Semantic Segmentation of Urban Scenes using Encoder-Decoder Architecture

Zheng Luo
Department of Computer Science
University of Virginia
zl5sv@virginia.edu

Zixuan Lin
Department of Biomedical Engineering
University of Virginia
zl7qk@virginia.edu

Abstract

Autonomous vehicles are assumed to be the key units of the next generation transportation system. Since vehicle safety is critical to real world, it is important to maintain that a vehicles understanding to environment is highly reliable. Real-world driving environment is a mixture of multiple objects, and can be dense especially on urban roads. Vehicles are assumed to not only recognize multiple objects in the whole scene but also know where exactly different objects occupy. Therefore, while cameras are widely used to recognize objects, image segmentation (or clustering pixels) from a mixture becomes the challenge at the first step, in order to understand the environment precisely.

1. Introduction

In this project, the plan is to cast existing different neural networks and fine tune them to segment the validation sets in Cityscapes Dataset. Finding a proper way to modify the network structures is the first important task. Moreover, we will find fine annotated training sets to train convolutional neural networks. Except playing different neural networks and compare (and try to improve) their performance of segmentation and recognition, another experiment can also be planned to test the adaptivity of trained neural networks. That is, the project is also able to study if a fine-tuned network for a single city can behave well on images from another city. Alternatively speaking, if the training set is a mixture of many different cities, it is interesting to know how much improvement can be achieved compared to only using a single or few cities.

2. Related Work

Proven classification architectures such as AlexNet and VGG (and their successors) are widely used for single-object recognitions. However, segmentation requires pixel-wise output to distinguish different objects rather than a single judgement. In 2015, Long et al. [4] introduced fully convolutional networks to adapt those classifiers for dense prediction, with in-network up-sampling and a pixel-wise loss. Therefore, by converting and fine tuning currently existing convolutional neural networks, it is possible to achieve a much better performance of segmentation than other simple methods such as k-means clustering and watershed transform.

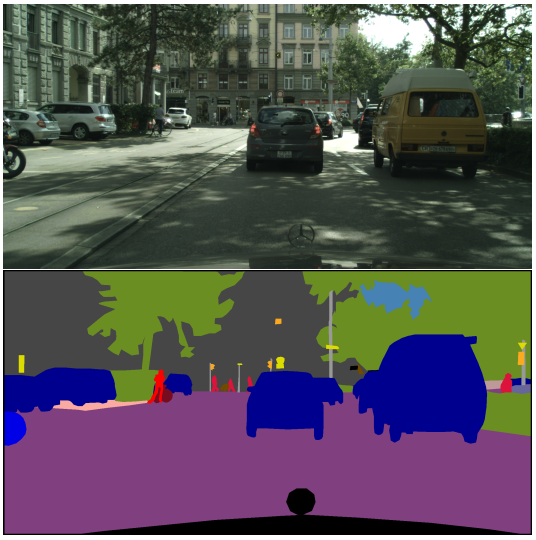

Figure 1. Here is an illustration of fine annotated urban scene from Cityscapes Dataset. The upper side is an original image with 2048×1024 resolution taken from Zuerich. The bottom side is its corresponding fine annotation. The overlayed color represents semantic classes defined by the dataset provider.

Since this project is going to play a role among the problems of autonomous driving, Cityscapes Dataset [2] will be used in this project. It has fine annotated images and more coarse annotated images of urban scenarios from multiple cities. We will firstly use the 5000-image volumn including 3475 pixel-level fine annotated images for training and validation, and 1525 dummy annotated images for testing. The dummy annotation means regions are ignored. Class labels are defined and also grouped semantically. This dataset can be very useful due to its high quality and images re-

sources (multiple cities). The dataset also have benchmarks for pixel-level and instance-level semantic labeling. They can be used to show segmentation performance quantitatively.

3. Model Structure

VGG-16 [5] has an outstanding performance on many datasets such as ImageNet for purposes of object recognition. Although casting VGG-16 to a fully convolutional network in the way described by [4] does handle the task for image segmentation, the resolution of segmentation is not good enough even if the training input has high resolutions. Therefore, we cast VGG-16 to an FCN with combinations of upsampling (unpooling) and convolutional layers (decoders) appended after convolutional layers, by following the method described in [1]. This architecture is named SegNet which contains an encoder followed by a decoder. Intuitively, we can consider the encoder part is responsible for feature extraction while the decoder handles the task of transforming deep features to RGB space.

3.1. Network Structure

First of all, we follow the structure of VGG-16 with batch normalization [3] but simply discard the fully connected layers of it. Then, we append a set of unpooling and convolutional layers with batch normalization symmetrically to the encoder part, but with the output channel equal to the number of classes for the last layer. At the end of the network, we add a softmax function to produce class predictions for each pixel in the input image. That is, for each pixel, the network will give a probability distribution telling the likelihood of belonging to each class. This is an analog to the case of image classification, but is actually performing pixel-wise classifications. The detailed structure is not showing here since this design closely follow the architecture introduced in [1].

