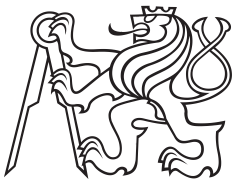
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Deep neural network for city mapping using Google Street View data

Varun Burde

Supervisor: Ing.Michal Reinštein,Ph.D.
May 2019

Acknowledgements

Declaration

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[1] 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[2]. Pretrained model of Mask RCNN[3] for matterport [].Implementation is done in Python[4] 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[5] with repository in GIT[6].

A pipeline was created for the task, user provide the parameters like coordinates for the Google StreetView API [2]. Python[4] script downloads the available images to that location then images were pre-processed to fit into classifier. Classification is done on the Images depending on the architecture of the Neural Network.A vectorized map is generated as a result of classified Images and put into map using Folium.

Map is represented in various form as heat-map for better understanding of scenario. Object markers in map are the objects classified with the Mask RCNN .

Keywords:

Supervisor: Ing.Michal Reinštein,Ph.D.
E225b,
Karlovo nam. 13,
121 35 Prague 2,
Czech Republic

Abstrakt

Rozvíjíme ...

Klíčová slova:

Contents

1 Introduction 1
1.1 Aim and objective of project ... 1
1.2 Overview of Project ... 1

2 SOTA 3

3 Theory 5
3.1 KERAS ... 5
3.2 TensorFlow ... 5
3.3 Google Colab ... 6
3.4 Mask R-CNN ... 6

4 Methodology 9
4.1 hardware and software setup ... 9
4.1.1 hardware setup ... 9
4.1.2 list of libraies used ... 9
4.2 Pretrained model details ... 10
4.3 Google street view ... 10
4.3.1 Basic overview ... 10
4.3.2 Parameters in Street view ... 10
4.3.3 Selection of parameters ... 11
4.4 Mask Rcnn ... 11
4.4.1 Description about Mask Rcnn 11
4.4.2 How Its implement in python with Keras and tensorflow backend with citiation ... 12
4.4.3 Visulization of output from Mask R-CNN ... 12
4.4.4 Classes used in map ... 13
4.4.5 some drawback of Mask R-CNN during classification ... 14
4.5 Google Colab ... 14
4.5.1 basic overview ... 14
4.5.2 advantages of using this ... 14
4.5.3 changes needed to implement this ... 14
4.5.4 how it improved the performance ,types of runtime ... 14
4.6 Folium ... 14
4.6.1 basic overview and its features 14
4.6.2 how i used it in project ... 14
4.6.3 classes and types i object I marked in the map and why did I skipped others ... 14
4.6.4 types of representation of maps ... 14
4.7 experimentation ... 15
4.7.1 use of depth images ... 15
4.7.2 difference in usage of depth in mapping and without depth ... 15
4.7.3 Algorithm about how the objects are localize in the 2d map 15
4.7.4 how i tackle with the localization of same object ... 15
4.7.5 algorithm to find the proper image under the polygon ... 15

5 Implementation 17

6 Experimentation evaluation 19
6.1 performance of Mask R-CNN on GSV dataset ... 19
6.2 performance of GSV Images ... 19
6.3 performance in maps ... 19
6.4 experimentaion of overpass api to get the street coordinates ... 19
6.5 use of depth Images ... 19

7 Conclusion 21
7.1 advantages ... 21
7.2 drawback ... 21
7.2.1 neccesary condition to have results ... 21
7.3 how it is similar with other sota projects above ... 21
7.4 google pricing ... 21
7.4.1 Google colab ... 21
7.4.2 google api ... 21
7.5 will training on other data set might changed results ... 21

8 Bibliography 23

Bibliography 25

Figures

3.1 Head architecture of Mask R-CNN 7

4.1 street view structure 10
4.2 List of classes 10
4.3 parameters of GSV images 11
4.4 pitch with 90 12
4.5 pitch with -90 12
4.6 Configurable parameter in Mask R-CNN 13
4.7 wronf classification with good confidence 13

Tables

Chapter 1

Introduction

1.1 Aim and objective of project

The aim is to design, implement and experimentally evaluate a deep neural network based solution for city mapping using Google Street View images. The proposed software solution should allow the user to request Google Street View imagery for any given location specified as geojson, perform analysis and feature extraction using deep neural network(s) and output vectorized description projected and visualized over an underlying map. User interface for the application execution, processing of the input images and visualization of the results should be realized using Google Colab to utilize Google TPUs. Existing pre-trained 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 work and should be presented in the final thesis. Recommendation: implementation should be done in Python, using Keras and TensorFlow frameworks.

