Kernel Methods for Pattern Analysis

Dr. C. Chandra Sekhar

Speech and Vision Laboratory
Dept. of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai-600036

chandra@cse.iitm.ac.in
Outline of the Talk

• Kernel methods for pattern analysis
• Support vector machines for classification
• Support vector regression
• Kernel based clustering
• Learning kernels from data
• Applications
  – Handwritten character recognition
  – Image classification
  – Speaker change detection
  – Speech recognition
  – Cryptanalysis
Speech Recognition

<table>
<thead>
<tr>
<th>is</th>
<th>bu</th>
<th>le</th>
<th>Tin</th>
<th>ki</th>
<th>mu</th>
<th>khya</th>
<th>sa</th>
<th>mA</th>
<th>chAr</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>nnAL</td>
<td>mu</td>
<td>da</td>
<td>la</td>
<td>mei</td>
<td>ccar</td>
<td>sel</td>
<td>vi</td>
<td>jey</td>
</tr>
<tr>
<td>I</td>
<td>rO</td>
<td>ju</td>
<td>vAr</td>
<td>ta</td>
<td>lo</td>
<td>lu</td>
<td>mu</td>
<td>khyam</td>
<td>sa</td>
</tr>
</tbody>
</table>
Speaker Segmentation

Multiple speakers

Multispeaker conversation

Speaker changes

Segmented speech
Handwritten Character Recognition

dha

dya

gha

ka

pha
Image Classification

- Residential Interiors
- Mountains
- Military Vehicles
- Sacred Places
- Sunsets & Sunrises
Pattern Analysis

• Pattern: Any regularity, relation or structure in data or source of data

• Pattern analysis: Automatic detection and characterization of relations in data

• Statistical and machine learning methods for pattern analysis assume that the data is in vectorial form

• Relations among data are expressed as
  – Classification rules
  – Regression functions
  – Cluster structures
Evolution of Pattern Analysis Methods

• **Stage I (1960’s):**
  - Learning methods to detect linear relations among data
  - Perceptron models

• **Stage II (Mid 1980’s):**
  - Learning methods to detect nonlinear patterns
  - Gradient descent methods: Local minima problem
  - Multilayer feedforward neural networks

• **Stage III (Mid 1990’s):**
  - Kernel methods to detect nonlinear relations
  - Problem of local minima does not exist
  - Methods are applicable to non-vectorial data also
    - Graphs, sets, texts, strings, biosequences, graphs, trees
  - Support vector machines
Artificial Neural Networks
Neuron with Threshold Logic Activation Function

\[
\sum_{i=1}^{M} W_i X_i - \theta
\]

- McCulloch-Pitts Neuron
Perceptron

- **Linearly separable classes**: Regions of two classes are separable by a linear surface (line, plane or hyperplane)
Hard Problems

- Nonlinearly separable classes
Neuron with Continuous Activation Function

\[ \sum_{i=1}^{M} w_i x_i - \theta \]

Input \( x_1 \), \( x_2 \), \( x_M \)
Weights \( w_1 \), \( w_2 \), \( w_M \)
Activation value \( a \)
Sigmoidal activation function \( s = f(a) \)
Output signal
Multilayer Feedforward Neural Network

• Architecture:
  – Input layer ---- Linear neurons
  – One or more hidden layers ---- Nonlinear neurons
  – Output layer ---- Linear or nonlinear neurons
Gradient Descent Method

- Error
- Weight
- Local minimum
- Global minimum
Artificial Neural Networks: Summary

- Perceptrons, with threshold logic function as activation function, are suitable for linearly separable classes.
- Multilayer feedforward neural networks, with sigmoidal function as activation function, are suitable for nonlinearly separable classes.
  - Complexity of the model depends on:
    - Dimension of the input pattern vector
    - Number of classes
    - Shapes of the decision surfaces to be formed
  - Architecture of the model is empirically determined
  - Large number of training examples are required when the complexity of the model is high
  - Local minima problem
  - Suitable for vectorial type of data
Kernel Methods
Kernel Methods for Pattern Analysis

• Supervised tasks:
  – **Classification**: Support vector machine
  – **Regression**: Support vector regression

• Unsupervised tasks:
  – **Clustering**:
    • Kernel based clustering
    • Support vector clustering
  – **Kernel principal component analysis** for dimension reduction
Key Aspects of Kernel Methods

- Kernel methods involve
  - **Nonlinear transformation** of data to a **higher dimensional feature space** induced by a **Mercer kernel**
  - Detection of **optimal linear solutions** in the kernel feature space

- Transformation to a higher dimensional space is expected to be helpful in conversion of nonlinear relations into linear relations (**Cover’s theorem**)
  - Nonlinearly separable patterns to linearly separable patterns
  - Nonlinear regression to linear regression
  - Nonlinear separation of clusters to linear separation of clusters

- Pattern analysis methods are implemented in such a way that the kernel feature space representation is **not explicitly required**. They involve computation of **pair-wise inner-products** only.

- The pair-wise inner-products are computed efficiently directly from the original representation of data using a **kernel function** (**Kernel trick**).
Illustration of Transformation

\[ \Phi(X) = \{ x^2, y^2, \sqrt{2}xy \} \]
Support Vector Machines for Pattern Classification
Concept of Support Vector Machine

\[ X \xrightarrow{\Phi(X)} Z \xrightarrow{W^T Z} \hat{y} \]

- **Cover's theorem:** A complex pattern classification problem cast in a high-dimensional space non-linearly is more likely to be linearly separable than in a low-dimensional space.

- Mapping from the input pattern space \( X \) to a high-dimensional feature space \( Z \) using a non-linear function \( \Phi(X) \).

- Constructing an *optimal hyperplane* as the decision surface to separate the examples of two classes in the *feature space*.
Optimal Separating Hyperplane
Maximum Margin Hyperplane
Learning problem: To find the optimal hyperplane

Margin: Distance of the nearest example to a hyperplane

Optimal hyperplane is the one for which the margin is maximum

Principle of structural risk minimization, i.e., minimization of the complexity of the model, can be used to find the optimal hyperplane
Optimal hyperplane specified by \((w_o, b_o)\) must satisfy the constraint:

\[
y_i(w_o^T z_i + b_o) \geq 1 \quad \text{for} \quad i = 1, 2, \ldots, N
\]  

Here \(z_i\) is the feature vector for \(x_i\), and \(y_i\) is the corresponding desired output. 

\(N\) is the number of training examples.
Linearly Separable Patterns

The training examples for which the constraint is satisfied with the equality sign are called support vectors.

