CEG499/699-13: INTELLIGENT SENSOR SYSTEMS

LABORATORY IV: PATTERN ANALYSIS

The purpose of this laboratory is to collect data with the complete sensor array system and perform multivariate analysis, including visual inspection, preprocessing, feature selection, dimensionality reduction and classification. You will be guided through a series of experiments and analyses that resemble a real-world situation. A few MATLAB routines have been posted on the course webpage to help you develop your analysis software. These include an implementation of Linear Discriminant Analysis, generation of labeled scatter plots and an example program.

SECTION I: DATA COLLECTION

The instructor should have given you a bubbler containing the washing agent (1/9 vol. dilution of rubbing alcohol in water) and four vials with pipe Tobacco, cedarWood chips, ground Coffee and a blAnk (labels T, W, C and A). Here is a list of recommendations for data collection:

- Analytes: Keep the vials closed while they are not connected to the sample port to prevent the aroma from evaporating too fast.
- Flow rate: Adjust the manual flowmeter to approximately 0.5 liters per minute when the system is in the reference cycle. When you sample from the washing line you should see the dilution bubbling, although not violently[1].
- Sampling rate: 10 samples per second should be fast enough to capture the sensor transient response, yet slow enough to be plotted in real time by your LabVIEW graphical user interface.
- Sampling times: 60 seconds for the wash and sample cycles should be enough time for the sensors to reach steady state. The reference cycle may have to be longer since the recovery dynamics for metal-oxide chemoresistors are notoriously slow.
- Database size: collect at least 10 samples per odor to ensure reasonable statistics for dimensionality reduction, classification and validation. The more data you collect, the better. Use simple file names (e.g., `Tij.txt` to denote the jth sample of Tobacco collected on the ith day) to store your "sniffs."
- Sampling rotations: presentation of the analytes should be randomized to minimize temporal correlations between samples. Do not cycle through the four analytes using the same sequence (e.g.: TWCA TWCA TWCA…). Instead, you may want to use something like TCWA WACT ACTW …

[1] Have the instructor check the flow rate before you connect the bubbler. Electromechanical hardware and sensors may be damaged if the wash bubbles are drawn into the instrument.

SECTION II: VISUAL INSPECTION

Once you have collected your data it is time to perform visual inspection of your results. Generate a MATLAB plots to display the complete response of each sensor (wash, reference and sample) for all the "sniffs" (hint: use the 'subplot' and 'hold on' command). To facilitate comparisons with other teams, plot the data from each analyte using the following color codes: tobacco (red), cedarwood (green), coffee (blue) and air (white). Can you visually tell the differences among samples from the four analytes?

SECTION III: PREPROCESSING

Extract the data from the final 60 seconds, which contain the response of the sensors to the analyte. Assuming a 10 Hz sampling rate, you have a 1200-dimensional feature vector for each 'sniff'. This data will be affected by two sources of noise:

- High-frequency noise caused by the electronics. This may be compensated for by processing the data with a mean filter, in which the response of sensor *s* at acquisition time *t* is averaged with the N previous and N subsequent acquisitions:

  $$s(t) = \frac{1}{2N+1} \sum_{k=-N}^{+N} s(t + k\Delta T) \qquad (1)$$

  Experiment with the value of N to find a reasonable value. You want to eliminate high-frequency noise while still preserving the shape of the sensor response, which may contain information relevant for classification. Generate a MATLAB plot of the filtered data.

- Low-frequency noise caused by sensor drift. Process the data by subtracting the baseline of the transient from all the samples in each transient:

  $$s(t) = s(t) - s(0) \qquad (3)$$

  Generate a MATLAB plot of the filtered data. Do the samples from each analyte 'look' more similar after removing the baseline?

Now that you have (digitally) filtered the data, it is time to select features from the sensor response that may be helpful to discriminate the different analytes. Look for differences in the response of the sensors to the various analytes and compute features that capture those differences: steady-state value, rise time, maximum slope, etc. Use your imagination! You could even sub-sample each sensor transient down to a few features[2], which is illustrated below[3]:

$$\vec{x} = \frac{1}{100}\left[\sum_{k=1}^{100} s(k) \quad \sum_{k=101}^{200} s(k) \quad \cdots \quad \sum_{k=501}^{600} s(k) \right]^T \qquad (2)$$

[2] Remember the curse of dimensionality: if you increase the number of features you also have to collect more data!
[3] Assuming two sensors, 10Hz sampling rate and a 60-second sample cycle, equation (2) would then generate a 12-dimensional feature vector for each "sniff".

Before you attempt to perform pattern classification, it is important to analyze possible correlations between the sensor response and environmental factors such as chamber temperature. Compute the average temperature of each "sniff" and plot it against the features you generated previously. What is the correlation coefficient between temperature and each feature? If there is a significant correlation, how could you compensate for it?

SECTION IV: DIMENSIONALITY REDUCTION

As a result of the feature-selection process that you performed in the previous section, you have obtained a feature vector of moderate size (10-20 dimensions). Now it is time to perform dimensionality reduction. In this section you are asked to implement Principal Components Analysis (PCA), and use the Linear Discriminant Analysis (LDA) routine posted on the website. Generate scatter plots of the PCA and LDA projections (hint: use the '`iss_plot_scatterx`' function). Which technique provides better results?

SECTION V: CLASSIFICATION

You are almost there… now it is time to build a classifier. You are asked to implement the k Nearest Neighbor rule covered in Lecture 12 according to the following procedure:

1. Split data set into training and test sets (80/20)
2. Compute [LDA or PCA] eigenvectors and eigenvalues using only training data
3. Sort eigenvectors by decreasing eigenvalues
4. Project the entire data set using these eigenvectors
    a. For LDA keep only the first N_C-1 eigenvectors
    b. For PCA keep 90-99% of the variance[4]
5. Classify the test data using the KNN rule on the low-dimensional projection
6. Compute the ratio of correctly classified test data

Generate scatter plots of training vs. test data (see '`iss_ex1`') for both PCA and LDA.

SUBMISSION OF RESULTS

Generate a brief report explaining how you collected your data, and the results that you obtained in sections II-V, documented with the appropriate MATLAB plots.

[4] The total variance in the data is the sum of the PCA eigenvalues. Therefore, the percentage of variance preserved by the M largest eigenvalues is $\frac{\sum_{k=1}^{M}\lambda_k}{\sum_{k=1}^{N}\lambda_k}$, where N is the total number of eigenvalues (or feature dimensions).