Machine Learining for Computer Vision Coursework 2
Visual Categorisation by BoW (K-means) and SVM

David Angelov
MEng Electrical and Electronic Engineering
Imperial College London
david.angelov12@imperial.ac.uk

Huaqi Qiu
MSc Communication and Signal Processing
Imperial College London
h.qiu15@imperial.ac.uk

1. Question 1

2. Question 2

2.1. Theory of SVM and multi-class extension

In this coursework, we applied Support Vector Machine (SVM) as the classifier. SVM is a binary classifier based on the discriminant function

$$y(\boldsymbol{x}) = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + b \tag{1}$$

where $\boldsymbol{w}$ is a weight vector and b is the bias. The dataset x is classified to one of the two classes according to the value of $y(\boldsymbol{x})$. However, there are cases when dataset is not linearly separable, in which the Kernel Trick is required. The Kernel Trick map the data points to a higher-dimensional feature space so that the dataset is linearly separable. This process is denoted by: $\boldsymbol{x} \rightarrow \phi(\boldsymbol{x})$. Now the binary SVM function (Eq.1) becomes

$$y(\boldsymbol{x}) = \boldsymbol{w}^{\boldsymbol{T}}\phi(\boldsymbol{x}) + b \tag{2}$$

The SVM separate data in the Kernel transformed space by a linear decision hyperplane. The best hyperplane is found by maximizing the margin of separation. The margin is evaluated by finding the distance of the closest training point to the decision hyperplane. The optimised weighting vector $\boldsymbol{w}$ takes the form of

$$\boldsymbol{w} = \sum_{n=1}^{N} a_n t_n \phi(\boldsymbol{x_n}) \tag{3}$$

where $\boldsymbol{x_n}$ is training data vector and t_n is the labels of the n^{th} training data vector. Then the SVM decision function becomes

$$y(\boldsymbol{x}) = \sum_{n=1}^{N} a_n t_n k(\boldsymbol{x}, \boldsymbol{x}_n) + b \tag{4}$$

where $k(\boldsymbol{x}, \boldsymbol{x}_n)$ is the Kernel function. In this course work we primarily consider the RBF kernel function, which is formulated as

$$k(\boldsymbol{x}, \boldsymbol{x'}) = exp(\frac{||\boldsymbol{x} - \boldsymbol{x'}||^2}{2\sigma^2}) \tag{5}$$

We considered the parameter σ in the kernel function as the primary factor that affects the performance of RBF kernel SVM. We also considered the parameter C, which is an error weighting constant that penalizes the misclassification during the supervised learning and effectively controls the margin.

Since SVM is two-class classifier, we extended it in order to solve multi-class classification problems. Two methods of extension were considered:

1. One versus the rest (OVR)
   In OVR method, we trained M separate SVMs for M class problem. In the training of the m^{th} SVM, the m^{th} class were labelled as positive class, while the rest $M-1$ classes were labelled as negative class. Upon testing a query image $\boldsymbol{x}$, the output from all M SVMs were compared. The query image was classified as the m^{th} image if the output of the m^{th} SVM was the highest, i.e.
   $$y(\boldsymbol{x}) = max_m\{y_m(\boldsymbol{x})\} \tag{6}$$
2. One versus one (OVO)
   In OVO method, we trained SVMs by pairs of the classes. Each of the $M(M-1)/2$ SVMs was trained by data from a pair of two classes, labelling data points from one of the class as the positive class. In testing, the query image was tested and the output from each SVM was treated as 'vote'. The class that had the majority of votes was the class that the query image was classified to.

2.2. Implementation on Toy Spiral data

In this coursework, we firstly tested the SVM theory on the Toy Spiral data, shown in Figure 1. Each data point is in

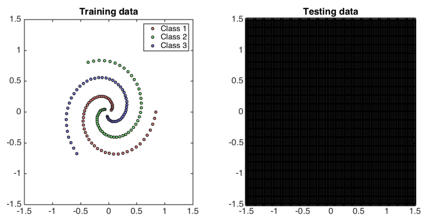


Figure 1: Toy Spiral training and testing data face

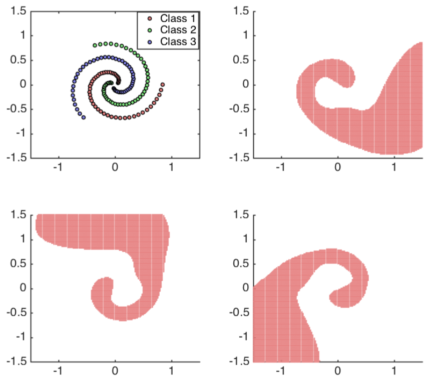


Figure 2: Result form each OVR extended SVMs

the form of $[(X_n, Y_n), t_n]$, where (X_n, Y_n) is 2-D Cartesian coordinates and t_n is the class label of the data point. The testing data is a 2D dense grid within the range of $X_n, Y_n \in [-1.5, 1.5]$. Testing with this testing data set can visualise the decision boundaries.

2.3. SVM extension

As introduced previously, we extended the binary class SVMs to multi-class SVMs by OVR and OVO. Figure 2 shows the classification results from each of the 3 SVMs extended by OVR method. In this Figure, red regions represents the positive class classified by the SVM. The combined classification of the 3 SVMs is shown in Figure 3. In all demonstrations in this part of the discussion, the kernel coefficients σ and C are kept unchanged ($\sigma = 0.8$ and $C = \infty$).

