Machine Learning Engineer Nanodegree

Capstone Proposal – Image Segmentation

Charlio Xu
November 1st, 2017

Proposal

(approx. 2-3 pages)

Please explore my github repo for this project:

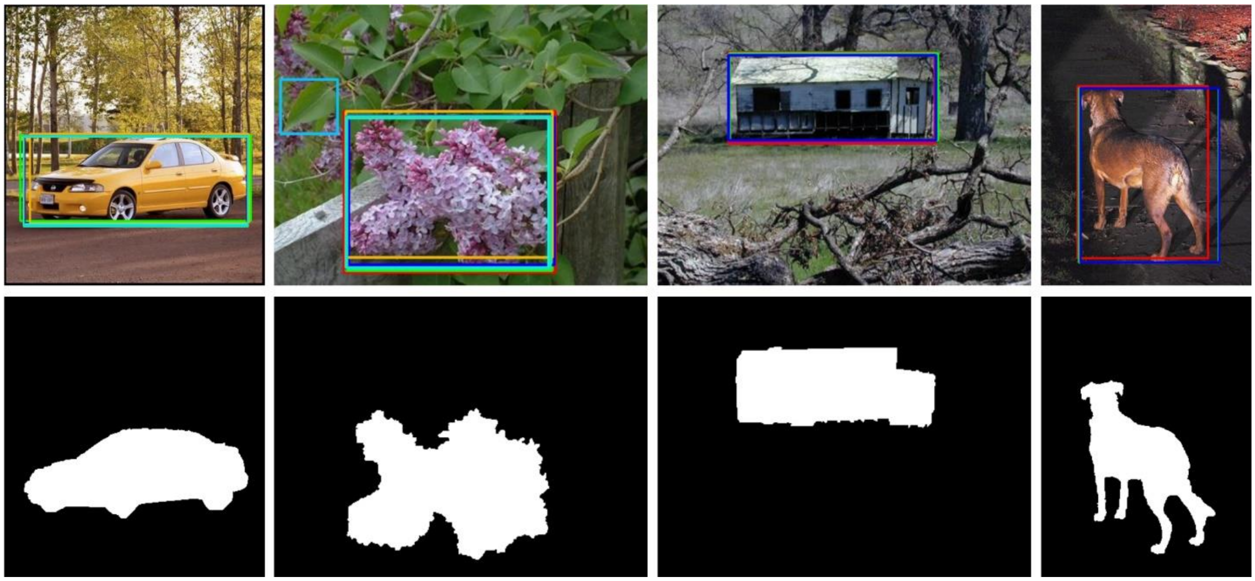
https://github.com/Charlio/image-classification-segmentation

I have trained the model and made sure I have the positive result for this project. At the end of exploration.ipynb, I gave one example. The result shows the model is implemented correctly, and the training works, though it is coarse. So I will focus on improving the result.

Domain Background

(approx. 1-2 paragraphs)

Image processing has always been a fascinating field for me to learn and experiment techniques in deep learning. In the nanodegree, I learned how to use convolutional neural networks to solve image classification problems. There are many excellent models like VGG and Inception series trained on ImageNet dataset. In this project, I will go one step further to train a model which is able to give pixel-wise classification called image segmentation. The following pictures show what this means. Given a picture, we are trying to recognize and locate the object in the image by generating a mask of it.



I will follow the idea in the following paper

https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

which uses a fully convolutional network fined tuned on the existing cnn models to accomplish the segmentation task.

Problem Statement

(approx. 1 paragraph)

Given a colored image of size (width = 224, height = 224, channels = 3) which may contain common objects like a human, a car, an airplane, an animal etc, generate a mask image of size (224, 224, 1). The value at each pixel in the mask image is either 0 or 1. 0 means this pixel is not in any object while 1 means this pixel is in some object.

Datasets and Inputs

(approx. 2-3 paragraphs)

I will use the PASCAL VOC 2012 segmentation dataset:

http://host.robots.ox.ac.uk/pascal/VOC/voc2012/

The inputs for our model will be two array of images. The array of original images (num_of_images=2913, width=224, height=224, channel=3) and the array of corresponding mask images (num_of_masks=2913, width=224, height=224). Notice that the original segmentation images in the VOC2012 dataset are given in colors, different color means different classes and there are 20 classes. However, I will simplify the segmentation images for our project so that each mask image only contain either the background (value=0), or the object(value=1) for each pixel.

Solution Statement

(approx. 1 paragraph)

First, I preprocess the PASCAL VOC 2012 segmentation data into the form of two arrays of images and corresponding masks.

Second, I implement the FCN32 model fine tuned on the VGG16 model to recognize and locate common objects in the input images. These common objects have already been learned by the VGG16 model.

Then I train the FCN32 model with the 2913 pairs of images and masks resized to 224*224 with Dice coefficient as the evaluation metric, negative Dice coeff as the loss function, adam optimizer.

Finally, I will use the trained FCN32 model to generate the mask of some input images.

If time admitted, I will also implement and train the FCN16 model which will given more accurate results.

