Urban Scene Segmentation for Autonomous Vehicles

Team: Hsiao-Chen Huang, Eddie Tseng, Ping-Chun Chiang, Chih-Yen Lin

Department of Electrical and Computer Engineering, ECE 228, University of California, San Diego, La Jolla, CA 92093, USA

Background & Motivation

With the rapid developing and the continuing evolution of autonomous vehicles, people begin to increase its reliability. To guarantee the safety of autonomous vehicles, we need to make AVs classify objects as quick as possible (e.g. 24 frames per second). Hence, we decided to solve semantic segmentation problem to localize objects in the images using Fully Convolutional Networks (FCNs) [1] and evaluate the advantages of different networks.

Dataset

We use Cityscapes Dataset [2] to train our network, and the dataset contains 19998 images which consist of street scenes of 50 cities and corresponding annotation images. The annotations represent 30 classes in different RGB color for each pixel. However, we just choose 19 classes in our task, and an example is shown in Figure 1.

Methods

Fully Convolutional Network can efficiently learn to make dense prediction in pixel-wise classification task.
- Using 1x1 convolutions to transfer feature map to pixel-to-pixel prediction.
- Deconvolutional layers are used to capture different level of shape details and expand the prediction size to input size.
- Skip connection allows lower level information to reach top level by adding deconvolutional layers to the previous layer.

The main goal is to design a network as a bunch of convolutional layers with downsampling and upsampling. The architecture of AlexNet and VGG net with skip connection are shown in Figure 2 and Figure 3.

Results

The dataset contains 18,000 training and 1,998 testing images. AlexNets were trained in 13k steps and VGG was trained in 93k steps (skip) vs 41k steps (non-skip).

Discussion

Skip Connection:
The accuracy of the model with skip connection should be higher and improve the segmentation detail because of fusing information we lose during pooling operation. AlexNet vs. VGG net:
VGG is similar to AlexNet, but more filters. Thus, VGG can extract higher features. That is why it is currently the most popular model, deep and simple.

Batch Size:
Due to the limited GPU and RAM size, we trained mini-batch gradient descent with batch size equal to 5 for AlexNet and 2 for VGG. Therefore, this is the reason for large fluctuations.

Learning Rate:
In this project we use learning rate = 0.001. For the future work, we will tune learning rate to optimize the training speed.

As the prediction result shown in Figure 4, green indicates road, blue indicates cars, etc.

Feature:

Input: \((x_1, x_2, \ldots, x_N) \in X\) (1)

Annotation:
\[\{(y_1, y_2, \ldots, y_N) \in Y\} \quad \forall N \in \mathbb{N} \] (2)

Sigmoid activation function:
Use sigmoid function (eq 3) as our activation function so that all the outputs will lie between 0 and 1.

\[ S(z) = \frac{1}{1 + e^{-z}} \] (3)

Cross-entropy loss function:
Evaluate cross-entropy as loss function (eq 4).

\[ L(y, \hat{y}) = -\sum_{i} y_i \log(\hat{y}_i) \] (4)

Future Work

- Real-time segmentation
- Implement different model (RNN, LSTM) [5]
- Try more classes or use different dataset.
- Calculate class mean accuracy which consider the performance of each object.
- Tuning parameters.

Reference

4. VGG model.