Optimum value of margin of separation: $\frac{2}{\|w_o\|}$

Maximizing the margin of separation is equivalent to minimizing the Euclidean norm of the weight vector $w$.

Learning problem: Given the training examples $\{(x_i, y_i)\}$, $i = 1, 2, ..., N$, find the values of $w$ and $b$ such that they satisfy the constraints

$$y_i(w^Tz_i + b) \geq 1 \quad for \quad i = 1, 2, ..., N$$

and the weight vector $w$ minimizes the cost function:

$$\Psi(w) = \frac{1}{2}w^Tw$$
**Linearly Separable Patterns**

The Lagrangian function is:

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^{N} \alpha_i [y_i (\mathbf{w}^T \mathbf{z}_i + b) - 1]$$

(4)

Conditions of optimality:

$$\frac{\delta J(\mathbf{w}, b, \alpha)}{\delta \mathbf{w}} = 0$$

(5)

$$\frac{\delta J(\mathbf{w}, b, \alpha)}{\delta b} = 0$$

(6)

Application of optimality conditions gives:

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{z}_i$$

(7)

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

(8)
Linearly Separable Patterns

Find the Lagrange multipliers \( \{\alpha_i\} \) that maximize the objective function

\[
Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j z_i^T z_j
\]  

(9)

subject to the constraints

(1) \( \sum_{i=1}^{N} \alpha_i y_i = 0 \)

(2) \( \alpha_i \geq 0 \) for \( i = 1, 2, ..., N \)

For support vectors only, the optimum Lagrange multipliers will take non-zero values.

For optimum Lagrange multipliers \( \alpha_{o,i} \), the optimum weight vector \( w_o \) is given by

\[
w_o = \sum_{i=1}^{N_s} \alpha_{o,i} y_i z_i
\]  

(10)
Linearly Nonseparable Patterns

- Some data points may fall inside the region of separation or on the wrong side of separation.
- $\beta_i$ is a measure of the deviation for $z_i$ from the ideal condition of pattern separability.
- Constraints to be satisfied:
  1. $y_i(w^Tz_i + b) \geq 1 - \beta_i$ and 2. $\beta_i \geq 0$ for all $i$
Cost function:

\[ \Psi(w) = \frac{1}{2}w^T w + C \sum_{i=1}^{N} \beta_i \]  \hspace{1cm} (11)

where \( C \) is a user-specified positive parameter.

Find the \( \{\alpha_i\} \) that maximize the objective function

\[ Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j z_i^T z_j \]  \hspace{1cm} (12)

subject to constraints: (1) \( \sum_{i=1}^{N} \alpha_i y_i = 0 \) and (2) \( 0 \leq \alpha_i \leq C \)

The optimum weight vector \( w_o \) is given by

\[ w_o = \sum_{i=1}^{N_s} \alpha_{0,i} y_i z_i \]  \hspace{1cm} (13)

where \( N_s \) is the number of support vectors.
Linearly Nonseparable Patterns

The optimal hyperplane is defined in terms of support vectors:

$$\mathbf{w}_o^T \mathbf{z} + b_o = \sum_{i=1}^{N_s} \alpha_{o,i} y_i \mathbf{z}^T \mathbf{z}_i + b_o$$  \hspace{1cm} (14)

The inner-product kernel $K(\mathbf{x}, \mathbf{x}_i)$ is defined as:

$$K(\mathbf{x}, \mathbf{x}_i) = \mathbf{z}^T \mathbf{z}_i$$  \hspace{1cm} (15)

Discriminant function of the optimal decision surface:

$$\sum_{i=1}^{N_s} \alpha_{o,i} y_i K(\mathbf{x}, \mathbf{x}_i) + b_o$$  \hspace{1cm} (16)

Examples of kernel functions:

- Polynomial kernel: $K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i + c)^d$
- Sigmoidal kernel: $K(\mathbf{x}, \mathbf{x}_i) = \tanh(\mathbf{a}^T \mathbf{x}_i + c)$
- Gaussian kernel: $K(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}_i||^2}{\sigma^2}\right)$
Inner-product Kernels

\[ K(x, y) = \Phi^T(x) \cdot \Phi(y) \]

Polynomial kernel: \[ K(x, y) = (x^T y + 1)^2 \]

For 2-dimensional patterns, \( x = [x_1, x_2]^T \) and \( y = [y_1, y_2]^T \),

\[
\Phi(x) = [1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2]^T
\]

\[
\Phi(y) = [1, \sqrt{2}y_1, \sqrt{2}y_2, y_1^2, y_2^2, \sqrt{2}y_1y_2]^T
\]

\[
(x^T y + 1)^2 = 1 + 2x_1y_1 + 2x_2y_2 + x_1^2y_1^2 + x_2^2y_2^2 + 2x_1x_2y_1y_2
\]

The dimension of the feature space vector \( \Phi(x) \) for the
polynomial kernel of degree \( d \) and for the pattern dimension of \( m \)
is given by \( \frac{(m+d)!}{m!d!} \)

For sigmoidal kernel and Gaussian kernel, the dimension of
feature space vectors is shown to be infinite

Finding a suitable kernel for a given task is an open research
problem
Kernel Functions

- Mercer kernels: Kernel functions that satisfy Mercer’s theorem
- Kernel gram matrix: The matrix containing the values of the kernel function on all pairs of data points in the training data set
- Kernel gram matrix should be positive semi-definite, i.e., the eigenvalues should be non-negative, for convergence of the iterative method used for solving the constrained optimization problem
- Kernels for vectorial data:
  - Linear kernel
  - Polynomial kernel
  - Gaussian kernel
Kernel Functions

- **Kernels for non-vectorial data:**
  - Kernels on graphs
  - Kernels on sets
  - Kernels for texts
    - Vector space kernels
    - Semantic kernels
    - Latent semantic kernels
  - Kernels on strings
    - String kernels
    - Trie-based kernels
  - Kernels on trees
  - Kernels from generative models
    - Fisher kernels
  - Kernels on images
    - Hausdorff kernels
    - Histogram intersection kernels
- **Kernels learnt from data**
- **Kernels optimized for data**
Architecture of a Support Vector Machine

\[ D(X) = \sum_{i=1}^{N_s} \alpha_i y_i K(X, X_i) + b_0 \]

\[ N_s: \text{Number of support vectors} \]
Multi-class Pattern Recognition

Multi-class pattern recognition for $M$ classes is solved using a combination of several binary classifiers and a decision strategy.
Approaches to Multi-class Pattern Recognition

One-against-the-rest

One-against-one

One-against-the-rest approach: An SVM is built for each class to form a boundary between the region of the class and the regions of all other classes. Number of SVMs is $M$.

