MACHINE LEARNING ENGINEER NANODEGREE

Capstone Project Report

Semantic segmentation on Cityscapes

Sharad Kakran

I. Definition

Project Overview

The project is to segment all the objects present in the images into categories (classes). Concretely, each pixel in the image is classified to one of the 20 classes present in the data. The task of semantic segmentation is also known as dense prediction task, as compared to image classification where the whole image is classified to one of the categories, here each pixel has to be classified. Formally this problem is known as semantic segmentation.

In contrast to an image classification problem where image is classified into one of the classes, here every pixel has to be classified, therefore it is also called a dense prediction task. This very reason also makes it quite a challenging task.

In order to accomplish the task of scene segmentation effectively, we need to distinguish some confusing categories and take into account objects with different appearance. For example, regions of ’field’ and ’grass’ are often indistinguishable, and the objects of ’cars’ may often be affected by scales, occlusion and illumination. Therefore, it is necessary to enhance the discriminative ability of feature representations for pixel-level recognition.

We use convolutional neural networks (CNN) to solve this task. CNNs have showcased very good performance on high dimensional data, they scale well with large data compared to other traditional algorithms. They consist of many layers whose work is to extract features from images, modern CNNs could be very deep and consist of 100s of layers. Depending upon the task, their architecture may differ, like the architecture we used to solve this task is different to the ones used for image classification or object detection.

The dataset used for this task is Cityscapes data, which is an urban scene understanding dataset and could be used for various tasks like pixel-level, instance-level and panoptic level semantic labelling.

There are 20 classes including background and images are divided in 3 categories (train/val/test). These are high quality images and for each image there is a segmented map which should be used as ground truth.

Note: Test data labels are not available, therefore evaluation is done on validation set in this project.

Problem Statement

In this project, given the image from cityscapes dataset, we aim to classify each pixel of the image. Meaning each pixel should be labelled with one out of 19 classes, because the dataset has 19 classes. To achieve this, we use CNN, in particular Unet network. The output of this network will be an image suppose of dimension (512,512), every pixel in this image will have a pixel value from 0 to 18. Each of these value correspond to a class in our dataset, for example
0:'car', 1:'sky', 2:'road' …..

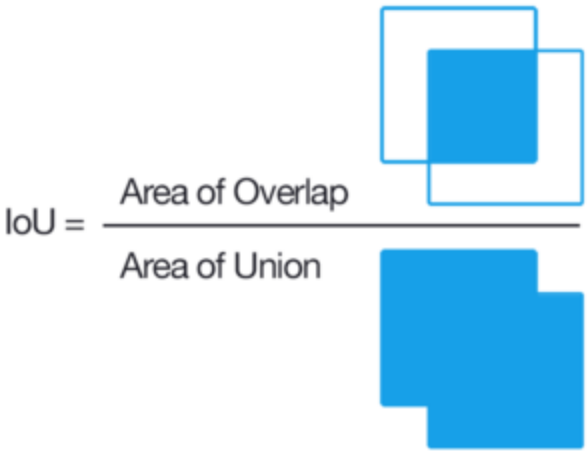
The following are the tasks to be carried out

- Data exploration and understanding
- Data preparation for training
- Implementation of CNN architecture
- Implementation of respective loss, scheduler and metric
- Training and evaluation

Metrics

I am using both pixel accuracy and mean Intersection over union (mIoU) as metrics for the project, but mIoU is the main metric for evaluation as pixel accuracy does not give the true result if a few classes are dominant in the data.

mIoU is calculated by taking the intersection between two images and dividing by union of those two images. Since there could be multiple classes present in these two images, we do this calculation for each class and in the end take mean over calculated IoU for all classes.



