Computer Vision I, SS13 - Homework Assignment 3

June 21, 2013

Prof. Stefan Roth
Qi Gao
Thorsten Franzel
Stephan Richter

This assignment is due on July 02nd, 2013 at 23:59.

Please see the previous assignments for general instructions and follow the handin instructions from there.

Problem 1 - Linear SVMs (15 points)

In class we have seen how formulating support vector machines (SVMs) involves implementing and solving a constrained quadratic optimization problem. In this problem, you will implement linear SVMs in the primal form. Conveniently, Matlab has a solver for constrained quadratic optimization problems built in (`quadprog`), which does essentially all of the work for you. This solver can solve arbitrary constrained quadratic optimization problems of the type

$$\arg\min_{\mathbf{z}} \quad \frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{H}\mathbf{z} + \mathbf{f}^{\mathrm{T}}\mathbf{z}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{z} \leq \mathbf{b}.$$

Your task it to convert the SVM formulation (see below) into this generic input format of the `quadprog` function, and then turn its output into a linear discriminant function.

Task A: Separable case.

- Load the training data from `a3p1_tdata.mat`, where data points are stored in rows, the first two columns are features and the last column gives the class. Make a scatter plot that allows to see which data point belongs to which class.
- Implement the following quadratic optimization problem:

$$\arg\min_{\mathbf{w},b} \quad \frac{1}{2}||\mathbf{w}||^2$$

$$\text{s.t.} \quad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1, \qquad \forall i.$$

  To do this, you have to find how to turn this constrained optimization problem into the generic form from above. Things you should ask yourself are: Which variable(s) does $\mathbf{z}$ need to represent? What do you need to set $\mathbf{H}, \mathbf{f}, \mathbf{A}$, and $\mathbf{b}$ to? If you are unsure about whether you got this right, state a brief answer to these questions in your solution to receive partial credit (but you do not need to answer these to receive full credit; a correct implementation suffices).
- Train the linear SVM on the training data and report what values of $\mathbf{w}$ and b you obtain.
- Find out how many support vectors the solution has. To do that, inspect the Lagrange multipliers that `quadprog` returns.
- Plot the decision boundary $y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$ as well as the margin lines (see illustrations in the lecture slides) on top of the scatter plot of the data. Note that the margin lines are also contour lines $y(\mathbf{x}) = c$. (Which values of c do you need?) Also mark the support vectors in the plot.

Task B: Non-separable case.

- Implement the quadratic optimization problem for this case, which now contains slack variables:

$$\arg\min_{\mathbf{w},\xi,b} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i$$

$$\text{s.t.} \quad y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \qquad \forall i$$

$$\xi_i \geq 0, \qquad \forall i.$$

  Again, ask yourself what the various terms in the generic formulation need to represent now. If you want, you can give a written answer to this to receive partial credit (in case your implementation is not correct).

- Train on the same data as in task A. The data is linearly separable, but you should still see the effect of the slack variables depending on the setting of C.

- Plot the discriminant function as well as the margin lines on top of the data for two different settings of C. Also mark the support vectors. You should choose the settings of C so that in one of the cases the margin lines look like in part A; in the other case several data points should violate the margin constraint. Report which values of C you found and how many support vectors you have in each case.

Problem 2 - Bag-of-Words Model and Categorization (30 points)

In this problem, you will implement a simplified Bag-of-Words model and try both naive Bayes classifier and SVM classifier for a simple categorization problem.

A: Train the codewords dictionary.

First of all, you should build a codewords dictionary – representative keypoints, over which each image could be represented by a distribution. Here we represent these keypoints using the k-means clustering centers of all *feature vectors* from the training images.

k-means clustering aims to partition n observations $(\mathbf{x}_1, \cdots, \mathbf{x}_n)$ into k ($k \leq n$) clusters $(s_1, \cdots, s_k)$ so as to minimize the within-cluster sum of squares

$$\sum_{i=1}^{k} \sum_{\mathbf{x}_j \in s_i} \|\mathbf{x}_j - \mu_i\|^2,$$

where μ_i is the mean of points in s_i. To find the means, you can start with some (random) initial k means $\mu_1^{(1)}, \cdots, \mu_k^{(1)}$, and then alternate between two steps:

- assign each observation to the cluster with the closest mean

$$s_i^{(t)} = \{\mathbf{x}_j : \|\mathbf{x}_j - \mu_i^{(t)}\| \leq \|\mathbf{x}_j - \mu_{i^*}^{(t)}\| \text{ for all } i^* = 1, \cdots, k\},$$

- and calculate the new means to be the centroid of the observations in the cluster

$$\mu_i^{(t+1)} = \frac{1}{|s_i^{(t)}|} \sum_{\mathbf{x}_j \in s_i^{(t)}} \mathbf{x}_j$$

until the assignments no longer change.

Tasks:

- Load all airplane and motorbike images from the `images/` subdirectory of the zip file. Randomly separate each image set into two equal-size parts for training and testing respectively.
- Apply the Harris detector and SIFT descriptor (sample code will be sent to you by email) to obtain the *feature vectors* from the training images. Use the parameters $\sigma = 1.4$ (for both Harris and SIFT), filter size 15×15, threshold 1500 and ignore the feature points within a 10-pixel wide boundary.
- Compute the codewords (keypoints) using k-means clustering with $k = 50$. During the iterations, if any of the cluster centers has no data points associated with it, replace it with a random data point.

B: Categorize images with naive Bayes classifier.

To categorize an image I_m with the naive Bayes classifier, first label each extracted feature descriptor from I_m with the keypoint *(i.e.*, cluster center) to which it lies closest in feature space, and count the number $n(i,m)$ of times of keypoint μ_i occurs in I_m; then apply Bayes rule and take the category C_j with the largest posterior as the prediction

$$p(C_j|I_m) \propto p(C_j) \cdot p(I_m|C_j) \propto p(C_j) \prod_{i=1}^{k} p(\mu_i|C_j)^{n(i,m)}. \tag{1}$$

Here $p(C_j)$ are the category priors. The class-conditional probabilities of keypoint μ_i given category C_j are estimated from the labelled training images. In order to avoid probabilities of zero, these estimates are computed with Laplace smoothing:

$$p(\mu_i|C_j) = \frac{1 + \sum_{I_m \in C_j} n(i,m)}{k + \sum_{t=1}^{k} \sum_{I_m \in C_j} n(t,m)}.$$

Tasks:

- Implement the naive Bayes classifier for separating motorbike and airplane images. Note you should also estimate the class priors $p(C_j)$ according to the training data.
- Reformulate (1) in log space. Compute and compare the log posteriors. Write a comment in your code why this is better than a direct computation.
- Compute the training error rate, *i.e.*, the percentage of incorrect categorizations (for training images only).
- Use the other set of images for testing to compute the testing error rate

C: Categorize images with linear SVMs.

Over the trained codewords dictionary each image is represented by a distribution, which could be estimated through normalizing the number $n(i,m)$ of times of keypoint μ_i occurs in I_m (the same $n(i,m)$ as above task). We can then use SVMs for separating these distributions to categorize images.

Tasks:

- Use the same training data from above tasks to train a linear SVM (first try separable case; if it is not working, try the non-separable case.) based on your code from previous problem. Find out how many support vectors the solution has.
- Implement the classifier using your trained SVM to separate motorbike and airplane images.
- Compute the training error rate as well as the testing error rate.