One-against-one approach: An SVM is built for every pair of classes to form a boundary between their regions. Number of pairwise SVMs is $M(M - 1)/2$. 
Support Vector Regression
Support Vector Regression

Non-linear regression in the input space

Linear Regression in the kernel feature space

Data point, $P$

Support vectors
Support Vector Machines for Nonlinear Regression

Consider a nonlinear regressive model in which the dependence of a scalar \( d \) on a vector \( x \) is described by

\[
d = f(x) + \gamma
\]  

(1)

Given a set of training data \( \{(x_i, d_i)\}_{i=1}^{N} \), where \( x_i \) is a sample value of the input vector \( x \) and \( d_i \) is the corresponding value of the model output \( d \). The problem is to provide an estimate of the dependence of \( d \) on \( x \).
Support Vector Machines for Nonlinear Regression

The constrained optimization problem is formulated by introducing two sets of nonnegative slack variables \( \{ \xi_i \}_{i=1}^{N} \) and \( \{ \xi'_i \}_{i=1}^{N} \) that are defined as follows:

\[
d_i - w^T \varphi(x_i) \leq \epsilon + \xi_i, \: i = 1, 2, ..., N
\]  \hspace{1cm} (5)

\[
w^T \varphi(x_i) - d_i \leq \epsilon + \xi'_i, \: i = 1, 2, ..., N
\]  \hspace{1cm} (6)

\[
\xi_i, \: i = 1, 2, ..., N
\]  \hspace{1cm} (7)

\[
\xi'_i, \: i = 1, 2, ..., N
\]  \hspace{1cm} (8)
Support Vector Machines for Nonlinear Regression

This constrained optimization problem may therefore be viewed as equivalent to that of minimizing the cost functional

$$\Phi(w, \xi, \xi') = C\left(\sum_{i=1}^{N}(\xi + \xi'_i)\right) + \frac{1}{2}w^Tw$$  \hspace{1cm} (9)

subject to the constraints given above.

Lagrangian function:

$$J(w, \xi, \xi', \alpha, \alpha', \gamma, \gamma') = C\sum_{i=1}^{N}(\xi + \xi'_i) + \frac{1}{2}w^Tw$$  \hspace{1cm} (10)

$$- \sum_{i=1}^{N}\alpha_i[w^T\varphi(x_i) - d_i + \epsilon + \xi_i]$$

$$- \sum_{i=1}^{N}\alpha'_i[d_i - w^T\varphi(x_i) + \epsilon + \xi_i] - \sum_{i=1}^{N}(\gamma_i\xi_i + \gamma'_i\xi'_i)$$

where the $\alpha_i$ and the $\alpha'_i$ are the Lagrange multipliers.
Support Vector Machines for Nonlinear Regression

The requirement is to minimize $J(w, \xi, \xi', \alpha, \alpha', \gamma, \gamma')$ with respect to the weight vector $w$ and slack variables $\xi$ and $\xi'$; it must also be maximized with respect to $\alpha$ and the $\alpha'$ and also with respect to $\gamma$ and $\gamma'$. By carrying out this optimization we have in respective ways:

$$w = \sum_{i=1}^{N} (\alpha_i - \alpha_i') \varphi(x_i)$$  \hspace{1cm} (11)

$$\gamma_i = C - \alpha_i$$  \hspace{1cm} (12)

and

$$\gamma_i' = C - \alpha_i'$$  \hspace{1cm} (13)
Support Vector Machines for Nonlinear Regression

Dual problem for nonlinear regression:
Given the training sample \( \{(x_i, d_i)\}_{i=1}^{N} \) find the Lagrange multipliers \( \{\alpha_i\}_{i=1}^{N} \) and \( \{\alpha_i^i\}_{i=1}^{N} \) that maximize the objective function

\[
Q(\alpha_i, \alpha_i^i) = \sum_{i=1}^{N} d_i (\alpha_i - \alpha_i^i) - \epsilon \sum_{i=1}^{N} (\alpha_i + \alpha_i^i)
\]

\[
-\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i - \alpha_i^i)(\alpha_j - \alpha_j^i)K(x_i, x_j)
\]

subject to the following constraints:
(1) \( \sum_{i=1}^{N} (\alpha_i - \alpha_i^i) = 0 \)
(2) \( 0 \leq \alpha_i \leq C, i = 1, 2, ..., N \) \( 0 \leq \alpha_i^i \leq C, i = 1, 2, ..., N \)

where \( C \) is a user-specified constant.
Support Vector Machines for Nonlinear Regression

An estimate of \( d \), denoted by \( y \), is expanded in terms of a set of nonlinear basis functions \( \{ \varphi_j(x) \}_{j=0}^{m_1} \) as follows:

\[
y = \sum_{j=0}^{m_1} w_j \varphi_j(x) = w^T \varphi(x)
\] (2)

where

\[
\varphi(x) = [\varphi_0(x), \varphi_1(x), ..., \varphi_{m_1}(x)]^T
\] (3)

and

\[
w = [w_0, w_1, ..., w_{m_1}]^T
\] (4)
Kernel Based Clustering
K-Means Clustering Method

- **Task:** Given N data points, form K clusters.
- **Algorithm:**
  1. Initialize K randomly chosen data points as the initial centers of the clusters.
  2. For each of the N data points,
     - compute its distance to each of the K centers
     - determine the nearest center
     - include the data point in the cluster of that center
  3. Compute the centroid of each cluster using the data points currently assigned to that cluster. The recomputed centroids are the new centers of the clusters.
  4. Repeat Steps 2 and 3 until there are no changes in assigning the clusters to the data points.
- **Criterion for clustering:** Minimize the trace of the scatter matrix
- **Linear separation of clusters**
Kernel based Clustering

- Within cluster scatter matrix in the input space:
  \[
  S_w = \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N} z_{kn} (x_n - m_k)(x_n - m_k)^T
  \]
  where \( z_{kn} = 1 \) if \( x_n \in C_k \)
  \( = 0 \) otherwise

- Within cluster scatter matrix in the feature space:
  \[
  S_w^\phi = \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N} z_{kn} (\Phi(x_n) - m_k^\phi)(\Phi(x_n) - m_k^\phi)^T
  \]
  \[
  m_k^\phi = \frac{1}{N_k} \sum_{n=1}^{N_k} \Phi(x)
  \]