Similarly, Figure 4 and 5 show the results of individual SVMs and their combination respectively. Notice that in the OVO case, the votes from all $M(M-1)$ might not have a single majority. Data points with these classification results were classified as 0 in this question, which corresponds to the 'white' area in Figure 5. For the recognition problem of Caltech 101 data in question 4, we randomly selected class labels from all classes for these type of data points by `randi`.

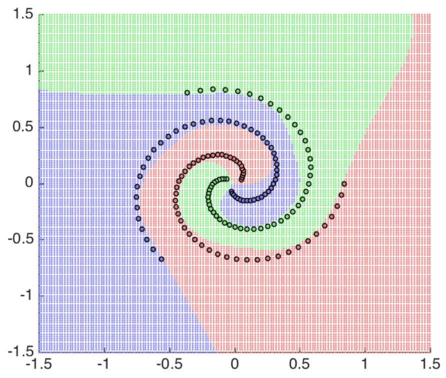


Figure 3: Combined result of OVR extended SVMs

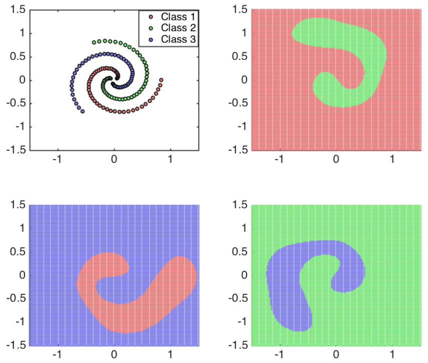


Figure 4: Result form each OVO extended SVMs

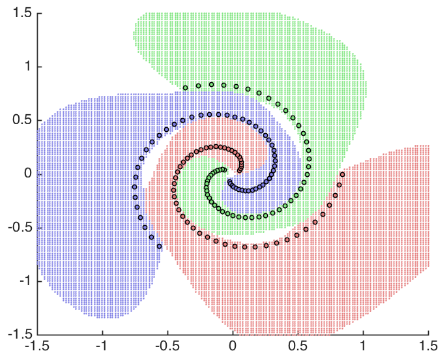
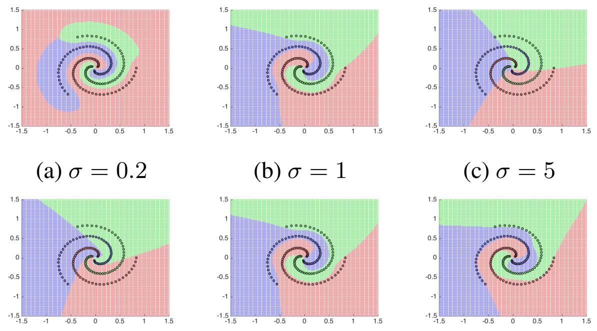


Figure 5: Combined result of OVO extended SVMs

2.4. SVM parameters

The parameters σ for the RBF kernel function and the parameter C for the misclassification penalty are problem



(a) $\sigma = 0.2$ (b) $\sigma = 1$ (c) $\sigma = 5$

(d) $C = 1$ (e) $C = 1000$ (f) $C = \infty$

Figure 6: Classification results with varying parameters σ and C for OVR

dependent. Here we discuss the effect of the two parameters using the Toy Spiral data test. The kernel parameter σ affects the generalization or complexity of the model. Simpler models could have better generalization to unseen data with the risk of underfitting. Complex models on the other hand could be overfitted to training data and results in high testing loss (i.e. lower classification accuracy). Figure (a), (b) and (c) in Figure 6 show the result of varying the parameter σ while keeping C unchanged. Result (a) demonstrates the case of overfitting and (c) demonstrates the case of underfitting.

The parameter C for each misclassified data point is proportional to its distance from the decision boundary. This indicates that C controls the margin allowed for the SVMs.

3. Question 3

References

[1] A. Alpher. Frobnication. *Journal of Foo*, 12(1):234–778, 2002.

[2] A. Alpher and J. P. N. Fotheringham-Smythe. Frobnication revisited. *Journal of Foo*, 13(1):234–778, 2003.

[3] A. Alpher, J. P. N. Fotheringham-Smythe, and G. Gamow. Can a machine frobnicate? *Journal of Foo*, 14(1):234–778, 2004.

[4] Authors. The frobnicatable foo filter, 2014. Face and Gesture submission ID 324. Supplied as additional material `fg324.pdf`.

[5] Authors. Frobnication tutorial, 2014. Supplied as additional material `tr.pdf`.

A. Appendix 1 Matlab Code

A.1. Init.m

```
% Please run this script under the root folder

 clearvars -except N;
 close all;

% addpaths
 addpath('./internal');
 addpath('./external');
% addpath('./external/libsvm-3.18/matlab');

% initialise external libraries
 run('external/vlfeat-0.9.18/toolbox/vl_setup.m'); % vlfeat library
% cd('external/libsvm-3.18/matlab'); % libsvm library
% run('make');
% cd('../../..');

% tested on Ubuntu 12.04, 64-bit, Intel  Core ?i7-3820 CPU @ 3.60GHz    8
```

B. matlab code part 2

```
% Please run this script under the root folder

 clearvars -except N;
 close all;

% addpaths
 addpath('./internal');
 addpath('./external');
% addpath('./external/libsvm-3.18/matlab');

% initialise external libraries
 run('external/vlfeat-0.9.18/toolbox/vl_setup.m'); % vlfeat library
% cd('external/libsvm-3.18/matlab'); % libsvm library
% run('make');
% cd('../../..');

% tested on Ubuntu 12.04, 64-bit, Intel  Core ?i7-3820 CPU @ 3.60GHz    8
```