Engineering and Applied Science Programs for Professionals
Whiting School of Engineering
Johns Hopkins University
685.621 Algorithms for Data Science

Data Processing

This document provides a rollup of the data processing methods covered in this module. The following list of topics is covered in this module allowing for an understanding of how data is processed to be used in a compact efficient manner for machine learning.

Contents

1 Introduction to Data Processing

Data is everywhere. Nowadays, it is most likely in digital form that can be processed by a machine. Data processing is a series of operations on data, including but not limited to retrieval, transformation, and/or classification to produce meaningful information. Figure 1 shows a generic example of a data processing system, or specifically a pattern recognition system.

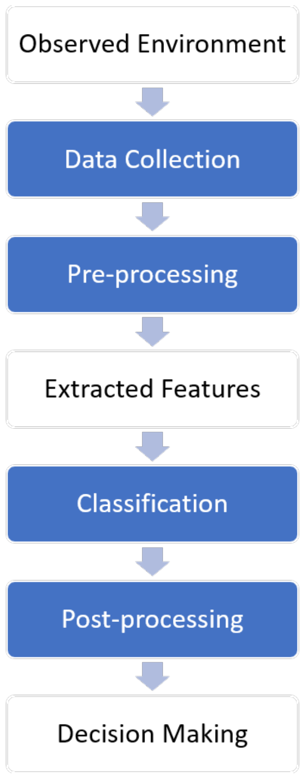


Figure 1: A data processing system

For each component in Figure 1, it may contain multiple subprocesses. First and foremost, what are the questions/problem sets that you would like to ask? Then, identify your environment. By observing environment, data can be collected and/or synthesized to simulate the environment. Collected data may or may not need to be cleaned depending on your problems and environment. Pre-processing may include cleaning data, generating primitive observations/features, ranking features, selecting features, extracting features, etc. Classification is the choice of model for machine learning or estimation. Post-processing includes actions for decision making, such as minimizing error rate or cost, evaluating classifiers, etc. All of these components may be constructed by data analysis, statistics, machine learning and their related techniques and methods.

2 Understanding of Data

In this section the types and format of data are described as raw data, such as account numbers, names, SSN, images, voice signals, videos, measurements, etc. This can also get even more complicated. Let's consider images that are jpeg, jpg, bmp, gif, tiff. In many cases, data can be the information held in headers of files, such as but not limited to, describing the name of the file, data format, compression type, location of the captured file, geographic location. The data in this case can be numeric (real, complex, positive) or symbolic (alphabet, ascii, words) in which values are [-10, -9, ...9, 10] or [easy, medium, difficult].

3 Data Cleansing

In some application areas of data science, data retrieval and data cleansing are critical to the entire analysis process. Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, or irrelevant parts of the data and then replacing, modifying, or deleting the noisy or coarse data. By doing so, it improves data quality and hopes to increase overall productivity and robustness along with reduce error rate. In addition, this step may take data scientists a very large potion of their time.

4 Data Transformations

Now let's discuss how we represent raw data in a manageable form. Features/attributes where large datasets, such as images, can be exploited with a much smaller set of features. We will call this feature generation, where the basic concept of generating features is to transform a given image, which contains an extensive number of data values in a two dimensional matrix, into a new set of features that will allow the image to be represented in a much smaller set of values. If the transform (transforms such as the discrete consine transform (DCT), fast Fourier transform (FFT), principal component analysis (PCA) will be discussed later) is suitably chosen the transform domain features can exhibit informational properties about the original input image in a compact vector form. This means that most of the classification related information is compressed in a relatively small number of values leading to a reduced feature space (Theodoridis and Koutroumbas, 2006). For example, consider a grayscale image that is of 512x512 pixels. This image would contain 262,144 pixel values, mapping the image into a new domain with the use of a transfer function can potentially represent the image with a significantly smaller number of values know as features or attributes. The basic reasoning behind transform-based features is that an appropriate chosen transform can exploit and remove redundancies that usually exist in digital images (Theodoridis and Koutroumbas, 2006). Consider the problem of character recognition, an input image that has been taken can be represented with a much smaller set of values to determine what the actual hand written character is. In the case of JPEG images a compression technique is used which is based on the discrete cosine transform (DCT). Generating features for discriminating between the numbers 0 - 9 using the DCT will eliminate redundant pixel information. When generating features derived from calculating the DCT, most of the energy lies in the frequency bands of the coefficients providing important information for class discrimination. This however leads to a large number of features, which for classification accuracy must be reduced.

5 Generating Features

In generating features it is important to understand the data being processed, images, video, audio, acoustic, radiometric, sonar, medical, banking, etc. In each of these data types the data can also be represented in various forms. Let consider an audio file, it can be represented as analog, various bits, frequencies, collection microphones, stereo, Dolby Digital, communication channel, etc. Based on the data type, the transformation needed will determine how well the features will represent the data in a compressed fashion. Let's consider taking the FFT of a standard 44.1kHz musical audio signal the vocals can be represented with a few real and imaginary coefficients.

5.1 Statistical methods (mean, moment, standard deviation, skewness, kurtosis)

The statistical methods can be used on raw data or by looking at larger sets of data produced by a metric or metrics, e.g., DFT, FFT, Wavelets, etc. The statistics calculated over the data vectors $\mathbf{x} = x_i = [x_1, x_2, \ldots, x_n] \in X_n$ are used to generate the features that represent the raw data in a much small form. The table below lists five statistics: mean, standard deviation, skewness, kurtosis, and entropy along with their calculation.

5.2 Principal Component Analysis (PCA) and the Karhunen-Loève Transform (KLT)

In this section the differences and uses for generating features is discussed using Principal Component Analysis and the Karhunen-Loève Transform. In PCA, there are assumptions made in the derivation that must be accounted

Table 1: Statistics for Generating Features

Test Statistics	**Statistical Function $\boldsymbol{F}(\cdot)$**
Mean	$F_\mu(\mathbf{x}) = \mu(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i$
Standard Deviation	$F_\sigma(\mathbf{x}) = \left(\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \mu(\mathbf{x})\right)^2\right)^{1/2}$
Skewness	$F_\gamma(\mathbf{x}) = \gamma(\mathbf{x}) = \dfrac{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \mu(\mathbf{x})\right)^3}{\sigma(\mathbf{x})^3}$
Kurtosis	$F_\kappa(\mathbf{x}) = \kappa(\mathbf{x}) = \dfrac{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \mu(\mathbf{x})\right)^4}{\sigma(\mathbf{x})^4}$
Entropy	$F_E(\mathbf{x}) = E(\mathbf{x}) = -\sum_{i=1}^{n}(x_i)\log(x_i)$

for when scaling of the variables and the applicability. We will see later that PCA can be used for dimensionality reduction, however, caution should be used to ensure information representing the data is preserved in the principal components. In the case of the Karhunen-Loève transform, the coefficients are random variables. The orthogonal basis functions in the KLT are determined by the covariance function and how it is processed. In this section, the goal is to generate features that are uncorrelated. Keep in mind that this is not a full explanation of the differences between PCA and KLT, rather an introduction is presented to ensure the use of PCA for dimensionality and KLT for generating features is introduced.

5.2.1 Principal Component Analysis (PCA)

The idea of feature extraction using PCA (Hotelling, 1933) is to represent a new space in a way to extract mutually uncorrelated features from the current space. The new features are known as the principal components after transform mapping. The dimensionality assessment is accomplished by extracting the principal components from the correlation matrix and retaining only the factors described in Kaiser's criterion (eigenvalues: $\lambda \geq 1$) (Kaiser, 1960). The criterion is used as a guide line to determine the number of principal components to retain by calculating the correlation matrix of the input features. Each observed variable contributes one unit of variance to the total variance in the data set. Hence, any principal component that has an eigenvalue, λ greater than one accounts for a greater amount of variance than had been contributed by one variable. Additionally, a principal component that displays an eigenvalue less than one indicates less variance than had been contributed by one variable. The covariance matrix, Σ, is used to extract eigenvectors, e, retaining only the number of principal components corresponding to Kaiser's criterion.

The basic concept of feature extraction using PCA is to map x onto a new space capable of reducing the dimensionality of the input space. The data is partitioned by variance using a linear combination of 'original' factors. To perform PCA, let $\mathbf{x} = [x_1, x_2, \ldots, x_n] \in X_n$ be a set of training vectors from the n-dimensional input space X_n. The set of vectors $\hat{\mathbf{x}} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_m] \in \hat{X}_m$ is a lower dimensional representation of the input training vectors $\mathbf{x}$ in the m-dimensional space $\hat{X}_m$. The vectors $\hat{x}$ are obtained by the linear orthonormal projection

$$\hat{\mathbf{x}} = \mathbf{A}^T(\mathbf{x} - \mu) \tag{1}$$

where $\mathbf{A}$ is an $[n \times m]$ matrix containing the top m eigenvectors and μ is the mean of the each set of features from $\mathbf{x}$.