- Trace of the scatter matrix:
  \[
  \text{Tr}(S_w^\phi) = \text{Tr}\left( \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N} z_{kn} (\Phi(x_n) - m_k^\phi)(\Phi(x_n) - m_k^\phi)^T \right)
  \]

- Optimization problem:
  \[
  Z = \arg \min_z \text{Tr} \left( S_w^\phi \right)
  \]
  \[
  Z = K
  \]
  \[
  \begin{bmatrix}
    0 & 0 & 1 & 1 & 0 & 1 & \ldots & 0 \\
    1 & 0 & 0 & 0 & 1 & 0 & \ldots & 1 \\
    \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \\
    0 & 1 & 0 & 0 & 0 & 0 & \ldots & 0
  \end{bmatrix}
  \]
Interlocking Clusters

Data to be clustered

Polynomial Kernel of degree 2

K-means clustering

Polynomial Kernel of degree 3

Gaussian Kernel with $\sigma = 1$

Gaussian Kernel with $\sigma = 0.18$
Clustering and Vector Quantization

Representation of a sequence of feature vectors extracted from a speech segment by a sequence of discrete symbols

Sequence of feature Vectors \( F_1 \ F_2 \ F_3 \ \text{………………….} \ F_T \)

Sequence of codebook indices (symbols) obtained by clustering and VQ in the input space \( S_1 \ S_2 \ S_3 \ \text{………………….} \ S_T \)

Sequence of codebook indices (symbols) obtained by clustering and VQ in the kernel feature space \( S_1^\phi \ S_2^\phi \ S_3^\phi \ \text{………………….} \ S_T^\phi \)

- Sequence of symbols can be considered as a string
- String kernel can be used to match the sequences of symbols corresponding to two speech segments
Applications of Kernel Methods
Applications of Kernel Methods

- Genomics and computational biology
- Knowledge discovery in clinical microarray data analysis
- Recognition of white blood cells of Leukaemia
- Classification of interleaved human brain tasks in fMRI
- Image classification and retrieval
- Hyperspectral image classification
- Perceptual representations for image coding
- Complex SVM approach to OFDM coherent demodulation
- Smart antenna array processing
- Speaker verification
- Kernel CCA for learning the semantics of text

Work on Kernel Methods at IIT Madras

Focus:
Design of suitable kernels and development of kernel methods for speech, image and video processing tasks

Tasks:
- Speech recognition (Hidden Markov models in kernel feature space)
- Speech segmentation
- Speaker segmentation
- Voice activity detection
- Handwritten character recognition
- Face detection
- Image classification
- Video shot boundary detection
- Content-based information retrieval for multimedia data
- Cryptosystem identification and decryption
- Network intrusion detection
- Time series data processing
On-line Handwritten Character Recognition for Indian Languages
Representation of a Character

• **Stroke**
  – Basic unit of on-line handwritten character recognition
  – Sequence of points captured between pen-down and pen-up

• **Character is a combination of strokes**

• **Rule list is used to identify the character from the recognized sequence of strokes**
Stroke Recognition using SVMs

- **Implementation Details**
  - Representation of a stroke: **120-dimensional vector** consisting of the x-y coordinates of 60 points
  - Gaussian kernel based SVM
  - Datasets:
    - Training data: Stroke data from 90 writers
    - Test data: Stroke data from 10 writers

- **Stroke Classification Performance**
  - Devanagari:
    - Number of strokes: **115**
    - Classification Accuracy: **96.83%**
  - Tamil
    - Number of strokes: **93**
    - Classification Accuracy: **91.07%**
  - Telugu:
    - Number of strokes: **253**
    - Classification Accuracy: **83.08%**
Image Classification
Kernel Functions for Image Classification

- Low-level descriptions of an image:
  - Pixels-based representations: Exploit spatial correlation
  - Histogram-based representations: Use information about relative frequencies

- Kernels for pixels-based representations:
  - Hausdorff distance based kernel
    \[
    \text{Hausdorff distance: } h_A(B) = \sum_{i=1}^{N} U(\varepsilon - |A[i] - B[s(i)]|) 
    \]

- Kernels for histogram-based representations:
  - Generalized Gaussian kernel:
    \[
    K(A,B) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{N}|A_i^a - B_i^a|^b\right)
    \]
  - Histogram intersection kernel:
    \[
    K(A,B) = \sum_{i=1}^{N} \min(A_i, B_i)
    \]
Wildlife (5-class) Image Data

AfricanSpecialityAnimals

AlaskanWildlife

ArabianHorses

NorthAmericanWildLife

WildlifeGalapagos
Animals and Birds (8-class) Image Data

ArabianHorses

BarnyardAnimals

Bears

FoxesCoyotes
Animals and Birds (8-class) Image Data

NorthAmericanDeer

Tigers

WildlifeGalapagos

NestingBirds
Miscellaneous (10-class) Image Data

- AdventureSailing
- AirShows
- AutoRacing
- Beaches
- LandPyramids
Miscellaneous (10-class) Image Data

- Residential Interiors
- Mountains
- Military Vehicles
- Sacred Places
- Sunsets & Sunrises
# Image Classification Performance

(Classification Accuracy in %)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Kernel Function</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Standard Gaussian</td>
<td>Generalised</td>
<td>Hausdorff</td>
<td>Histogram</td>
</tr>
<tr>
<td></td>
<td>(Gaussian)</td>
<td>Gaussian</td>
<td></td>
<td>Intersection</td>
</tr>
<tr>
<td>Wildlife (5-class)</td>
<td>78.0</td>
<td>90.0</td>
<td>78.0</td>
<td>80.0</td>
</tr>
<tr>
<td>Animals and Birds (8-class)</td>
<td>74.4</td>
<td>82.5</td>
<td>70.6</td>
<td>83.1</td>
</tr>
<tr>
<td>Miscellaneous (10-class)</td>
<td>73.5</td>
<td>85.0</td>
<td>----</td>
<td>83.5</td>
</tr>
</tbody>
</table>
Speaker Change Detection
Speaker Segmentation

Multiple speakers

Multispeaker conversation

• Multispeaker speech
  – Casual conversations, meetings, news broadcasts
• Detection of speaker change points
Fixed Duration Window based Patterns

**Extraction of Positive Examples**

\[ scp_{k-1} \quad scp_k \quad scp_{k+1} \]

**Time**

**Extraction of Negative Examples**

\[ scp_{k-1} \quad scp_k \quad scp_{k+1} \]

