Uniform Approximation of Functions with Random Bases

Ali Rahimi
Intel Research

Ben Recht
UW Madison
• **Goal:** Find a class $\mathcal{F}$ which is easy to search over, but can approximate complex behavior.

Typically a list of example inputs

$\min_{f \in \mathcal{F}} \text{fitness}(f, \text{data})$

Which space of functions? dictated by application

classes covariate state
Approximation Schemes

- Approximate \( f(x) \) by
  \[
  f_n(x) = \sum_{k=1}^{n} c_k \phi_k(x; \theta_k)
  \]

- Jones (1992), \( \tilde{f} \in L_1(\mathbb{R}^d) \)
  \( \phi(x; w, b) = \cos(w^* x + b) \)

- Barron (1993), \( \nabla \tilde{f} \in L_1(\mathbb{R}^d) \)
  \( \phi(x; w, b) = \sigma(w^* x + b) \)

- Girosi & Anzellotti (1995), \( f \in W_{2,s}(\mathbb{R}^d) \)
  \( \phi(x; z) = \exp(-\|x - z\|^2) \)
  with \( 2s > d \)

- Using nearly identical analysis, all of these schemes achieve
  \[
  \|f - f_n\|_2 = O\left( \frac{1}{\sqrt{n}} \right)
  \]
Approximation Schemes

- Approximate $f(x)$ by $f_n(x) = \sum_{k=1}^{n} c_k \phi_k(x; \theta_k)$

- Parameter tuning is tricky...

- (Can achieve "algorithm").

\[ \|f - f_n\|_2 = O\left(\frac{1}{\sqrt{n}}\right) \quad \text{via a greedy} \]
Randomize, don’t optimize

- Approximate $f(x)$ by $f_n(x) = \sum_{k=1}^{n} c_k \phi_k(x; \theta_k)$

- For which functions can we achieve $\|f - f_n\| = O\left(\frac{1}{\sqrt{n}}\right)$?

- How are these functions related to objects we already know and love?

- Practical Implementations
Function Class

- Fix parameterized basis functions $\phi(x; \theta)$
- Fix a probability distribution $p(\theta)$

Our target space will be:

$$
\mathcal{F}_p \equiv \left\{ f = \int \alpha(\theta)\phi(\cdot; \theta)d\theta \mid \sup_{\theta} \left| \frac{\alpha(\theta)}{p(\theta)} \right| < \infty \right\}
$$

- With the convention that

$$
\left| \frac{\alpha(\theta)}{0} \right| = \begin{cases} 0 & \alpha(\theta) = 0 \\ \infty & \text{otherwise} \end{cases}
$$
Random Features: Example

- Fourier basis functions: \( \phi(x; \omega, b) = \cos(\omega^* x + b) \)

- Gaussian parameters \( \omega \sim \mathcal{N}(0, \sigma^2 I) \) \( b \sim \text{unif}([0, 2\pi]) \)

- If \( \tilde{f}(\omega) = \int_{\mathbb{R}^d} f(x)e^{-i\omega^* x} \, dx \), then \( \sup_{\omega} \left| \frac{\tilde{f}(\omega)}{p(\omega)} \right| \leq \infty \) means that the frequency distribution of \( f \) has subgaussian tails.
\[ \mathcal{F}_p \equiv \left\{ f = \int \alpha(\theta) \phi(\cdot; \theta) d\theta \mid \sup_{\theta} \left| \frac{\alpha(\theta)}{p(\theta)} \right| \leq \infty \right\} \]

- **Thm:** Let \( f \) be in \( \mathcal{F}_p \) with \( \|f\|_p = \sup_{\theta} \left| \frac{\alpha(\theta)}{p(\theta)} \right| \). Let \( \theta_1, \ldots, \theta_n \) be sampled iid from \( p \). Then with probability at least \( 1 - \delta \):

\[
\min_{c_k} \left\| f - \sum_{k=1}^{n} c_k \phi(x; \theta_k) \right\|_2 \leq \frac{\|f\|_p}{\sqrt{n}} \left( \sqrt{2} + \frac{\sqrt{2}}{2} \log\left(\frac{1}{\delta}\right) \right)
\]

- If additionally, \( \phi(x; \theta) = \phi(\theta'x) \), with \( \phi: \mathbb{R} \to \mathbb{R} \) L-Lipschitz, \( \phi(0) = 0 \), and \( |\phi| < 1 \) and \( p \) has a finite second moment, then with probability at least \( 1 - \delta \)

\[
\min_{c_k} \left\| f - \sum_{k=1}^{n} c_k \phi(x; \theta_k) \right\|_{\infty} \leq \frac{\|f\|_p}{\sqrt{n}} \left( \sqrt{\log \frac{1}{\delta}} + 4LB\sqrt{\mathbb{E}\theta'\theta} \right)
\]

where \( B = \sup_{x \in X} \|x\|_2 \)
Reproducing Kernel Hilbert Spaces

- A symmetric function $k: X \times X \to \mathbb{R}$ is a positive definite kernel if for all $N$
  $$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j k(x_i, x_j) \geq 0$$

