A random matrix analysis of random fourier features

beyond the Gaussian kernel, a precise phase transition, and the corresponding double descent

Kailong Wang[1]

[1]Ph.D. of ECE
Rutgers University

ECE 539 HDP, May 16, 2023

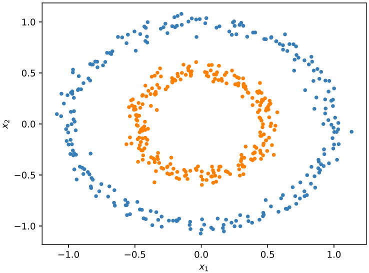
Table of Contents

1. Motivation
2. Random Fourier Features
3. An analysis of RFF

Linear Classification with Non-linear Input

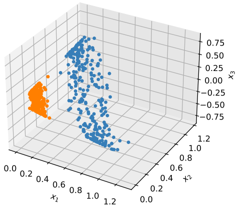
Consider a binary classification problem with non-linear (*e.g.* polynomial) samples. This is not separable with linear function.

$$(e.g.\ \mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \\ & \dots \\ x_{N,1} & x_{N,2} \end{bmatrix} \in \mathbb{R}^{N\times 2}.)$$



Lifting

One idea is to **LIFT** the samples into a higher dimensional space in which the samples are linearly separable.



The Lifting function in this case is $\phi(\mathbf{X}) = \begin{bmatrix} x_{1,1}^2 & x_{1,2}^2 & \sqrt{2}x_{1,1}x_{1,2} \\ x_{2,1}^2 & x_{2,2}^2 & \sqrt{2}x_{2,1}x_{2,2} \\ & \cdots & \\ x_{N,1}^2 & x_{N,2}^2 & \sqrt{2}x_{N,1}x_{N,2} \end{bmatrix}$.

Curse of Dimensionality

Consider solving the above problem with *support vector machine* (SVM).

$$\mathcal{L}(\mathbf{w}, \alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n}^{N} \sum_{m}^{N} \alpha_n \alpha_m y_n y_m (\mathbf{x_n}^{\mathsf{T}} \mathbf{x_m}).$$

The $\mathbf{w}$ is the linear decision boundary and α is a vector of Lagrange multipliers.

Curse of Dimensionality

Consider solving the above problem with *support vector machine* (SVM).

$$\mathcal{L}(\mathbf{w}, \alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n}^{N} \sum_{m}^{N} \alpha_n \alpha_m y_n y_m (\mathbf{x_n}^{\mathsf{T}} \mathbf{x_m}).$$

The $\mathbf{w}$ is the linear decision boundary and α is a vector of Lagrange multipliers.

We need to use lifting function $\phi(X)$ to make the samples linearly separable. Specifically, we replace $(\mathbf{x_n}^{\mathsf{T}} \mathbf{x_m})$ with $(\phi(\mathbf{x_n})^{\mathsf{T}} \phi(\mathbf{x_m}))$.

$$\begin{aligned}\phi(\mathbf{x_n})^{\mathsf{T}} \phi(\mathbf{x_m}) &= \begin{bmatrix} x_{n,1}^2 & x_{n,2}^2 & \sqrt{2} x_{n,1} x_{n,2} \end{bmatrix} \begin{bmatrix} x_{m,1}^2 & x_{m,2}^2 & \sqrt{2} x_{m,1} x_{m,2} \end{bmatrix}^{\mathsf{T}} \\ &= x_{n,1}^2 x_{m,1}^2 + x_{n,2}^2 x_{m,2}^2 + 2 x_{n,1} x_{n,2} x_{m,1} x_{m,2}\end{aligned}$$

Curse of Dimensionality

Consider solving the above problem with *support vector machine* (SVM).

$$\mathcal{L}(\mathbf{w}, \alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n}^{N} \sum_{m}^{N} \alpha_n \alpha_m y_n y_m (\mathbf{x_n}^\mathsf{T} \mathbf{x_m}).$$

The $\mathbf{w}$ is the linear decision boundary and α is a vector of Lagrange multipliers.

