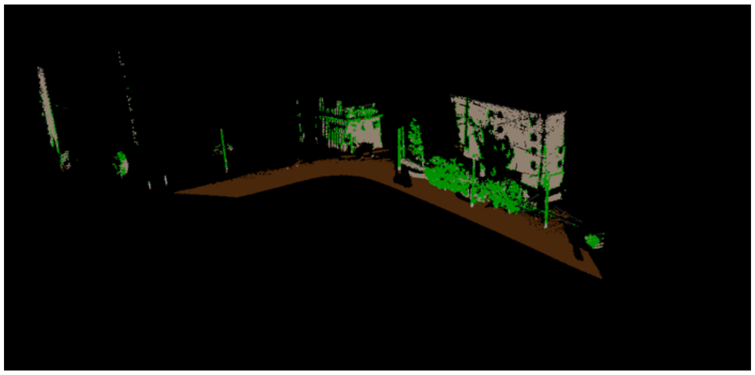
Figure 2: classified Dataset 1

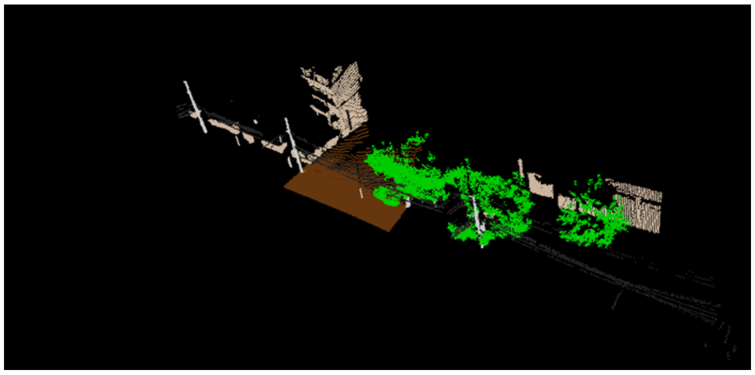
Figure 3: given Dataset 2

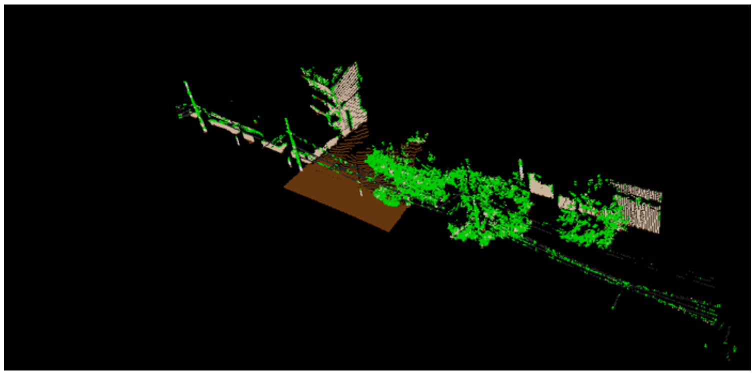
Figure 4: classified Dataset 2

Fig. **??** shows another test case. In this case, we learn with Dataset 1 and use W to classify Dataset 2.

Wires and poles do not get classified very well for many potential reasons. One reason is they are consideraly "thinner features". Poles and wires have very little feature content in transverse direction. Other features are easier to classify because they tend to be clumped together. Size, density, and color information of clumped features are easier to extract. We also note that we have much less samples of poles and wires in the training set. This is especially true in our second test case where we learn using Dataset 1 (which do not have many wires) and attempt classify Dataset 2. Fig. **??** shows that wires are classified very poorly.

As often the case, choice of implemention is a direct tradeoff between implementation difficulty and CPU time. Our choice of C++ makes the learning and classification very fast. However, some implementation such as matrix multiplication is not trivial. Overall the algorithm is not difficult to implement.

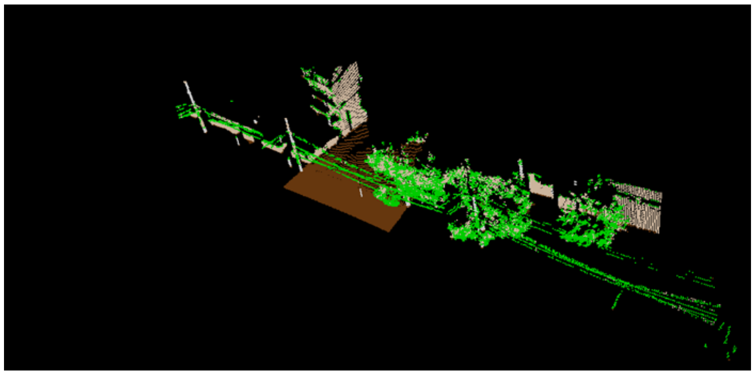

Figure 5: classification of Dataset 2 by learning Dataset 1

To evaluate robustness of the learned data, we apply 1σ noise into Dataset 2. The classification result is shown in Fig **??**. Despite many misclassifications, facade and ground are still mostly correctly classified .

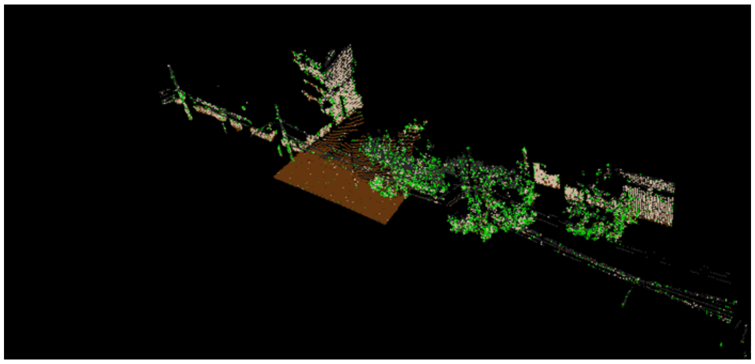

Figure 6: classification results after noisy Dataset 2