5.2.2 Karhunen-Loève Transform (KLT)

The below writeup is from Theodoridis and Koutroumbas, 2006, pp. 266-270.

Let

$$\hat{\mathbf{x}} = \mathbf{A}^T \mathbf{x} \tag{2}$$

From the definition of the correlation matrix we have

$$R_y \equiv E[\mathbf{y}\mathbf{y}^T] = E[\mathbf{A}^T \mathbf{x}\mathbf{x}^T \mathbf{A}] = \mathbf{A}^T R_x \mathbf{A} \tag{3}$$

Where R_x is a symmetric matrix, and hence its eigenvectors are mutually orthogonal. Thus, if the matrix A is chosen so that its columns are the orthonormal eigenvectors $\mathbf{a}_i$, $i = 0, 1, \ldots, N-1$, of R_x, then R_y is diagonal

$$R_y = \mathbf{A}^T R_x \mathbf{A} = \Lambda \tag{4}$$

where Λ is the diagonal matrix having as elements on its diagonal the respective eigenvalues λ_i, $i = 0, 1, \ldots, N-1$, of R_x. Furthermore, assuming R_x to be positive definite the eigenvalues ar positive. The resulting transform is known as the Karhunen-Loève transform, and it achieves our original goal of generating mutually uncorrelated features. The KLT is of fundamental significance in pattern recognition and in a number of signal and image processing applications. Let us look at some of its important properties.

Mean square error approximation. From the definition of unitary matrices we have

$$\mathbf{x} = \sum_{i=0}^{N-1} y(i)\mathbf{a}_i \tag{5}$$

and

$$y(i) = \mathbf{a}_i^T \mathbf{x} \tag{6}$$

Let us now define a new vector in the m-dimensional subspace

$$\hat{\mathbf{x}} = \sum_{i=0}^{m-1} y(i)\mathbf{a}_i \tag{7}$$

where only m of the basis vectors are involved. Obviously, this is nothing but the projection of $\mathbf{x}$ onto the subspace spanned by the m (orthonormal) eigenvectors involved in the summation. If we try to approximate $\mathbf{x}$ by its projection $\hat{\mathbf{x}}$, the reslting mean square error is given by

$$E\left[\|\mathbf{x} - \hat{\mathbf{x}}\|^2\right] = E\left[\left\| \sum_{i=0}^{N-1} y(i)\mathbf{a}_i \right\|^2\right] \tag{8}$$

Our goal now is to choose the eigenvectors that result in the minimization MSE. From the above equation and taking into account the orthonormality property of the eigenvectors, we have

$$E\left[\left\| \sum_{i=0}^{N-1} y(i)\mathbf{a}_i \right\|^2\right] = E\left[\sum_i \sum_j (y(i)\mathbf{a}_i^T)(y(j)\mathbf{a}_j)\right] = \sum_{i=m}^{N-1} E[y^2(i)] = \sum_{i=m}^{N-1} \mathbf{a}_i^T E[\mathbf{x}\mathbf{x}^T]\mathbf{a}_i \tag{9}$$

Combining the previous two equations and the eigenvector definition, we finally get

$$E\left[\|\mathbf{x} - \hat{\mathbf{x}}\|^2\right] = \sum_{i=m}^{N-1} \mathbf{a}_i^T \lambda_i \mathbf{a}_i = \sum_{i=m}^{N-1} \lambda_i \tag{10}$$

Thus, if we chose in equation (7) the eigenvectors corresponding to the m largest eigenvalues of the correlation matrix, then the error in equation (10) is minimized, being the sum of the $N - m$ smallest eignevalues. Furthermore, it can be shown that this is also the minimum MSE, compared with any other approximation of x and an m-dimensional vector. This is the reason the KLT is also know as PCA.

A difference from the KLT results if we compute A in terms of the eigenvectors of the covariance matrix. This transform diagonalizes the covariance matrix Σ_y,

$$\Sigma_y = A^T \Sigma_x A = \Lambda \tag{11}$$

In general, the two are different and coincide for zero mean random vectors. In practice this is usually the case, because if it is not true one can replace each vector by $\mathbf{x} - E\mathbf{x}$. Despite that, it is still interesting to point out a difference between the two varants of the KLT. It can be shown that in this case, the resulting orthonormal basis (Σ_x eigenvectors $\hat{a}_i$) guarantees that the mean square error beetween $\mathbf{x}$ and its approximation given by

$$\hat{\mathbf{x}} = \sum_{i=0}^{m-1} y(i)\hat{\mathbf{a}}_i + \sum_{i=m}^{N-1} E\big[y(i)\hat{\mathbf{a}}_i, \qquad y(i) \equiv \hat{\mathbf{a}}_i^T \mathbf{x} \tag{12}$$

in minimum. In words, the last $N-m$ components are not random but are frozen to their respective mean value.

The optimality of the KLT, with respect to the MSE approximation, leads to excellent information packing properties and offers us a tool to select the m dominat features out of N measurement samples. However, although this may be a good criterion, in may cases it does not necessarily lead to maximum class seperability in the lower dimesional subspace. This is reasonable, since the dimensionality reduction is not optimized with respect to class separability. This is demonstarted via the example of Figure 2. The feature vectors in the two classes follow the Gaussian distribution with the same covariance matrix.

The ellipses show the curves of constant pdf values. We have computed the eigenvectors of the overall correlation matrix, and the resulting eigenvectors are shown in the figure. Eigenvector $\mathbf{a}_i$ is the one that corresponds to the largest eigenvalue. It does not take time for someone to realize that projection on $\mathbf{a}_1$ makes the two classes almost coincide. However, projecting on $\mathbf{a}_2$ keeps the teo classes separable.

Total Variance. Let $E[\mathbf{x}]$ be zero. If this is not the case, the mean can always be subtracted. Let $\mathbf{y}$ be the KLT vector of $\mathbf{x}$. From the respective definitions we have that $\sigma^2_{y(i)} \equiv E[y^2(i)] = \lambda_i$. That is, the eigenvalues of the input correlation matrix are equal matrix are equal to the variances of the transformed features. Thus, selecting those features, $y(i) \equiv \mathbf{a}_i^T\mathbf{x}$, corresponding to the m largest eigenvalues makes their sum variance $\sum_i \lambda_i$ maximum. In other words, the selected m features retain most of the total variance associated with the original random variables $x(i)$. Indeed, the latter is equal tot he trace of R_x, which we know from linear algebra to be equal to the sum of the eigenvalues $\sum_{i=0}^{N-1} \lambda_i$. It can be shown that this is a more general property. That is, from all possible sets of m features, obtained via any orthogonal linear transformation on $\mathbf{x}$, the ones resulting from the KLT have the largest sum variance.

Entropy. We know that the entopy of a process is defined as

$$H_y = -E[\ln p_y(\mathbf{y})] \tag{13}$$

and it is a measure of the randomness of the process. For a zero mean Gaussian multivariable m-dimensional process the entropy becomes

$$H_y = \frac{1}{2}E[\mathbf{y}^T R_y^{-1}\mathbf{y}] + \frac{1}{2}\ln|R_y| + \frac{m}{2}\ln(2\pi) \tag{14}$$