We need to use lifting function $\phi(X)$ to make the samples linearly separable. Specifically, we replace $(\mathbf{x_n}^\mathsf{T}\mathbf{x_m})$ with $(\phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m}))$.

$$\begin{aligned}\phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m}) &= \left[x_{n,1}^2 \ x_{n,2}^2 \ \sqrt{2}x_{n,1}x_{n,2}\right]\left[x_{m,1}^2 \ x_{m,2}^2 \ \sqrt{2}x_{m,1}x_{m,2}\right]^\mathsf{T} \\ &= x_{n,1}^2 x_{m,1}^2 + x_{n,2}^2 x_{m,2}^2 + 2x_{n,1}x_{n,2}x_{m,1}x_{m,2}\end{aligned}$$

Calculate the inner product in the $\mathbb{R}^3$ across all N pairs of samples is acceptable. However, the lifting function $\phi(X)$ is usually very high dimensional.

Kernel Trick

Consider the following derivation,

$$\begin{aligned}
(\mathbf{x_n}^\mathsf{T}\mathbf{x_m})^2 &= \left([x_{n,1}\ x_{n,2}][x_{m,1}\ x_{m,2}]^\mathsf{T}\right)^2 \\
&= (x_{n,1}x_{m,1} + x_{n,2}x_{m,2})^2 \\
&= x_{n,1}^2 x_{m,1}^2 + x_{n,2}^2 x_{m,2}^2 + 2x_{n,1}x_{n,2}x_{m,1}x_{m,2} \\
&= \phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m})
\end{aligned}$$

Kernel Trick

Consider the following derivation,

$$\begin{aligned}
(\mathbf{x_n}^\mathsf{T}\mathbf{x_m})^2 &= \left([x_{n,1}\ x_{n,2}][x_{m,1}\ x_{m,2}]^\mathsf{T}\right)^2 \\
&= (x_{n,1}x_{m,1} + x_{n,2}x_{m,2})^2 \\
&= x_{n,1}^2 x_{m,1}^2 + x_{n,2}^2 x_{m,2}^2 + 2x_{n,1}x_{n,2}x_{m,1}x_{m,2} \\
&= \phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m})
\end{aligned}$$

Instead of computing inner product in the high dimensional space, we compute the inner product in the original space.

Kernel Trick

Consider the following derivation,

$$\begin{aligned}
(\mathbf{x_n}^\mathsf{T}\mathbf{x_m})^2 &= \left([x_{n,1}\ x_{n,2}][x_{m,1}\ x_{m,2}]^\mathsf{T}\right)^2 \\
&= (x_{n,1}x_{m,1} + x_{n,2}x_{m,2})^2 \\
&= x_{n,1}^2 x_{m,1}^2 + x_{n,2}^2 x_{m,2}^2 + 2x_{n,1}x_{n,2}x_{m,1}x_{m,2} \\
&= \phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m})
\end{aligned}$$

Instead of computing inner product in the high dimensional space, we compute the inner product in the original space.

The function

$$K(\mathbf{x_n}, \mathbf{x_m}) = (\mathbf{x_n}^\mathsf{T}\mathbf{x_m})^2 = \phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m})$$

is called a **kernel function**.

There must be disadvantages. . .

Given training data $(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \dots, (\mathbf{x_N}, y_N) \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$. Consider *Kernel Ridge Regression* (KRR), with $\phi(\mathcal{X}) \subseteq \mathbb{R}^k$, where $k \to \infty$

$$\mathcal{L}(\mathbf{w}, \lambda) = \underset{\mathbf{w}}{\text{argmin}} \sum_{n}^{N} (y_n - \mathbf{w}^\intercal \phi(\mathbf{x}_n))^2 + \lambda \mathbf{w}^\intercal \mathbf{w}.$$

Solving it with Lagrange multipliers α, which is the solution of

$$(\mathbf{K} + \lambda \mathbf{I}_k)\alpha = \mathbf{y},$$

requires $\Theta(k^3)$ time and $\Theta(k^2)$ memory. Here $\mathbf{K} \in \mathbb{R}^{k \times k}$ is the kernel matrix or Gram matrix defined by $\mathbf{K}_{nm} \equiv K(\mathbf{x_n}, \mathbf{x_m})$.

There must be disadvantages. . .