1.2 Overview of Project

The project is software solution with the aim of achieving the task described above. There are few important part of the project which is implemented as a individual task in order to make it easy to debug.

The core implementation of project is depends on the Google Street View [2] imaginary. The API consist of helper function and parameter to get the images at certain location available in Google street view.Task was to Use API with Python [4] script to download the required images under the area by the coordinates provided .

Then this Images are processed with Deep Neural Network for classification and segmentation of objects, Mask R-CNN [3] with pre-trained model consist of 82 classes from matterport [**?**]s used . The resulted Images are classified and segmented with the bounding box with there confidence. For purpose of project ,the confidence threshold is kept 0.7 for classification.

Experimentation is done to localized the object in map.Map is created with Folium [**?**]nd represented in different ways ,the detected objects are put with

different marker in the map and view is represented as heatmap to get better understanding of density of object and its type in the scenario

Chapter 2

SOTA

Smart way of Google street view images[2] as data set and using Machine learning algorithm specifically deep learning to develop a software solution has been seen in last few years. some such application are [7] where images of house from GSV is annotated manually and and this data is use to predicts car accident of its resident. Other such application is taking use of GSV and satellite images to extract features like age,size and accessibility using Deep Neural Network to estimate the house prices.

With the buzz of autonomous driving Layered Interpretation of Street View Images [**?**] was developed in Mitsubishi Electric Research Labs with deep neural network on GSV images.In paper ,they proposed a layered street view model to encode both depth and semantic information on street view images for autonomous driving. They propose a 4-layer street view model, layers encode semantic classes like ground, pedestrians, vehicles, buildings, and sky in addition to the depths. The only input to our algorithm is a pair of stereo images. Deep neural network was used to extract the appearance features for semantic classes.

Another example of application developed with GSV and deep neural network is Building instance classification using street view images[8] ,Land-use classification based on spaceborne or aerial remote sensing images has been extensively studied over the past decades. Proposed a general framework for classifying the functionality of individual buildings. The proposed method is based on Convolutional Neural Networks (CNNs) which classify facade structures from street view images, such as Google StreetView[], in addition to remote sensing images which usually only show roof structures. Geographic information was utilized to mask out individual buildings, and to associate the corresponding street view images. In addition, the method was applied to generate building classification maps on both region and city scales of several cities in Canada and the US

One of the such exampple which is similar to work done in the project is Automatic Discovery and Geotagging of Objects from Street View Imagery [**?**] This paper describe about solution to localize the object from multiple view using geometry.To geolocate the object in image they developed Markov Field model to perform object triangulation.They use 2 SOTA FCNN for semantic segmentation and monocular depth estimation.The geolocalization

is done with Google street view images with Triangular based MRF model descriped in paper. End Result of the projects looks similar to the output of the resulting output but lacks the precision which is done with the help of depth image and Triangulation-based MRF model.The end result is a vectorized map with overlay of tags in map.

Chapter 3

Theory

3.1 KERAS

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. For the usage in project, it allows user to use the model of different architecture just by defining class and pretrained weights in arguments. For the purpose of project pretrained weights which were trained in ImageNet database were used . There are total of 1,281,167 images for training. The number of images for each (category) ranges from 732 to 1300. There are 50,000 validation images, with 50 images per category. There are 100,000 test images. All images are in JPEG format. Also, it has 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.Below Image is example of how to make a model class in python using the Imagenet weights.

have to add picture

3.2 TensorFlow

TensorFlow is an interface for expressing machine learning algorithms, and an implementation for executing such algorithms. A computation expressed using TensorFlow can be executed with little or no change on a wide variety of heterogeneous systems, ranging from mobile devices such as phones and tablets up to large-scale distributed systems of hundreds of machines and thousands of computational devices such as GPU cards. For the purpose of project TensorFlow GPU backend is used to do all the computation to accelerate the progress cause its highly compatible with the Google colab and Keras [14].

3.3 Google Colab

It's a free research tool for machine learning education and research tool which uses Jupyter notebook Environment which can be run maximum of 12 hours in single runtime. Which can be useful in evaluation and testing cause of high-end Hardware but on the other side makes it not proper tool for training of Neural network. Google colab have support for deep learning application like Keras,TensorFLow ,Pytorch and Opencv[28] .This make task easier and save time for installation and configuration of all libraries .Also the other dependencies could be easily installed using pip installer . It also allows 3 different kind of runtime Hardware's like GPU, CPU and TPU which can accelerate the process. It just needs little changes from the developed code in local machine to work in Colab.

