COMS4036A & COMS7050A
Computer Vision

Devon Jarvis, Nathan Michlo, Dr Richard Klein

Lab 3: Principal Component Analysis

In this lab, we will continue working on our background classifiers. In particular we will be reducing the dimensionality of our feature space using Principal Component Analysis (PCA) as well as include the foreground distribution in classifying pixels.

1 Principal Component Analysis

From Lab 2 you should have your 3 images (training, validation and test) and as well as the MR8 filter bank.

1. Read in your training image as an RGB image. Convert your training image to HSV and apply the MR8 filter bank to both the RGB and HSV image. This should give you 54 features (3 from the RGB and HSV channels respectively, as well as the output of the 8 MR8 filters applied to each image channel). Before we will create our classifier we are first going to reduce the dimensionality of our features.

2. For your training image, calculate the covariance of the features using all of the pixels (remember to flatten the length and width of the image).

3. Calculate the eigenvectors of the covariance matrix from Question 2. Hint: `np.linalg.eigh`$(\cdot)$ (this function returns your eigenvectors is ascending order with the columns being the eigenvectors. So you first principal component will be at `eig_vecs[:,-1]`. You may want to have your eigenvectors in descending order. For this use `np.flip(eig_vecs, axis=1))`.

4. Make sure that you now have the eigenvectors in a matrix with each column as an eigenvector (principal component) with the eigenvectors in descending order (the eigenvector corresponding to the largest eigenvalue must be in the first column). Use $\hat{X} = X \cdot E_{0:n}$ where X is your $P \times 54$ matrix containing the P pixels with 54 features, $E_{0:n}$ is your $(54 \times n)$ matrix of eigenvectors using only the first n columns, $\hat{X}$ is pixels after the dimensionality reduction and $\cdot$ denotes matrix multiplication. n determines the number of principal components we use and can be set to $n = 3$ to start. Note that for $\hat{X} = X \cdot E_{0:n}$ we are multiplying a $(P \times 54)$ matrix with a $(54 \times n)$ matrix and so we have $\hat{X}$ as a $(P \times n)$ matrix where n is also the new number of features.

2 Including a Foreground Distribution

1. Using the mask for your training data, separate your pixels from the output of Question 1 (the output image from the dimensionality reduction) into the foreground and background pixels.

2. Compute the mean (μ) and covariance matrix (Σ) for both the foreground (μ_f, Σ_f) and background (μ_b, Σ_b) pixels. Remember that the mean and covariance matrix are the parameters of

a normal distribution, $P(X|\mu_b, \Sigma_b)$, which tells us the likelihood of seeing a specific set of features, X, for a pixel given that it is a background pixel. We can consider another similar model $P(X|\mu_f, \Sigma_f)$, which tells us the likelihood of seeing a specific set of features, X, for a pixel given that is in the foreground. For now, we will use multivariate normal distributions for both models.

3. For your validation image, calculate the 54 original features and project this onto the n principal components from Section 1 using $\hat{V} = V \cdot E_{0:n}$ where V is the validation image's pixels and $\hat{V}$ is the dimensionality reduced validation image. Note that we still use the principal components from our training data for the projection.

4. Now infer the probability density (i.e. evaluate the probability density function) for each pixel's reduced dimensional representations. This indicates the probability of seeing these features given that it is a background or foreground pixel. For the foreground and background models, separately use the means and variances found in Question 2.

5. For each pixel in the validation image compute the value of

$$C = \frac{P(X|\mu_b, \Sigma_b)}{P(X|\mu_f, \Sigma_f) + P(X|\mu_b, \Sigma_b)}.$$

   Note that $(P(X|\mu_f, \Sigma_f) + P(X|\mu_b, \Sigma_b))$ normalizes the likelihood $P(X|\mu_b, \Sigma_b)$ so that it gives the probability of X being a background pixel, given the training data that we have seen.[1] If $C \geq 0.5$ classify it as a background pixel, else classify it as a foreground pixel. Does your model achieve better accuracy than when you just used a distribution for the background pixels with the 54 features?

6. Using this new approach, with any combination of features that we have implemented so far. Does this model perform better than the previous model? Does this model generalise better than the previous model? How do your thresholds compare between the two approaches? How do the ROC and Precision-Recall curves compare with the previous approach? What is the maximum value for C in this approach? What was the maximum value returned by the probability density function previously?

3 Submission

There is no submission for this lab, although future submissions will be based on this work so ensure that you complete this work.

[1]We will cover this maths formally in lectures soon.