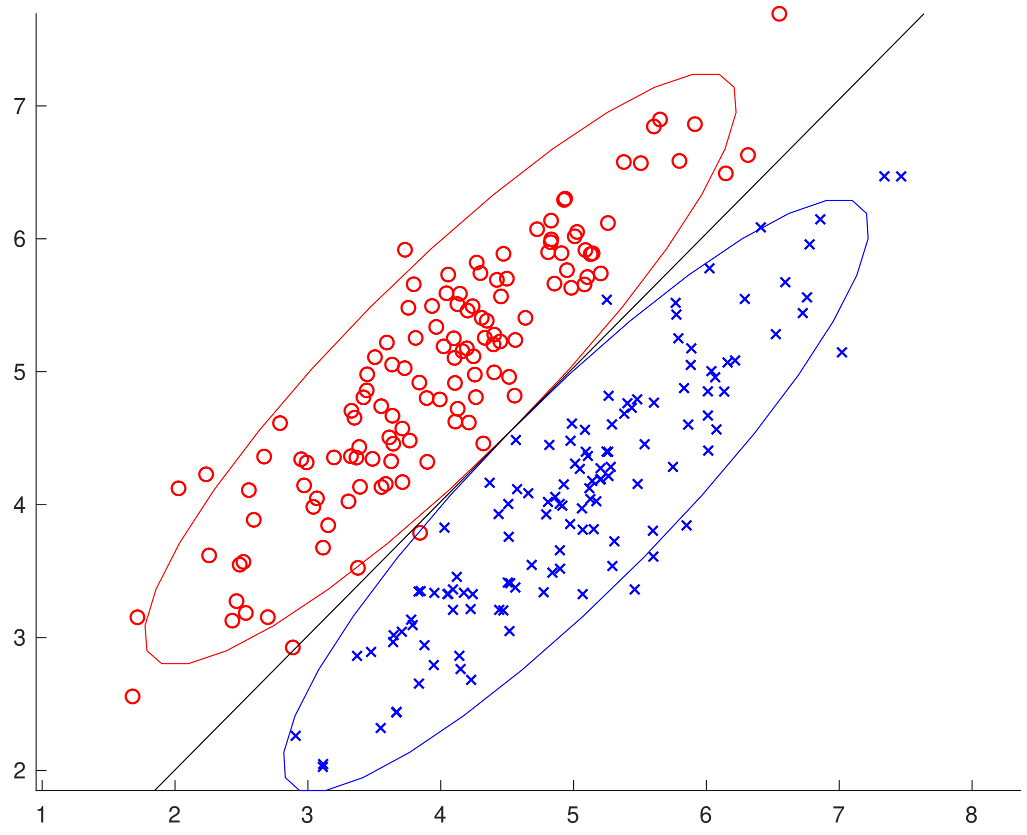
However,

$$E[\mathbf{y}^T R_y^{-1}\mathbf{y}] = E[\mathrm{trace}(\mathbf{y}^T R_y^{-1}\mathbf{y})] = E[\mathrm{trace}(R_y^{-1}\mathbf{y}\mathbf{y}^T)] = \mathrm{trace}(I) = m \tag{15}$$

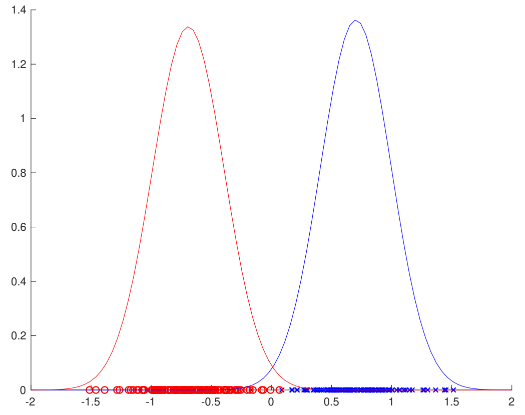
and using the known property from linear algebra the determinant is

$$\ln|R_y| = \ln(\lambda_0\lambda_1 \cdot \lambda_{m-1}) \tag{16}$$

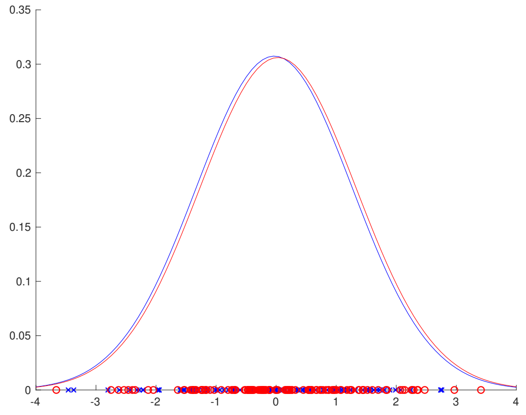
In words, selection of the m features that correspond to the m largest eigenvalues maximizes the entropy of the process. This is expected, because variance and randomness are directly related.



(a) Generated Data



(b) Smaller eigenvector projection



(c) Larger eigenvector projection

Figure 2: The KLT is not always best for pattern recognition. In this example, projection on the eigenvector withthe larger eigenvalue makes the teo classes coincide. On the other hand, projection on the other eigenvector keeps the class separated.

5.3 Discrete Fourier Transforms (DFT)

The Discrete Fourier transform (DFT) is the most widely used application in the fields of science and engineering, such as mathematics (linear systems, random process, probability, boundary-value systems), physics (quantum mechanics, optics, acoustics, astronomy), chemistry (spectroscopy, crystallography), and engineering (digital signal and image processing, telecommunications, computer vision). The theory of trigonometric series can be dated back to the beginning of 18th century. In 1747, Euler represented the movements of the planets in the form of a trigonometric series, which actually contained what is now called the Fourier series (Euler, 1760). In 1754, d'Alambert represented the reciprocal value of the mutual distance of two planets as a series of cosine functions (d'Alambert, 1754). H. H. Goldstine attributed to Carl Friedrich Gauss an algorithm, developed in 1805, similar to the fast Fourier transform (FFT) for the computation of the coefficients of a finite Fourier series (Goldstine, 1977). In 1807, Fourier discovered that a wide class of signals could be generated by summing scaled sine and cosine functions. These early techniques provided ideal tools for analyzing both periodic signals and the more general class of stationary signals. In 21st century, discrete Fourier transform remains one of the most frequently applied tools in science and technology. This is also because of the development in fast Fourier transform algorithms. The history of FFT development, dates back to Gauss, can be found in ((Goldstine, 1977; Caglayan, 2008).

Consider the N-point discrete Fourier Transform (DFT) of a signal $x(n), n = 0, 1, 2, \cdot, N - 1$ is defined by

$$Y(k) = \sum_{n=0}^{N-1} x(n)w(n,k), \qquad k = 0, 1, \cdot, N-1 \tag{17}$$

Similarly, the inverse Fourier transform of the sequence $x(n) = (x_1, x_2, \cdot, x_{N-1}), n = 0, 1, 2, \cdot, N - 1$ can be given by

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} Y(k)w^{-1}(n,k), \qquad n = 0, 1, \cdot, N-1 \tag{18}$$

where $w(n, k)$ are the complex roots of unity, that is

$$w(n,k) = e^{-2i\pi k/N}, \qquad n, k = 0, 1, \cdot, N-1 \tag{19}$$

Both equations (17) and (18) can be expressed in vector matrix form as

$$Y_N = \mathbf{F_N} x_N \tag{20}$$

$$x_N = \frac{1}{N} \mathbf{F_N^{-1}} Y_N \tag{21}$$

DFT matrix of size $N \times N$ composed of the twiddle factors as:

$$\mathbf{F_N} = \begin{bmatrix} w(0,0) & w(0,1) & w(0,2) & \cdots & w(0,N-1) \\ w(1,0) & w(1,1) & w(1,2) & \cdots & w(1,N-1) \\ w(1,0) & w(1,1) & w(1,2) & \vdots & w(2,N-1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w(N-1,0) & \cdots & \cdots & \cdots & w(N-1,N-1) \end{bmatrix} \tag{22}$$

Figure 3 illustrates the basis function of the discrete Fourier transform when $N = 16$, in which DFT matrix coefficients in each row is represented by a linear combination of each element of Fourier space, i.e. sines and cosines.

5.4 Discrete Cosine Transforms (DCT)

The standard DCT used in JPEG compression has two properties, i.e., the directional and frequency distributions of 8×8 blocks within an image (Rao and Yip, 1990). In JPEG compression on a two dimensional (2-D) signal, the zig-zag scan shown in Figure 3.3a is used to take advantage of the frequency distributions of the DCT shown

