Number Plate Recognition Using SVM and Neural Networks (pg.161-pg.188; 28 pages)

My ideas :
- as this is for Spain. Try for Germany.
- find some more number plates of Spain and try to add it to the available dataset

- Algorithms used: Support Vector Machines, Artificial Neural Networks

- Topics covered: Auto(ANPR), Plate detection, Plate recognition

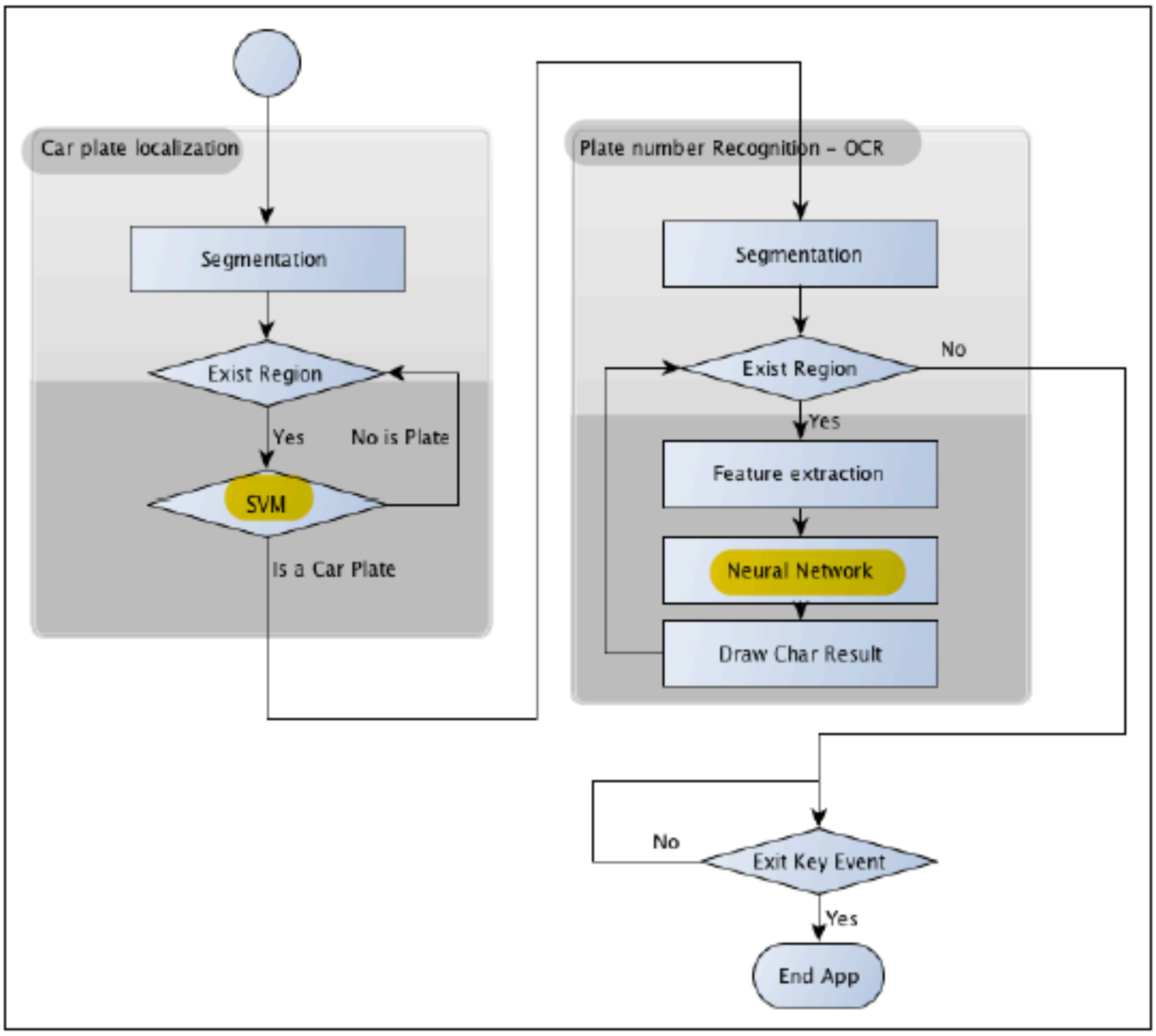
- Surveillance method: Optical Character Recognition - easy, clean and minimise errors

- Light principle: Retro-reflection

- Number plate country: Spain
  - Dimensions: license plate 520 x 110 mm , two groups of characters separated by 41mm space and 14 mm width between two characters, first group has 4 number values, second has 3 letters without vowels, dimensions of all characters is 45 x 77 mm

- ANPR algorithm
  - Two main steps: plate detection and plate recognition
  - Plate detection: detect plate in the whole camera frame
  - Plate recognition: OCR algorithm to determine the alphanumeric characters