Given training data $(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_N}, y_N) \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$. Consider *Kernel Ridge Regression* (KRR), with $\phi(\mathcal{X}) \subseteq \mathbb{R}^k$, where $k \to \infty$

$$\mathcal{L}(\mathbf{w}, \lambda) = \operatorname*{argmin}_{\mathbf{w}} \sum_{n}^{N} (y_n - \mathbf{w}^\intercal \phi(\mathbf{x}_n))^2 + \lambda \mathbf{w}^\intercal \mathbf{w}.$$

Solving it with Lagrange multipliers α, which is the solution of

$$(\mathbf{K} + \lambda \mathbf{I}_k)\alpha = \mathbf{y},$$

requires $\Theta(k^3)$ time and $\Theta(k^2)$ memory. Here $\mathbf{K} \in \mathbb{R}^{k \times k}$ is the kernel matrix or Gram matrix defined by $\mathbf{K}_{nm} \equiv K(\mathbf{x_n}, \mathbf{x_m})$.

Intuition: Can we find a kernel function which lifts $\mathcal{X}$ to $\mathbb{R}^s$, where $d < s \ll k$, while not sacrifices model performance?

Some Prerequisites

Shift Invariant Kernel (Radial Basis Function (RBF))

A kernel function $K(\mathbf{x_n}, \mathbf{x_m})$ is called **shift invariant** if it can be written as $K(\mathbf{x_n}, \mathbf{x_m}) = g(\mathbf{x_n} - \mathbf{x_m})$ for some function $g(\cdot)$ (*e.g.* $K_{Gaussian}(\mathbf{x_n}, \mathbf{x_m}) = \exp(-\gamma\|\mathbf{x_n} - \mathbf{x_m}\|_2^2)$).

Mercer's Theorem

A continuous function $K(\mathbf{x_n}, \mathbf{x_m})$ is a valid kernel function if and only if the kernel matrix $\mathbf{K}$ is **positive semi-definite**.

Bochner's Theorem

A continuous function $g(\cdot)$ is **positive semi-definite** if and only if it is the Fourier transform of a non-negative measure.

Random Fourier Features

Conclusion

A continuous **shift invariant** kernel $K(\mathbf{x_n}, \mathbf{x_m})$, which is **positive semi-definite** (Mercer's Theorem), is the Fourier transform of a non-negative measure $p(\cdot)$.

$$
\begin{aligned}
\phi(\mathbf{x_n})^\top \phi(\mathbf{x_m}) &= K(\mathbf{x_n}, \mathbf{x_m}) = K(\mathbf{x_n} - \mathbf{x_m}) && (1)\\
&= \int_{\mathbb{R}^d} p(\omega) \exp(i\omega^\top (\mathbf{x_n} - \mathbf{x_m}))\, \mathrm{d}\omega && (2)\\
&= \mathbb{E}_\omega \left[\xi_\omega(\mathbf{x_n})^\mathsf{H} \xi_\omega(\mathbf{x_m}) \right] && (3)
\end{aligned}
$$

Here $\xi_\omega(\mathbf{x}) = \exp(i\omega^\top \mathbf{x}) = \begin{bmatrix} \cos(\omega^\top \mathbf{x}) \\ \sin(\omega^\top \mathbf{x}) \end{bmatrix}$ and hence $\xi_\omega(\mathbf{x_n})^* \xi_\omega(\mathbf{x_m})$ is an unbiased estimator of $K(\mathbf{x_n}, \mathbf{x_m})$ when ω is drawn from $p(\cdot)$.

Random Fourier Features

Since both the $p(\cdot)$ and $K(\triangle)$ are real-valued, we can replace $\xi_\omega(\mathbf{x})$ with $z_\omega(\mathbf{x}) = [\sqrt{2}\cos(\omega^\intercal \mathbf{x} + b)]$ where ω is drawn from $p(\omega)$ and b is uniformly drawn from $[0, 2\pi]$. Then eq. (3) becomes $\mathbb{E}_\omega[z_\omega(\mathbf{x_n})^\intercal z_\omega(\mathbf{x_m})]$

Random Fourier Features

Since both the $p(\cdot)$ and $K(\triangle)$ are real-valued, we can replace $\xi_\omega(\mathbf{x})$ with $z_\omega(\mathbf{x}) = [\sqrt{2}\cos(\omega^\mathsf{T}\mathbf{x} + b)]$ where ω is drawn from $p(\omega)$ and b is uniformly drawn from $[0, 2\pi]$. Then eq. (3) becomes $\mathbb{E}_\omega[z_\omega(\mathbf{x_n})^\mathsf{T} z_\omega(\mathbf{x_m})]$

Note: $z_\omega(\mathbf{x_n})^\mathsf{T} z_\omega(\mathbf{x_m})$ is an unbiased estimator of $\phi(\mathbf{x_n})^\mathsf{T}\phi(\mathbf{x_m})$. The $z_\omega(\mathbf{x})$ is not a lifting function.

