Mini Project 2
--Machine Learning Models for Segment Object Detection
By Jie Lu

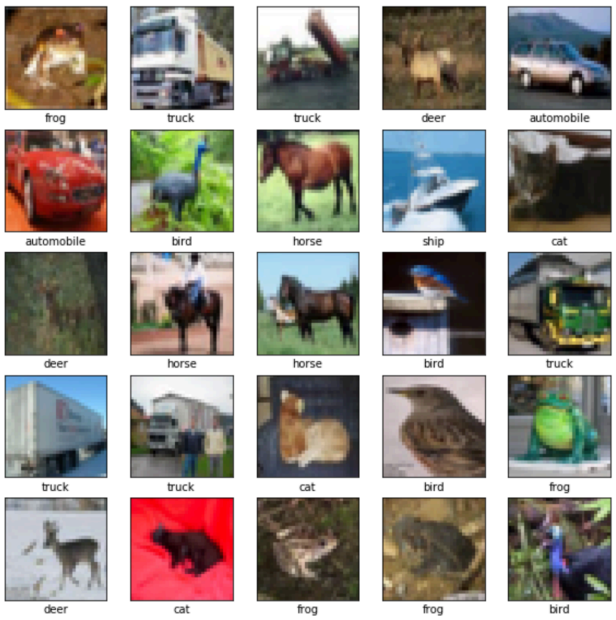
Introduction

Nowadays Machine Learning plays an elemental role in today's development. The application of the Machine Learning on some field of our world is making our life better. It has been used on the face recognition and self-driving (e.g. Tesla's Autopilot). Object Segment Detection is one of the extensions of the Object Detection in Machine Learning. TensorFlow, powered by Google, provides a lot of collection of workflows and pre-trained models. Here, I would like to introduce the TensorFlow to analyze our results of object detection.

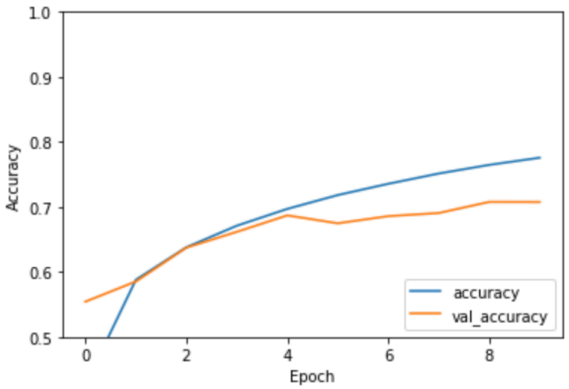
Analysis of Results

Convolution Neural Network (CNN)

The information from an image could be large and it needs much calculation to analyze its data. The CNN can decrease the calculation by reducing the number of dimensions to a rational level. In the TensorFlow, the CIFAR10 dataset, contains over 60,000 images in 10 classes; These classes are exclusive and there is no overlap between them.



1-1. 25 images from the CIFAR10 Dataset

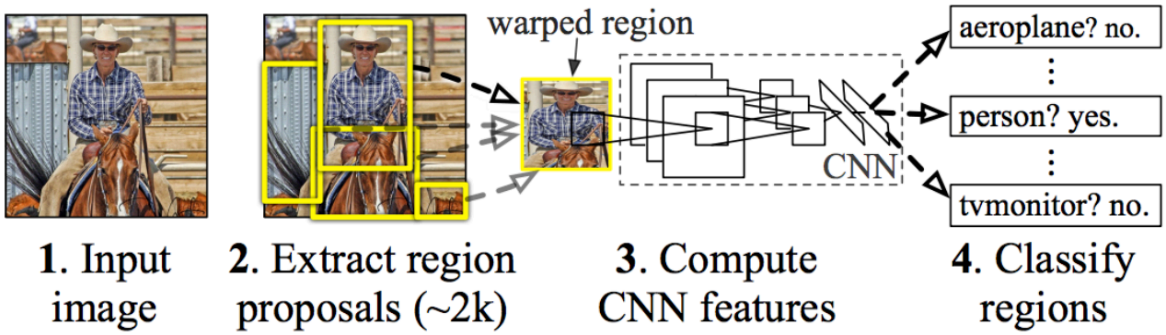


1-2. Chart of Accuracy

From the above chart, the accuracy with CNN by TensorFlow could reach about 77%.