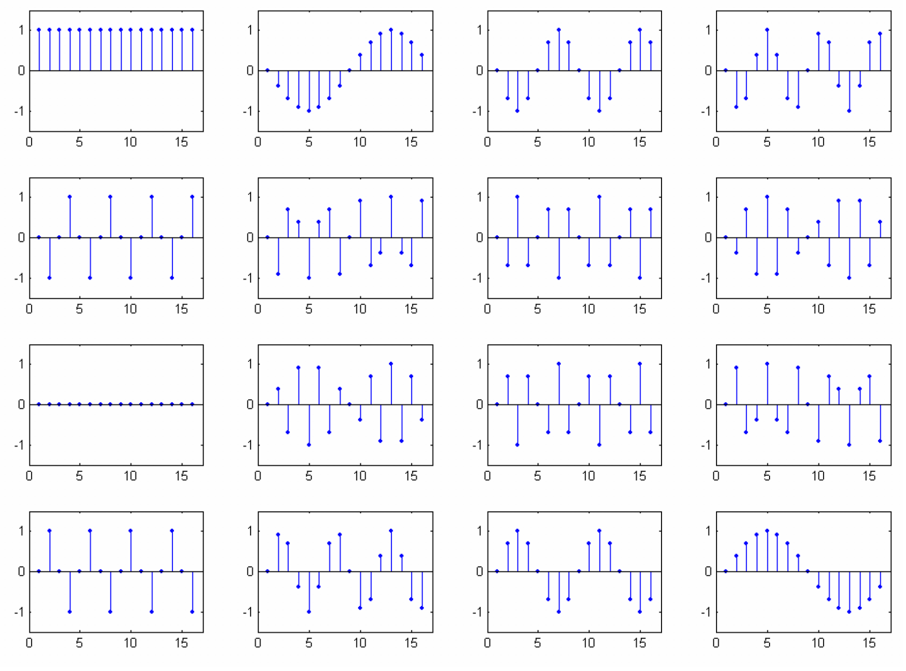


Figure 3: 16-point basis functions of discrete Fourier transform

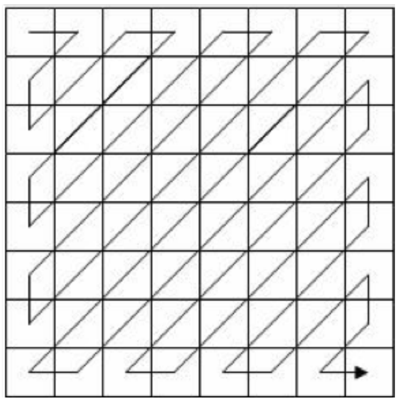
in Figure 3.3b (Brown and Shepherd, 1995, pp. 224). The DCT decomposition divides the coefficients into low, medium and high frequencies. Figure 3.3c shows the breakdown of the vertical, diagonal and horizontal directions of the coefficients. In this research both the frequencies and directions of the DCT are investigated to generate features. Figure 3.3d shows an 8×8 image with a horizontal edge between black and white pixels. The corresponding 2-D DCT of Figure 3.3d is shown in Figure 3.3g which has coefficients that are prominent along the first column. In Figure 3.3e an image is shown with a diagonal edge between black and white pixels with a corresponding 2-D DCT shown in Figure 3.3h which has coefficients located along the diagonal. In Figure 3.3f an image is shown with a vertical edge between black and white pixels with a corresponding 2-D DCT shown in Figure 3.3i which has coefficients located along the first row.

5.4.1 JPEG Image Representation Background

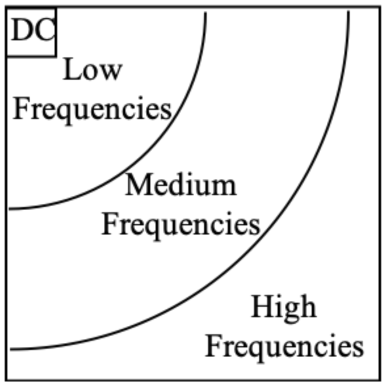
In this section, the basic structure of the JPEG image format and the steps in the compression process are described. This is followed by a brief introduction of JPEG image embedding methods.

The Joint Photographic Experts Group (JPEfG) format uses lossy compression to achieve high levels of compression on images with many colors (Elysium Ltd., 2004). JPEG is an international standard for still image compression, and is widely used for compressing gray scale and color images. JPEG images are commonly used for storing digital photos, and publishing Web graphics; tasks for which slight reductions in the image quality are barely noticeable. Due to the loss of quality during the compression process, JPEGs should be used only where image file size is important (Murry and vanRyper, 1994; Brown and Shepherd, 1995).

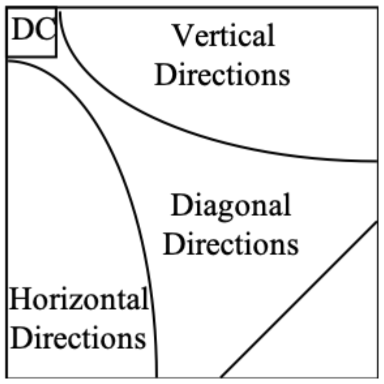
The JPEG encoder, shown in figure below, performs compression with the following sequential steps: image preprocessing (divides the input image into 8×8 blocks), forward DCT of each 8×8 block, quantization with scaling factor, separation of DC and AC coefficients, prediction of the DC coefficient and zig-zag scan the AC coefficients and Huffman encoder (there is a separate encoder for the DC and AC coefficients).



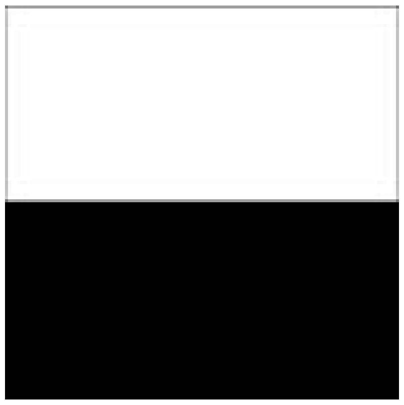
(a) JPEG Zig-Zag Pattern



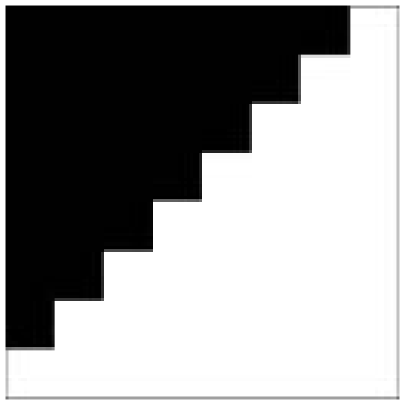
(b) DCT Frequencies



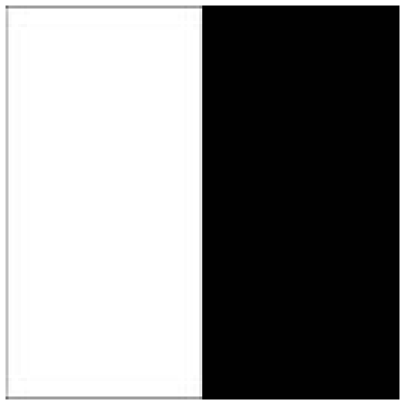
(c) DCT Directions



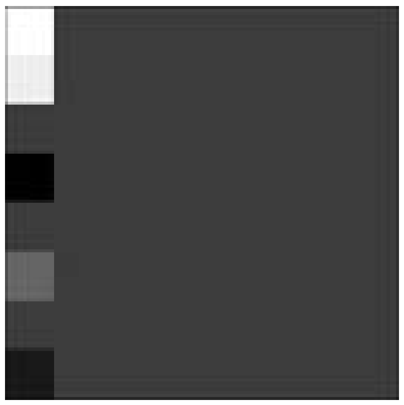
(d) Horizontal Image



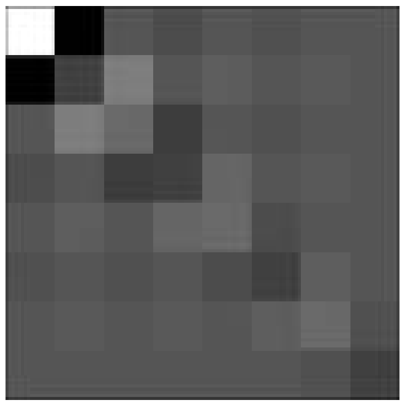
(e) Diagonal Image



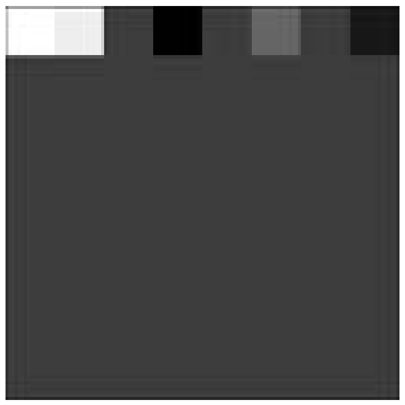
(f) Vertical Image



(g) DCT Horizontal



(h) DCT Diagonal



(i) DCT Vertical

Figure 4: DCT decomposition (a) zig-zag scan pattern (b) low, medium and high frequency distributions (c) vertical, diagonal and horizontal directions (d) 8×8 image with a horizontal edge between pixels (e) 8×8 image with a diagonal edge between pixels (f) 8×8 image with a vertical edge between pixels (g) 2-D DCT representation of horizontal image (h) 2-D DCT representation of diagonal image (i) 2-D DCT representation of vertical image.

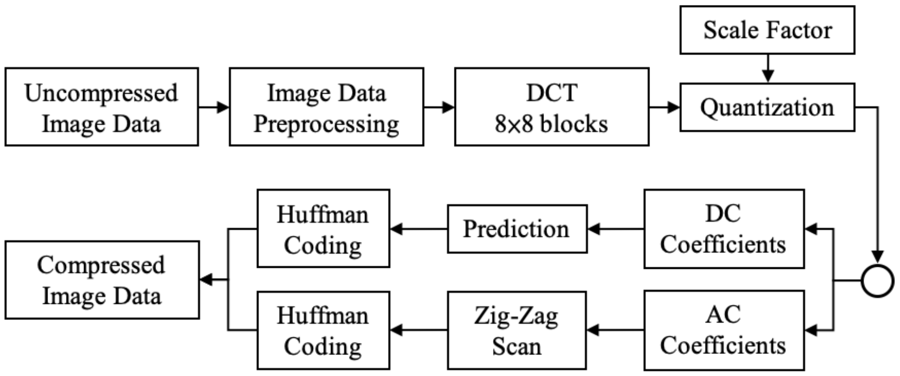


Figure 5: Block Diagrams of Sequential JPEG Encoder.