3.4 Mask R-CNN

Mask R-CNN[3] is state of the art convolution Neural network which can do Instance segmentation. We start with Faster R-CNN[29] .It consist of two stages the first Region proposal Network(RPN) stage scans the image and generates proposals (areas likely to contain an object) and other stage which is essence of fast R-CNN[30] classifies the proposals and generates bounding boxes and masks. Anchors are the fixed bounding boxes of defined shape and sizes which are places into images which will be used for reference when localizing the object in the image. Region Proposal Network A Region Proposal Network (RPN) takes an image (of any size) as input and outputs a set of rectangular object proposals, each with an object score with the help of anchors. The offsets of the image from the anchors is taken as input and propose an object location in Image. Second stage of feature extraction extracts features using RoIPool from each candidate box and performs classification and bounding-box regression. Output of Regression gives determines the a predicted bounding box in form of x,y,w,h (x coordinate , y coordinate , width ,height) and output of classification is a probability whether the predicted bounding box contain an object or not. Along with these two stages, In second stage of Mask R-CNN in parallel to predicting he class and box offset it also outputs a binary mask for each Region of Interest Bilinear interpolation is an extension of linear interpolation for interpolating functions of two variables

Roi Align: RoIPool is a standard operation for extracting a small feature map from each RoI. RoIPool first quantizes a floating-number RoI to the discrete granularity of the feature map, this quantized RoI is then subdivided into spatial bins which are themselves quantized, and finally feature values covered by each bin are aggregated (usually by max pooling). In each ROI bin, the value of the four regularly sampled locations are computed directly through bilinear interpolation. Thus, avoid the misaligned problem.

Network Architecture: Mask R-CNN have multiple architecture 1) Con-

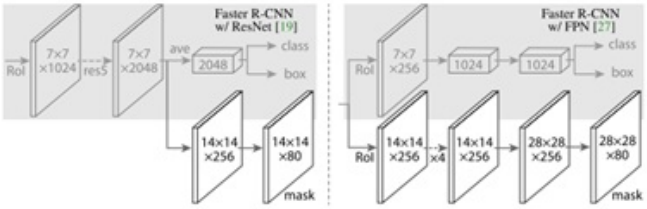


Figure 3.1: Head architecture of Mask R-CNN

volution backbone architecture used for feature extraction over an entire image and 2) network head for bounding box recognition(classification and regression) .

Figure 3.1 shows the head architecture of Mask R-CNN with resnet as Backbone.

Backbone: Its standard convolution network that serve as feature extractor, convolution network Resnet50 with the introduction of Feature Pyramid Network [9]. Feature pyramid Network: The Feature Pyramid Network (FPN) was introduced in Mask R-CNN as an extension that can better represent objects at multiple scales. FPN improves the standard feature extraction pyramid by adding a second pyramid that takes the high-level features from the first pyramid and passes them down to lower layers. By doing so, it allows features at every level to have access to both, lower and higher-level features.

Chapter 4

Methodology

4.1 hardware and software setup

4.1.1 hardware setup

The uses the Tensorflow gpu backend along with Nvdia cuda dnn ver ..... and python as a scripting language. The local hardware used with GPU unit Nvdia Gtx 1060 and CPU unit - intel core i5 6300 CPU

4.1.2 list of libraies used

Many different python packages used for different purpose .

Google street view

Google street view library is used for the using google street view api which initiate the url request to the server with the parametes and developer key to download the images and metadata.

Figure 4.2 structure of how google street view api works

Folium

This package allow to create a world map place the marker of classified and detected objects into 2d world map with overlay of markers and also to create heatmaps

TensorFlow

This package enables user to create high dimensional data type which can be efficiently use to perform high computation process on huge and dense data

Keras

This package enable user to define the architecture of the Deep Neural network, defining the order of different layer there activation functions ,learning rate etc.

```
Google Street View Image API      <-- API for Street View Images
            |
    google_streetview.api         <-- URL Request with query string
            |
         request                  <-- Download URLs and images
```

Figure 4.1: street view structure

```
class_names = ['BG', 'person', 'bicycle', 'car', 'motorcycle', 'airplane',
               'bus', 'train', 'truck', 'boat', 'traffic light',
               'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird',
               'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear',
               'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie',
               'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball',
               'kite', 'baseball bat', 'baseball glove', 'skateboard',
               'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup',
               'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple',
               'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza',
               'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed',
               'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote',
               'keyboard', 'cell phone', 'microwave', 'oven', 'toaster',
               'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors',
               'teddy bear', 'hair drier', 'toothbrush']
```

Figure 4.2: List of classes

4.2 Pretrained model details

The architecture and pre-trained model use in this project is taken from Matterport [10].The pre-trained model was trained on COCO [11] data set with 81 classes.