RCNN

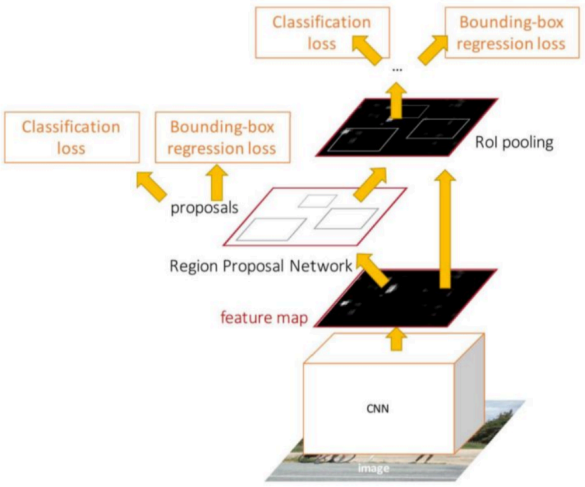
The RCNN is the upgrade of the CNN. It uses the following algorithm: 1. Get region proposals for object detection (using selective search). 2. For each region crop the area from the image and run it thorough a CNN which classify the object.



In RCNN, step 2 is to help determine the possible regions we need to extract; Step 3 is to use CNN to determine features of the image. However, the RCNN has a low efficiency, since it takes more time on the selective search.

Faster RCNN

Faster R-CNN has two networks: region proposal network (RPN) for generating region proposals and a network using these proposals to detect objects. The main different here with Fast R-CNN is that the later uses selective search to generate region proposals. The time cost of generating region proposals is much smaller in RPN than selective search, when RPN shares the most computation with the object detection network. Briefly, RPN ranks region boxes (called anchors) and proposes the ones most likely containing objects.

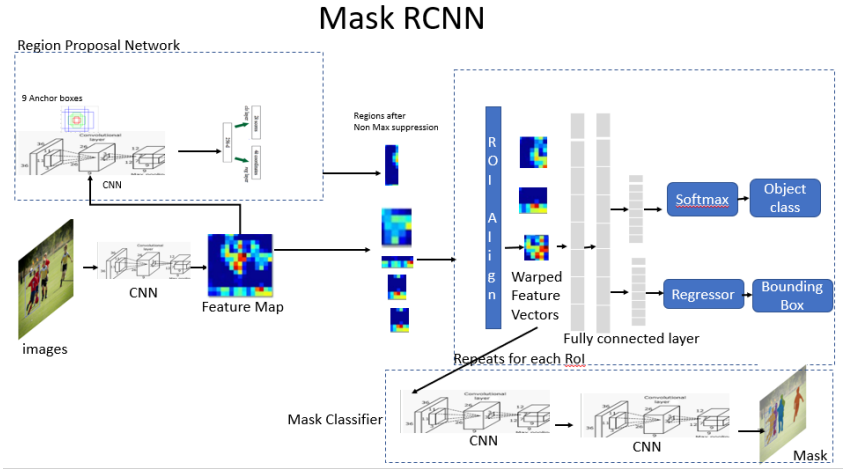


Faster-RCNN

Mask-RCNN

"This is an implementation of Mask RCNN on Python 3, Keras and TensorFlow. The models generate bounding boxes and segmentation masks for instances of an object in the image. It's based on Feature Pyramid Network (FPN) and a ResNet101 backbone."

In Mask-RCNN, there are two stages: First, it generates proposals about the regions where there might be an object based on the input image. Second, it predicts the class of the object, refines the bounding box and generates a mask in pixel level of the object based on the first stage proposal. Both stages are connected to the backbone structure.



Mask-RCNN Architecture

Mask-RCNN generate with Bounding boxes

YOLO in TensorFlow (You Only Look Once)
"TensorFlow implementation of 'YOLO: Real time object detection', with training and an actual support for real time running on the mobile devices."

Using YOLO in TensorFlow can determine the location on the image where certain objects are present, as well as classifying those objects. Previous methods for this, like R-CNN and its variations, used a pipeline to perform this task in multiple steps. This can be slow to run and also hard to optimize, because each individual component must be trained separately. YOLO, does it all with a single neural network.