In JPEG decoding, all steps from the encoding process are reversed. The following procedure is a short description of the JPEG baseline systems.

Preprocessing block - Subdivides the image into blocks of 8×8 pixels and level-shift the original pixel values from the range $[0, 225]$ to the range $[-128, +127]$ by subtracting 128. The shifting procedure is a preprocessing step for the DCT calculation.

Forward DCT block - Perform a two dimensional discrete cosine transform (DCT) on each level-shifted block B from the Preprocessing block step. The two dimension DCT is defined as

$$C(n_1, n_2, k_1, k_2) = \begin{cases} \frac{1}{\sqrt{N_1 N_2}} & \text{if } k_1 = k_2 = 0 \\ \frac{1}{\sqrt{N_1 N_2}} \cos\left(\frac{\pi(2n_1+1)k_1}{2N_1}\right) \cos\left(\frac{\pi(2n_2+1)k_1}{2N_2}\right) & \text{if } 1 \leq k_1 \text{and} 1 \leq k_2 \leq N_2 - 1 \end{cases} \tag{23}$$

where $0 \leq n_1 \leq N_1$ and $0 \leq n_2 \leq N_2 - 1$. In the JPEG encoding process, $N_1 = N_2 = 8$. The transform is performed on the two dimensional matrix B as CBC^T.

The transform helps to remove data redundancy by mapping data from a spatial domain to the frequency domain. No compression has been achieved in this stage, but by changing representation of the information contained in the image block it makes the data more suitable for compression.

Quantization - Quantize the DCT coefficients block obtained from the previous step using the quantization table Q. The quantization table is a matrix used to divide the transformed block for compression purpose by reducing the amplitude of the DCT coefficient values and increasing the number of zero valued coefficients. The Huffman encoder takes advantage of these quantized values. When Qs is represented the value s is a scalar multiple, called the scale (or quality) factor, which defines the amount of compression within the image. Higher values of s yield higher compression. The below figure shows an instance of the typical quantization matrix Qs.

$$Qs = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 6 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \tag{24}$$

A set of four quantization tables are specified by the JPEG standard (Independent JPEG Group, 1998). After quantization, most of the DCT coefficients in the 8×8 blocks are truncated to zero values. It is the principal of lossiness in the JPEG transform-based encoder.

DC Coefficient Coding - The first coefficient, coefficient 1 (upper left) in figure b) below is called the "DC coefficient", short for the direct current coefficient, and represents the average brightness (intensity) of the component block. To encode the DC coefficient, the JPEG standard utilizes a Huffman difference code table that categorizes the value according to the number of k bits that are required to represent its magnitude. The value of the element is encoded with k bits.

AC Coefficients Coding - The remaining 63 coefficients are the "AC coefficients", short for the alternating current coefficients. The Huffman code assigns short (binary) codewords to each AC coefficient. The AC coefficient encoding scheme is slightly more elaborate than the one for the DC coefficient. For each AC array, a run-length of 0 elements is recorded. When encountering a non-zero element, the length of $0s$ is recorded and the number of k bits to represent the magnitude of the element is determined. The run-length and k bits are used as a category in the JPEG default Huffman table for assigning a code.

Using a zig-zag run encoder converts the 8×8 array of DCT coefficients into a column vector of length k (zig-zag goes from left to right and top to bottom). The "zig-zag" scan attempts to trace the DCT coefficients according to their significance, shown in figure below.

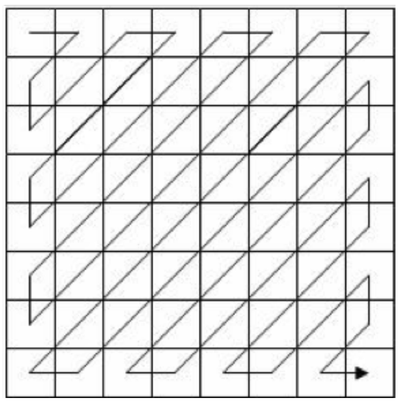


Figure 6: DCT decomposition zig-zag structure for an 8×8 block.

The coefficient ordering sequence is shown below.

$$\begin{bmatrix} 1 & 2 & 6 & 7 & 15 & 16 & 28 & 29 \\ 3 & 5 & 8 & 14 & 17 & 27 & 30 & 43 \\ 4 & 9 & 13 & 18 & 26 & 31 & 42 & 44 \\ 10 & 12 & 19 & 25 & 32 & 41 & 45 & 54 \\ 11 & 20 & 24 & 33 & 40 & 46 & 53 & 55 \\ 21 & 23 & 34 & 39 & 47 & 52 & 56 & 61 \\ 22 & 35 & 38 & 48 & 51 & 57 & 60 & 62 \\ 36 & 37 & 49 & 50 & 58 & 59 & 63 & 64 \end{bmatrix} \tag{25}$$

The Huffman encoding reduces the number of bits needed to store each of the 64 integer coefficients. For example, when a true color uncompressed image of size 512×512 pixels is stored the file size is 769 kilobytes. However, this same image store as a JPEG at a quality factor of 75, the image is stored in 200 kilobytes or smaller. The Huffman encoding tables for the DC and AC coefficients can be found in Gonzalez and Woods (2007), Elysium Ltd.

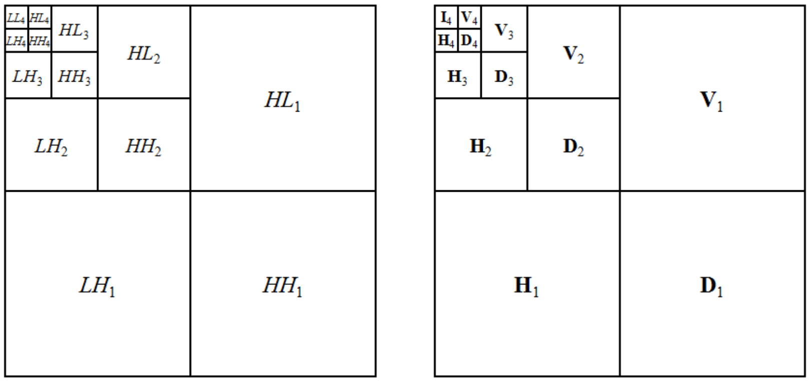
(2004), Independent JPEG Group (1998), and JPEG (1994).

One of the primary reasons using image embedding methods for creating stego files is due to the number of redundant portions within a digital image. The vast number of JPEG images on the Internet makes them ideal cover images for hiding secrete data and transmitting them as stego images. In JPEG steganography, the stego message is converted to binary values and embedded into DC and AC coefficients prior to Huffman encoding. By embedding at this stage, the stego message can be extracted without losing the message. The embedding methods range from simple embedding techniques that alter the least significant bits (LSB) of the coefficients such as JP Hide (Latham, 1999) and JSteg (Upham, 1993) to more complicated embedding techniques that maintain natural histograms of the coefficients such as; F5 (Westfeld, 2001; 2003), JP Hide (Latham, 1999), JSteg (Upham, 1993), Model-base (Sallee, 2003; 2006), Model-based Version 1.2 (Sallee, 2008a), OutGuess (Provos, 2004), Steganos (2008), StegHide (Hetzl, 2003) and UTSA (Agaian et al., 2006). The six tools selected provide a set of embedding methods that differ in embedding strategy. Investigation of these methods has provided an insight into six different and unique embedding capacities, embedding patterns and the appearance of the individual feature spaces. Another reason for selecting these particular tools is in previous research and existing steganalysis tools, these 6 embedding methods have been used for analysis (Provos and Honeyman, 2003; Lyu and Farid, 2004; Kharrazi et al., 2005; Shi et al., 2005; Xuan et al., 2005; Fu et al., 2006; Pevny and Fridrich, 2007).

In summary, a useful property of JPEG is that the degree of lossiness can be varied by adjusting the quality factor s (scale of the quantization table), shown in Figure 2.3. The ease of file sharing with JPEG images and its popularity over the internet has made JPEG image format a desirable cover file for many stego methods. Each embedding method leaves a signature that can be identified by various statistical measures. The next section describes feature generations methods used to identify changes made to a JPEG image.

5.5 Wavelets

This research employs wavelet transforms as a preprocess for deriving image features. The image decomposition employed here is based on separable quadrature mirror filters (QMF) originally used in (Farid, 2002). Applying the wavelet transform on a given image maps the image pixel from the spatial domain to the wavelet transform domain, i.e., to the frequency space. The image decomposition splits the frequency space into multiple orientations and levels. For each level, four subbands, LL, LH, HL, and HH, are created inheriting the characteristics of the image, identical (**I**), vertical (**V**), horizontal (**H**) and diagonal (**D**) information, correspondingly. The symbol L is denoted as a lowpass filter being used, while H would be the case of a highpass filter.