**Time**

\[ scp: \text{ speaker change point} \]
Performance of Speaker Change Detection System

- Number of actual speaker change points in test dataset: 282
- Number of frames in the test dataset: about 16000
- Number of speaker change points missed (not detected): $M$
- Number of false alarms: $FA$

<table>
<thead>
<tr>
<th>Window length (in msec)</th>
<th>After speaker change hypothesization</th>
<th>After smoothing</th>
<th>After false alarm reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$M$</td>
<td>$FA$</td>
<td>$M$</td>
</tr>
<tr>
<td>100</td>
<td>14</td>
<td>2488</td>
<td>26</td>
</tr>
<tr>
<td>200</td>
<td>27</td>
<td>2269</td>
<td>37</td>
</tr>
<tr>
<td>300</td>
<td>30</td>
<td>3277</td>
<td>39</td>
</tr>
<tr>
<td>400</td>
<td>9</td>
<td>5276</td>
<td>24</td>
</tr>
</tbody>
</table>
Recognition of Consonant-Vowel (CV) Units of Speech
Speech Recognition
Recognition of Subword Units of Speech using Kernel Methods

Main issues:

• Varying durations of segments of subword units of speech
• Representation of a segment by a sequence of feature vectors
• Classification of varying length sequences of feature vectors
• Mapping the varying length segment to a fixed length pattern
  • Split and average Method
  • Deletion and replication of least varying frames
  • Anchor point based method
  • Linear compaction and elongation method
• Mapping the sequence of feature vectors to a sequence of discrete symbols
  • Clustering and vector quantization
• Classification of varying length sequences of discrete symbols
  • String kernel based SVMs
  • Discrete HMMs in kernel feature space
Extraction of Fixed Length Patterns by Linear Compaction and Elongation Method

(a) Compaction

(b) Elongation
### Performance in Classification of CV Segments

<table>
<thead>
<tr>
<th>Language</th>
<th>No. of Classes</th>
<th>No. of CV segments</th>
<th>Model</th>
<th>$k$—best accuracy (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$k=1$</td>
</tr>
<tr>
<td>Tamil</td>
<td>123</td>
<td>10,293</td>
<td>HMM</td>
<td>50.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SVM</td>
<td>50.18</td>
</tr>
<tr>
<td>Telugu</td>
<td>138</td>
<td>11,347</td>
<td>HMM</td>
<td>46.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SVM</td>
<td>50.61</td>
</tr>
<tr>
<td>Hindi</td>
<td>103</td>
<td>4,137</td>
<td>HMM</td>
<td>40.09</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SVM</td>
<td>41.04</td>
</tr>
<tr>
<td>Multi-Lingual</td>
<td>196</td>
<td>25,777</td>
<td>HMM</td>
<td>41.32</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SVM</td>
<td>45.31</td>
</tr>
</tbody>
</table>
Recognition of CV Segments Represented using Sequences of Codebook Indices

<table>
<thead>
<tr>
<th>Method for vector quantization</th>
<th>Classification model</th>
<th>Classification accuracy (in %) for CVs in Indian Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Tamil</td>
</tr>
<tr>
<td>VQ in the input space</td>
<td>DHMMs</td>
<td>50.55</td>
</tr>
<tr>
<td></td>
<td>SVMs using polynomial kernel</td>
<td>53.05</td>
</tr>
<tr>
<td></td>
<td>SVMs using Gaussian kernel</td>
<td>56.43</td>
</tr>
<tr>
<td></td>
<td>SVMs using string kernel</td>
<td><strong>63.46</strong></td>
</tr>
<tr>
<td>VQ in the polynomial kernel feature space</td>
<td>DHMMs</td>
<td>52.73</td>
</tr>
<tr>
<td></td>
<td>SVMs using polynomial kernel</td>
<td>58.15</td>
</tr>
<tr>
<td></td>
<td>SVMs using Gaussian kernel</td>
<td>59.92</td>
</tr>
<tr>
<td></td>
<td>SVMs using string kernel</td>
<td><strong>74.82</strong></td>
</tr>
<tr>
<td>VQ in the Gaussian kernel feature space</td>
<td>DHMMs</td>
<td>54.28</td>
</tr>
<tr>
<td></td>
<td>SVMs using polynomial kernel</td>
<td>61.67</td>
</tr>
<tr>
<td></td>
<td>SVMs using Gaussian kernel</td>
<td>62.84</td>
</tr>
<tr>
<td></td>
<td>SVMs using string kernel</td>
<td><strong>79.65</strong></td>
</tr>
</tbody>
</table>
Cryptanalysis

Work done as part of Projects sponsored by SAG, DRDO
Paradigms for Cryptanalysis Tasks

- **Classification paradigm:**
  - Identification of the encryption method

- **Hetero-association paradigm:**
  - Decryption without the knowledge of key

- **Clustering paradigm:**
  - Reduction of the key space
Approach to Cryptanalysis

Identification of Encryption Method

Switch

Hetero-association Model 1

Hetero-association Model 2

Hetero-association Model m

Decryption without the knowledge of the key

Cipher Text

Plain Text
Approach to Cryptanalysis (contd.)

Cipher Text → Identification of Encryption Method → Key Space Reduction → Decryption → Plain Text
Identification of Encryption Method
Identification of Encryption Method using Pattern Classification Paradigm

- No visible patterns in the cipher texts
- Classification models should capture the implicit patterns for different encryption methods
Representation of a Cipher Text

- Bag-of-words based representation
- Raw data representation
- $n$-gram analysis based representation
- Block-level similarity based representation
- Pair-wise distance based representation
Document Vector Representation using Bag-of-Words Approach

• Let N be the size of the dictionary built using a corpus of documents.
• A document $d$ is represented by an $N$-dimensional vector, $\Phi(d)$, given as follows:

$$\Phi(d) = [tf(t_1,d), tf(t_2,d), \ldots, tf(t_N,d)]$$

$tf(t_i,d)$ : Frequency of occurrence for term $t_i$ in document $d$
Document Vector Representation of a Cipher Text

• A cipher text is considered as a document
• The bit sequence of a cipher text is divided into $k$-bit subsequences
• A symbol is assigned to each unique subsequence, leading to an alphabet of $2^k$ symbols
• Fixed length word representation:
  – **Term**: A sequence of fixed number ($l$) of symbols
Illustration of Representation of a Cipher Text using Bag-of-Words approach

