Individual Project Report

*Name*: Varun Burde (burdeva1@fel.cvut.cz)

*Course*: Individual Project BE3MPROJ8

*Supervisor*: Ing. Michal Reinštein Ph.D. (reinstein.michal@fel.cvut.cz)

*Title*: Deep neural network for classification and segmentation of Google StreetView imagery

February 2019

# **Abstract**

With the advancement computation power and large datasets finally made a huge improvement of Deep neural network leading many widespread applications. One of such application is solving computer vision problems like classification and segmentation. Also Competition like ImageNet Large Scale Visual Recognition Challenge, took the solution to next level , in some cases classification is better than Human .

This report describes the evaluation of pre-trained deep neural network on Google StreetView Images. Pretrained model used are Mask RCNN, Xception , VGG16, VGG19, Inception v3, Inception resnet v2, Resnet50, MobileNet , MobileNet v2, DenseNet, NASNet. Implementation is done in Python using Keras and TesnsorFlow-GPU framework. User interface for the application execution, processing of the input images and visualization of the results is realized using Google Colab with repository in GIT.

A pipeline is created for the task, user provide the parameters like coordinates, heading, field of view for the Google StreetView API ,Python script downloads the available images to that location ,Pre Processing of Images to fit into classifier is done, Classification is done on the Images depending on the architecture of the Neural Network.

The results are evaluated based on architecture of different network. Mask RCNN turns out to be best performer among all other Pre-trained model cause of its architecture of performing classification and segmentation side by side. Still there were some misclassification of objects which is visualized in report and the solution which might get the network to get better results.

# **Acknowledgement**

I would like to acknowledge the help of all of those which made this project possible. I would like to express my deep gratitude to my supervisor **Ing. Michal Reinštein, Ph.D** , for his time ,patience, guidance and also for allowing the idea to be persuaded originally, and made this project successful. I would also like to thank Jana Zichová who let me to register for this project.

Furthermore, I would like to thank to all those people who works on all open source projects mentioned in reference, and all wonderful people who post the discussion and blogs for all useful learning resources, specially to Stanford open course CS231n: Convolutional Neural Networks for Visual Recognition which. I am also thankful for Google free resources like Google Collab laboratory and Street View API which made the project possible.

**Contents**

[**Abstract** 2](#_Toc536402859)

[**Acknowledgement** 3](#_Toc536402860)

[**1.** **Introduction** 6](#_Toc536402861)

[I. Aims and objective of the Project 6](#_Toc536402862)

[II. Overview of the project progress 6](#_Toc536402863)

[III. Overview of report 6](#_Toc536402864)

[**2.** **Background Research** 7](#_Toc536402865)

[**3.** **Implementation** 8](#_Toc536402866)

[Google Street view Api 8](#_Toc536402867)

[Keras 8](#_Toc536402868)

[Tensorflow 9](#_Toc536402869)

[Pre trained model in Keras 9](#_Toc536402870)

[xception, 9](#_Toc536402871)

[vgg16, 9](#_Toc536402872)

[vgg19, 9](#_Toc536402873)

[resnet50, 9](#_Toc536402874)

[inception\_v3, 9](#_Toc536402875)

[inception\_resnet\_v2, 9](#_Toc536402876)

[mobilenet, 9](#_Toc536402877)

[mobilenet\_v2, 9](#_Toc536402878)

[densenet, 9](#_Toc536402879)

[nasnet 9](#_Toc536402880)

[Mask RCNN 9](#_Toc536402881)

[Google Colab 9](#_Toc536402882)

[**4.** **Evaluation** 10](#_Toc536402883)

[**5.** **Future Scope** 11](#_Toc536402884)

[**6.** **Conclusions** 12](#_Toc536402885)

[**7.** **References** 13](#_Toc536402886)

[**8.** **Appendix** 14](#_Toc536402887)

[**List of Tables** 15](#_Toc536402888)

[**List of Figures** 16](#_Toc536402889)

# **Introduction**

## Aims and objective of the Project

The aim is to design and implement deep neural network-based solution for online semantic segmentation of Google StreetView images. The proposed software solution should allow the user to request Google StreetView imagery for any given location and output scene description as a list of detected objects (with confidences) and pixel-wise scene segmentation mask. User interface for the application execution, processing of the input images and visualization of the results should be realized using Google Colab and should be easily executable (handling required packages and dependencies on code repositories). Pre-trained existing models should be explored first, thorough experimental evaluation on publicly available datasets should follow. Comparison with related state-of-the-art work is integral part of the project and should be presented in the final report. Recommendation: implementation should be done in Python, using Keras and TensorFlow frameworks.