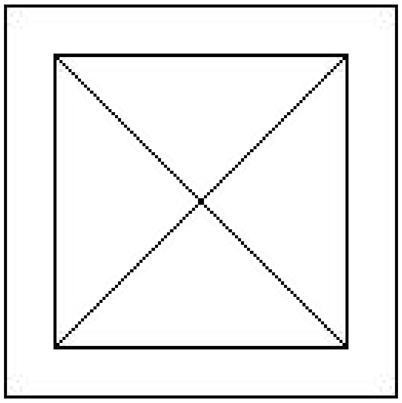


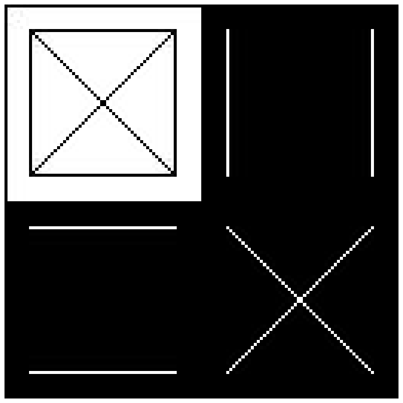
(a) Subband information (b) Edge information

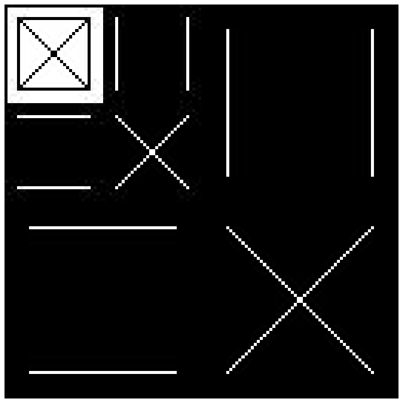
Figure 7: A four-scale wavelet decomposition

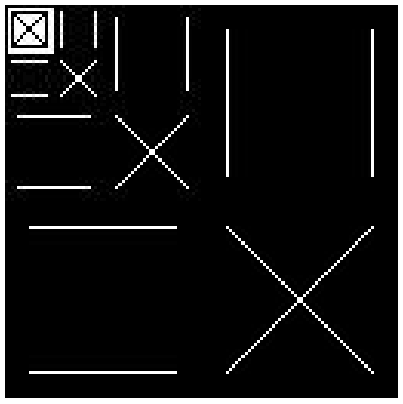
The number of iterations applied, also named as a scale or level, is denoted by a subscript i.

Recursively applying QMF to LL_i, Figure 7 presents an i^{th}-scale wavelet decomposition, where $i = 1, 2, 3, 4$. Suppose, there is a set of n images. For each image, a four-scale three-orientation QMF is utilized. A coefficient located within a wavelet transformed image is denoted by its edge characteristics and its position, i.e., $v_{j,i}(x, y)$, $h_{j,i}(x, y)$, $d_{j,i}(x, y)$, where $i = 1, 2, 3, 4$, $j = 1, 2, \ldots, n$, and (x, y) is the row and column index within a characteristic block. The coordinate (x, y) denotes the coefficient location within a block. Given an example image with vertical,


(a) Scale 0


(b) Scale 1


(c) Scale 2


(d) Scale 3

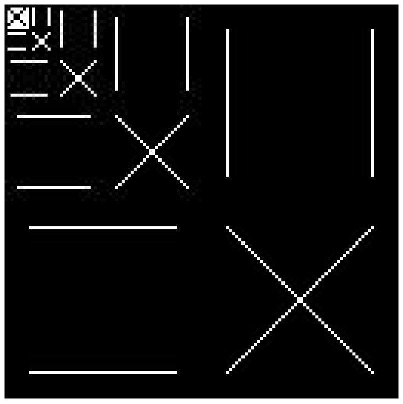

(e) Scale 4

Figure 8: Wavelet decompositions on an example image

horizontal, and diagonal lines as in Figure 8(a), Figure 8(b) to 8(e) demonstrate the wavelet decomposition results for each iteration from 1 to 4. The various scales of the wavelet coefficients are normalized for visualization purposes in Figure 8.

6 Feature Preprocessing

After features are generated it is necessary to preprocess the features that are to be used for classification. In many practical situations the classification model may receive input features whose values lie within different dynamic ranges. Thus, features with large values may inadvertently influence classification over features with small values. Another problem arises when a particular sample is not within the same area as the other features. To resolve these issues the feature preprocessing methods used in this research are data normalization (Theodoridis and Koutroumbas, 2006, pp. 214-215), data standardization (Dillon and Goldstein, 1984, pp. 12-13) and outlier removal (Barnett and Lewis, 1994). The training vectors in this section are represented by $x_i = [x_1, x_2, \ldots, xn] \in X_n$ with a dimension of n and the number of sample defined as m.

6.1 Data Preparation

Data preparation scales the features so that they have similar magnitudes. Some of the procedures used for data preparation are feature standardization (Dillon and Goldstein, 1984, pp. 12-13), feature min-max normalization (Theodoridis and Koutroumbas, 2006, pp. 214-215), min-max global normalization (Guyon et al., 2006, pp. 254), sigmoid normalization (Theodoridis and Koutroumbas, 2006, pp. 214-215) and softmax scaling (Theodoridis and Koutroumbas, 2006, pp. 214-215). We use zero-mean normalization (feature standardization) and min-max normalization (feature normalization) and describe them in more details.

1. **Min-max normalization**
   The Min-max normalization performs a linear scaling on the original data. The normalization is calculated by estimates of the minimum and maximum of the values. The normalization technique is defined for the m available data samples and the k^{th} feature as:

$$\hat{x}_{ik} = \frac{x_{ik} - min(x_k)}{max(x_i) - min(x_k)}((b-a)+a), \qquad k = 1,2,...,n \tag{26}$$

   where a and b are scaling factors. When a = 0 and b = 1 the individual feature values are in the range of [0,1]. In the event that the denominator of Equation (2.18) is equal to zero that feature is removed, avoiding the potential of normalizing a feature of constants.

2. **Z-score Normalization (Standardization)**
   The Z-score normaliztion is based on the mean and standard deviation of each feature. Each feature in this method is separately standardized by subtracting its mean and dividing by the standard deviation as follows:

$$\hat{x}_{ik} = \frac{x_{ik} - \mu_k}{\sigma_k}, \qquad k = 1,2,...,l \tag{27}$$

   where μ_k and σ_k are defined as:

$$\mu_k = \frac{1}{l}\sum_{i=1}^{l} x_{ik} \tag{28}$$

$$\sigma_k^2 = \frac{1}{l-1}\sum_{i=1}^{l}(x_{ik} - \mu_k)^2 \tag{29}$$

   and m is the number of samples. In the event that the standard deviation of a particular feature is zero (e.g., each element of the observed feature is a constant value), the feature is discarded.

6.2 Outlier Removal

An outlier is defined as a sample that is inconsistent with the existing sample distribution. The inconsistency is defined by the analyst observing the input data. Outliers can be discarded if the number of samples is small in comparison with the remaining samples, e.g., one or two samples. Various guides are provided by Barnett and Lewis (1994) to determine a small number of outliers. When a large number of outliers exist, care must be taken by the analyst. In this case, the classification model may have to be trained to accommodate the presence of outliers, e.g., expectation maximization can be trained using ellipsoids. Two outlier removal techniques are used in the case of multivariate outliers in this section. The first is a technique in which the mean is used to identify an upper and lower boundary of a confidence interval to identify an outlier and remove the sample (Barnett and Lewis, 1994). The second is a multivariate outlier technique presented by Wilks (1963).

1. **Confidence Interval Outlier Removal**
In confidence interval outlier removal, any sample outside of the confidence interval is considered an outlier. This method assumes the data is normally distributed and generates a confidence interval for each feature. The first step identifies an upper and lower limit means from the global mean as follows:

$$\mu_{upper} = \frac{1}{S_{upper}} \sum_{i \in S_{upper}} x_i, \qquad for \quad x_i > \mu \tag{30}$$

$$\mu_{lower} = \frac{1}{S_{lower}} \sum_{i \in S_{lower}} x_i, \qquad for \quad x_i < \mu \tag{31}$$

where μ is the global mean vector

$$\mu = \frac{1}{l} \sum_{i=1}^{l} x_i \tag{32}$$

where m is the number of samples, Supper and Slower are the number of samples meeting the criteria $x_i > \mu$ and $x_i < \mu$, respectively. The term $i \in S_{lower}$ and $i \in S_{upper}$ indicate the indices when the criteria $x_i < \mu$ and $x_i > \mu$ are met. This now leads to the confidence interval defined as:

$$[\mu_{lower} - (\alpha(\mu - \mu_{lower})), \mu_{upper} + (\alpha(\mu_{upper} - \mu))] \tag{33}$$

where m is the parameter set by the user. A good starting point is $\alpha = 0.5$ allowing the parameter to be adjusted based on the data set being analyzed. The terms multiplied by α in the above equation can be replaced by the critical of the t- distribution as described by Barnett and Lewis (1994, page 74) providing robustness of validity for the confidence interval. Another alternate modification to the above equation is to simply replace $(\mu - \mu_{lower})$ by the standard deviation of μ_{lower} and $(\mu_{upper} - \mu)$ by the standard deviation of μ_{upper} allowing the standard deviation to determine the confidence interval.

