# Image classification with SVM

Andrea Portscher, Juan Pablo Stumpf, Philip Wille

February 4, 2021

## Introduction

For this project we implemented an image classifier using three different feature extraction approaches: Scale-invariant feature transform (SIFT), Speeded up robust features (SURF) and Histogram of Gradients (HOG). First, we prepared an image dataset containing five different image classes, each comprising training and testing data. After extracting features using the above algorithms, we trained a Support-Vector Machine (SVM) and testetd, how well it was able to classify images.

## Approaches

In the following section we will shortly explain the functionality of the feature extraction algorithms we applied to the dataset - namely SIFT, SURF and HOG. In general, feature extraction algorithms work in three steps [1]:

1. They select interest points at distinctive locations, such as corners.

2. The neighbourhood of the interest points is represented by a feature vector.

3. The descriptor vectors are matched between different images.

### SIFT

### SURF

SURF is based on similar properties as SIFT but is - as the name already takes away - faster and more robust. In the detection phase, SURF approximates the Laplacian of Gaussian with a box filter, and a convolution with a box filter can can easily be calculated with the help of integral images, and can be parallelized for different scales. Another improvement is the use of the sign of the Laplacian for underlying interest points in the matching phase. Since it is already calculated during the detection phase, it comes at no additional cost, but speeds up the process by only comparing features if they have the same type of contrast[1].

---

[1]https://docs.opencv.org/3.4/df/dd2/tutorial_py_surf_intro.html

Unlike SIFT, the SURF algorithm is currently still patented and can therefore only be used by including the `opencv_contrib` package[2] and allowing non-free algorithms.

## HOG

The purpose of Histogram of Gradients (shortly HOG) is to count the occurrence of gradient orientations in a given image. The first time HOG was described by Robert K. McConnell of Wayland Research Inc. in a patent application in 1986 without using the term. It has become more famous when Navneet Dalal and Bill Triggs [2] presented their work on HOG descriptors in their 2005 published paper.

### Functionality of HOG

HOG decomposes an image into small squared cells, computes a histogram of oriented gradients in each cell, normalizes the result using a block-wise pattern, and return a descriptor for each cell. Stacking the cells into a squared image region can be used as an image window descriptor for object detection, for example by means of an SVM.

## Implementation

### Data

The image data we used stems from the Caltech-256 Image Set[3], which consists of 256 sets of images of a certain class. We randomly selected 5 of these, namely: cactus, dice, raccoon, spaghetti and sushi. The preaparation of the image data was comprised of the following steps, which we implemented in the file `utils.py`:

- All images were resized to the same size, converted to grayscale and normalized (which removes noise from the image).

- All images are associated with a label (their image class).

- The images are split into a training set (80% of images per class) and a testing set (20% of images).

### Pipeline

We use a pipeline[4] which sequentially applies a list of transforms and a final estimator. In our case the transformers are the application of the respective feature extractor and finally the SVM. The SIFT and SURF transformers are initialized in the same manner: After the features have been extracted, they are clustered. In this step we used the MiniBatchKMeans clustering algorithm[5]. We used the Elbow Method to approximate

---

[2]https://github.com/opencv/opencv_contrib
[3]http://www.vision.caltech.edu/Image_Datasets/Caltech256/, accessed on the 22nd of December 2020
[4]https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
[5]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html

the ideal size for K. This approach works by simply calculating and plotting the distortions as a function of the number of clusters. Then the "elbow" of the graph - meaning the point in which the number of distortions starts to decrease too slowly to justify the additional cost of an increase in the number of clusters. The resulting graph for SIFT can be seen on the following figure. On this base we chose the value 500.
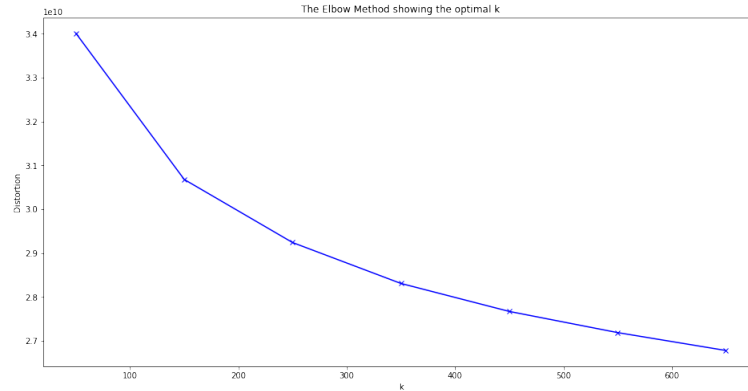


After having fitted the MiniBatchKMeans, the transformation begins and histograms are generated. The next step in the pipeline is the training of the SVM. We chose the LinearSVC algorithm provided by the `sklearn` library[6].

For the HOG feature descriptor we only resize the incoming images to a size of 64 times 128 to be compliant with the paper of Navneet Dalal and Bill Triggs [2]. The following steps are equal to these from SIFT or SURF.

## Discussion of Results

In order to be able to evaluate our results we calculated metrics such as precision and recall and generated confusion matrices. Furthermore, we used cross-validation. We checked the results for 2, 3, 4 and 5 image classes. Naturally, all of the algorithms showed the best results with only 2 image classes and their reliability decreased with a rsing number of classes.

## References

[1] Herbert Bay, Tinnie Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". In: *European Conference on Computer Vision*. 2006.

[2] Navneet Dalal and Bill Triggs. "Histograms of Oriented Gradients for Human Detection". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005)* 2 (June 2005).

---

[6]https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html