## Overview of the project progress

Project is done in small steps ,there were many unexpected results and problems faced during projects ,this section briefly describe about steps of implementation ,problem faced and and future scope

### Pipeline

1. Use the Google Streetview API to get the Images
2. Preprocess the Images to fit for the model
3. Load the model
4. Get the results
5. Compare the results
6. Finding the misclassification

### Problems faced during project

1. Preprocessing of image
2. Street view api key
3. Fixing the parameter of Google street view api
4. Implementation of script in Google colab

### Future Scope

## Overview of report

This report fully describes the work done in this project along with basic learning of State-of-the-art Neural Networks

1. Introduction: This gives the Introduction of project, its aim and overview of the project work done.
2. Background Research: Analysis of project done in Image segmentation and attempt to define what is state of art. Description of revolution of Computer vision with time.
3. Implementation: The description of how the implementation of the software solution is done in steps
4. Evaluation: The misclassification and other factor which gives the evaluation of the pre-trained model
5. Future Scope: This project will be continued as Master thesis to design and implement the State of art network which will outperform the results of pre-trained model got from the evaluation.
6. Conclusion: Results about how the implementation should take place and what changes should be made to get better results

Appendix – A list of all packages used in the script and the list of all pretrained model used to get the results on the images .

# **Background Research**

The project describes about the two sub computer vison problem

Image Classification :Classification is process of classifying the object or categorizing objects in some class .In Computer vision and machine learning, It’s the task of recognition of object in which pre-determined class it belongs .It generally described by class label along with the confidence level .



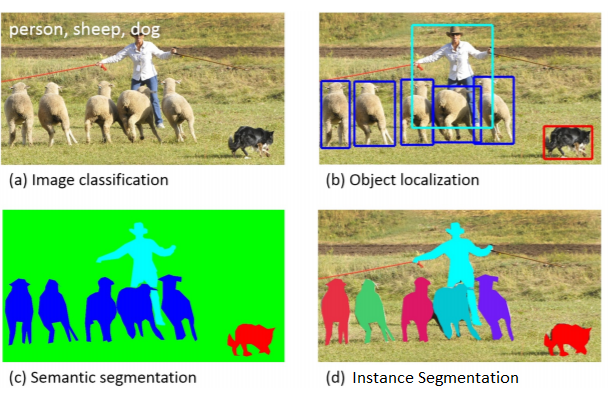
Figure 1 Example of Image classification

Sementic Segmentation: In computer vision ,Image segmentation is process of partitioning of digital image into multiple segment .Having different Segments of Image with different boundaries between multiple objects make it easier to analyze and differentiate



Figure 2Example of Image segmentation

4)Instance segmentation: is the combination of sub problems of Object classification, Object Localization and Semantic segmentation. Combing all together we get bounding box from object localization and a mask pixel wise segmentation .



It would be out of scope to how the neural network work for this project, but it would be interesting to know how convolution neural network make this computer vison problem possible.

For a start we can think of every image as a matrix

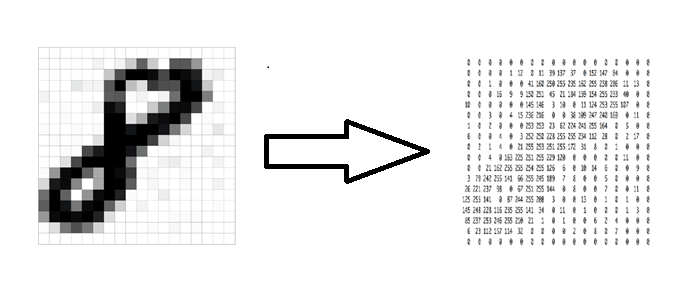


Figure 3 Image as matrix

More typically three channel colored picture can be considered as 3-dimensional array with intensity of color ranging from 0 - 255.

