Kernel Dependency Estimation
Presented by Alex Ainslie | Advanced Machine Learning | CS 6784 | February 18, 2010

Motivation

- Learning problem
- Find a dependency between a general class of objects and another
- Relies on kernel functions because it uses similarity measures in both input and output spaces
- Encodes complex costs and outputs

Learning

- Inputs $x \in \mathcal{X}$
- Outputs $y \in \mathcal{Y}$
- Learn the function $f(x, \alpha^*)$
- Minimum value of risk function

$$R(\alpha) = \int_{\mathcal{X} \times \mathcal{Y}} L(y, f(x, \alpha))dP(x, y)$$

- Requires a priori knowledge of similarity measure (the loss function for outputs)

Complex Cost

- This loss function can be simple:
  - pattern recognition (zero-one loss)
  - regression (squared loss)
- or more complicated:
  - mapping to images
  - mixture of drugs

Kernel Functions

- A kernel $k$ is:
  - a symmetric function
  - an inner product in some Hilbert space $\mathcal{F}$
    (same class: high, different class: low)
    $\Phi_k : \mathcal{X} \rightarrow \mathcal{F}$ such that $k(x, x') = (\Phi_k(x) \cdot \Phi_k(x'))$
- EX: $k(x, x') = (x \cdot x' + 1)^p$

Kernel Examples

- M-class pattern recognition
  $\ell_{\text{pat}}(y, y') = \frac{1}{2} |y - y'|$
  $\Phi_k(y) = (0, 0, \ldots, 0, \frac{x_{i1}^2, \ldots, 0}$ where the $y^{th}$ coordinate is nonzero
- Regression estimation
  $\ell_{\text{reg}}(y, y') = (y - y')$
- Strings
  $\ell(s, t) = \sum_{u \in \Sigma^r} \sum_{w \in \Sigma^{|x|}} \chi^{(i)}(u) \chi^{(j)}(w)$
  exponential decay

$\Sigma$ ordered subsequences of length $r$
Algorithm (KDE)

- Minimize the risk function using the feature space \( F \) induced by the kernel \( k \) and the loss function measured in the space \( L \) induced by the kernel \( l \)
- Decomposition of outputs
- Learning the map
- Solving the pre-image

Decompose

- Construct kernel matrix \( L \) on training data
- Perform kernel PCA
  \[ L' = (I - \frac{1}{m}1_m1_m^T)L(I - \frac{1}{m}1_m1_m^T) \]
  \( n^{th} \) principal component
  \[ (\psi^n \cdot \Phi(x)) = \sum_{i=1}^{m} \alpha_i^n \Phi_i(x) \]
  \( (\psi^n \cdot \Phi_{\ell}(y)) = \sum_{i=1}^{m} \alpha_i^n \ell(y_i, y) \).

Map

- Using the \( p \) principal components
- Perform kernel ridge regression
- Estimator:
  \[ f_{\beta}(x) = \sum_{i=1}^{m} \beta_i k(x_i, x), \quad \beta = (K + \gamma I)^{-1} \psi^n \]

Pre-Image

- During testing to find estimate for \( y \) for a given \( x \), we need the pre-image \( \Phi(x) \)
  \[ y(x) = \text{argmin}_{y \in Y} ||(\psi \cdot \Phi(y), \ldots, \psi \cdot \Phi(y)) - (f_1(x), \ldots, f_p(x))|| \]

Experiment: Images

- USPS handwritten 16 pixel digit database
- Classification

<table>
<thead>
<tr>
<th>Method</th>
<th>RBF classification loss</th>
<th>k-NN classification loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>KDE</td>
<td>0.0878 ± 0.0067</td>
<td>0.1847 ± 0.0064</td>
</tr>
<tr>
<td>k-NN</td>
<td>0.1293 ± 0.0075</td>
<td>0.0805 ± 0.0072</td>
</tr>
</tbody>
</table>

Experiment: Images

- Image Reconstruction
- Estimate using first 8 rows
KDE Mistakes

Original, KDE, KNN

KNN Mistakes

Original, KDE, KNN

Toy Problem: Strings

- Predict output string from input string
- Almost classification with three classes

<table>
<thead>
<tr>
<th>input string</th>
<th>output string</th>
<th>KDE</th>
<th>4-NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>cecececececececece</td>
<td>aabe</td>
<td>0.070 ± 0.030</td>
<td>0.085 ± 0.029</td>
</tr>
<tr>
<td>dcecececececececece</td>
<td>abc</td>
<td>0.125 ± 0.012</td>
<td>0.205 ± 0.026</td>
</tr>
<tr>
<td>addcddddddddddd</td>
<td>bb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bbdddddddbddbd</td>
<td>aebad</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cdeccacacedddcdd</td>
<td>abad</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

String Subsequence Kernel

- Compare text documents by substrings (not necessarily contiguous) \( \lambda \in (0,1) \)
- c-a-r is in card and custard
- Used for both inputs and outputs

KNN Mistakes

Original, KDE, KNN

Toy Problem: Strings

- Alphabet (a,b,c,d)
- Input: random length (10 -15)
- Three classes of strings
  - transitions equally likely : abad
  - 0.7 repeat, 0.1 other : dbbd
  - 0.7 repeat, 0.1 other, only c,d : aabc
- Outputs corrupted with noise

String Subsequence Kernel

- Compare text documents by substrings (not necessarily contiguous) \( \lambda \in (0,1) \)
- c-a-r is in card and custard
- Used for both inputs and outputs

Toy Problem: Strings

- cat, car, bat, bar \( \lambda = 0.01 \)
- ca, ct, at, ca, cr, ar, ba, bt, at, ba, br, ar

\[
\begin{align*}
\phi(\text{cat}) & = \lambda^2 \\
\phi(\text{car}) & = \lambda^2 \\
\phi(\text{bat}) & = \lambda^2
\end{align*}
\]

\[
K(\text{car}, \text{cat}) = \lambda^4
\]
Toy Problem: Strings

\[ k(x, x') = k(x, x')/(\sqrt{k(x, x)} \sqrt{k(x', x')}) \exp\left(-k(x, x) + k(x', x') - 2k(x, x')/2\sigma^2\right) \]

- Find this distance (similarity) measure for each pair in inputs and outputs
- Then using kernel ridge regression to finding a mapping
- Pre-image: closest training example output to the given solution

Image: Joachims, SIGIR03 Tutorial Slides

Kernel Dependency Estimation

Presented by Alex Ainslie | Advanced Machine Learning | CS 6784 | February 18, 2010