2. **Wilks' Outlier Removal**
Wilks' outlier removal technique uses an upper bound for detection of a single outlier from a set of normal multivariate samples in which the maximum squared Mahalanobis distance (as shown in the equation below) approaches an F distribution (Wilks, 1963).

$$D_i^2 = (x_i - \mu)\Sigma^{-1}(x_i - \mu)^T \tag{34}$$

In multivariate outlier detection the normality between samples is assessed. A partial mathematical description is provided by Rencher (2002, pp. 101-104) and expanded in application by Trujillo-Ortiz, et al. (2008).

Determining the threshold is defined by the F distribution critical value (inverse of F cumulative distribution function) with n and $(m - n - 1)$ degrees of freedom using the Bonferroni correction (Bonferroni, 1935; 1936). The final critical value is defined by:

$$C_v = \frac{n(l-1)^2}{l(l-n-1) + (lnF)} \tag{35}$$

The index of an outlier(s) is identified by the following criteria:

$$D_i^2 \geq C_v \tag{36}$$

This method is provided in full detail by Trujillo-Ortiz, et al. (2008).

7 Feature Ranking and Selection

The major task in feature selection, given a large number of features, is to select the most important features and reduce the dimensionality while retaining class discriminatory information. This procedure is important when determining which features are to be used to train the classification model. If features with little discrimination power are selected the subsequent classification model will lead to poor classification performance. On the other hand, if information rich features are selected the design of the classifier can be greatly simplified. In a more quantitative description, feature selection leads to large between-class distances and small within-class distances in the feature space. That is, features should separate different classes by a large distance, and should have small distance values between objects in the same class. Several methods are available to identify individual features with linear separation, a few ranking and selection methods include; divergence measure (Fukunaga, 1990; Theodoridis and Koutroumbas, 2006), Bhattacharyya distance (Bhattacharyya, 1943; Fukunaga, 1990) and Fisher's linear discriminant ratio (Fisher, 1936; 1943; Dillon and Goldstein, 1984; Fukunaga, 1990; van der Heijden et al., 2004; Bishop, 1995, 2006; Theodoridis and Koutroumbas, 2006).

When measuring nonlinear class separability, care must be taken when using feature ranking methods. Ranking methods developed for specific classifiers are often best suited for determining the best set of ranked features. For neural network classifiers features are ranking and selected based on a saliency metric (Ruck et al., 1990; Belue and Bauer, 1995) and signal-to-noise ratio (Bauer et al., 2000). For kernel based classifiers, such as kernel Fisher's discriminant and support vector machines, method-specific techniques are best suited for ranking. These techniques include recursive feature elimination (Guyon et al., 2002; Guyon, 2007), zero-norm feature ranking (Weston et al., 2003), gradient calculations using recursive feature elimination (Rakotomamonjy, 2003), and kernel Fisher's discriminant using recursive feature elimination (Louw and Steel, 2006).

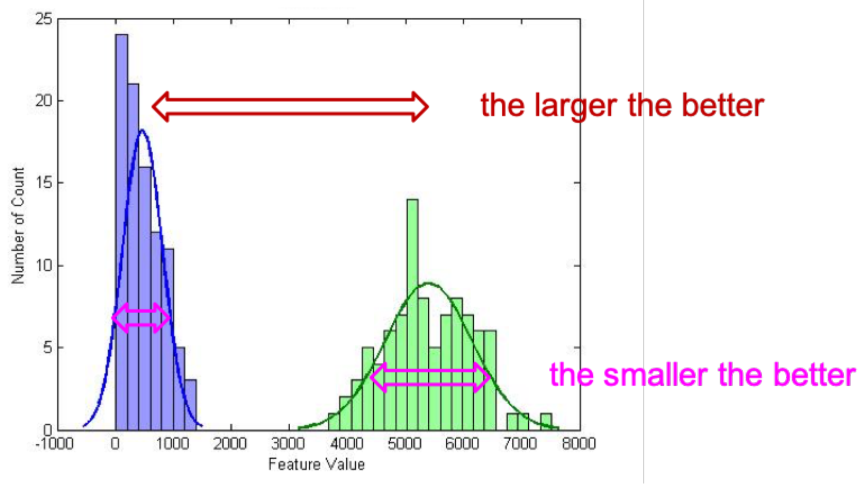


Figure 9: Feature Selection - ranking method based on between class means and within class variances.

7.1 Bhattacharyya Distance

The Bhattacharyya distance is used as a class separability measure. For two-class normal distributions the Bhattacharyya distance is defined as:

$$B = \frac{1}{8}(\mu_{-1} - \mu_{+1})^T \left(\frac{\Sigma_{-1} + \Sigma_{+1}}{2} \right)^{-1} (\mu_{-1} - \mu_{+1}) + \frac{1}{2} ln \left(\frac{|\frac{\Sigma_{-1}+\Sigma_{+1}}{2}|}{\sqrt{|\Sigma_{-1}||\Sigma_{+1}|}} \right) \tag{37}$$

where $||$ denotes the determinant of the respective matrix. The Bhattacharyya distance corresponds to the optimum Chernoff bound when $\Sigma_{-1} = \Sigma_{+1}$. It is readily seen that in this case the Bhattacharya distance becomes proportional to the Mahalanobis distance between the means. It should be noted that the Bhattacharya distance consists of two terms. The first term gives the class separability due to the mean difference and disappears when $\mu_{-1} = \mu_{+1}$. The second term gives the class separability due to the covariance difference and disappears when $\Sigma_{-1} = \Sigma_{+1}$ (Fukunaga, 1990).

The Bhattacharyya distance for the multi-class case is represented as:

$$B_{ij} = \frac{1}{8}(\mu_i - \mu_j)^T \left(\frac{\sigma_i + \sigma_j}{2}\right)^{-1} (\mu_i - \mu_j) + \frac{1}{2} ln\left(\frac{\sigma_i^2 + \sigma_j^2}{2\sigma_i \sigma_j}\right), \qquad for \quad i \neq j \tag{38}$$

where $i, j \in \mathbb{Z}$ in this case corresponding to the classes $C = C_j = [C_1, C_2, \ldots, C_c], j = 1, 2, \ldots, c$. In this case for each feature an individual class is compared to the remaining classes based on distance. The features are assigned a ranking value according to the greatest distance between classes.

7.2 Fisher's Linear Discriminant Ratio (FDR/F-Score)

The FDR is used to quantify the separability capabilities of individual features (Fisher, 1936). FDR is a simple technique which measures the discrimination of sets of real numbers. The within-class scatter matrix is defined as

$$S_W = \sum_C P_C S_C \tag{39}$$

where S_c is the covariance matrix for class $\mathbf{C} \in -1, +1$

$$S_W = \sum_{\substack{i=1, \\ i \in C}}^{l_C} (\mathbf{x} - \mu_C)(\mathbf{x} - \mu_C)^T \tag{40}$$

and P_C is the *a priori* probability of class $\mathbf{C}$. That is, $P_C \approx \lambda_C / \lambda$, where λ_C is the number of samples in class $\mathbf{C}$, out of a total of λ samples. The between-class scatter matrix is defined as

$$S_B = \sum_{\mathbf{C}} (\mu_C - \mu_C)(\mu_C - \mu_C)^T \tag{41}$$

where μ is the global mean vector

$$\mu = \frac{1}{l} \sum_{i=1}^{l} \mathbf{x}_i \tag{42}$$

and the class mean vector μ_C is defined as

$$\mu_C = \frac{1}{l_C} \sum_{\substack{i=1, \\ i \in C}}^{l_C} \mathbf{x}_i \tag{43}$$