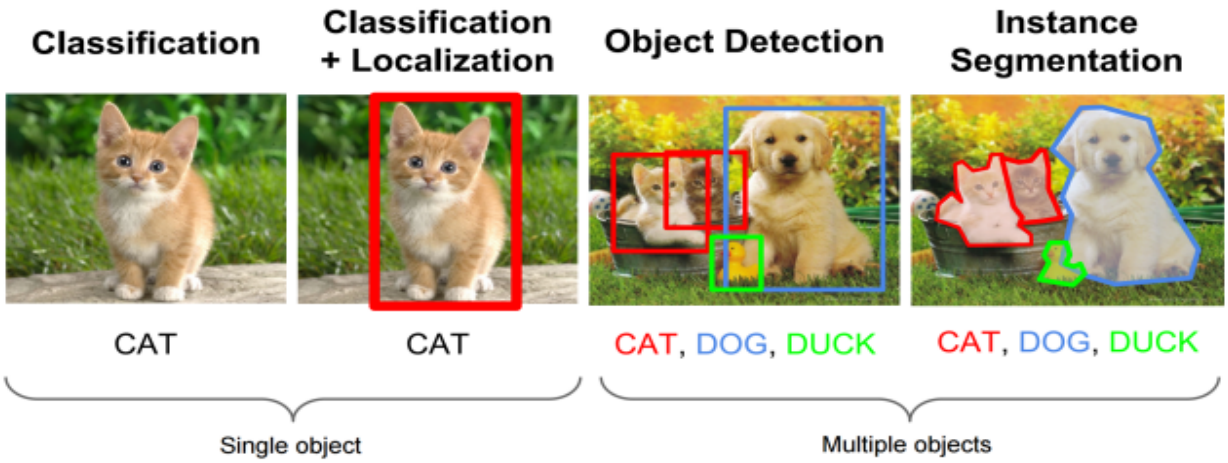


Image Classification and Object Detection with YOLO

"We reframe the object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities."

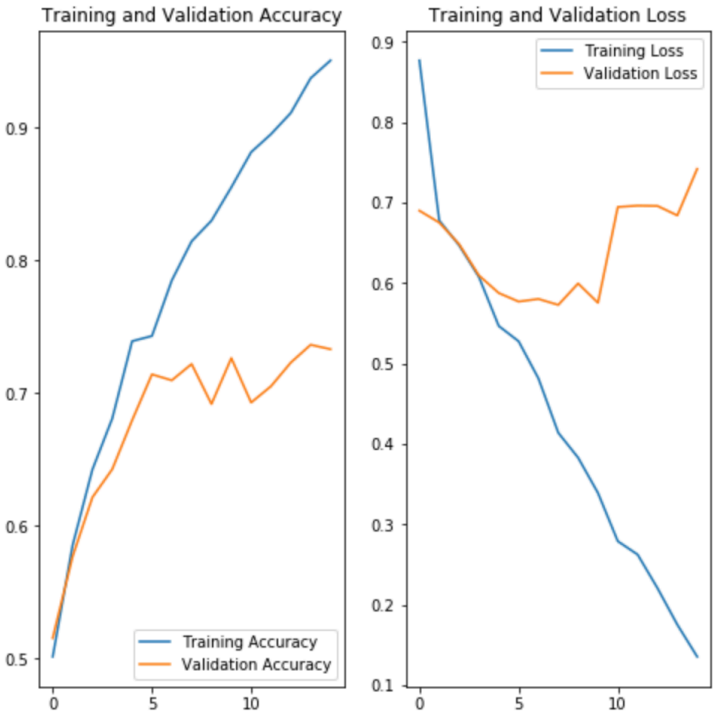
TensorFlow Solve with Images: Image Classification and Segmentation

Image Classification

TensorFlow uses a **tf.keras.Sequential** model to classify images. For example, classifying cats and dogs from images by this model. Download the dataset and use the **ImageDataGenerator** class to build the input pipelines. After that, we need to build the model, train the model and finally test our model to make sure it reaches our expectation.

Download *Dogs VS Cats dataset* from Kaggle

After loading the dataset and data preparation finished, we start to create our model, then compile our model. We can also use the *summary* method to view all layers. Finally, use the *fit_ generator* method of the **ImageDataGenerator** class to train this model. After all these steps done, we can visualize our results as the following plots.



As we can see from the plots, the training accuracy and the validation accuracy are off by large margin and in the validation set, this model only achieve about 70% accuracy. Since we are using a small number of examples, the models learn from some noises and some unwanted

details. This phenomenon is known as **overfitting.** To fix the overfitting, we need to use the Data argumentation and Dropout. After we train a new model, it has less overfitting than before.