In addition, we also add "links" between each corresponding pooling and unpooling layers. This means the unpooling layers are deterministic with respect to max-pooling layers. When data goes through pooling layers, we record the indices of preserved pixels (or tensor elements). When doing unpooling, we place tensor elements back to the same places according to the previously recorded indices.

Therefore, the image input to the network is a tensor of size B×C×H×W while the label (mask) input to it is a tensor of size B×H×W, where B, C, W, H stand for batch size, number of channels, image width, and image height respectively. The output of the network is a tensor of size B×N×H×W where N stands for number of classes.

3.2. Loss Function

The loss function for this segmentation purpose is 2D cross-entropy loss. The idea is the same as the case of image classifications. Let P be the output (predictions) of the network, and have size N×H×W, and M of size H×W be the ground-truth label of the segmented image; then the loss is defined as the following:

$$\text{Loss} = -\sum_{i=0}^{W}\sum_{j=0}^{H} logP(M(j,i),j,i).$$

With this loss function, we use Adam to train the network model for 10 epochs.

4. Experiments

When training the network defined above, we use about 3100 fine-annotated images as the training set and the rest about 100 200 fine-annotated images as the validation set. We have written a dataloader for CityScapes that can handle the whole fine-annotated dataset, including image transformations. There are also a fine-annotated test set but the ground-truth labels are only available on the server of Cityscapes website. Therefore, this test set is not used for our experiments.

The experiments break down into two main parts which will be described in this section. The first part is to train the network on a set of cities and validate it on another city that is exactly not included in the training set. The second part is to train a network among all of cities and validate it on the same set of cities, where training set and validation set are always mutual exclusive. The purpose of this experimental design is to explorer the adaptivity of neural networks with different prior knowledge. The details of dataset content are shown in Table 1.

Germany	Aachen(174) Bochum(96) Bremen(316) Cologne(154) Darmstadt(85) Dusseldorf(221) Erfurt(109) Frankfurt(267) Hamburg(248) Hnaover(196) Jena(119) Krefeld(99) Lindau(59) Monchengladbach(94) Munster(174) Stuttgart(196) Tubingen(144) Ulm(95) Weimar(142)
France	Strasbourg(365)
Switzerland	Zurich(122)

Table 1. The set of cities, their countries, and the number of corresponding images in the fine-annotated dataset.

4.1. Dataset Modifications

As mentioned in [2], only 19 classes frequently appear in those images, therefore the benchmark will only con-

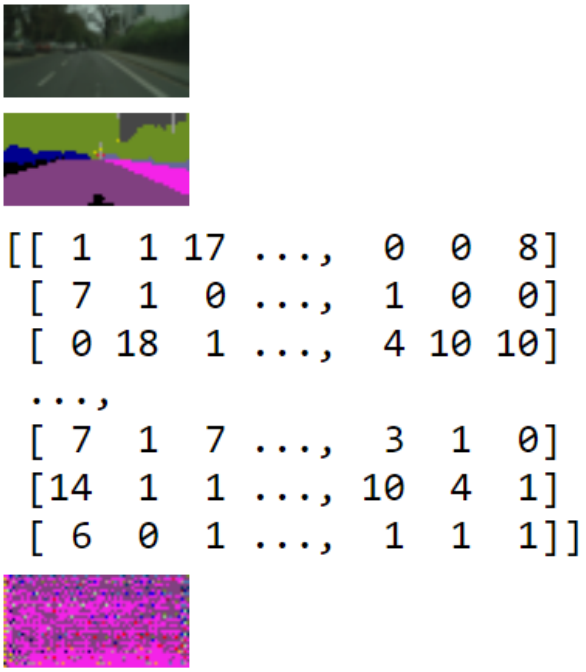

Figure 2. Here is a bad sample segmentation result of the trained neural network. The top is the original image. The second is the ground-truth label. The third (matrix) is the class predictions for each pixel on the original image. The bottom is the visualized class predictions.

sider those 19 classes to avoid ambiguity. In this project, we discard labels other than the 19 classes by setting those discarded labels to the same value 255. When doing segmentation, those classes will be considered as the same "unknown" class.

Since the images in the dataset have a high resolution (2048×1024), we resize them to 256×128 in order to save time and graphic memories. When resizing the images, we use the default bicubic method. When resizing the masks (labels), we use nearest neighbors.

4.2. Training on All Cities

4.3. Training without Frankfurt

5. Future Work

6. Conclusion

References

[1] V. Badrinarayanan, A. Handa, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv preprint arXiv:1505.07293*, 2015.

[2] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[4] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.