Support Vector Machines: Kernels

CS4780/5780 – Machine Learning
Fall 2012

Thorsten Joachims
Cornell University

Reading: Schoelkopf/Smola Chapter 7.4, 7.6, 7.8
Cristianini/Shawe-Taylor 3.1, 3.2, 3.3.2, 3.4
• Transform a linear learner into a non-linear learner
• Kernels can make high-dimensional spaces tractable
• Kernels can make non-vectorial data tractable
Problem:
• some tasks have non-linear structure
• no hyperplane is sufficiently accurate
How can SVMs learn non-linear classification rules?
Extending the Hypothesis Space

Idea: add more features

Learn linear rule in feature space.

Example:

The separating hyperplane in feature space is degree two polynomial in input space.
Example

- Input Space: \( \tilde{x} = (x_1, x_2) \) (2 attributes)
- Feature Space: \( \Phi(\tilde{x}) = (x_1^2, x_2^2, x_1, x_2, x_1 x_2, 1) \) (6 attributes)
Dual SVM Optimization Problem

• Primal Optimization Problem

\[
\text{minimize: } P(\vec{w}, b, \xi) = \frac{1}{2} \vec{w} \cdot \vec{w} + C \sum_{i=1}^{n} \xi_i \\
\text{subject to: } \forall i=1^n : y_i [\vec{w} \cdot \vec{x}_i + b] \geq 1 - \xi_i \\
\forall i=1^n : \xi_i > 0
\]

• Dual Optimization Problem

\[
\text{maximize: } D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j (\vec{x}_i \cdot \vec{x}_j) \\
\text{subject to: } \sum_{i=1}^{n} y_i \alpha_i = 0 \\
\forall i=1^n : 0 \leq \alpha_i \leq C
\]

• Theorem: If \( w^* \) is the solution of the Primal and \( \alpha^* \) is the solution of the Dual, then

\[
\vec{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \vec{x}_i
\]

\( \Phi(\vec{x}) \)
• **Problem:**
  – Very many Parameters! Polynomials of degree \( p \) over \( N \) attributes in input space lead to \( O(N^p) \) attributes in feature space!

• **Solution:**
  – The dual OP depends only on inner products
  \[ \Rightarrow \text{Kernel Functions } K(\vec{a}, \vec{b}) = \Phi(\vec{a}) \cdot \Phi(\vec{b}) \]

• **Example:**
  – For \( \Phi(\vec{x}) = (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, 1) \) calculating
  \[ K(\vec{a}, \vec{b}) = [\vec{a} \cdot \vec{b} + 1]^2 \]
  computes inner product in feature space.

\[ \Rightarrow \text{no need to represent feature space explicitly.} \]
SVM with Kernel

• Training:

maximize: \( D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j K(\vec{x}_i, \vec{x}_j) \)

subject to: \( \sum_{i=1}^{n} y_i \alpha_i = 0 \)
\( \forall_{i=1}^{m} : 0 \leq \alpha_i \leq C \)

• Classification:

\[
h(\vec{x}) = \text{sign} \left( \sum_{i=1}^{n} \alpha_i y_i \Phi(\vec{x}_i) \cdot \Phi(\vec{x}) + b \right)
\]

\[
= \text{sign} \left( \sum_{i=1}^{n} \alpha_i y_i K(\vec{x}_i, \vec{x}) + b \right)
\]

• New hypotheses spaces through new Kernels:
  
  – Linear: \( K(\vec{a}, \vec{b}) = \vec{a} \cdot \vec{b} \)
  
  – Polynomial: \( K(\vec{a}, \vec{b}) = [\vec{a} \cdot \vec{b} + 1]^d \)
  
  – Radial Basis Function: \( K(\vec{a}, \vec{b}) = \exp \left( -\gamma [\vec{a} - \vec{b}]^2 \right) \)
  
  – Sigmoid: \( K(\vec{a}, \vec{b}) = \tanh(\gamma [\vec{a} \cdot \vec{b}] + c) \)
Examples of Kernels

Polynomial

\[ K(\tilde{a}, \tilde{b}) = (\tilde{a} \cdot \tilde{b} + 1)^2 \]

Radial Basis Function

\[ K(\tilde{a}, \tilde{b}) = \exp\left(-\gamma [\tilde{a} - \tilde{b}]^2\right) \]
What is a Valid Kernel?