- Reproducing Kernel Hilbert Space: $\{ f(x) = \sum_{i=1}^{n} c_i k(x_i, x) \}$
  $$\left\langle \sum_{j=1}^{n} c_j k(x_j, \cdot), \sum_{k=1}^{m} d_k k(z_k, \cdot) \right\rangle_k := \sum_{j=1}^{n} \sum_{k=1}^{m} c_j d_k k(x_j, z_k)$$

- Extensive Applications: Support Vector Machines, Kernel Machines, etc.
\[ k(x, y) = \int p(\theta) \phi(x; \theta) \phi(y; \theta) d\theta \]

- RKHS generated by \( k \): \( \mathcal{H} = \{ f(x) = \sum_{i=1}^{n} c_i k(x_i, x) \} \)

\[ \mathcal{F}_p \equiv \left\{ f = \int \alpha(\theta) \phi(\cdot; \theta) d\theta \mid \sup_{\theta} \left| \frac{\alpha(\theta)}{p(\theta)} \right| \leq \infty \right\} \]

- \( \mathcal{F}_p \) is dense in \( \mathcal{H} \), and for any \( f \in \mathcal{F}_p \)

\[ \| f \|_k \leq \sup_{\theta} \left| \frac{\alpha(\theta)}{p(\theta)} \right| \]
Gaussian RKHS vs Random Features

• **Representer Theorem**: for many applications, the optimal function in an RKHS is of the form

\[ \sum_{i=1}^{n} c_i k(x_i, x) \]

• **RKHS form is preferred**: when number of data points is small or the function is not smooth

• **Random Features are preferred**: when number of data points is very large or the Representer theorem doesn’t apply
Fourier Random Features

\[ \omega \sim \mathcal{N}(0, 1) \]
\[ b \sim \text{unif}[-\pi, \pi] \]
\[ \phi(x; \omega, b) = \cos(\omega'x + b) \]

\[ k(x, y) = \exp\left(-\gamma\|x - y\|^2\right) \]

 RKHS is dense in continuous functions
Random Decision Stumps

\[ t \sim \text{unif}([a, b]) \]
\[ i \sim \text{unif}([1, \ldots, d]) \]

\[ \phi(x; t, i) = \text{sign}(x_i - t) \]

Boosting Features

\[ k(x, y) = 1 - 2 \frac{\|x - y\|_1}{b - a} \]
Binning Random Features

Lay a random grid so that for any \( x \) and \( y \)

\[
Pr[x \text{ and } y \text{ are binned together}] = k(x, y)
\]

\( \phi(x) \) is the bin ID, encoded as a binary indicator vector.

\[
k(x - y) = \prod_{d} \int_{0}^{\infty} k_{\hat{}}(x - y; \delta) p(\delta) \, d\delta
\]

\[
k_{\hat{}}(x - y; \delta) = \max \left( 0, 1 - \delta^{-1} |x - y| \right)
\]
% Approximates Gaussian Process regression
% with Gaussian kernel of variance gamma
% lambda: regularization parameter
% dataset: X is dxN, y is 1xN
% test: xtest is dx1
% D: dimensionality of random feature

% training
w = randn(D, size(X,1));
b = 2*pi*rand(D,1);
Z = cos(sqrt(gamma)*w*X + repmat(b,1,size(X,2)));
alpha = (lambda*eye(size(X,2)+Z*Z')\(Z*y);

% testing
ztest = alpha(:)'*cos( sqrt(gamma)*w*xtest(:) + ... 
+ repmat(b,1,size(X,2)) );
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Fourier+LS</th>
<th>Binning+LS</th>
<th>Exact SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU regression</td>
<td>3.6%</td>
<td>5.3%</td>
<td>11%</td>
</tr>
<tr>
<td>6500 instances 21 dims</td>
<td>20 secs</td>
<td>3 mins</td>
<td>31 secs</td>
</tr>
<tr>
<td></td>
<td>( D = 300 )</td>
<td>( P = 350 )</td>
<td>ASVM</td>
</tr>
<tr>
<td>Census regression</td>
<td>5%</td>
<td>7.5%</td>
<td>9%</td>
</tr>
<tr>
<td>18,000 instances 119 dims</td>
<td>36 secs</td>
<td>19 mins</td>
<td>13 mins</td>
</tr>
<tr>
<td></td>
<td>( D = 500 )</td>
<td>( P = 30 )</td>
<td>SVM Torch</td>
</tr>
<tr>
<td>Adult classification</td>
<td>14.9%</td>
<td>15.3%</td>
<td>15.1%</td>
</tr>
<tr>
<td>32,000 instances 123 dims</td>
<td>9 secs</td>
<td>1.5 mins</td>
<td>7 mins</td>
</tr>
<tr>
<td></td>
<td>( D = 500 )</td>
<td>( P = 30 )</td>
<td>SVM light</td>
</tr>
<tr>
<td>Forest Cover classification</td>
<td>11.6%</td>
<td>2.2%</td>
<td>2.2%</td>
</tr>
<tr>
<td>522,000 instances 54 dims</td>
<td>71 mins</td>
<td>25 mins</td>
<td>44 hrs</td>
</tr>
<tr>
<td></td>
<td>( D = 5000 )</td>
<td>( P = 50 )</td>
<td>lib SVM</td>
</tr>
</tbody>
</table>
\[ f(x) = \sum_{k=1}^{n} c_k \sigma(w_k^* x + b_k) \]