- Pattern recognition algorithm:
  - step 1: Segmentation - detects and removes the region of interest
  - step 2: Feature extraction - extract set of characteristics
  - step 3: Classification - classify each character



- Two more important tasks:
  - how to train a pattern recognition system
  - how to evaluate such a system

- **<u>Plate detection:</u>**
  - detect plates in a camera frame : 1. Segmentation 2. Segment classification
  - 1. Segmentation : filters, morphological operators, contour algorithm - retrieve parts of image that could have a plate

- 2. Segment classification: apply SVM to each image patch(feature)- train with plate and non plate classes

- Segmentation:
  - dividing image into multiple segments
  - simplify the image for analysis and make feature extraction easier
  - lots of vertical edges present in number plates
  - eliminate regions that don't have vertical edges
  - remove noise by applying a gaussian blur or else you get fake vertical edges
  - apply sober filter
  - apply threshold using OTSU's method
  - closing morphological operation - possible regions that contain plates - most won't contain plate
  - find connected components using contours algorithm
  - draw minareaRectangle around the contours found
  - make preliminary validations while drawing rectangles - check if they are proper(needed) rectangles - check area and aspect ratio - 520/110 = 4.727272 with error margin of

40%
- https:/
www.dipolnet.comlicense_plate_recognition_lpr_systems__part_1_camera_positioning_bib318.htm
- more info on the formulae and its construction
- the code to compare the plate detected to our set limits is decoded. Check the PDF comments for explanation
- more code was given. I studied it and tried to understand what it meant. Understood some and should observe the result while coding

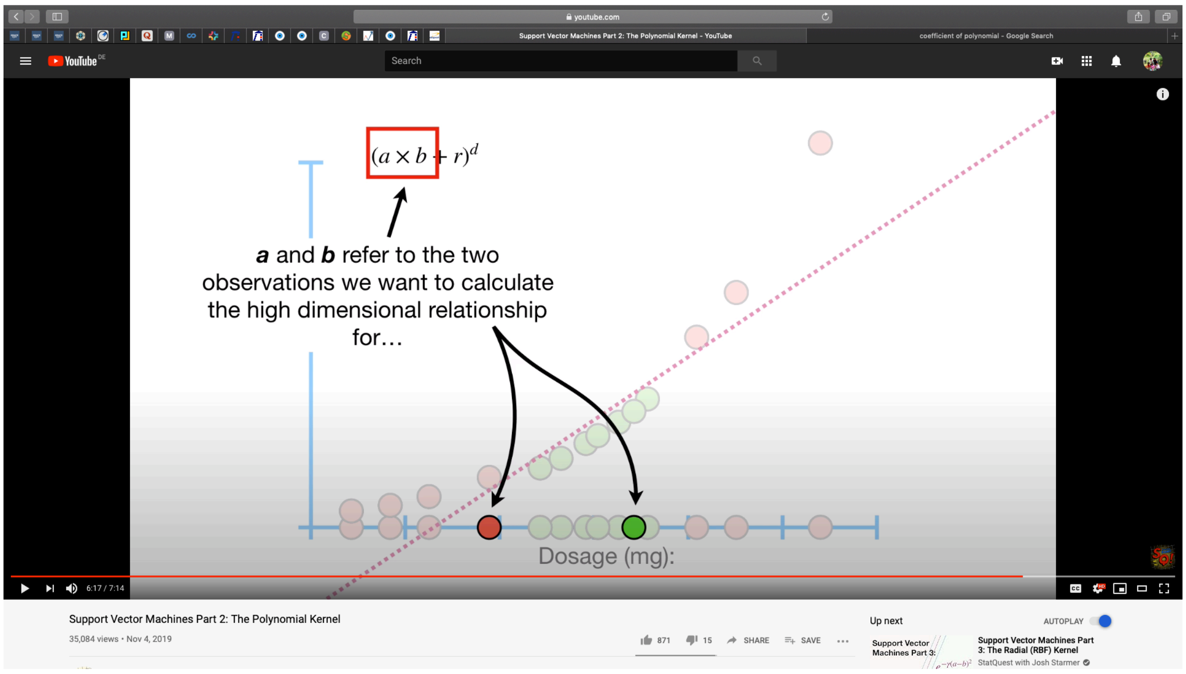
———UNDERSTAND THE CODE STRUCTURE —————

- **<u>Classification:</u>**
  - use **SVM** to classify
  - first task is to train our classifier but not easy. Need a large dataset but doesn't mean good results.
  - take hundreds of photos, pre-process and then segment all the photos
  - training done on 75 license-plate image and 35 images without licence plates - 144 x 33 pixels

- real world application needs more training data
- trainSVM.cpp creates a .XML file which has the trained data
- Training data for ML algorithm for OPENCV is saved as NxM matrix with N samples and M features
- OPENCV easily manages data file in XML or JSON format - training data from SVM.xml can be extracted by using the FIleStorage class
- using the CvSVMParams structure define basic SVM parameters to be used in the algorithm
- SVM is used here to classify whether the image has a plate or not
- NEED TO UNDERSTAND THE SVM(Support Vector Machines) ALGORITHM