Random Fourier Features

Since both the $p(\cdot)$ and $K(\triangle)$ are real-valued, we can replace $\xi_\omega(\mathbf{x})$ with $z_\omega(\mathbf{x}) = [\sqrt{2}\cos(\omega^\intercal \mathbf{x} + b)]$ where ω is drawn from $p(\omega)$ and b is uniformly drawn from $[0, 2\pi]$. Then eq. (3) becomes $\mathbb{E}_\omega[z_\omega(\mathbf{x_n})^\intercal z_\omega(\mathbf{x_m})]$

Note: $z_\omega(\mathbf{x_n})^\intercal z_\omega(\mathbf{x_m})$ is an unbiased estimator of $\phi(\mathbf{x_n})^\intercal \phi(\mathbf{x_m})$. The $z_\omega(\mathbf{x})$ is not a lifting function.

Note: To further reduce the variance of the estimator, we can randomly draw s samples of ω and normalize each corresponding $z_\omega(\mathbf{x})$ by $\sqrt{s}$. Then the inner product $z(\mathbf{x_n})^\intercal z(\mathbf{x_m}) = \frac{1}{s}\sum_{j=1}^{s} z_{\omega j}(\mathbf{x_n})^\intercal z_{\omega j}(\mathbf{x_m})$

Algorithm

Algorithm Random Fourier Features

Require: A shift invariant kernel $K(\mathbf{x_n}, \mathbf{x_m}) = K(\mathbf{x_n} - \mathbf{x_m})$.

Ensure: A randomized feature map $z(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^s$ so that $z(\mathbf{x_n})^\intercal z(\mathbf{x_m}) \approx K(\mathbf{x_n}, \mathbf{x_m})$.

Compute the Fourier transform $p(\cdot)$ of the kernel $K : p(\omega) = \frac{1}{2\pi} \int \exp(-i\omega^\intercal \triangle) K(\triangle) \, \mathrm{d}\triangle$

Draw s i.i.d. samples $\omega_1, \omega_2, \ldots, \omega_s \in \mathbb{R}^d$ from $p(\cdot)$ and s i.i.d. samples $b_1, b_2, \ldots, b_s \in [0, 2\pi]$.

Let $z(\mathbf{x}) \equiv \sqrt{\frac{2}{s}} [\cos(\omega_1{}^\intercal \mathbf{x} + b_1) \ \cos(\omega_2{}^\intercal \mathbf{x} + b_2) \ \ldots \ \cos(\omega_s{}^\intercal \mathbf{x} + b_s)]^\intercal$

Convergence

Bound for a *fixed* pair of samples $\mathbf{x_n}$ and $\mathbf{x_m}$

Given z_ω is bounded random variable between $[-\sqrt{2}, \sqrt{2}]$, with Hoeffding's Inequality, we have

$$\mathbb{P}\left(|z(\mathbf{x_n})^\mathsf{T} z(\mathbf{x_m}) - K(\mathbf{x_n}, \mathbf{x_m})| \geq \epsilon\right) \leq 2\exp\left(-\frac{s\epsilon^2}{4}\right).$$

Convergence

Bound for *all* pair of samples $\mathbf{x_n}$ and $\mathbf{x_m}$

Let $\mathcal{M}$ be a compact subset of $\mathbb{R}^d$ with diameter $\text{diam}(\mathcal{M})$. Then, for the mapping z defined in Algorithm 1, we have

$$\mathbb{P}\left(\sup_{x,y\in\mathcal{M}} |z(\mathbf{x_n})^\mathsf{T} z(\mathbf{x_m}) - K(\mathbf{x_n}, \mathbf{x_m})| \geq \epsilon\right) \leq 2^8 \left(\frac{\sigma_{p(\cdot)}\text{diam}(\mathcal{M})}{\epsilon}\right)^2 \exp\left(-\frac{s\epsilon^2}{4(d+2)}\right).$$

