Letter Localization

Markus Köhler

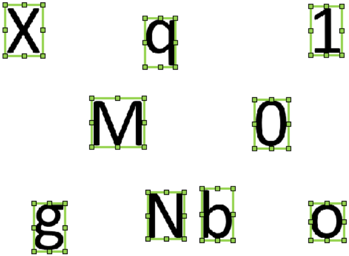
Project *Machine Learning using MATLAB*

13.02.2019

Outline

Description of my project

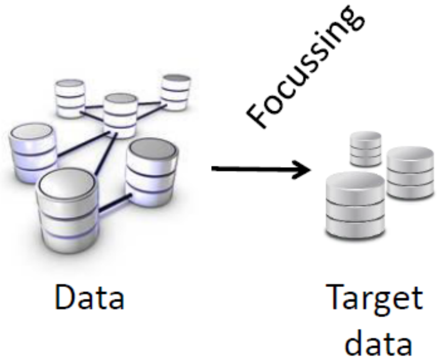
- only detect the positions of letters in images, not the class
- letters: a-z, A-Z, 0-9 (62 classes)
- try out different ways of data preprocessing and different models



Possible challenges

- diversity of the same letter (font, texture, orientation)
- diversity of the 62 classes of letters
- background, size of the area around the positive training examples
- accuracy of bounding boxes, aspect ratio, very small/large letters
- overlapping of letters, simplicity vs. inflexibility of rectangular bounding boxes

Data Selection



Training data

Positive examples:

- both from the Chars74k image dataset

(a) Chars74k, from natural scenes (7705 images)

(b) Chars74k, synthetically created (62992 images)

Training data

Negative examples:

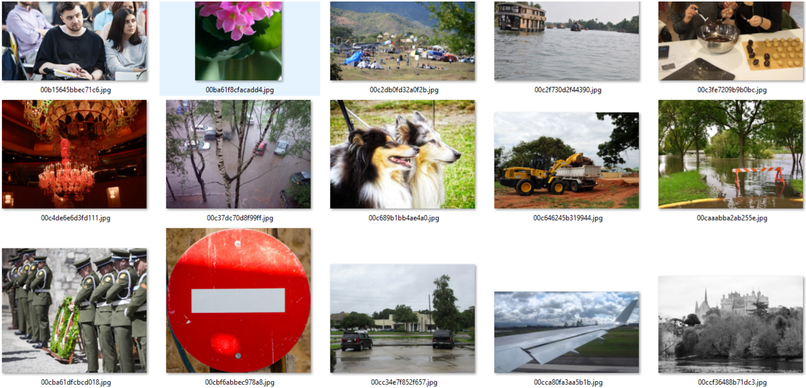
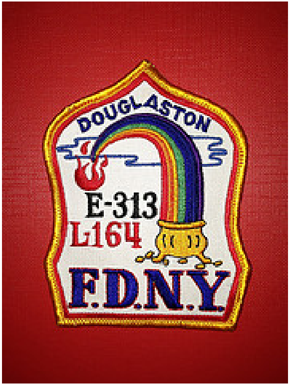


Figure 2: part of the Google Open Image dataset (2627 resp. 100 images)

Test data



(a) CTW1500 (1500 images)

(b) Google Open Image (those with text)