-> **SVM Algorithm**
- uses a kernel to change data from 1D to 2D
- if it were a polynomial kernel and if data was in 2D, then the polynomial kernel finds the 2-D relationship between each pair of observations
- polynomial kernel: (a x b + r)^d; ‘a’ and ‘b’ refer to two different observations in the

dataset; ‘r’ determines to the co-efficient of the polynomial; ‘d’ sets the degree of the polynomial; ‘r’ and ‘d’ are determined using Cross Validation


$(a \times b + r)^d$

a and ***b*** refer to the two observations we want to calculate the high dimensional relationship for…

Support Vector Machines Part 2: The Polynomial Kernel

35,084 views • Nov 4, 2019

youtube.com
Support Vector Machines Part 2: The Polynomial Kernel - YouTube
coefficient of polynomial - Google Search
YouTube
Search
Dosage (mg):
6:17 / 7:14
871
15
SHARE
SAVE
Up next
AUTOPLAY
Support Vector Machines Part 3: The Radial (RBF) Kernel
StatQuest with Josh Starmer

- Linear SVM Classification (Source: Hands on Machine Learning; pg.274)
  - the decision boundary should not only seperate two classes but also stay as far away from the closest training instance as possible. Also called *large margin classification*. Widest possible street between two classes. SVMs are sensitive to feature scaling - meaning that both the x and y axes have to be scaled properly
  - *soft margin classification* - the objective of this process it to find a good balance between keeping the street as large as

possible and limiting the margin violations(i.e., instances that end up in the middle of the street or even on the wrong side)

svm_clf = Pipeline([("scaler", StandardScaler()),("linear_svc", LinearSVC(C=1, loss="hinge"))])

The 'C' parameter increases or reduces the size of the street

'StandartScaler()' scales the features accordingly

'linear_svc' linear support vector classification

- Nonlinear SVM Classification
  - many datasets are not so linear

  polynomial_svm_clf = Pipeline([("poly_features", PolynomialFeatures(degree=3)),("scaler", StandardScaler()),("svm_clf", LinearSVC(C=10, loss="hinge"))])

  - https://github.com/ageron/handson-ml2/blob/master/05_support_vector_machines.ipynb -> how to draw the graphs
  - Polynomial Kernel ->a low polynomial degree, this method(SVM) cannot deal with very complex datasets, and with a high

polynomial degree it creates a huge number of features, making the model too slow.

- Cross Validation : Say if there is a dataset and say if 75% of the data is used for training the data and 25% is used for testing the data. A question arises on how do I decide it s a 75:25 ratio. Cross validation uses all possible ratios and find the results. It compares the results and whichever ratio gives the best result then it is used.
  - Coming back to the PDF(pg.176), we label a plate class with 1 and no plate class with 0.

- **Plate recognition:**
  - it is the second step
  - this section retrieves the characters of the plate and the algorithm used is the **optical character recognition**(OCR)
  - after the plate is detected in the previous step, we proceed to segment the plate to get each character
  - artificial neural network is used to recognise the character
- **OCR segmentation:** pg.177
  - plate image patch is received as input
  - apply equalise histogram algorithm

- apply a threshold filter
  - find the contours using the findContours() algorithm - use the CV_THRESH_BINARY_INV parameter in the algorithm to invert the binary image - contour algorithm only see the white pixels as contours
  - size verification after the contour algorithm
  - remove all regions which do not meet the desire size or aspect ratio
- **<u>Feature extraction:</u>** pg.178
  - After the characters are segmented, extract the features for training and classifying the ANN algorithm
  - Count number of pixels with non zero values
- **<u>OCR Classification:</u>** pg.181
  - Classification of the digits and alphabets is done by ANN machine learning algorithm called Multi-Layer Perceptron(MLP)
  - MLP consists of network of neurons which an input layer, output layer and one or more hidden layers
  - Each neuron calculates the output value as a **sum of the weighted inputs** <u>plus</u> **a bias term** and is transformed by a selected activation function
  - Some of the selected activation functions are: Identity, Sigmoid(default), and Gaussian

    - Weight of each layer and neutron is computed and learned by **training** the ANN algorithm
    - Training is done creating a *N x M* matrix where N is training/samples data and M is the classes (10 digits + 20 letters in our case) and the matrix is set to 1 if the data matches with the label
    - After training is done we can classify any segmented plate feature using OCR::classify
- **<u>Evaluation:</u>** pg.185
    - We should know how to correct the classification, recognition, and detection errors in the system
    - Evaluation should be done on different situation and parameters
    - Evaluation done on the OCR task using the following variables: the size of the low-level resolution image features and the number of hidden neurons
    - Testing done on random samples - if result is == classes ; if not increment error + 1, divide by the number of sample to get error ratio
    -