II. Analysis

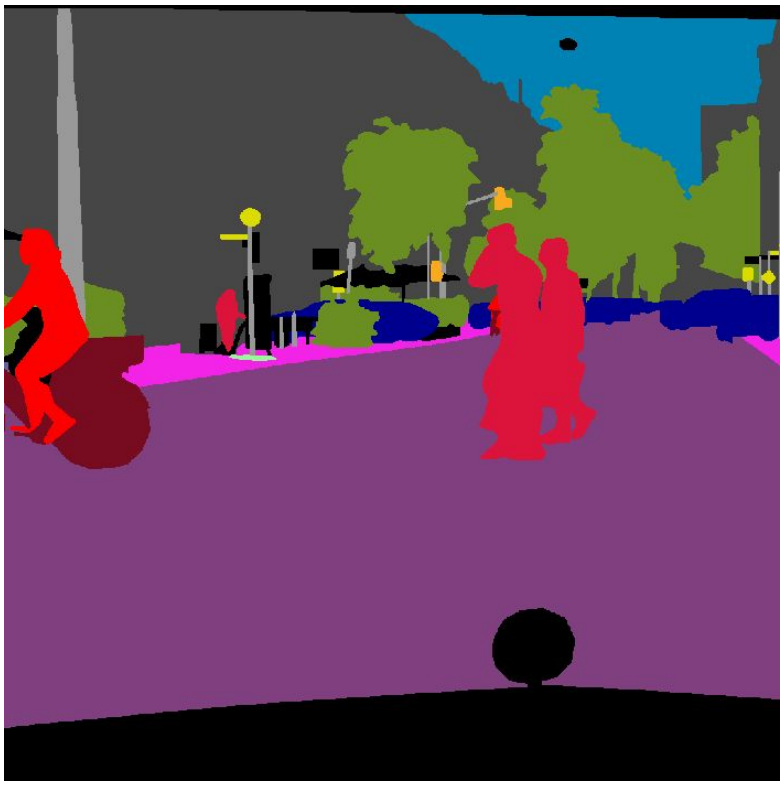
Data Exploration

As mentioned above the dataset is Cityscapes dataset, it has 5,000 images captured from 50 different cities. Each image has 2048 × 1024 pixels, which have high quality pixel-level labels of 19 semantic classes (excluding background). There are 2,979 images in the training set, 500 images in the validation set and 1,525 images in the test set. There are coarse images present as well but we do not use coarse data in our project.

The classes present in the dataset are:

'road', 'sidewalk', 'building', 'wall', 'fence', 'pole', 'traffic light', 'traffic sign', 'vegetation', 'terrain', 'sky', 'person', 'rider', 'car', 'truck', 'bus', 'train', 'motorcycle', 'bicycle'.

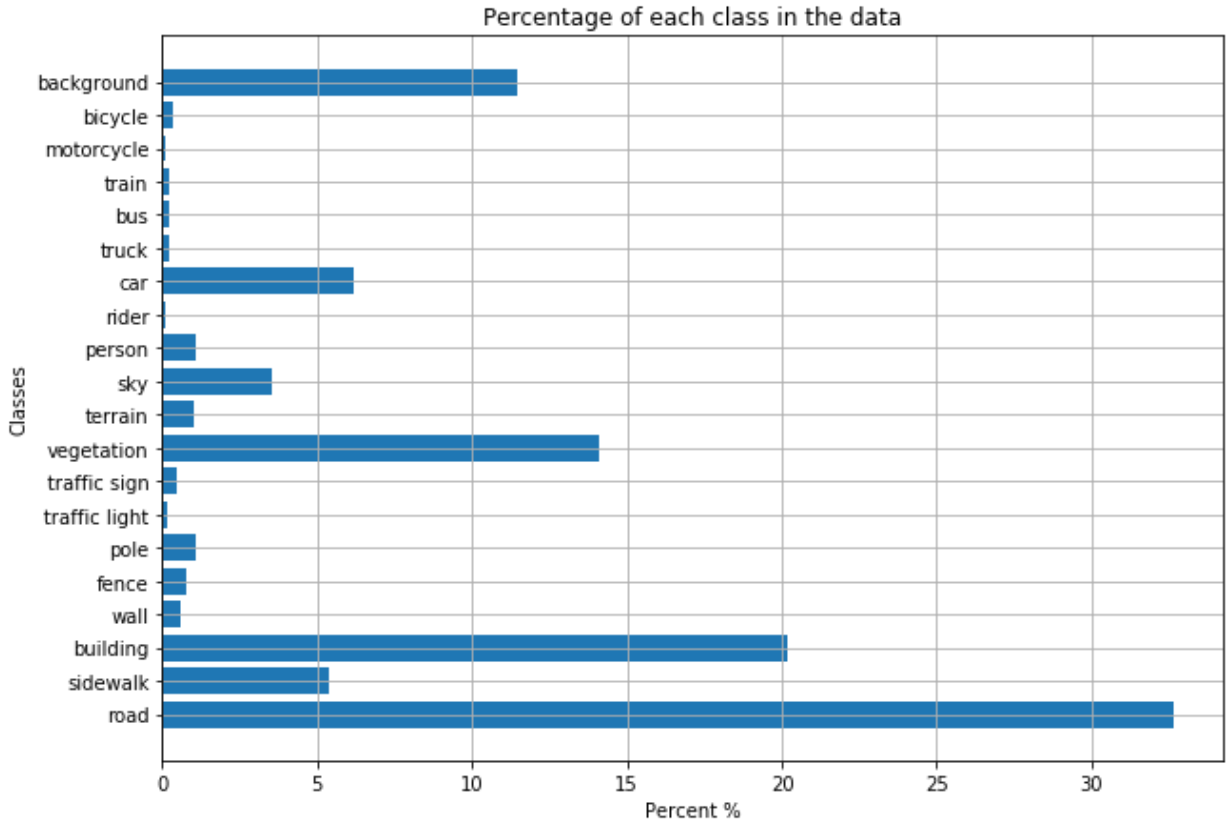
Anything which is not part of these classes is considered background. Therefore it makes up to 20 classes.



