Mobile Object Detection Using 2D and 3D Basic Geometric Figures in Colour and Grayscale

Ari Ernesto Ortiz Castellanos
Electrical Engineering and Computer Science, National Taipei University of Technology, 1, Sec. 3, Zhongxiao E. Rd., Taipei 10608 Taiwan
t107999401@ntut.org.tw

Abstract. In this paper, we use TensorFlow Mobile Lite for Object Detection with datasets of basic geometric figures on iOS mobile devices. Additionally, we trained 4 datasets in 2D and 3D and we compare the accuracy of detection between using colour and grayscale image data. Also, we evaluate the detection rate using 2D and 3D for some kind of normal objects in precision and label output. We used Convolutional Neural Networks (CNN) for build the datasets and OPENCV for convert into grayscale. We value the result relation between flat and volume datasets in way of label and numeric detection, also the affectionation of TensorFlow Mobile with this kind of datasets. We make comparisons based on the results of the different experiments object the detection and, in this work, TensorFlow Mobile Lite implementation does not have pseudo boxes and the reason is explained for detection purposes and accuracy adjusted to this kind of experiments.

1. Introduction
Smart mobile devices have been changed the life of humanity, and the integration of deep and machine learning models have been increasing the target devices. The simplification of computer technologies and the adaptation for mobiles have open a new way of possibilities. Object detection is developing from the single object recognition to the multi-object recognition [1]. The increase in functionality of mobile devices makes transitioning from an Internet society towards a mobile one [2]. An example of this kind of evolution is TensorFlow, since early a version for Android, Raspberry and iOS were released. This framework of machine learning provides some samples of speech and object pattern recognition with their own dataset model for testing allows create knowledge transference for training own data. The most common datasets are QuocNet, AlexNet, Inception(GoogLeNet) and BN-Inception; those dataset models have different features and definitions.

We train a dataset using Learning Transference to dataset model SqueezedNet and the source images were created based on it. This model is a based on ResNet model and Squeeze and the documentation website suggest the similarity to AlexNet accuracy. The datasets were tested on the same objects using the camera stream for checking the accuracy of detection. The dataset 2D consist in four geometric shapes from kaggle.com which contains, images (star, circle, square and triangle) in PNG format and the 3D is our own built with the same number of images in same format (Cube, Pyramid, Sphere and Star) also in PNG format. Images from different viewing angles were extracted from each model to build the 2D dataset [3]. The input of images is from 200x200 pixels for the dataset. In addition, the 2D and 3D images were converted into grayscale and retrained for create two new datasets. The reason is we want to know if the accuracy increases or decreases when the source...
dataset is 3D or 2D and when it is colour or grayscale. The way to segmentation of images into shapes and background within HIS representation of the image [4] [5]. The rest of paper is organized as the following way. Section 2, it explains with more details the process of image selections. Section 3, it shows the process of training of datasets and the model implemented. Section 4, it is the experiment settings and the detection process. Section 5, Results of our experiment and comparisons between the datasets. Section 6, it is the conclusions of this work.

2. Geometric figures
In this work, the datasets trained are from geometric figures, it allows to reduce the detection in only for elements for a better measure and as basic geometric figures more objects can be treated the contain those figures. Other reason that we consider this kind of dataset is because we can have expected to have more accuracy in shapes detections than the conventional sample datasets. The model is the target extension for our Mobile TensorFlow Lite Sample App and we generated this extension directly.