As any other Neural Network Convolutional network have layer inside the deep architecture specifically have convolutional layer, pooling layer

Convolutional layer: what it does is transform input into form of some filter and pass it to other layer

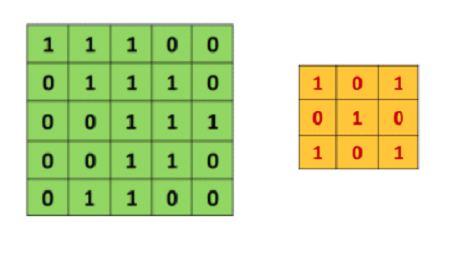


Figure 4 Image and filter matrix

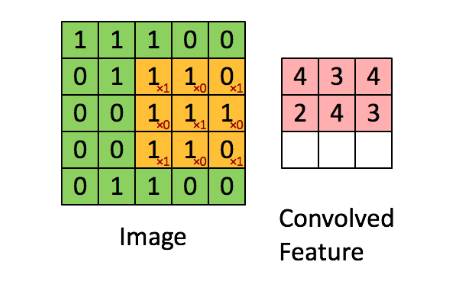


Figure 5 resulted convoluted Feature

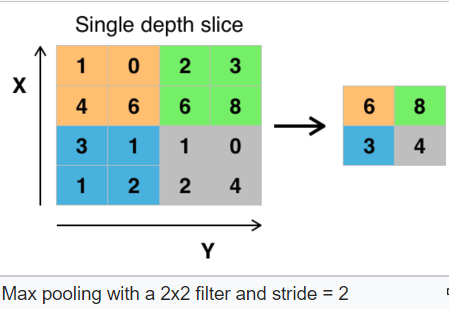
This convolved Feature can be imagined as small filter like line, curve, circle etc. detector which become more and more detailed as we go deeper and deeper into networks. These convolved features also called Feature map which is the main detection unit of neural network.

Feature map is controlled by some parameters which can really decide about how the Feature map will be generated and hence the performance of our Neural Network.

Depth: There could be more than one 2d Feature map back to back arranged together. The number of 2d matrices together is the depth.

Stride: Stride is number of pixels we slide our filter matrix over input matrix.

Pooling layer: One of the important concept of convolution network is pooling, which is form of down sampling .Most common nonlinear function to implement pooling is max pooling in which max value from the frame is taken and move forward.



Relu Layer: Relu stands for Rectified Linear unit and it’s a non- linear operation. Its purpose is to replace all negative pixel values from feature map by zero and make the operation less computational expensive. Since most of the real-world data we feed in CNN would be non-linear, so removing the negative pixel value will introduce non-linearity.

Fully connected layer: Its like the same layer from basic perceptron in which every neuron in previous layer is connected to every neuron of next layer. It also consists of SoftMax activation function at the output layer which finally classify the class on their scores.

