Udacity Machine Learning Nanodegree

Capstone Proposal

Andreas Börzel

March 13, 2020

German License Plate Recognition

Domain Background

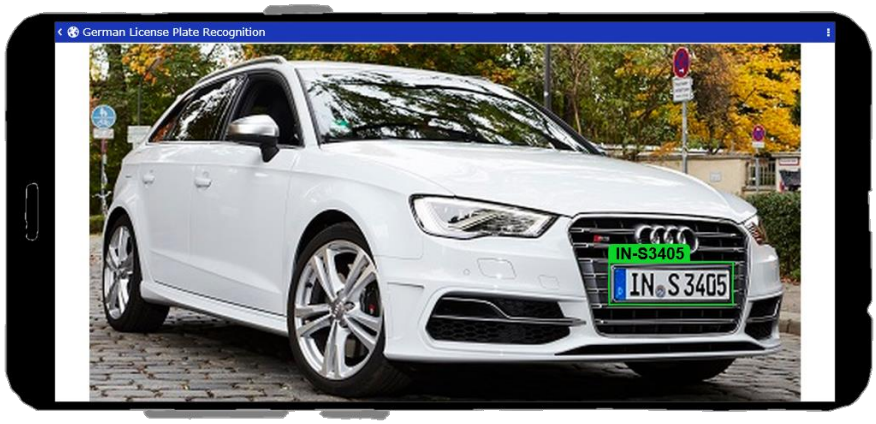
Machine learning methods are used today in many different areas. One particularly exciting and important area is machine vision, which is about the ability to understand and interpret image content with a computer. The range of tasks is roughly divided into classification, object detection and localization, segmentation and recognition. Today, machine learning, especially deep learning, makes it possible to detect, identify (identity) and localize objects within an image with the help of a computer, and even to read gestures, feelings or behavioral patterns from images - abilities that until a few years ago were reserved for humans alone.

The practical range of application of these abilities is almost unlimited, here are some examples:

- Autonomous systems (e.g. automated driving, robotics) for perception of the environment with cameras
- Industrial production, e.g. for computer-aided quality control
- Medicine, e.g. for computer-aided diagnosis of X-ray, MRT or CT images
- Public administration, e.g. for computer-based access control or personalisation for authorities, contracts, etc.
- Security, e.g. computer-aided monitoring (detection of dangerous situations)
- Mobile applications, e.g. face recognition, automatic grouping of images, etc.

Problem Statement

The goal of this project is to create a small Android app that allows to recognize the license plate of a car quickly and easily with the camera of a smartphone or tablet and translate the license into plain text. The app should mark the recognized license plate within the camera image with a bounding box and display the determined license in plain text as annotation above or below the bounding box, as outlined in the following screen mockup:



For this project, the app is limited to the recognition and display of the license text, but this is an essential basis for many practical mobile applications, such as:

- Automatic traffic controls with cameras, e.g. determination of the vehicle owner, check wanted list, ...
- Access control and documentation for parking lots or car parks
- Mobile registration of the vehicle owner during parking checks by the public order office
- Mobile registration of the license plate or vehicle holder as support in car garages, e.g. for repair or maintenance acceptance within the scope of digitalisation

For simplicity, the project is initially limited to the recognition of German license plates. A later extension to European or even worldwide license plates is possible!

Datasets and Inputs

Regarding machine learning, the project is roughly divided into two tasks, the detection and localization of the license plate and the recognition of the license as plain text. In order to train models for the two problems mentioned above, data collections with images of cars with recognizable license plates are needed. For reasons of data protection law, such images are very rare on the Internet and ready-to-use, publicly accessible data collections are not to be found at all. This means that an essential task of this project is to create 2 data collections, which can be used for the training of the license plate detection and the license recognition as plain text.

1) License plate detection and localization

For the data collection, as many pictures as possible showing cars with license plates are collected from the Internet and labelled accordingly (class and bounding box) using the LabelImg tool. LabelImg stores the label annotations in PASCAL VOC format, which can be used to create a data collection in TF-Record format, which is used by the TensorFlow Object Detection API. As a rule of thumb, the data collection should have about 1000 images per class. Although it is difficult to get so many images, the data collection should have at least 300 to 500 images to get useful results.