Exploratory Visualization

It is very important to check the class distribution of the dataset first.

This is what the class distribution of the train set looks like.



As it could be seen from the plot above that a few classes are more prevalent in the data such as “road, building, car, vegetation”. So that these classes don’t have more influence on predictions we use class weights to limit their influence.

Algorithms and Techniques

For this project, we are using the Unet network for segmentation. Unet is encoder-decoder architecture and is particularly good for segmentation due to its design. In semantic segmentation, there are a few challenges such as different scales of objects, global context. Conventional CNNs such as resnet are very good for detecting features of images by adding more layers but they fail to keep the spatial

information about the objects present in the images. Spatial information is very important in semantic segmentation and Unet tries to use this information by encoder-decoder architecture. Encoder is like our typical CNN which extracts features at different scales by having different receptive fields at each level, after the encoder, decoder takes the information extracted from the encoder at a respective level and upsample the image to the original size. This upsampling could be done using transposed convolution or normal cubic interpolation.

2

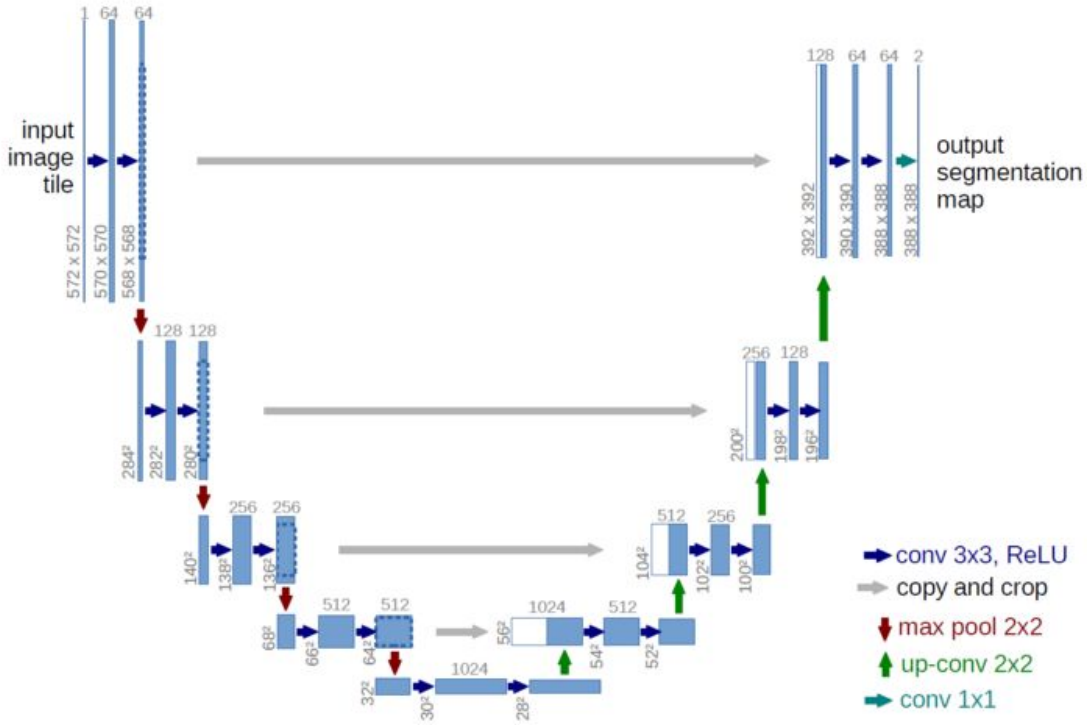


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Another advantage of Unet is that it could be used with arbitrary size, one can use image size of (512,512) for training and then use image size of (710, 1024) during evaluation because there is no fully connected layer present in it.