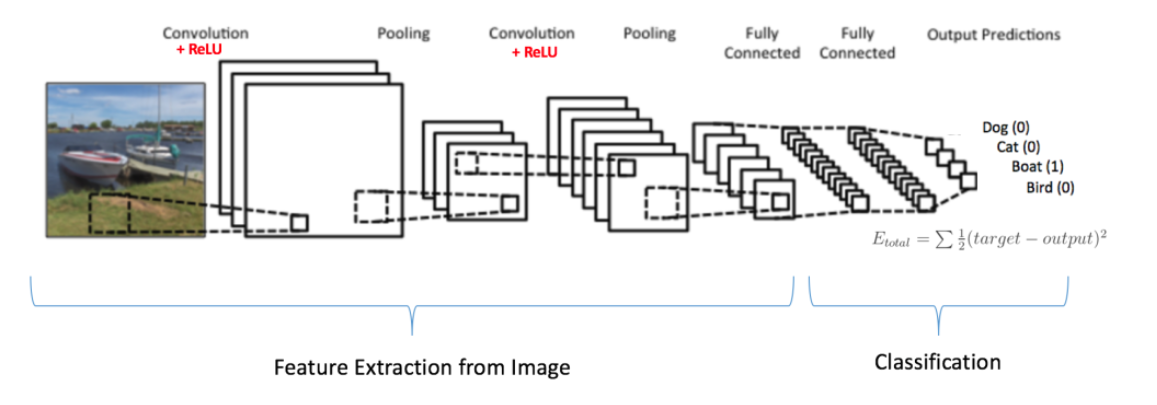


Figure 6 Convolutional network structure

## Keras

Keras Is High Level Neural network API, which run on top of TensorFlow, Thaneo or CNTK. It a simple to use open source deep learning library which makes the Implementation of Neural Network much easier. It can be use to create architecture, load/save model, training of neural network in easy way. Also it have support for CPU and GPU as well TPU in case we use Google Colab. It’s a powerful library with lots of option in tweaking the parameter of layer, loss functions etc.

## Tensorflow

TensorFLow is open-source software library for developed by Google Brain Team. It have well documentation along with example source code to start with. TensorFlow is a symbolic math library for dataflow programming in which data flow is represented by Tensor .The most advantageous feature about TensorFlow is visualization tool Tensor Board which have features like What if and Visualization Graphs which can make a work lot easier in optimizing the learning of Neural network.

## Pre trained model in Keras

### xception,

### vgg16,

### vgg19,

### resnet50,

### inception\_v3,

### inception\_resnet\_v2,

### mobilenet,

### mobilenet\_v2,

### densenet,

### nasnet

## Google Colab

One of the important part of task was to realize the developed pipeline in Google Collab. It’s a free research tool for machine learning education and research tool which uses Jupyter notebook Environment which can be rum maximum of 12 hours in single go, which makes it not proper tool for training of Neural network, though all evaluation and testing are done in the Google Colab.

Good thing abour Google colab is it have support for deep learing application like Keras, TensorFLow ,Pytorch and Opencv .This make task easier and other dependencies could be easily installed using pip installer .

## Mask RCNN

ROI Align

# **Implementation**

## Google Street view Api

The First part of the implementation of this project was to use Google Street view Api to download Images for any given location and store it for further detection and Image processing.

Google street view Api gives user many parameters to work with, example of parameters is given below,

apiargs = {  
 **'location'**:**'50.100471, 14.392636'**,  
 **'size'**: **'640x640'**,  
 **'heading'**: **'0;45;90;135;180;225;270'**,  
 **'fov'**: **'90'**,  
 **'key'**: **XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'**,  
 **'pitch'**: **'0'**}

Parameters were chosen on basis will give us the best images with orientation, less noise and perfect fit for our model to test. For ex – having pitch to -90 or 90 gives the images which make no sense.

Figure 7 pitch with 90 Figure 8pitch with -90

Google street view Api requires Developer key to access the resources by Google, higher resolution images can be downloaded using premium plan.

The task was to find the images which not only consist of normal streets consist of vehicle, traffic lights on which Models were trained but also park, public and general places which can get us the proper evaluation of Detection. To start working with API, it needs to install and import in the script.

