SGN-13006 Introduction to Pattern Recognition and Machine Learning
TAU Computing Sciences
Exercise 3 *Visual classification (CIFAR-10 dataset)*

Be prepared for the exercise sessions. You may ask TA questions regarding your solutions, but don't expect them to show you how to start from the scratch. Before the end of the session, demonstrate your solution to TA to receive exercise points.

1. **CIFAR-10 – Bayesian classifier** (40 points)

   We continue with the CIFAR-10 dataset and adopt the Bayesian classifier approach this time.

   First we need to select suitable features. Let's use the average color of an image as a simple color feature. That means that the feature vectors of the length $\boldsymbol{x} = 32 \times 32 \times 3 = 3,072$ are coverted to three mean color values o $\boldsymbol{f} = (m_R, m_G, m_B)$ values where m_R is the mean value of the red channel, m_G green and m_B blue.

   Let's assume that channels are independent from a normal distribition. The Bayesian probability can be computed from

   $$P(class_1|\boldsymbol{f}) = \frac{P(\boldsymbol{f}|class_1)P(class_1)}{P(\boldsymbol{f}|class_1)P(class_1) + P(\boldsymbol{f}|class_2)P(class_2) + \ldots} \ .$$

   Note that you may omit the demoninator parts since it is same for all classes. Now,

   $$P(\boldsymbol{f}|class_1)P(class_1) = \mathcal{N}(m_R; \mu_{R,c_1}, \sigma_{R,c_1})\mathcal{N}(m_G; \mu_{G,c_1}, \sigma_{G,c_1})\mathcal{N}(m_B; \mu_{B,c_1}, \sigma_{B,c_1})P(c_1)$$

   where μ and σ are the mean and variance of each class and for each feature.

   Write a function *f=cifar_10_features(x)* that forms the mean color feature of x. Write another function *[mu,sigma,p]=cifar_10_bayes_learn(F,labels)* that computes the normal distribution parameters for the samples in F (one line per sample) for each category (category specific mean and variance). Note that the size of mu and sigma must be $num\_of\_classes \times 3$. The p (prior) probabilities are a vector of $num\_of\_classes \times 1$, one for each class.

   Finally write function *c=cifar_10_bayes_classify(f,mu,sigma,p)* that returns the Bayesian optimal class c for the sample f.

   Run your classifier for all CIFAR-10 test samples and report the accuracy (write another script for this).

   Hints: mean(), std(), normpdf()

2. **CIFAR-10 – Bayesian classifier with better pdf** (20 points)

   In this experiment we continue the previous Bayesian classifier, but we relax the naive assumption and switch to the full *multivariate normal distribution* (see the *mvnpdf* function in Matlab). That means that you need to put all three features (image mean of the Red, Green and Blue channels) into a single feature vector $\boldsymbol{f}_{1\times3}$ and represent class specific pdf's using the mean vectors $\boldsymbol{\mu}_{3\times1}$ and covariance $\boldsymbol{\Sigma}_{3\times3}$.

   Compute the classification accuracy and compare it to the naive version - which one is better and why?

3. **CIFAR-10 – Bayesian with extended features** (40 points)

   Perhaps you can improve your classification with better features?

   In the previous exercises you used the three mean values for images of size 32×32 as the image feature $\boldsymbol{f}_{3\times1}$. However, you may divide each image to four 16×16 sub-images which provide $4\times$ more features $\rightarrow \boldsymbol{f}_{12\times1}$. You may continue the process and divide each sub-image to another four 8×8 sub-sub-image providing now $16\times$ more features $\rightarrow \boldsymbol{f}_{48\times1}$. Note that the image division to sub-images can be done in multiple ways. You can see the sub-images as 4 squares forming the original image, or 4 rows or columns side by side forming the image. The way the image i.e. the pixels are divided into sub-images can slightly affect the results.

   Extend your feature extraction function *f=cifar_10_features(x,N)* such that now N defines the sub-window size (32 would be the original) and the function returns these new advanced *spatial features.* Test at least values $N = 32, 16, 8, 4$ and plot a graph where accuracy is plotted as a function of the sub-window size. Use full multivariate normal distribution in your experiments.

   What kind of behaviour you will find and what explains that?

   Hints: mvnpdf(), cov(), mean()