Figure 4.2 shows the list of classes which can be used with pre - trained model caption.

4.3 Google street view

4.3.1 Basic overview

Street View, by Google Maps, is a virtual representation of our surroundings on Google Maps, consisting of millions of panoramic images. Street View's content comes from two sources - Google and contributors. Through our collective efforts, we enable people everywhere to virtually explore the world.[2]

Images form street view was the major dataset for the project.Street view images are in form of 360 panorama,for the purpose of projects 640x640 image is used with field of view 90 to process the DNN efficiently.

4.3.2 Parameters in Street view

Google street view api uses python sent a url request to server in order to download the images , request is in form of string made up of parameters needed to fulfill the query. server returns back with the metadata with the information of images such as date , location in form of latitude and lattitude , panorama id ,status file name.

parameter string is in form of python dictionary with certain key like location , size ,heading , field of view, pitch and developer key . Developer

```
# 50.10291748018805, 14.39132777985096              dejvice
# 50.0795436,14.3907308                             Strahov
# 50.0746767,14.418974                              Karlovo namesti
apiargs = {
  'location': '50.0753397,14.4189888 ; '
              '50.0795436,14.3907308 ;'
              '50.10291748018805, 14.39132777985096 ;'
              '50.0753575,14.4060179',
  'size': '640x640',
  'heading': '0;45;90;135;180;225;270',
  'fov': '90',
  'key': '[redacted]',
  'pitch': '0'
}
```

Figure 4.3: parameters of GSV images

key is one of the most important key in parameter to make this Google street api works.

Figure 4.3 Example of street view api parameters to showing apiargs dictionary with key and values .

4.3.3 Selection of parameters

Location describes the coordinates of location in form of numbers of latitude, longitude. Multiple location can be fitted in arguments. If no image is available then API will return a generic image with text “Sorry, we have no imagery here." Size is the resolution of downloaded image. Highest resolution can be downloaded is 2048x2048 which is available with google premium account services. In project standard 640x640 image is used for the classification.

Heading indicated the compass heading of the camera, accepted values are 0 to 360. North is indicated by 0. seven different heading angles were chosen for project cause each angle to have something different scenario along with some similarity with adjacent image makes easy to detect the confused objects.

FOV is set to 90 for the project for optimum setting to avoid pixilation in case of zoom in and to less info per pixel in case of zoom out. For the privacy purpose the developer key is hidden.

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 headed up to sky and bottom to floor.

Figure 4.5 and 4.5 Example images with pitch 90 and -90.

4.4 Mask Rcnn

4.4.1 Description about Mask Rcnn

Mask R-CNN[3] is state of the art Deep neural network proposed by Facebook research scientist Kaiming HE in 2017 which can perform instance segmentation and object detection together.I can have several different backbone architecture like Inception V2 ,ResNet 50 ,Resnet101 and Inception-Rennet2.For project ,the pre-trained model on COCO dataset with Resnet

Figure 4.4: pitch with 90

Figure 4.5: pitch with -90

101 architecture was used.Mask R-CNN with Resnet-101-FPN backbone was able to outperform other state of the art network like MNC and FCIS winner of COCO 2015 and 2016 segmentation challenge.

4.4.2 How Its implement in python with Keras and tensorflow backend with citiation

Implementation of architecture of Mask R-CNN was done in Keras framework and TensorFlow backend ,Its taken from the open source repository from Matterport [10] .Implementation of Mask R-CNN is done in python and can be seen in script model.py ,It consist definition of all the layer and architecture of Mask R-CNN in TensorFlow.

4.4.3 Visulization of output from Mask R-CNN

Implementation of creation of masks and image processing is done with the Opencv [12].The output from the results are in form of image with overlay of text(class name) with number(confidence of being of that class,Form 0 to 1 ,being 1 means full confidence) bounding box to localize the classified object in Image and mask with color to cover pixel wise segmentation of detected object in image.

```
BACKBONE                        resnet101
BACKBONE_STRIDES                [4, 8, 16, 32, 64]
BATCH_SIZE                      1
BBOX_STD_DEV                    [0.1 0.1 0.2 0.2]
IMAGE_MAX_DIM                   1024
IMAGE_META_SIZE                 93
IMAGE_MIN_DIM                   800
IMAGE_MIN_SCALE                 0
IMAGE_RESIZE_MODE               square
IMAGE_SHAPE                     [1024 1024    3]
LEARNING_MOMENTUM               0.9
LEARNING_RATE                   0.001
POOL_SIZE                       7
DETECTION_MAX_INSTANCES         100
DETECTION_MIN_CONFIDENCE        0.7
VALIDATION_STEPS                50
NUM_CLASSES                     81
```

Figure 4.6: Configurable parameter in Mask R-CNN

Figure 4.7: wronf classification with good confidence

Parameters in Model

With ResNet 101 as back bone architecture, model provide several other parameters tweak to fit for application .Some of the easily configurable parameters are Minimum confidence detection , learning momentum , learning rate .etc

Figure 4.6 shows some of the configurable parameter of Mask R-CNN .Out of which number of classes depends on the pre trained model during testing. Minimum confidence decide creation of bounding box if the minimum 0.7 confidence is achieved at classification of that object.