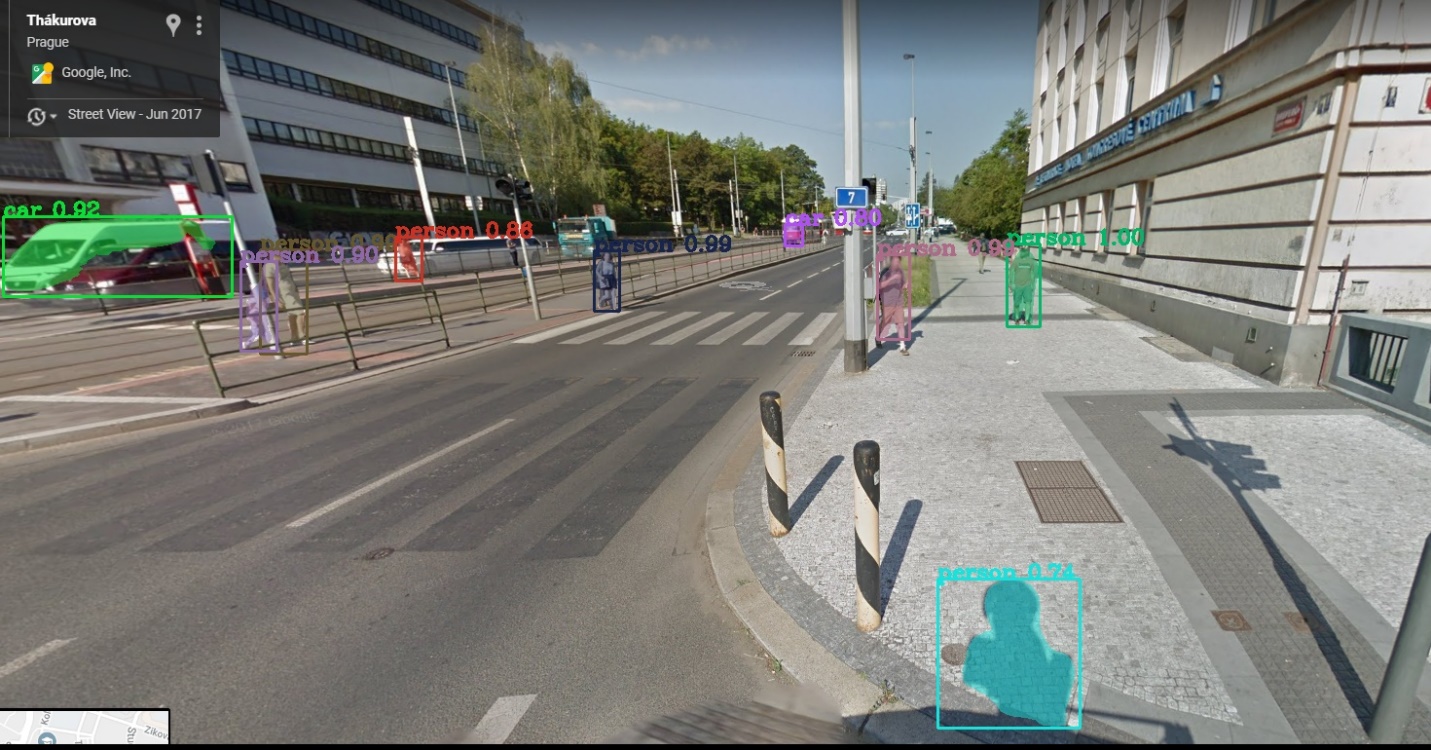
# **Evaluation**

Here are the Results of some misclassified Images with in Google Street Downloaded Images





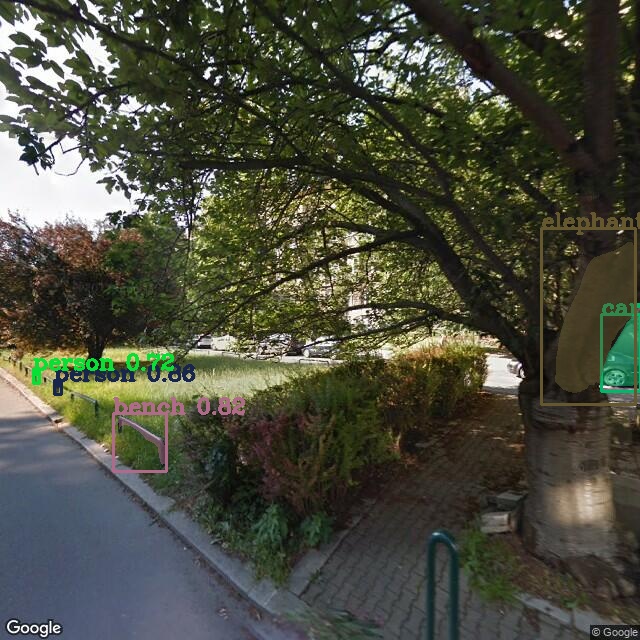
















# **Future Scope**

This part of report tells about what can be done with model to make it more accurate in google street view.

## Transfer learning

# **Conclusions**

# **References**

# **Appendix**

# **List of Tables**

# **List of Figures**