Definition: Let $X$ be a nonempty set. A function is a valid kernel in $X$ if for all $n$ and all $x_1, \ldots, x_n \in X$ it produces a Gram matrix

$$G_{ij} = K(x_i, x_j)$$

that is symmetric

$$G = G^T$$

and positive semi-definite

$$\forall \vec{\alpha}: \vec{\alpha}^T G \vec{\alpha} \geq 0$$
How to Construct Valid Kernels

Theorem: Let $K_1$ and $K_2$ be valid Kernels over $X \times X$, $\alpha \geq 0$, $0 \leq \lambda \leq 1$, $f$ a real-valued function on $X$, $\phi:X \rightarrow \mathbb{R}^m$ with a kernel $K_3$ over $\mathbb{R}^m \times \mathbb{R}^m$, and $K$ a symmetric positive semi-definite matrix. Then the following functions are valid Kernels:

$$K(x,z) = \lambda K_1(x,z) + (1-\lambda) K_2(x,z)$$
$$K(x,z) = \alpha K_1(x,z)$$
$$K(x,z) = K_1(x,z) K_2(x,z)$$
$$K(x,z) = f(x) f(z)$$
$$K(x,z) = K_3(\phi(x),\phi(z))$$
$$K(x,z) = x^\top K z$$
Kernels for Sequences: Two sequences are similar, if they have many common and consecutive subsequences.

Example [Lodhi et al., 2000]: For $0 \leq \lambda \leq 1$ consider the following features space:

<table>
<thead>
<tr>
<th></th>
<th>c-a</th>
<th>c-t</th>
<th>a-t</th>
<th>b-a</th>
<th>b-t</th>
<th>c-r</th>
<th>a-r</th>
<th>b-r</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi(\text{cat})$</td>
<td>$\lambda^2$</td>
<td>$\lambda^3$</td>
<td>$\lambda^2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\phi(\text{car})$</td>
<td>$\lambda^2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$\lambda^3$</td>
<td>$\lambda^2$</td>
<td>0</td>
</tr>
<tr>
<td>$\phi(\text{bat})$</td>
<td>0</td>
<td>0</td>
<td>$\lambda^2$</td>
<td>$\lambda^2$</td>
<td>$\lambda^3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\phi(\text{bar})$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$\lambda^2$</td>
<td>0</td>
<td>0</td>
<td>$\lambda^2$</td>
<td>$\lambda^3$</td>
</tr>
</tbody>
</table>

$\Rightarrow K(\text{car,cat}) = \lambda^4$, efficient computation via dynamic programming.
Kernels for Non-Vectorial Data

• Applications with Non-Vectorial Input Data
  → classify non-vectorial objects
  – Protein classification (x is string of amino acids)
  – Drug activity prediction (x is molecule structure)
  – Information extraction (x is sentence of words)
  – Etc.

• Applications with Non-Vectorial Output Data
  → predict non-vectorial objects
  – Natural Language Parsing (y is parse tree)
  – Noun-Phrase Co-reference Resolution (y is clustering)
  – Search engines (y is ranking)

→ Kernels can compute inner products efficiently!
Properties of SVMs with Kernels

• Expressiveness
  – SVMs with Kernel can represent any boolean function (for appropriate choice of kernel)
  – SVMs with Kernel can represent any sufficiently “smooth” function to arbitrary accuracy (for appropriate choice of kernel)

• Computational
  – Objective function has no local optima (only one global)
  – Independent of dimensionality of feature space

• Design decisions
  – Kernel type and parameters
  – Value of C
SVMs for other Problems

• Multi-class Classification
  – [Schoelkopf/Smola Book, Section 7.6]

• Regression
  – [Schoelkopf/Smola Book, Section 1.6]

• Outlier Detection

• Structured Output Prediction