In addition to this, we have included two losses, cross entropy and dice loss. Dice loss is the overlap loss between prediction and actual labels just as Intersection over union.

Benchmark

Since there is no benchmark available of Unet network on cityscapes, I decided (as mentioned in the proposal) to achieve better than .50 mIoU. Given the complexity of semantic segmentation task and Unet was proposed for the medical domain and there is large evidence that it performs well for segmentation in the medical domain and by no means State-of-the-art on datasets such as Pasval VOC, cityscapes, 0.50 seems reasonable.

But other benchmarks on Cityscapes data could be seen here (https://www.cityscapes-dataset.com/benchmarks/)

III. Methodology

Data Preprocessing

A few steps are used as preprocessing while creating a data pipeline for the network.

1. Since images are very big, they should be resized to smaller and reasonable sizes. To achieve this we used random crop of (704 x 704)
2. As augmentation on train set, horizontal flip is used with probability of 0.5
3. Then images are normalized to have pixel values between 0 and 1 with the mean of `[.485, .456, .406]` and standard deviation `[.229, .224, .225]`

No augmentation is applied on the validation set.

As mentioned in the above section, classes are very unbalanced in the data, hence class weights are calculated for each class as per ENet paper.

Basically first calculate the class percent for each class in the training set and then calculate each class weight based on

```
Weight = 1/np.log(1.02 + trainId_percent)
```

Implementation

The whole project is carried out using pytorch and a few other libraries which are mentioned clearly in the README on github.

For upsampling in the decoder part of the Unet I used transposed convolution with 3x3 filter size. At each level the feature maps channels are doubled starting from 64 to 1024.

There are five levels in both encoder and decoder, at each level in encoder there are two conv layers each followed by a batch norm layer and relu activation. In decoder at each level, the first feature map is upsampled by using a transposed layer which is then followed by two conv layers same as the encoder layer.

In the last layer a conv layer is used with 1x1 filter to return the output map of size (B, Classes, H, W)

B - batch size
Classes - number of classes (20)

Dice loss and metric calculation is also implemented. The training is carried out for 200 epochs with batch size of 12 images.

Usually it is very normal to train semantic segmentation models for 200 epochs, in fact the State-of-the-art models train for longer.

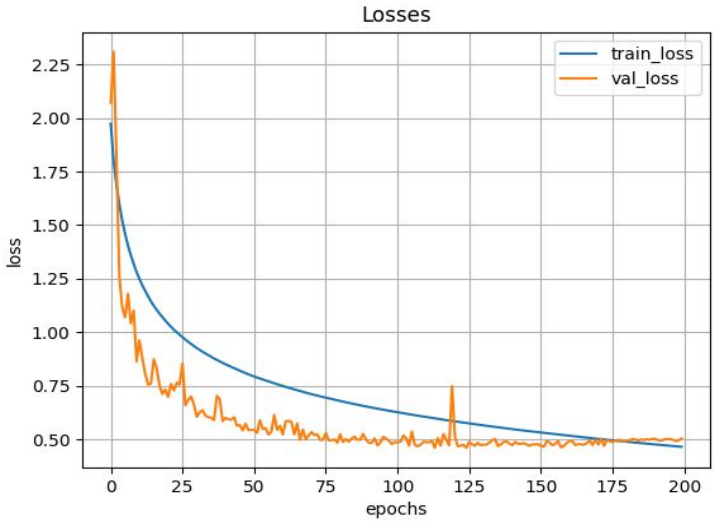
Refinement

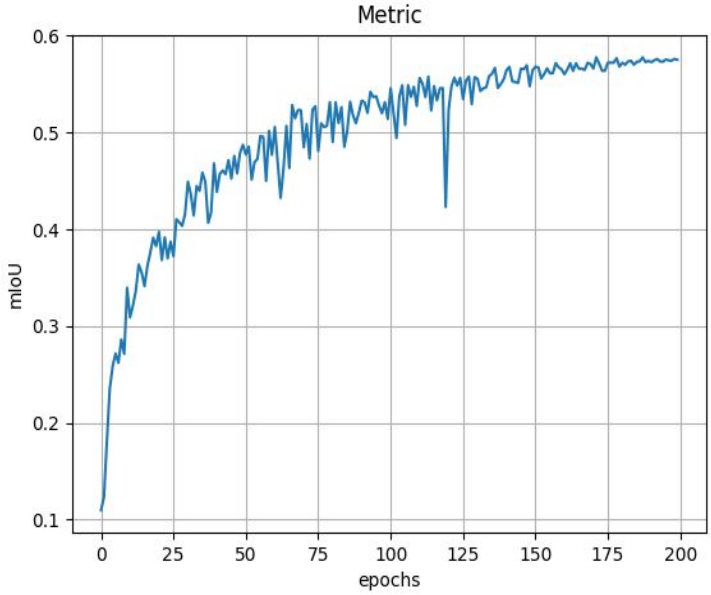
The results were poor initially as we did not use the class weights and the network was not able to learn the representation of objects with less frequent classes. Per class IoU was less for less frequent classes. After adding class weights to the loss calculation, per class IoU improved by a lot.