- Cipher Text: _û_j \ È't
- 4-bit subsequences and symbols for cipher text ($k=4$):

```
0 0 0 1 0 1 1 1 1 1 0 1 1 0 1 1 1 0 1 1 1 0 0 0 0 1 0 1 1 1 1 0 0
b  h  n  l  l  i  f  m
```

```
1 0 0 0 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 0 1 1 1 0 1 1 1 0 1 0 0
i  p  m  i  c  h  h  e
```

- Fixed length word representation:
  - No delimiters used
  - Term: Sequence of a fixed number ($l = 4$) of symbols
  - Terms in the cipher text
    - bhnl, lifm, ipmi, chhe
Generation of Document Vector: Class Specific Dictionary Method

- A separate corpus consisting of cipher texts obtained using a particular encryption method is used to build the dictionary for that class.
- Class specific dictionaries are used to obtain different document vectors for a given cipher text.
Raw Data Representation of a Cipher Text

- A cipher text is represented as a vector of integer values
- The bit-sequence of a cipher text is divided into $k$-bit subsequences
- Each $k$-bit subsequence is considered as an unsigned integer where $k$ can be 8, 16, 24 or 32
- Dimension of the vector is fixed (500, 250, 166, 125) for a given size of cipher text (500 characters)
- Encryption method identification system:
Representation of a Cipher Text using n-gram Analysis

- A group of $k$ bits is represented by a character
- The size of alphabet is $2^k$
- An $n$-gram is a group of $n$ consecutive characters
- Number of possible $n$-grams is $2^{kn}$
- A cipher text is represented by the frequency of occurrence of different $n$-grams

- 4-bit subsequences and symbols for cipher text ($k=4$):

```
0 0 0 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 0 0 0 0 1 0 1 1 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 1 1 1 0 0 0 0 1 0 1 1 0 1 1 0 1 1 0 0
b h n l l i f m
```

```
1 0 0 0 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 0
i p m i c h h h e
```

- Symbol sequence in the cipher text: `bhnllifmipmichhe`
- $n$-grams in this cipher text for $n=4$ are:
  `bhnl, hnl1, nlli, lifm, ifmi, fmip, mipm, ipmi, pmic, mich, ichh, chhe`
Representation of a Cipher Text by a Block-level Similarity based Feature Vector

- Let the training data set for an encryption method consist of $N$ cipher texts, $CT_1, CT_2, \ldots, CT_N$.

- A cipher text, $CT_i$, is represented by an $N$-dimensional feature vector, $Z_i$. The $j$th element of $Z_i$ is the total block-level similarity between $CT_i$ and $CT_j$, $S_{ij}$, defined as follows:

$$S_{ij} = \sum_{k=1}^{M} s(B_{ik}, B_{jk})$$

where $M$ is the number of blocks in a cipher text, and $s(B_{ik}, B_{jk})$ is the similarity of blocks.

- Measure of similarity between blocks
  - Hamming distance
  - String kernel
String Kernel

- **String**: A finite sequence of characters
- **Subsequence**: Ordered sequence of noncontiguous characters in a string
- **Length of a subsequence**, \( u \), given by \( L(u) \) is defined in terms of the position of the first character \( (i_1) \) and the position of the final character of the subsequence \( (i_{|u|}) \) in the string:
  \[
  L(u) = i_{|u|} - i_1 + 1
  \]
- **Example**:
  - String: identification
  - Substrings of 3 characters: ide, den, ent, nti, …
  - Subsequences of 3 characters, and their lengths:
    - Subsequence dnf: Length = 7-2+1 = 6
    - Subsequence dtc: Length = 9-2+1 = 8
- **String kernel** computes a measure of similarity between two strings based on subsequences common between them and the lengths of the common subsequences.
Document Distance based Representation

- Feature vector: Euclidean distance between the bag-of-words based document vectors for every pair of documents.

- Document vector:

\[ \phi(D) = [tf(t_1, D), tf(t_2, D), \ldots, tf(t_N, D)]^T \]

where,

- \( \phi(D) \) is the document vector
- \( tf(t_i, D) \) is the frequency of occurrence of \( t_i \) in a given document \( D \)
- \( N \) is the size of the dictionary
- \( t_i \) is the \( i^{th} \) word or term in the dictionary
Decryption of Cipher Texts
Decryption using Hetero-Association Paradigm

- The aim of this paradigm is to associate a block of cipher text with the corresponding block of plain text.

- A cipher text block and the corresponding plain text block are considered as an input-output pattern vector pair.

- Hetero-association is considered as multiple nonlinear function approximation tasks
  - Multilayer feedforward neural network (MLFFNN)
  - Support vector regression
Hetero-Association using Support Vector Regression
Studies on Decryption using Support Vector Regression

- Each cipher text and plain text blocks are $k$ bit length
- **Encryption methods**: DES, 3DES, Blowfish, AES, and RC5
  - Mode of operation: CBC
- **A plain text block is predicted from its corresponding cipher text block**
- Number of plain texts for training: 100
- Number of plain texts for testing: 50
- Each plain text has about 500 characters
- Single key is used for encryption of the plain texts in the training data
- Number of blocks in training data: 6200
- Number of blocks in test data: 3100
**Performance of Support Vector Regression**

- **Bit-level error rate** (in %) in decryption of cipher texts generated using different encryption methods

<table>
<thead>
<tr>
<th>Plain text domain for test data</th>
<th>Keys for Test data</th>
<th>Encryption Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>DES (CBC)</td>
</tr>
<tr>
<td>Same as the domain for training data</td>
<td>Same as the key for training data</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Different from the key for training data</td>
<td>31.12</td>
</tr>
<tr>
<td>Different from the domain for training data</td>
<td>Same as the keys for training data</td>
<td>31.26</td>
</tr>
<tr>
<td></td>
<td>Different from the keys for training data</td>
<td>33.53</td>
</tr>
</tbody>
</table>
**Performance of Support Vector Regression**

- **Character-level error rate** (in %) in decryption of cipher texts generated using different encryption methods

<table>
<thead>
<tr>
<th>Plain text domain for test data</th>
<th>Keys for Test data</th>
<th>Encryption Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>DES (CBC)</td>
</tr>
<tr>
<td>Same as the domain for training data</td>
<td>Same as the key for training data</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Different from the key for training data</td>
<td>95.82</td>
</tr>
<tr>
<td>Different from the domain for training data</td>
<td>Same as the keys for training data</td>
<td>96.03</td>
</tr>
<tr>
<td></td>
<td>Different from the keys for training data</td>
<td>96.55</td>
</tr>
</tbody>
</table>
Performance of Support Vector Regression

