Neural Networks
Group 04
Andrea Jemmett (2573223)
Bas van der Drift (1707507)

Assignment 4
RBF networks and Support Vector Machines

Section A

The code for this section is contained in 'ass4a.m'.

We tested the RBF model against the 'pics.mat' dataset (we used a Gabor filter to preprocess the images, like in Assignment 3B) with a different number of kernels (1, 2, 3, 4, 5, 7, 10, 15, 25, 50, 100, 200, 360) and measured the accuracy by means of a confusion matrix and 10-fold cross validation. We have chosen this set of numbers because some are the same used for the MLP and others have proven by experimentation to increase the accuracy. The maximum number of kernels is 360 because with 400 examples we have a training set size of 360 (due to 10-fold cross validation) and the maximum number of kernels is equal to the number of training examples; with K-means clustering this means that each training example will fall in its own cluster.

To create, train and test an RBF network we used the following Netlab functions:

- 'rbf' to create the network with gaussian hidden unit activation function;
- 'rbfsetbf' (the teacher's K-means version) to find and set the kernels;
- 'rbftrain' to train the network;
- 'rbffwd' to feed-forward the test examples and measure the error.

In the following table we show the average confusion matrix (over different cross validation runs) and the accuracy (obtained by summing the elements on the diagonal and dividing by the number of examples in test set) for different numbers of hidden nodes.

K	Confusion matrix	Accuracy
1	0 11.9000 0 28.1000	**0.7025**
2	3.3000 8.6000 2.4000 25.7000	**0.725**
3	3.6000 8.3000 3.0000 25.1000	**0.7175**
4	3.8000 8.1000 2.9000 25.2000	**0.725**
5	3.7000 8.2000 2.9000 25.2000	**0.7225**
7	4.4000 7.5000 3.7000 24.4000	**0.72**

10	5.5000 6.4000 1.9000 26.2000	**0.7925**
15	6.7000 5.2000 2.6000 25.5000	**0.805**
25	7.4000 4.5000 2.2000 25.9000	**0.8325**
50	9.2000 2.7000 1.6000 26.5000	**0.8925**
100	10.6000 1.3000 0.9000 27.2000	**0.945**
200	10.9000 1.0000 1.6000 26.5000	**0.935**
360	6.1000 5.8000 12.4000 15.7000	**0.545**

From the table is easy to see that increasing the number of kernels affects positively the accuracy until 100, where we reach 94.5%. With 360 hidden nodes the accuracy on the test set drops to 54.5%; this is due to the fact that the number of kernels is equal to the number of training examples, in this way each kernel will calculate the distance by its input example. This produces overfitting on the training set.

With the Radial Basis Function network we have been able to achieve a better accuracy compared to the Multi-layer Perceptron. The reason to this is that with the MLP we were limited by Netlab to use a two-layer network (that can produce convex open or closed regions), instead RBF networks can have only a single hidden layer and with that are able to produce more complex separating regions.

Section B

The code for this section is contained in ‘ass4b.m’.

For this section we used the LS-SVMlab toolkit as requested. To complete the task we used the following functions:

- ‘trainlssvm’ to train an SVM model;
- ‘simlssvm’ to test the results of the training;
- ‘plotlssvm’ to plot the dataset with the found decision boundary.

We created an SMV model for classification with regularization parameter 1, radial basis kernels with distance 0.5. We then tried the whole training and testing process with and without preprocessing. We noticed that the preprocessing normalizes the input and output data and their mean is 0 and variance 1. The results can be observed in Figure 1.

The picture shows that with data preprocessing the separation boundary found resembles more a straight line that maximizes the margin between the two clusters. Without

preprocessing (Figure 1b) we can see that the separation boundary tends to “embrace” one data cluster (due to the use of RBF kernels).

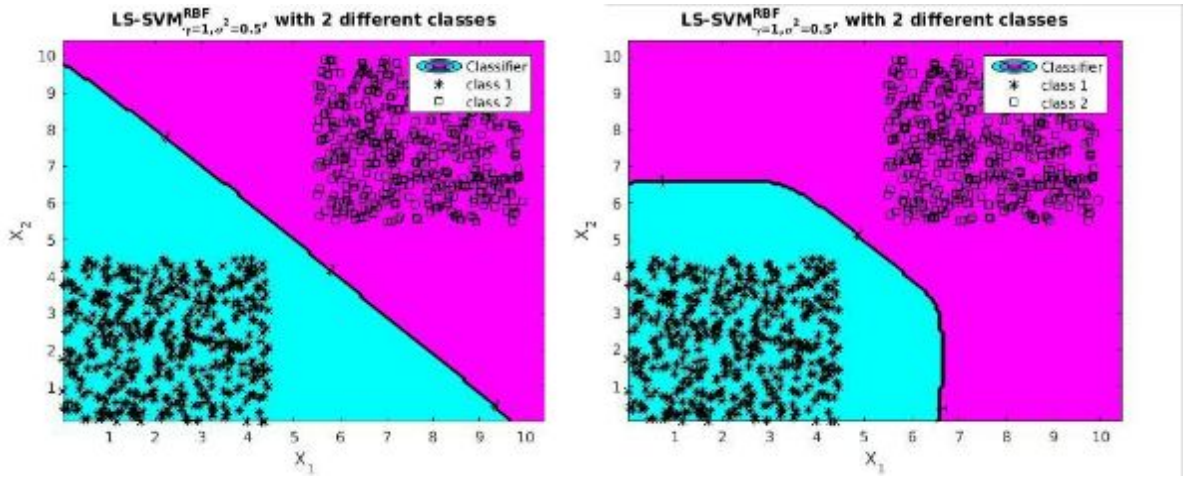

Figure 1. Separation boundary found by SVM a) with and b) without preprocessing.

Section C

The code for this section is contained in ‘ass4c.m’.

We applied the SMV method to the pictures dataset and used a 10-fold cross validation like in section A to asses the model accuracy. We trained an SVM model using different combinations of parameter settings for the regularization parameter (‘gam’) and the width of the Gaussian kernel functions (‘sig2’). In the following table are presented the results for sig2 equal to the number of input features (2576) and gam equals to the number of examples (400). Those parameters have been found by means of experimentation to produce the best accuracy both in training and test set.

Training	107.1 0 0 252.9	**1**
Test	11.5 0.4 0.1 28	**0.9875**

With RBF networks, we have to find the centers of the kernels with an algorithm like K-means, instead with an SVM with Gaussian kernels the support vectors are the centers of the functions. So with RBF we had to find the right number of hidden nodes (kernels) by cross validation, in SVM the support vectors are discovered by training.

Another important aspect to note is that to achieve overfitting with an RBF network, you have to use the maximum number of hidden units, with an SVM overfitting is hard to achieve. Regardless of the parameters set, the SVM produces 100% accuracy on the training set and higher accuracy on the training set by increasing the values of the parameters until the aforementioned values (gam=400, sig2=2576) are reached. We can conclude that SVM is more robust than RBF networks.