Data Preprocessing

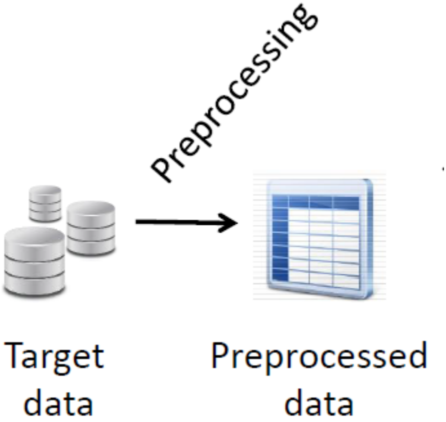


Image Labelling

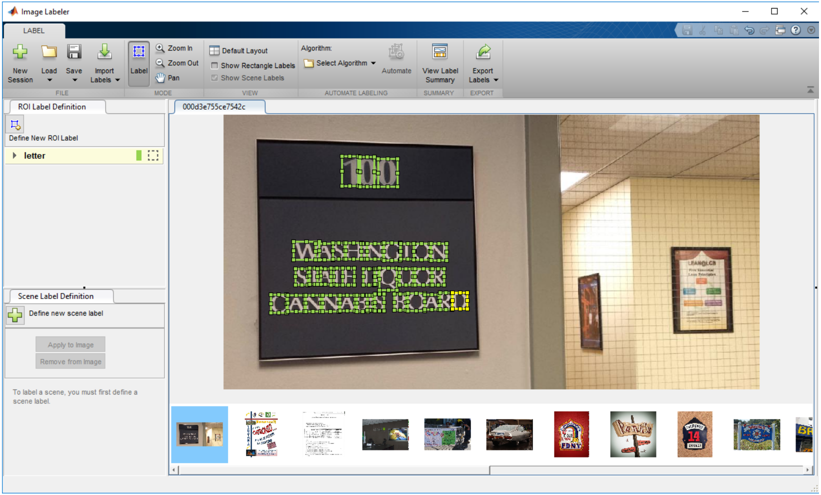


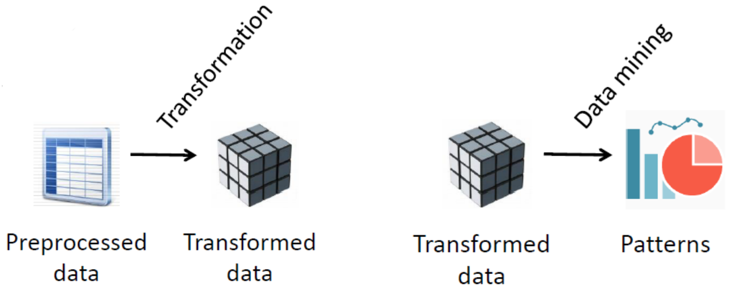
Figure 4: using MATLAB's Image Labeler

SVM

steps:

- resize all training data (positive and negative examples) to the same size
- extract features by using HOG
- train the SVM model
- evaluate the SVM model
  - use k-fold cross validation on training data
  - predict bounding boxes on test data
- find all (hyper-)parameters in the previous steps and optimize them

Data Transformation and Data Mining



Data Transformation using HOG feature extraction

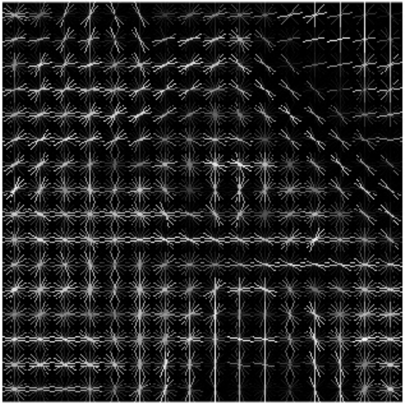
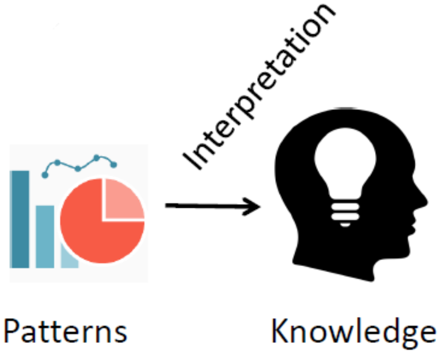


Figure 6: HOG Descriptors

hyper-parameters of HOG feature extraction

- CellSize: size [height width] of a HOG cell in pixels
- BlockSize: number [num_y num_x] of cells in each block
- BlockOverlap: overlap between 2 neighbour blocks (specified by the overlapping number of cells [num_y num_x])
- NumBins: number of bins of the orientation histogram (n bins means that the orientation angles can be $0^\circ : \frac{180^\circ}{n} : 180^\circ$)
- UseSignedOrientation: true if the orientation may show in both directions (so the angles can be in $-180^\circ : \frac{360^\circ}{n} : 180^\circ$)

Bounding Box Generation



Bounding Box Generation

steps:

- iterate over the image with a certain stride and size S, for each sub-image:
  - extract features by using HOG
  - predict the output (hypothesis and certainty) with the SVM model
- greedily select the sub-images with the highest predicted certainty that there is a letter
  - ignore the sub-images which already overlap (e.g. at least 50%) with a previously selected sub-image
- draw a bounding box for every selected sub-image

Cascade Object Detector

There is also something in MATLAB which does the same task with a different concept.

Hyper-parameters:

- ObjectTrainingSize: size of the images to be resized
- NegativeSamplesFactor: factor (multiply with number of positive examples) for amount of the generated sub-images from the negative examples
- NumCascadingStages: number of stages
- FalseAlarmRate: acceptable rate for false positive classifications in each step
- TruePositiveRate: min. true positive rate required in each step
- FeatureType: type of feature extraction (HOG, LBP, Haar)

Error measures

Intersection over Union:

- compare the predicted bounding box P with the ground truth bounding box G:

$$IoU = \frac{|P \cap G|}{|P \cup G|}$$

- do this for all reasonable pairs of bounding boxes

  $\Rightarrow$ can be difficult if there are many ground truth bounding boxes and if they overlap each other

Rate of detected objects:

- might be easier to use
- needs reasonable definitions of when an element is detected correctly

Current progress of my project

What is done/not done so far:

- (in progress) SVM with HOG features
- (in progress) Cascade Object Detector
- (to do) Convolutional Neural Networks
- (to do) Transfer Learning
- (in progress) Hyper-parameter tuning, improve data selection