- Probability of bit mismatch for different bit positions of the decrypted characters

<table>
<thead>
<tr>
<th>Encryption Method</th>
<th>Bit position</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>DES (CBC)</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>0.30</td>
</tr>
<tr>
<td>3DES</td>
<td>0.00</td>
</tr>
<tr>
<td>Blowfish</td>
<td>0.00</td>
</tr>
<tr>
<td>AES</td>
<td>0.00</td>
</tr>
<tr>
<td>RC5</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Identification of Encryption Method using Decrypted Text Data

- A support vector regression based model is trained for each encryption method to predict a block of plain text from a block of cipher text.
- Output of the regression model is the decrypted text.
- For cipher texts generated for different encryption methods using the same key and the same plain text, the decrypted texts are not the same.
- Training the classification models:
  - Decrypted texts are represented by feature vectors using the methods for representation of cipher texts.
  - Feature vectors derived from the decrypted texts are used as training data for classification models.
- Classification of a test cipher text:
  - Obtain the decrypted texts using the regression models for different encryption methods
  - Give feature vectors for the decrypted texts as input to the classification models
Identification of Encryption Method using Decrypted Text Data

Cipher text → Decrypted text

Hetero-Association Model for DES → Representation → SVM for DES
Hetero-Association Model for 3DES → Representation → SVM for 3DES
Hetero-Association Model for Blowfish → Representation → SVM for Blowfish
Hetero-Association Model for AES → Representation → SVM for AES
Hetero-Association Model for RC5 → Representation → SVM for RC5

Feature vector → Decision Logic → Class
Studie on Identification of Encryption Method

- Training data: Cipher texts of 100 plain texts
- Test data set: 40 plain texts from the same domain
- Plain text size: 512 bytes
- Encryption methods: DES, 3DE, Blowfish, AES, RC5
- Representation of a cipher text or a partially decrypted text:
  - Bag-of-words representation
  - Raw data representation
  - $n$-gram analysis based representation
  - Block-level similarity based representation
  - Pair-wise distance based representation
Comparison of the performance of identification of encryption method for the input cipher text using (1) Raw data representation (2) N-gram bit level (3) N-gram symbol level (4) Bag-of-words approach (5) Document distance based feature vector and (6) Block level similarity based feature vector.
Comparison of the performance of identification of encryption method for the input partially decrypted text using (1) Raw data representation (2) N-gram bit level (3) N-gram symbol level (4) Bag-of-words approach (5) Document distance based feature vector and (6) Block level similarity based feature vector.
Key Space Reduction using Clustering
Key Space Reduction using Clustering Methods

Cipher texts generated using different keys → Clustering method → Cluster 1, Cluster 2, ..., Cluster N

Test cipher text → Cluster Identification

Cluster label k → Decrypt test cipher texts with keys in $c_k$ → Plain text
Studies on Key Space Reduction

- **Encryption method**: DES(ECB)
- **Types of keys**: Random and Pattern
- **Number of Keys**: 3000 and 4096
- **Representation of ciphertexts**:
  - Raw data representation
  - N-gram analysis based representation
- **Clustering methods**:
  - K-means clustering
  - Agglomerative clustering
  - Kernel based clustering
- **Cluster validity index**: CS validity index
- **Keyspace reduction achieved (Number of clusters formed)**:
  - Random keys: 3000 to 75
  - Pattern keys: 4096 to (100-200)
- **Study of relation among the keys in each cluster has been carried out to evaluate the effectiveness of clustering**
The key distribution of the random keys in a few clusters obtained using the \textit{K}-means clustering algorithm.
The key distribution of the **pattern keys** in a few clusters obtained using the *K*-means clustering algorithm

**Studies on Key Space Reduction**
The key distribution of the random keys in a few clusters obtained using the agglomerative clustering algorithm.
The key distribution of the pattern keys in a few clusters obtained using the agglomerative clustering algorithm.
Learning Kernels from Data

Dileep A.D.
Ph.D. Scholar
Choice of Kernel Function

- The choice of the **suitable kernel** for a given task and for a given type of data is crucial for the kernel methods to perform well.

- The commonly used kernels need not give the best performance.

- It is desirable to either **learn** the kernel functions from the data or **optimize** the kernel functions for a given task and for a given type of data.

- Kernel optimization is considered as
  - A multiple kernel learning task
  - Optimization of kernel by maximizing the measure of separability in the kernel feature space.
Work on Learning Kernels from Data


Multiple Kernel Learning (MKL)

- **Issues**
  - Construction of an optimized kernel function from multiple base kernels
  - Choice of base kernel
  - Suitable technique for optimization

- Optimized kernel can be obtained as a convex combination of finitely many base kernels
  - Each base kernel is a Mercer kernel

- Optimized kernel function is described as a weighed combination of base kernels:
  \[ K(x_i, x_j) = \sum_{l} \beta_l K_l(x_i, x_j) \]

Here \( \beta_l > 0 \) represents the weight of the \( l^{th} \) base kernel \( K_l \)
Multiple Kernel Learning in SVM Framework

- The multiple kernel learning task is considered as a problem of optimizing the kernel weights while training the SVM
- The dual form of the SVM optimization problem changes to:

\[
\max \left\{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \sum_{l=1}^{m} \beta_l K_l (x_i, x_j) \right\}
\]

\[
\begin{align*}
\sum_{i=1}^{n} y_i \alpha_i &= 0 \\
0 &\leq \alpha_i \leq C \quad i = 1, \ldots, n \\
\beta_l &> 0, \quad l = 1, \ldots, m
\end{align*}
\]

- Both the Lagrange coefficients \(\alpha_i\) and the base kernel weights \(\beta_l\) need to be optimized
Examples of Multiple Kernel Learning

- Combining data from the heterogeneous sources [1]
  - **Task**: Text categorization
  - **Base kernels**:
    - Kernel from bag-of-words representation
    - Kernel from document-concept-term graphical model

- Combining different types data[1]
  - **Task**: Protein function prediction
  - **Different types of data**:
    - Amino acid sequence, protein-protein interactions, Genetic interactions, Protein complex data, Expression data
  - A base kernel is associated with each type data

- Combining different feature sets [2, 3]
  - A base kernel is associated with each set of features

---

Optimization of Data-Dependent Kernel

• **Issues**
  – Use a measure of separability in the kernel feature space
  – Use a suitable transformation function to convert a primary kernel into an optimized kernel in a data-dependent way

• A conformal mapping is used to enlarge the spatial resolution around the separating boundary surface leading to a larger margin [4]. This conformal transformation of a kernel function is data-dependent.