After use the data argumentation and dropout

Image Segmentation

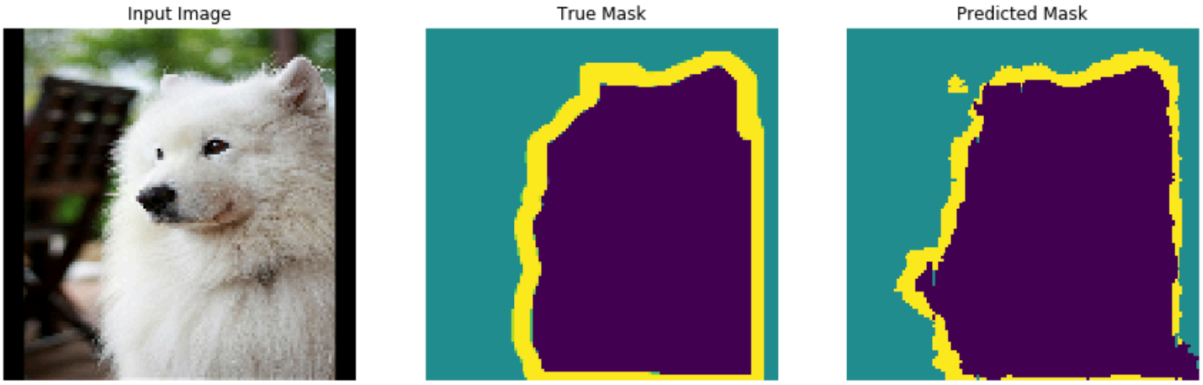
To detect an object in an image, we need to know where the object located in the image, the shape of the object and which pixel belong to which object, etc. In this case, the image needs to be segmented. The purpose of the image segmentation is to train a neutral network to output a pixel-wise mask of the image. It helps in understanding the image in a much lower level. The application of the image segmentation is among self-driving, medical imaging and satellite imaging.

Here the dataset would use the Oxford-IIIT Pet Dataset in the TensorFlow datasets. It contains images, pixel-wise masks and corresponding labels of these images. These masks labels for each pixel and each pixel is given one of three classes: 1. Pixel belong to the pet; 2. Pixel boarding the pet; 3. None of above pixel.

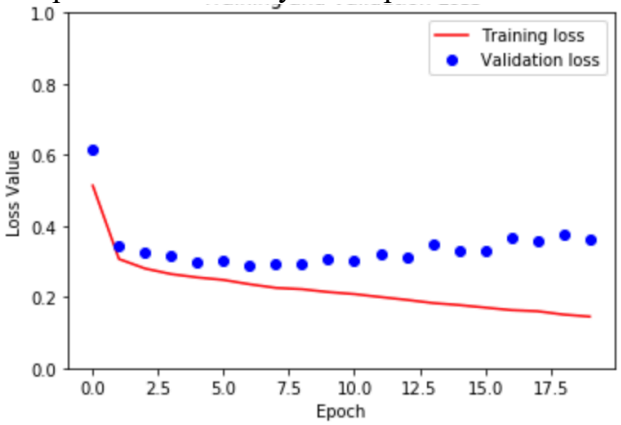
Input image

True Mask

The model used here is a modified U-Net. A U-Net consists of an encoder and decoder. The encoder we used a pretrained model, which is a pretrained MobileNet V2 model. The decoder used here is the unsampled block implemented in the TensorFlow.



The loss function also used here is because the network tries to label each pixel, like the multi-class prediction. It helps improve the accuracy of the prediction.



Training and violation loss

Pres & Cons

Pres:

1. Graphs. TensorFlow has better computational graph visualizations, which are indigenous when compare to Theano and Torch.
2. Library. TensorFlow are backed by Google. It is also a large open source library, has a collection of datasets and releases quite updates with features.
3. Scalability. TensorFlow can be used in devices from cell devices to computers.

Cons:

1. TensorFlow need to upgrade frequently.
2. TensorFlow support computation across from CPUs and GPUs. But the TensorFlow doesn't provide GPU support on the MacOS.

Recommendation

TensorFlow is a friendly tool to start machine learning for beginners. It also fully supports Python and easy to run on the python environment. Make sure your hardware met the requirement, or you can't run the training model. Apply for a cloud computing platform is also a good choice to run your model.

TensorFlow also provides powerful experimentation for research. Provided a large collection of datasets and most of pretrained models, uses can exploit these models on opensource platform, cloud and personal devices.

Conclusion

TensorFlow is friendly to users to learn the machine learning. The guide from the TensorFlow documentation is very detailed and easy to access. Uses can uses API from the TensorFlow Zoom, install these API and take advantage of those pretrain models also provided by TensorFlow. Besides, TensorFlow allowed neutral network computation on devices, including cell device, opensource platform, and cloud. Usually, the accuracy of the original pretrained models is pretty high enough.

Reference

[1]. GitHub: Machine Learning Object Detection Model Zoo
https://github.com/tensorflow/models/tree/master/research/object_detection
[2]. Image Segmentation TensorFlow Tutorial
https://www.tensorflow.org/tutorials/images/segmentation
[3]. Image Classification TensorFlow Tutorial
https://www.tensorflow.org/tutorials/images/classification