IV. Results

Model Evaluation and Validation

As it can be seen from the plot below that loss over both training and validation set decrease well, hence the network converged well .The Unet network gives mIoU of 0.576 upon training for 200 epochs. The network was kept in evaluation mode while evaluating the performance on validation set at each epoch.





Justification

The proposed benchmark in proposal was .50 mIoU and after training for 200 epochs Unet trained network has surpassed that accuracy. 0.50 mIoU is a quite decent benchmark because the current SOTA on cityscapes is 0.85 mIoU which is pretty good, in my opinion, given the complexity of the task.

I also calculated each class IoU which could be seen using a log file in the github repository.

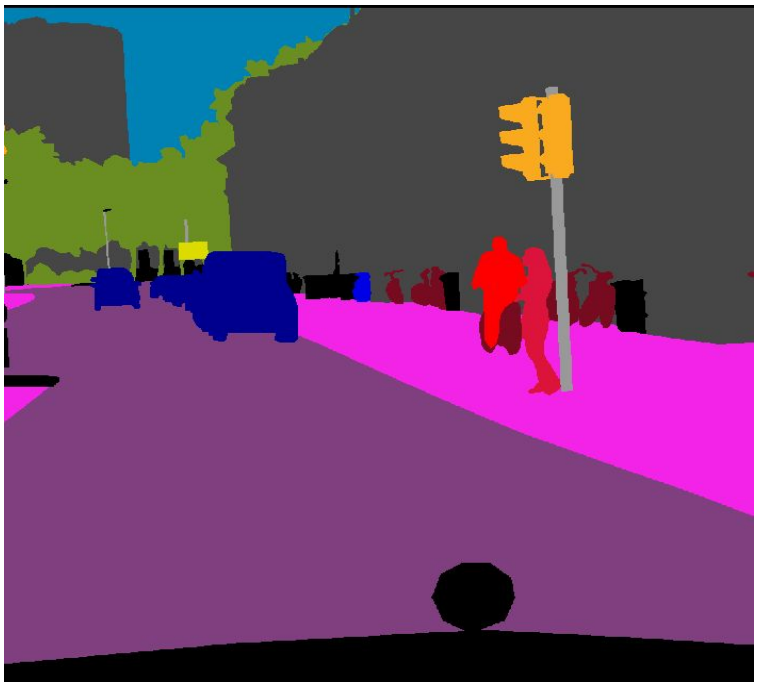
V. Conclusion

Free-form Visualization

Here are a few input images and their predicted masks.

Actual image

Actual Mask:



Predicted mask

Reflection

The process used for this project can be summarized using the following steps:

1. After formulating the problem of semantic segmentation in capstone proposal, the respective dataset was downloaded.
2. Once the data was available, data exploration was carried out.
3. Depending on the results of data exploration, the data pre processing pipeline was finalized.
4. The Unet architecture was implemented
5. Required losses, metrics and utilities functions were implemented.
6. Changes were made to improve the results.
7. Code refactoring was done and the code was clearly documented in the form README.

The most interesting part of the project for me was to deal with class imbalance to improve the results. Initially I had bad results as I was not using class weights, later I had to review some literature to find the right procedure to calculate these weights. Implementation of dice loss also took some time, especially incorporating the “ignore index” and use of class weights to calculate dice loss.

Improvement

There could be a few improvements over this work

- One could use more recent networks such as deeplab v3+, DANet which uses attention mechanisms to achieve very good results.
- The State-of-the-art networks carry out mutli-scale training, where they use images for different scales to train the network. So this could be incorporated as well.

Note: Due to not sufficient time, only the Unet network is used. In the proposal, I also mentioned about using PSPnet along with Unet.

References:

Unet (https://arxiv.org/abs/1505.04597)
ENet (https://arxiv.org/abs/1606.02147)