These criteria take a special form in the one-dimensional, two-class problem. In this case, it is easy to see that for equiprobable classes $|S_W|$ is proportional to $\sigma_i^2 + \sigma_j^2$ and $|S_B|$ proportional to $(\mu_{-1} - \mu_{+1})^2$. Combining SB and SW, the Fisher's Discriminant ratio results in the following equation

$$\boldsymbol{FDR} = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{44}$$

FDR is sometimes used to quantify the separability capabilities of individual features. For the multi-class case, averaging forms of FDR can be used. One possibility is

$$\boldsymbol{FDR} = \sum_{i}^{M} \sum_{j \neq i}^{M} \frac{(\mu_i - \mu_j)^2}{\sigma_i^2 + \sigma_j^2} \tag{45}$$

where the subscripts i, j refer to the mean and variance corresponding to the feature under investigation for the classes C_i, C_j, respectively. For the one-dimensional multi-class case, the Fisher's discriminant ratio is modified as:

$$FDR_{ij} = \frac{(\mu_i - \mu_j)^2}{\sigma_i^2 + \sigma_j^2} \tag{46}$$

7.3 Signal-to-Noise Feature Selection

One method for neural networks feature selection uses a signal-to-noise ratio (SNR) saliency measure (Bauer et al., 2000). This measure directly compares the saliency of a feature to that of an injected noise feature. The SNR saliency measure is computed using the following:

$$SNR_i = 10 \log_{10} \left(\frac{\sum_{j=1}^{J} (w_{i,j}^1)^2}{\sum_{j=1}^{J} (w_{N,j}^1)^2} \right) \tag{47}$$

where SNR_i is the value of the SNR saliency measure for feature i, J is the number of hidden nodes, $w_{i,j}^1$ is the first layer weight from node i to node j, and $w_{N,j}^1$ is the first layer weight from the injected noise node N to node j. The weights, for the noise feature are initialized and updated in the same fashion as the weights, emanating from the other features in the first layer. The injected noise feature is created such that its distribution follows that of a Uniform $(0, 1)$ random variable. The SNR screening method potentially requires only a single training run, because the SNR saliency measure appears highly robust relative to the effects of weight initialization. For the classification method probabilistic neural network described in the classification section, this method is used to determine the appropriate subset of features.

8 Dimensionality Reduction

Another approach to reducing the dimension of the input features is to use a transformed space instead of the original feature space. For example using a transformation $f(x)$ that maps the data points $\mathbf{x}$ of the input space, $x[n]$, into a reduced dimensional space $x'[p]$, where $n > p$, creates features in a new space that may have better discriminatory properties. Classification is based on the new feature space rather than the input feature space. The advantage of feature extraction with the use of dimensionality reduction as described here over feature selection is that no information from any of the elements of the measurement vector is removed. In some situations feature extraction is easier than feature selection. A disadvantage of feature extraction is that it requires the determination of a suitable transformation $f(x)$. Some methods include principal component analysis (Hotelling, 1933; Dillon and Goldstein, 1984) and kernel principal component analysis (Scholkopf et al., 1998; Bishop, 2006). If the transformation chosen is too complex, the ability to generalize from a small data set will be poor. On the other hand, if the transformation chosen is too simple, it may constrain the decision boundaries to a form that is inappropriate to discriminate between classes. Another disadvantage is that all features are used, even if some of them have noise like characteristics. This might be unnecessarily expensive in term of computation (van der Heijden et al., 2004). It should be noted that the transformation used for the input features in the training of the classification model should also be used for the testing features.

Previously in Section 3 PCA was used to generate features, in this section the reduction in dimensionality can also be accomplished using PCA. How ever PCA alone may not be sufficient, it may require the used of factor analysis in which the factors are rotated in order to group the appropriate variables based on their correlations. Examples will be given, however the exact details are outside of the scope of this document.

8.1 Kernel PCA

The Kernel Principal Component Analysis (Kernel PCA) is the non-linear extension of the ordinary linear PCA (Scholkopf et al., 1998). The input training vectors is $\mathbf{x} = [x_1, x_2, \cdots, x_\lambda] \in \Upsilon^n$ are mapped by a nonlinear transformation $\phi(\cdot) : X \to F$ to a new dimensional feature space $F \in \Upsilon^\lambda$. The mapping $\phi(\cdot)$ is represented in the kernel PCA by a kernel function $K(\cdot,\cdot)$ which defines an inner product in Υ^λ. This yields a non-linear (kernel) projection of data which has a general definition as

$$\hat{\mathbf{z}} = \hat{\mathbf{A}}^T K(\mathbf{x}_i, \mathbf{x}_j) + \mathbf{b} \tag{48}$$

where $\hat{A}$ is an $[\lambda \times p]$ matrix containing the top p values, $\mathbf{b}$ is a bias vector and $\hat{\mathbf{z}} \in \Upsilon^p$ is the vector of extracted features. The eigenvectors are not computed directly from the kernel matrix $K(\cdot,\cdot)$. The kernel matrix must be centered as follows:

$$\mathbf{K}_c = K(\mathbf{x}_i, \mathbf{x}_j) - 1_{[\lambda\times\lambda]} K(\mathbf{x}_i, \mathbf{x}_j) - K(\mathbf{x}_i, \mathbf{x}_j) 1_{[\lambda\times\lambda]} + 1_{[\lambda\times\lambda]} K(\mathbf{x}_i, \mathbf{x}_j) 1_{[\lambda\times\lambda]} \tag{49}$$

where $1_{[\lambda\times\lambda]}$ is a $[\lambda \times \lambda]$ matrix in which every value is $1/\lambda$. The eigenvalues, λ, and eigenvectors, $\mathbf{e}$, are determined with the use of $\mathbf{K}_c$. The bias vector $\mathbf{b}$ is computed as:

$$\mathbf{b} = \hat{\mathbf{A}}^T \left(1_{[\lambda\times\lambda]} K(\mathbf{x}_i, \mathbf{x}_j) 1_\lambda - K(\mathbf{x}_i, \mathbf{x}_j) 1_{[\lambda\times\lambda]} \right) \tag{50}$$

where 1_λ is an $[\lambda \times 1]$ vector with each element equal to $1/\lambda$.

9 References

[1] Bishop, Christopher M., *Pattern Recognition and Machine Learning*, Springer, 2006

[2] Brown, W. and Shepherd, B. J., *Graphics File Formats: Reference and Guide*, Greenwich, CT: Manning Publications, 1995

[3] Caglayan, O., *Digital Multimedia System Security Based on Transform Domain Techniques*, Dissertation, 2008

[4] Cormen, Thomas H., Leiserson, Charles E., Rivest, Ronal L., and Stein, Clifford, *Introduction to Algorithms*, 3rd Edition, MIT Press, 2009

[5] d'Alambert, J.R., *Researchers sur diferentes points importants du systeme du monde*, vol. 2, pp. 66, 1754

[6] Duda, Richard O., Hart, Peter E., and Stork, David G., *Pattern Classification*, 2nd Edition, John Wiley & Sons, Inc., 2001

[6] Duhamel, P., Hollman, H., *Existence of a 2n FFT algorithm with a number of multiplications lower than* 2^{n+1} Electron. Lett., vol 20, pp. 690-692, 1984

[8] Duin, Robert P.W., Tax, David and Pekalska, Elzbieta, *PRTools*, http://prtools.tudelft.nl/

[9] Elysium Ltd., *Home of the JPEG committee—*, Retrieved February 20, 2005, http://www.JPEG.org/, 2004

[10] Euler, L., *Nova Acta Acad. Sci. Petrop.*, v(1), 14, 435-542-84, publ. 1760

[11] Franc, Vojtech and Hlavac, Vaclav, *Statistical Pattern Recognition Toolbox*, https://cmp.felk.cvut.cz/cmp/software/stprtool/index.html

[12] Goldstine, H.H., *A History of Numerical Analysis from the 16th Through the 19th Century*, pp. 249-253, New York, Springer-Verlag, 1977 (See also M. T. Heideman, D. H. Johnson, and C. S. Burrus, "Gauss and the history of the fast Fourier transform," IEEE ASSP Magazine 1 (4), 14–21 (1984))

[13] Gonzalez, R. C. and Woods R., *Digital Image Processing*, 3rd Edition. Upper Saddle River, NJ: Prentice Hall, 2007

[14] Hotelling, H., *Analysis of a complex of statistical variables into principal components*, Journal of Educational Psychology, Number 24, pp. 417–441, 1933

[15] Independent JPEG Group, *Independent JPEG Group*, Retrieved February 20, 2005, http://www.ijg.org/, 1988

[16] JPEG, *Information technology - Digital compression and coding of continuous-tone still images: Requirements and guidelines*, ISE/IEC IS 10918-1, American National Standards Institute, Retrieved June 15, 2008 http://www.w3.org/Graphics/JPEG/itu-t81.pdf., 1994

[17] Kaiser, H. F., *The application of electronic computers to factor analysis*, Educational and Psychological Measurement, Volume 20, Number 1, pp. 141-151, 1960

[18] Machine Learning at Waikato University, *WEKA*, https://www.cs.waikato.ac.nz/ ml/index.html

[19] Murry, J. D. and vanRyper, W., *Encyclopedia of Graphics File Formats*, Sebastopol, CA: O'Reilly & Associates, Inc., 1994

[20] Rao, K. P. and Yip, P., *Discrete Cosine Transform Algorithms, Advantages, Applications*, San Diego, CA: Academic Press, Inc., 1990

[21] Theodoridis, S. and Koutroumbas, K.. *Pattern Recognition Third Edition*, San Diego, CA: Academic Press, 2006