4.4.4 Classes used in map

As the end result of project results in heatmap of the objects detected in streets and the pre-trained model consist of several classes with doesnt show any significance to result ,rather can make some false results in order to avoid those. Only few classes were used for heatmap which describe the scenario of the area under test.

Figure 4.7 show example of class knife which is classified wrong and make less significance in street view images.

4.4.5 some drawback of Mask R-CNN during classification

misclassification and correct classification how I dealt with too classification and eliminate the wrong one which parameters I changed

4.5 Google Colab

4.5.1 basic overview

4.5.2 advantages of using this

4.5.3 changes needed to implement this

4.5.4 how it improved the performance ,types of runtime

4.6 Folium

4.6.1 basic overview and its features

4.6.2 how i used it in project

4.6.3 classes and types i object I marked in the map and why did I skipped others

4.6.4 types of representation of maps

will describe the representation in form of heat-map , markers and what different kind of Information I can get from different representation

4.7 experimentation

4.7.1 use of depth images

4.7.2 difference in usage of depth in mapping and without depth

4.7.3 Algorithm about how the objects are localize in the 2d map

4.7.4 how i tackle with the localization of same object

4.7.5 algorithm to find the proper image under the polygon

like why i need proper image and what will happen if i dont use proper images

Chapter 5

Implementation

Project is done is small steps to maintain flow, simplicity, under stability and easy to debug. These small parts are represented in Google colab by small cells which can be run individually and have the output alone. Pipeline is created to explain the flow of data and process.

description of pipeline

5.1 downloading of libraries

important libraries to download in google colab

5.2 downloading mask Rcnn model

downloading and preparing mask Rcnnn. how to download it

5.3 preparing google street view api

description of algorithm to download the images under area

5.4 visuallizing the results

outputs from mask rcnn

5.5 algorithm of using follium to put marker in map

description of how to avoid the duplicate items in map

5.6 visualization of maps

visualization of detection in form of marker in maps and heatmaps , how to do that

Chapter 6

Experimentation evaluation

6.1 performance of Mask R-CNN on GSV dataset

table on number of detection ,number of no detection ,number of false detection small pie chart graph of number of detection .number of false detection , cases of no detection .

6.2 performance of GSV Images

availability of images, costing per imagae by google , problems in street view images ,few example of them

6.3 performance in maps

Real number of objects in map vs number of marker in the map ,there difference .

6.4 experimentaion of overpass api to get the street coordinates

its drawback , kiind of results ,its usability

6.5 use of depth Images

use of depth for creing maps , its difference with simple algorithm

Chapter 7

Conclusion

7.1 advantages

7.2 drawback

7.2.1 neccesary condition to have results

7.3 how it is similar with other sota projects above

7.4 google pricing

7.4.1 Google colab

7.4.2 google api

7.5 will training on other data set might changed results

Chapter 8

Bibliography

Bibliography

[1] Stanford Vision Lab, "Large Scale Visual Recognition Challenge (ILSVRC)," 2015.

[2] Developers.google.com, "Developer Guide," 2018.

[3] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 2980–2988, 2017.

[4] Python Software Foundation, "Python," 2019.

[5] Colab.research.google.com, "Welcome to Colaboratory!."

[6] S. F. Conservancy, "git –fast-version-control."

[7] K. Kita-wojciechowska and E. Sciences, "Google Street View image of a house predicts car accident risk of its resident,"

[8] J. Kang, M. Körner, Y. Wang, H. Taubenböck, and X. X. Zhu, "Building instance classification using street view images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 145, pp. 44–59, 2018.

[9] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 936–944, 2017.

[10] W. Abdulla, "Mask r-cnn for object detection and instance segmentation on keras and tensorflow." `https://github.com/matterport/Mask_RCNN`, 2017.

[11] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft {COCO:} Common Objects in Context," *CoRR*, vol. abs/1405.0, 2014.

[12] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, "A brief introduction to OpenCV," *MIPRO, 2012 Proceedings of the 35th International Convention*, pp. 1725–1730, 2012.