3. Training Dataset
The implementation of any method of recognition requires pre-processing of images [6]. Each dataset has many images, each image is 2KB of weight, those were trained for iOS mobile devices. In comparison with Android OS devices TensorFlow mobile has more examples and more documentation but in case of this work we selected iOS devices. The datasets have category names divided in folders with shape names for example: Cube, Sphere, etc (See Figure 1). The CNN in the moment of train the dataset and makes the transferring knowledge puts the folder category name implicit, it does not need like model PB extension a notepad with objects names or categories. The model has the benefit of small model size, good efficiency and good accuracy due to the fact that it’s fully convolutional and only contains a single forward pass [7]. CNNs lead to many great achievements on a series of vision tasks such as object recognition [8]. Neural networks are currently achieving things that no other machine learning algorithm can achieve [9] [10]. The datasets are not overpassing MB for efficiently implementing the data in the requires balancing computation, communication, and storage [11]. TensorFlow is an incredible tool that offers us a very powerful framework, but that greatly simplifies the internal complexity involved in the management of deep learning algorithms. In addition, all pre-trained models have the step of additional normalization that must subtract the average pixel of each pixel in each image. By retaking the data from a prediction generated by the model, it can be compared with the real data that was used to feed the algorithm. Thus, a performance metric can be obtained to decide if the model is satisfactory or if the process should be iterated. When a neural network becomes more complex, full of layers and parameters, the process becomes much more complex. This is where TensorFlow comes in, a numerical computing library that computes gradients automatically. The advantage of transfer learning is that it reduces the number of images required for training and the training time [12].

![Figure 1. Image sample grey and colour from dataset [13].](image)
Our training dataset is built on SqueezeNet with TuriCreate for iOS and it is readable by Mobile TensorFlow Lite App capable with Inception version model (See Figure 2). The SqueezeNet model has only 4.8 MB of parameters (50x smaller than AlexNet), and it matches or exceeds AlexNet-level accuracy on ImageNet [14].

![Diagram of SqueezeNet with TuriCreate integration for TensorFlow Mobile Lite iOS.](image)

**Figure 2.** Shows the Diagram of SqueezeNet with TuriCreate integration for TensorFlow Mobile Lite iOS.

4. Experiment
The experiment was implemented in iOS devices. The app was tested on different objects with the different datasets. Those objects were used for the trained data and app for a better measure of precision detection. The geometric datasets will detect the closest shape from objects. The hardware used for create the datasets has the following hardware: 16GB of RAM, CORE 7i, GPU NVIDIA GeForce of 6GB and 1TB of HDD. The dataset for mobile devices cannot be very large, we developed and trained the datasets with many images but with less weight, because deploy those are limited for mobile computation resources [15]. The built Mobile TensorFlow Lite was running in a MacBook Air with 8GB of RAM, CORE i5, Intel HD Graphics 4000 and 250GB of HDD.

Our sample of TensorFlow has an average precision meter which starts on 0.0000 to 1.0000, when it is closer to 1.0000 it seems it is very accurate and when it is decreasing it is less accurate. In [15] observe that the generated pseudo boxes have good localization accuracies, but cannot detect every object in complex images. For that reason, our sample does not have boxes and it uses labels in the control layout.

5. Results
The result of experiments shows comparing that detection of geometric figures in colour between 2D and 3D, 2D is more accurate in detection terms (See figure 3). In grey scale in 2D and 3D decrease dramatically, it suggests when the dataset is converted in a grayscale when it is trained it loses accuracy (See figure 4), detection in colour is better than grayscale in terms of number of precisions. In case of terms of output colour label, the colour 3D detection was better than 2D; otherwise the grey scale experiments showed the 2D object results were for a minimum more accurate than 3D (See figure 5 and 6). In figure 7 is showed the level of detection of specific shapes between grey and colour and it stills almost equal.

 Apparently, the volume of the geometric figures of the volume affects the detection process in relation to the flat figures. Comparing the 2D in colour and grey dataset results the grayscale was more accurate in a comparison of colour (Figure 8). In case of 3D the colour was more accurate than grayscale, it decreased close to a percent. Figures showed the detection errors in objects with grey and...
colour. We could see in some cases the precision was very high but the out-label showed a wrong object but with a high accuracy (See figures 9 and 10).

Figure 3. Shows the result detection in colour between 2D and 3D.

Figure 4. Shows the out-label accuracy in grey between 2D and 3D.
**Figure 5.** Shows grayscale was more accurate in 2D

**Figure 6.** Shows detection in colour and grayscale in 3D
Figure 7. Shows the result detection in grayscale between 2D and 3D.

