**Object Detection using TensorFlow**

CPS-584 Advanced Intelligent Systems and Deep Learning

Department of Computer Science

University of Dayton

Professor: Mehdi R Zargham

**Shravya Reddy Akmy – 101709698**

**Surya Venkatesh Vijjana – 101709607**

**Introduction**

This project involved implementing Object Detection using TensorFlow and deploying the trained model to mobile environment. We are using Pre-Trained Models to detect Cat(s) and Dog(s) in the given input image.

**Description**

To develop a model that can do object detection, we are using 1000 images of cat and dogs. Each image may contain the picture of a cat, or a dog, or both and 80% of the dataset is used as training data and the rest 20% used as the test data for the model training.

We used the **ssd\_mobilenet\_v2\_fpnlite\_320x320\_coco17\_tpu-8** pre-trained model from TensorFlow Model Zoo with 90 detection classes which is configured to detect 2 classes i.e., Cat and Dog as per our requirement.

**Purpose of the project**

The purpose of this project is to detect cats and dogs in any image given as an input to the trained model. The input can be an image from a local machine or from a live webcam footage or when deployed to mobile environment the input should be live camera footage from the primary camera.

**Dataset**

For the preparation of data set we used few images from the kaggle data set provided and from different web sources like Google Images, Pixabay and Unsplash.

**Sample Images**

Timeline

Description automatically generated

**Implementation**

To train the model, first we need to download the pre-trained model from the TensorFlow model zoo and extract the pre trained model zip file to our workspace.

* Create a virtual environment and install necessary dependencies including tensorflow-gpu and install CUDA and CUDNN to run the model using gpu.
* **Using LabelImg application**, label the cats and dogs in each image present in both train and test data folders and generate xml files.

A picture containing text, indoor, screenshot

Description automatically generated

* Create a pbtxt file(labelmap.pbtxt) with labelname and id’s for both cat and dog.
* Using the python script from this github repo ( [Generate TFRecord](https://github.com/nicknochnack/GenerateTFRecord) ), generate the tfrecord files for both train and test data with images, xml files and labelmap.pbtxt as input.
* Modify the config file of pre trained model as per our requirements.
  + Change the number of classes to 2 as we need to detect Cat and Dog.
  + Add the path(s) of labelmap.pbtxt file, test and train tf records to the config file.

Text

Description automatically generated

* Train the model to get minimum loss values and good mAP and recall values. We trained the model for 20,000 steps.

python Tensorflow\models\research\object\_detection\model\_main\_tf2.py --model\_dir=Tensorflow\workspace\models\my\_ssd\_mobnet --pipeline\_config\_path=Tensorflow\workspace\models\my\_ssd\_mobnet\pipeline.config --num\_train\_steps=20000

* After training is completed, evaluate the model to get the metrics related to the model.

python Tensorflow\models\research\object\_detection\model\_main\_tf2.py --model\_dir=Tensorflow\workspace\models\my\_ssd\_mobnet --pipeline\_config\_path=Tensorflow\workspace\models\my\_ssd\_mobnet\pipeline.config --checkpoint\_dir=Tensorflow\workspace\models\my\_ssd\_mobnet

Text

Description automatically generated with medium confidence

* Load saved pipeline config file and build the object detection model and restore the latest checkpoint file.

Text

Description automatically generated

* To Detect the Cat/Dog, pass the image path as an input.

Text

Description automatically generated

**Output**

**A picture containing text, mammal

Description automatically generated**

**Webcam Implementation**

**Text

Description automatically generated**

**Output**

A picture containing text, screen, cat, television

Description automatically generated

**Training Plots**

**Chart

Description automatically generated**

**Testing Plots**

Precision

**Chart

Description automatically generated**

Recall

**Chart

Description automatically generated**

Loss

**Chart

Description automatically generated**

**Evaluation**

**Graphical user interface, application

Description automatically generated**

**Mobile Deployment**

* Freeze our current trained model and export the model and convert the model to tflite model.

**Freeze the model to export using this command**

python Tensorflow\models\research\object\_detection\exporter\_main\_v2.py --input\_type=image\_tensor --pipeline\_config\_path=Tensorflow\workspace\models\my\_ssd\_mobnet\pipeline.config --trained\_checkpoint\_dir=Tensorflow\workspace\models\my\_ssd\_mobnet --output\_directory=Tensorflow\workspace\models\my\_ssd\_mobnet\export

**Convert into tflite model using these 2 commands**

python Tensorflow\models\research\object\_detection\export\_tflite\_graph\_tf2.py --pipeline\_config\_path=Tensorflow\workspace\models\my\_ssd\_mobnet\pipeline.config --trained\_checkpoint\_dir=Tensorflow\workspace\models\my\_ssd\_mobnet --output\_directory=Tensorflow\workspace\models\my\_ssd\_mobnet\tfliteexport

tflite\_convert --saved\_model\_dir=Tensorflow\workspace\models\my\_ssd\_mobnet\tfliteexport\saved\_model --output\_file=Tensorflow\workspace\models\my\_ssd\_mobnet\tfliteexport\saved\_model\detect.tflite --input\_shapes=1,300,300,3 --input\_arrays=normalized\_input\_image\_tensor --output\_arrays='TFLite\_Detection\_PostProcess','TFLite\_Detection\_PostProcess:1','TFLite\_Detection\_PostProcess:2','TFLite\_Detection\_PostProcess:3' --inference\_type=FLOAT --allow\_custom\_ops

* Clone the Tensorflow Android Deployment repository and import the directory as a project in the Android Studio and in the assets folder replace the tflite and labelmap.txt files with our exported tflite model and labelmap.txt file with cat and dog as labels.
* Install the required dependencies and connect your android device and run the build to install the apk file to the device connected.

**Output**

**Graphical user interface, application

Description automatically generated**

**Issues Encountered**

* While generating the tfrecord files, we encountered an error because in few xml files file format is missing (.jpg extension for filename tag in xml file). To fix this issue, we used **xml.etree** library to add the file format to the filename tag where it was missing.

Text

Description automatically generated

* Missing few dependencies while training our model. To fix this issue, installed those missing dependencies.

pip install <missing library>

* When we tried different object detection models like resnet, inceptionResNet, efficientDet we got a version mismatch error and out of memory errors. So used **MobileNet** for training.
* While deploying the tflite model to the mobile, application keeps on crashing and when referred to the github issues section found that metadata for tflite model is missing. To fix this issue, we used tflite\_support library and add metadata to our tflite model.

Text

Description automatically generated

**References**

* Object Detection using Tensorflow Tutorial by [Nicholas Renotte](https://www.youtube.com/channel/UCHXa4OpASJEwrHrLeIzw7Yg) - [Tutorial](https://www.youtube.com/watch?v=yqkISICHH-U&ab_channel=NicholasRenotte)
* Error Guide [ Github Repo ]: [TFOD Error Guide](https://github.com/nicknochnack/TFODCourse)
* TF Record Generator Script: [GenerateTFRecord](https://github.com/nicknochnack/GenerateTFRecord)
* Android Deployment: [Tensorflow : Android](https://github.com/tensorflow/examples/tree/master/lite/examples/object_detection/android)
* Generate Metadata for TFLITE Model : [Metadata Generator](https://www.tensorflow.org/lite/convert/metadata_writer_tutorial#object_detectors)
* Dataset : [Kaggle](https://www.kaggle.com/datasets/tarunbisht11/yolo-animal-detection-small?resource=download), [pixabay](https://pixabay.com/), [unsplash](https://unsplash.com/)