• Conformal transformation of a primary kernel function $K_0(x_i, x_j)$ by a positive data-dependent factor $q(x)$ is:

$$K(x_i, x_j) = q(x_i)q(x_j)K_0(x_i, x_j)$$

The primary kernel function $K_0(x_i, x_j)$ is any Mercer kernel

---

**Data-Dependent Kernel SVM**

- One choice for the factor $q(x)$ to be constructed in a data-dependent way is:

$$q(x_i) = \sum_{k=1}^{n_{sv}} \alpha_k \exp(-\gamma \| x_i - x_k \|^2)$$

where $n_{sv}$ is the number of support vectors

$\alpha_k$ are the non-zero Lagrange coefficients

$\gamma$ is the width parameter

- Training the data-dependent kernel SVM consists of the following steps:
  - Train an SVM with a primary kernel $K_0$ to determine the support vectors
  - Use the conformal transformation to obtain the data-dependent kernel $K$
  - Train another SVM with the data-dependent kernel $K$
Optimizing the Data-Dependent Kernel Function

**Objective:** To choose the data-dependent factor that maximizes the class separability measure

- The factor function $q(x)$ is constructed as follows:

  $$q(x) = \beta_0 + \sum_{k=1}^{m} \beta_k K_1(x, a_k)$$

  Here
  - The base kernel function $k_1(x, a_k) = \exp(-\gamma \| x - a_k \|^2)$
  - $\beta_k$’s are the combination coefficients to be optimized
  - The set $\{a_k\}_{k=1}^{m}$ is chosen from the training data
Optimizing the Data-Dependent Kernel Function

- Fisher discriminant criterion is used as the class separability measure

- Kernel gram matrix can be written as

\[
K = \begin{bmatrix}
K_{11} & K_{12} \\
K_{21} & K_{22}
\end{bmatrix}
\]

- Between-class scatter matrix, B in the kernel feature space:

\[
B = \begin{bmatrix}
\frac{1}{n_1}K_{11} & 0 \\
0 & \frac{1}{n_2}K_{22}
\end{bmatrix} - \begin{bmatrix}
\frac{1}{n}K_{11} & \frac{1}{n}K_{12} \\
\frac{1}{n}K_{21} & \frac{1}{n}K_{22}
\end{bmatrix}
\]

- Within-class scatter matrix, W in the kernel feature space:

\[
W = \begin{bmatrix}
k_{11} & \cdots & 0 \\n\vdots & \ddots & \vdots \\n0 & \cdots & k_{nn}
\end{bmatrix} - \begin{bmatrix}
\frac{1}{n_1}K_{11} & 0 \\
0 & \frac{1}{n_2}K_{22}
\end{bmatrix}
\]
Optimizing the Data-Dependent Kernel Function

- Fisher discriminant ratio in the kernel feature space can be obtained as a function of $\beta$

$$J(\beta) = \frac{1^T B_1}{1^T W_1} = \frac{q(\beta)^T B_0 q(\beta)}{q(\beta)^T W_0 q(\beta)}$$

- Here $W_0$ and $B_0$ are the within-class and between-class scatter matrices in the feature space of the base kernel $K_0$
- Maximizing $J(\beta)$ means increasing the separability of training data in the feature space
- Maximize the class separability $J$ using the gradient ascent approach

$$\beta^{(t+1)} = \beta^{(t)} + \eta^{(t)} \left( \frac{\partial J(\beta)}{\partial \beta} \right)$$

where $\eta$ is the learning rate
Optimizing the Kernel in the Empirical Feature Space

- Let $\{x_i\}_{i=1}^n$ be a $d$-dimensional training data set
- Let $X$ be the $n \times d$ sample matrix
- Let $K$ be the $n \times n$ kernel gram matrix of rank $r$
- Since $K$ is a symmetric, positive-semidefinite matrix, $K$ can be decomposed as

$$K_{n \times n} = P_{n \times r} \Lambda_{r \times r} P^T_{r \times n}$$

where $\Lambda$ is a diagonal matrix containing only the $r$ positive eigenvalues of $K$ in the decreasing order

$P$ consists of the corresponding eigenvectors as its columns
Optimizing the Kernel in the Empirical Feature Space

• Empirical feature space: [5]
  - The empirical kernel map $\phi_r^e$ from the input data space to an $r$-dimensional Euclidean space is defined as:
    $$\phi_r^e(x_i) = \begin{bmatrix} k(x_i, x_1), \ldots, k(x_i, x_n) \end{bmatrix} \Lambda^{-1/2}$$
  - The embedding space of the empirical kernel map is called the empirical feature space

• Empirical feature space preserves the geometrical structure of $\phi_r^e(x_i)$ in the kernel feature space

• Let $Y$ be an $n \times r$ matrix with each $\phi_r^e(x_i)$ as a row in it, i.e., $Y = K \Lambda^{-1/2}$

• It can be shown that $YY^T = K$

Study on Optimized Kernels for Artificial Data

Data: Two dimensional artificial data

Primary Kernel: Polynomial kernel

$J$ for Polynomial kernel: 0.0728
Study on Optimized Kernels for Artificial Data

Data: Two dimensional artificial data
Primary Kernel: Gaussian kernel
J for Gaussian kernel: 0.5149
Study on Optimized Kernels for Speech Data

Data in empirical feature space for Gaussian kernel, $\sigma = 0.001$

Class separability measure as a function of the iteration number

Speech Data: vowels /a/ and /i/

Primary Kernel: Gaussian Kernel

Data in empirical feature space for optimized kernel, $\eta = 0.005$
Study on Optimized Kernels for Speech Data

Speech Data: vowels /u/ and /o/

Primary Kernel: Gaussian Kernel
Kernel Methods: Summary

• Kernel methods involve
  – Nonlinear transformation of data to a higher dimensional feature space induced by a Mercer kernel
  – Construction of optimal linear solutions in the kernel feature space

• Complexity of model is not dependent on the dimension of data

• Models can be trained with small size datasets

• Kernel methods can be used for non-vectorial type of data also

• Performance of kernel methods is dependent on the choice of the kernel. Design of a suitable kernel is a research problem.

• Approaches to the design of kernels by learning the kernels from data or by constructing the optimized kernels for a given data are being explored
Text Books


Web resource: www.kernelmachines.org

Thank You