Figure 8. Shows output label 2D in grayscale
Figure 9. Shows the out-label accuracy in grayscale between 2D and 3D.

Figure 10. Shows 3D output label detection in 3D in grayscale.

6. Conclusion

We tested detection of geometric shapes using Mobile Lite TensorFlow in iOS devices. The version of the app of Mobile TensorFlow for iOS can read SqueezeNet Models. 3D geometric shapes were better in terms of precision numbers and label output than 2D geometries in grayscale and colour. In 2D detections the colour was more accurate than grayscale for minimum measure but in contrast with 3D, between grayscale decreased dramatically around a percent than the colour version dataset.

The training data does not have a high computational cost because the amount of data is relatively low and easy to manage. Some of the output labels were not accurate but in contrast of numeric precision it was the opposite. We applied the experiment on specific objects with the goal of understand deeper the behaviour of the datasets trained. In future work we could run this experiment in Android Devices and try detect between different models if the accuracy increases or decrease in comparison of iOS version, and increase the dataset of basic geometric shapes with more objects.
References

[1] Dong X, Zheng L, Ma F and Meng D 2017 Few-Example Object Detection with Model Communication(Australia: University of Technology Sydney).

[2] Rexha B, Shala G and Xhafa V 2018 Increasing Trustworthiness of Face Authentication in Mobile Devices by Modeling Gesture Behavior and Location Using Neural Networks(Kosovo and Germany: University of Prishtina, Freiburg University and Technical University of Munich).

[3] Tzitzilonis V, Kappatos V, Dermatas E and Apostolopoulos G 2018 Classification of occluded 2D objects using deep learning of 3D shape surfaces(Greece and Denmark: University of Patras and University of Southern Denmark).

[4] Sawarkar V and Katherkar SJ A 2018 Review: Rose Plant Disease Detection Using Image Processing(India: Dr. Babasaheb Ambedkar Marathwada University).

[5] Tripathi G and Save J 2015 An Image Processing and Neural Network Based Approach for Detection and Classification of Plant Leaf Diseases(India: Fr. Conceicao Rodrigues College of Engineering Fr. Agnel Ashram).

[6] Zhou X, Gong W, Fu WF and Du FT 2017. Application 6 of Deep Learning in Object Detection(India: Kalinga Institute of Industrial Technology (KIIT) University).

[7] Li Z and Rao Z 2014 Object Detection and Its Implementation on Android Devices(India: NMIT and NITTE).

[8] Zhang Y, Bi S, Dong M and Lui Y 2016 The Implementation of CNN-based Object Detector on ARM Embedded Platforms(United Kingdom and Cyprus: Imperial College London and University of Cyprus).

[9] Low L R and Tumakov D 2018 Method of Selecting an Optimal Activation Function in Perceptron for Recognition of Simple Objects(Russian Federation: Kazan Federal University).

[10] Phadnis R, Mishra J and Bendale S Objects Talk - Object detection and Pattern Tracking using TensorFlow(India: Vidyalankar Institute of Technology).

[11] Low Y, Gonzalez J, Kyrola A, Brickson D, Guesting C, 2012 Hellerstein J Distributed Graph Lab: A Framework for Machine Learning and Data Mining in the Cloud(United Stated of America: Carnegie Mellon University and UC Berkley).

[12] Liu W, Lu R and Lui X 2018 Traffic Sign Detection and Recognition via Transfer Learning(People’s Republic of China: Chan’an University and Harbin Institute of Technology).

[13] Geometric Shape Example, Retrieved February 28, 2019, from https://www.clipartmax.com/middle/m2i8H7Z5b1Z5d3K9_target-building-construction-has-a-great-capacity-to-rubik-cube-colouring-page/

[14] Wu B, Wan A, Iandola F, Jin P and Keutzer 2016 SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving(United States of America: UC Berkeley).

[15] Wu Y, Liu HF, Li J and Fu Y 2017 Deep Face Recognition with Center Invariant Loss(United States of America: Northeastern University).