# Traffic sign recognition with android app

## by Titania&co

### About the project

The purpose of this project is to build a neural network based android app for traffic sign detection and classification. The learning process runs on PC. The resulting neural network runs in an android app and processes real time camera stream.

For the neural network we use a tutorial [1]. We use an ssd model downloaded from the model zoo, to achive tflite compatibility [2]. The android app used is the tflite example app [3]. The softwares we used for this and their version are listed below [5]. The training database we tried to use [4] failed to give good results, so now it is our own, with screenshots from Google Maps.

We trained the neural network to recognise five traffic signs in the first round, with the public database. The results were unusable, but the conversion to tflite and embedding it into the mobile app worked well.

Our second attempt was with our own pictures, three traffic signs. The results, while not as good as a faster rcnn model gives, are decent. Converting to tflite with the scripts below [6] seems to work fine this time too, but when inserted in the app, it keeps crashing. We couldn't find a difference in our first and second attempt, we tried to use the same version of Android Studio, the same app with only the frozen graph and the label map changed. Couldn't find a change in tflite versions either.

In our third attempt, we corrected the label map, which caused our app to crash. The app now starts, but doesn't detect anything. The modifications done in the example app are only the uploading of our detect.tflite and labelmap.txt files in the assets (lite\examples\object\_detection\android\app\src\main\assets) directory, and commenting out the download assets line from the build.gradle file (lite\examples\object\_detection\android\app\build.gradle).

I have uploaded the detect.tflite and labelmap files to the github link below [7]. The training files soon to be uploaded.

### References

1. [https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10#1-install-anaconda-cuda-and-cudnn](https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10%231-install-anaconda-cuda-and-cudnn)
2. <http://download.tensorflow.org/models/object_detection/ssd_mobilenet_v1_coco_2018_01_28.tar.gz>
3. [https://www.tensorflow.org/lite/models/object\_detection/overview#get\_started](https://www.tensorflow.org/lite/models/object_detection/overview%23get_started)
4. <http://benchmark.ini.rub.de/?section=gtsdb&subsection=dataset>
5. Software versions:

|  |  |
| --- | --- |
| Software | Version |
| Anaconda3 | 5.1 |
| CUDA | 10.0 |
| CUDNN | 7.4 |
| Python | 3.5 |
| Tensorflow | 1.13.1 |

1. TFlite conversion:
   1. python export\_tflite\_ssd\_graph.py --pipeline\_config\_path training/ssd\_mobilenet\_v1\_pets.config --trained\_checkpoint\_prefix C:/tensorflow1/models/research/object\_detection/inference\_graph/model.ckpt --output\_directory exported\_model
   2. tflite\_convert --input\_shape=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 --allow\_custom\_ops --graph\_def\_file=exported\_model/tflite\_graph.pb --output\_file=C:/tensorflow1/lite/examples/object\_detection/android/app/src/main/assets/detect.tflite
2. Github: <https://github.com/TitaniaTosca/TrafficSign>