2) License recognition

In order to be able to train a model for a reliable recognition of the licenses, a corresponding number of images of license plates with corresponding variance are required. In the absence of these images, the data collection for this purpose is created from generated (synthetic) images of license plates, which are converted into realistic images for training using data augmentation. This procedure enables you to control the data distribution of the licenses very well. However, there is a risk that data augmentation does not correspond to the real world and that the trained model will perform poorly in practice. In order to depict realistic scenarios, influencing factors such as scaling, rotation, brightness and interference must be taken into account in data augmentation.

Solution Statement

The goal is to detect license plates within an image and to mark them with a bounding box, as well as to convert and display the license in plain text.

The first task is to detect license plates as such, this is a typical classification problem. The second task is to determine the position and dimensions of the license plate within the image in order to mark it with a bounding box, this is a regression problem. Both tasks can be solved well with the TensorFlow Object Detection API.

Last task is to convert the license into plain text, this is an OCR (Optical Character Recognition) problem, which can be solved with a RCNN (Recurrent Convolutional Neural Network).

Overall it is about supervised learning, i.e. during training the error between the labeled training data (ground truth data) and the model predictions are minimized.

Benchmark Model

Since this is a problem of its own and the models are based on data collections that have been created specifically for this purpose, there is unfortunately no possibility of direct comparison with other solutions at first. This means that to evaluate the performance, a part of the data collection is split off as test data, which is only used to evaluate performance with the common performance metrics, but not for training. Likewise, the completed mobile application is subjected to an extensive practical test with real camera images and the result is evaluated accordingly.

Evaluation Metrics

In object detection, evaluation is non trivial, because there are two distinct tasks to measure:

- Determining whether an object exists in the image (classification)
- Determining the location of the object (bounding box, a regression task)

The evaluation method used is the “mean average precision” or “mAP score”, this is a commonly accepted evaluation method for object detectors, which has also been used in object detection competitions, such as for the PASCAL VOC, ImageNet, and COCO challenges.

Here is a detailed description about the mAP score.

Project Design

The project is roughly divided into 3 tasks:

Training of a model for detection and localization of license plates in images

For the detection and localization of license plates I plan to use the TensorFlow Object Detection API. As the model will be executed from an Android app on a mobile device, the model will be trained in the first approach based on the pre-trained model `ssdlite_mobilenet_v2_coco`, as it has a good balance between `speed` and `accuracy`, see TensorFlow detection model zoo.

For the data collection about 300 pictures of German license plates and about 300 pictures of other license plates are needed to train the model to distinguish German license plates from other license plates. For this purpose, corresponding images are collected from the internet and labelled with the tool LabelImg. LabelImg stores the label annotations in the PASCAL VOC format. From the images and the label annotations a data collection in TF-Record format is generated, which is used by the TensorFlow Object Detection API for the model training.

Training of a model to transform the license plate image into plain text

The License Plate Detection above provides an image section which only contains the corresponding license plate. To transform the image section into plain text, another RCNN (Recurrent Convolutional Neural Network) model is trained. The model is created with TensorFlow / Keras and the Keras OCR example serves as a template. To achieve good performance on mobile devices, the trained Keras model is converted into a TensorFlow Lite model.

For the data collection, synthetic images of car license plates are generated using the web service heisnbrg.net. However, since realistic images are required for training, real images are simulated from the synthetic images using data augmentation. Factors such as scaling, rotation, brightness, interference, etc. must be taken into account. It is planned to use about 60000 license plates for the training. Since front and rear license plates are slightly different, both images will always be created for each license plate, i.e. the data collection will contain about 120000 images. When generating the license plates, you must ensure that there are approximately the same number of license plates for all 30 cities/counties in Germany.

The trained model gets the image section with the recognized license plate as input and delivers the license plate in plain text as output.

Creating the Android App

The last step is to create an Android app that enables the user scenario described in the “Problem Statement” section above, using the two trained models. Template for the creation of the app is the Tensorflow Lite Object Detection example app. As development environment Android-Studio and the programming language Kotlin is used.

App workflow:

Camera Image => License Plate Detection => Boundary Box => License Recognition => License Text Annotation

Tools and libraries:
Python, Numpy, Jupyter-Notebook, Pandas, Scikit-Learn, Scikit-Image, Pillow, Matplotlib, TensorFlow/Keras, TensorFlow Object Detection API, OpenVC, Android-Studio, Kotlin...