Benchmark Model

(approximately 1-2 paragraphs)

Completed benchmark models by one of the authors of the paper are given in the following github repository:

https://github.com/shelhamer/fcn.berkeleyvision.org

The author also developed caffe so used caffe to implement the models. Besides FCN32, more refined models FCN16, FCN8 are also implemented in the repo. One can use their models to work on images to see their results.

Evaluation Metrics

(approx. 1-2 paragraphs)

During the training, I will dice_coef for as the evaluation metric and the corresponding dice_coef_loss as the loss function. The definitions are given below:

```
def dice_coef(y_true, y_pred):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = K.sum(y_true_f * y_pred_f)
    return (2. * intersection + 1.0) / (K.sum(y_true_f) + K.sum(y_pred_f) + 1.0)


def dice_coef_loss(y_true, y_pred):
    return -dice_coef(y_true, y_pred)
```

Recall that we want to generate a mask image of size (224, 224) and each pixel takes values of either 0 or 1. The corresponding label mask also has this form. So these two functions summarize the difference between the mask prediction and the ground-true mask.

Project Design

(approx. 1 page)

The make the model less complicated, I transferred the VOC segmentation images into masks which only contain two values 0 and 1. So that the model will only need to predict between two classes.

Then I will implement the FCN32 model fine tuned on the VGG16 model. Recall in the VGG16 model, there are 13 convolutional layers after which follow 3 dense layers. In the FCN32, we copy the first 13 convolutional layers as in VGG16, then instead of flatten the intermediate features and feed them into dense layers, we continue to feed them into convolutional layers. This is what FCN means: fully convolutional networks. At the end, we will get features of size (7, 7, 2) where 2 is the number of classes, 7*7 is the filter size. We then apply a deconvolutional or conv2dtranpose layer for it to transfer it into the original size (224, 224, 2), and finally we apply the sigmoid function to get the final prediction mask image of size (224, 224). To make it clearer, please read the following definition of FCN32 I implemented in keras:

```
def FCN32(classes=2):
    x = Sequential()
    # Block 1
    x.add(Conv2D(64, (3, 3), input_shape=(224, 224, 3), activation='relu', padding='same', name='block1_conv1'))
    x.add(Conv2D(64, (3, 3), activation='relu', padding='same', name='block1_conv2'))
    x.add(MaxPooling2D((2, 2), strides=(2, 2), name='block1_pool'))


    # Block 2
    x.add(Conv2D(128, (3, 3), activation='relu', padding='same', name='block2_conv1'))
    x.add(Conv2D(128, (3, 3), activation='relu', padding='same', name='block2_conv2'))
    x.add(MaxPooling2D((2, 2), strides=(2, 2), name='block2_pool'))


    # Block 3
    x.add(Conv2D(256, (3, 3), activation='relu', padding='same', name='block3_conv1'))
    x.add(Conv2D(256, (3, 3), activation='relu', padding='same', name='block3_conv2'))
    x.add(Conv2D(256, (3, 3), activation='relu', padding='same', name='block3_conv3'))
    x.add(MaxPooling2D((2, 2), strides=(2, 2), name='block3_pool'))


    # Block 4
    x.add(Conv2D(512, (3, 3), activation='relu', padding='same', name='block4_conv1'))
    x.add(Conv2D(512, (3, 3), activation='relu', padding='same', name='block4_conv2'))
    x.add(Conv2D(512, (3, 3), activation='relu', padding='same', name='block4_conv3'))
    x.add(MaxPooling2D((2, 2), strides=(2, 2), name='block4_pool'))


    # Block 5
    x.add(Conv2D(512, (3, 3), activation='relu', padding='same', name='block5_conv1'))
```

```
x.add(Conv2D(512, (3, 3), activation='relu', padding='same', name='block5_conv2'))
x.add(Conv2D(512, (3, 3), activation='relu', padding='same', name='block5_conv3'))
x.add(MaxPooling2D((2, 2), strides=(2, 2), name='block5_pool'))

# Fully convolutional layers
x.add(Conv2D(4096, (7, 7), activation='relu', padding='same', name='fcn1'))
x.add(Dropout(0.5))
x.add(Conv2D(4096, (1, 1), activation='relu', padding='same', name='fcn2'))
x.add(Dropout(0.5))
x.add(Conv2D(classes, (1, 1), padding='valid', name='fcn3'))

x.add(Conv2DTranspose(classes, (64, 64), padding='valid', strides=(32, 32), name='deconv1'))
x.add(Cropping2D(cropping=((16, 16), (16, 16))))
x.add(Dense(classes, activation='softmax', name='predictions'))

return x
```

Before submitting your proposal, ask yourself. . .

- Does the proposal you have written follow a well-organized structure similar to that of the project template?
- Is each section (particularly **Solution Statement** and **Project Design**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?
- Would the intended audience of your project be able to understand your proposal?
- Have you properly proofread your proposal to assure there are minimal grammatical and spelling mistakes?
- Are all the resources used for this project correctly cited and referenced?