Common RFF

Kernel	$K(\triangle)$	$p(\omega)$
Gaussian	$\exp(-\gamma\|\triangle\|_2^2)$	$(2\pi)^{-\frac{s}{2}}\exp -\gamma\|\omega\|_2^2$
Laplacian	$\exp(-\|\triangle\|_1)$	$\prod_d(\pi(1+\omega_d^2))^{-1}$
Cauchy	$\prod_d 2(1+\triangle_d^2)^{-1}$	$\exp(-\|\triangle\|_1)(?)$

The challenge that RFF faces in the learning regime

Consider a machine learning system with d parameters, trained on a dataset of size N, asymptotic analysis has

Classical regime: either focuses on the (statistical) population $N \to \infty$ limit, for d fixed, or the over-parameterized $d \to \infty$ limit, for a given N.

Modern regime: modern learning system (*e.g.* Neural Network) usually has model complexity and data size increase together. A double asymptotic regime where $N, d \to \infty, d/N \to c$ is established.

RFF has been shown that entry-wise the Gram matrix $\xi(\mathbf{x})$ converges to the Gaussian kernel matrix as $s \to \infty$ and this property remains in modern regime.
However, the convergence $\|\mathbf{\Xi}^\mathsf{T}\mathbf{\Xi}/s - \mathbf{K}\| \to 0$ no longer holds in spectral norm (blow-up). Here $\mathbf{\Xi}$ is the matrix formed by stacking $\xi(\mathbf{x})$ for all samples.

Setup

$$0 < \liminf_{N} \min\{\tfrac{s}{N}, \tfrac{d}{N}\} \leq \limsup_{N} \max\{\tfrac{s}{N}, \tfrac{d}{N}\} < \infty.$$

$$\limsup_{N} \|\mathbf{X}\|_2 < \infty \qquad \limsup_{N} \|\mathbf{y}\|_\infty < \infty$$

In classical regime $\|\mathbf{\Xi}^\mathsf{T}\mathbf{\Xi}/s\| \equiv \mathbf{K} \equiv \mathbf{K}_{\cos} + \mathbf{K}_{\sin}$

Training MSE: $\mathcal{L}_{train} = \frac{1}{N}\|\mathbf{y} - \mathbf{\Xi}^\mathsf{T}\mathbf{w}\|_2^2 = \frac{\lambda^2}{N}\|\mathbf{Q}(\lambda)\mathbf{y}\|_2^2$ where

$$\mathbf{Q}(\lambda) \equiv \left(\frac{1}{N}\mathbf{\Xi}^\mathsf{T}\mathbf{\Xi} + \lambda\mathbf{I}_N\right)^{-1}$$

We want to assess the asymptotic $\mathcal{L}_{train}$ by expectation which is equivalent to assess the asymptotic $\mathbb{E}_{\mathbf{\Omega}}\{\mathbf{Q}(\lambda)\}$ where $\mathbf{\Omega}$ is the matrix form of ω, which is numerically hard.

Object: Find an asymptotic “alternative” for $\mathbb{E}_{\mathbf{\Omega}}\{\mathbf{Q}(\lambda)\}$ when $d, s, N \to \infty$.

Some Vague Idea from me...

We want to show that with consideration of d, s, N

$$\|\mathbb{E}_{\boldsymbol{\Omega}}\{\mathbf{Q}(\lambda)\} - \hat{\mathbf{Q}}(\lambda)\|_2 \to 0$$

$$\hat{\mathbf{Q}}(\lambda) \equiv \left(\frac{s}{N}\left(\frac{\mathbf{K}_{\cos}}{1+\delta_{\cos}} + \frac{\mathbf{K}_{\sin}}{1+\delta_{\sin}}\right) + \lambda \mathbf{I}_N\right)^{-1}$$

$$\delta_{\cos} = \frac{1}{N}\operatorname{tr}\left(\mathbf{K}_{\cos}\hat{\mathbf{Q}}\right) \qquad \delta_{\sin} = \frac{1}{N}\operatorname{tr}\left(\mathbf{K}_{\sin}\hat{\mathbf{Q}}\right)$$

When $\frac{s}{N} \to \infty$, $\delta_{\cos}, \delta_{\sin} \to 0$ and thus $\hat{\mathbf{Q}} \simeq \left(\frac{s}{N}\mathbf{K}\right)^{-1}$