

Learning Kernels with Random Features

Aman Sinha John Duchi

Stanford University

NIPS, 2016

Presenter: Ritambhara Singh

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

Motivation

- Randomized features computationally efficient for approximating kernels.

Motivation

- Randomized features computationally efficient for approximating kernels.
- Existing methods require a user defined kernel.

Motivation

- Randomized features computationally efficient for approximating kernels.
- Existing methods require a user defined kernel.
- **Weakness:** Poor choice of user defined kernel can lead to a useless model.

Motivation

- Randomized features computationally efficient for approximating kernels.
- Existing methods require a user defined kernel.
- **Weakness:** Poor choice of user defined kernel can lead to a useless model.
- **Goal:** Combine kernel learning with randomization.

Motivation

- Randomized features computationally efficient for approximating kernels.
- Existing methods require a user defined kernel.
- **Weakness:** Poor choice of user defined kernel can lead to a useless model.
- **Goal:** Combine kernel learning with randomization.
- **Idea:** Exploit computational advantage of randomized features for supervised kernel learning.

Outline

1 Introduction

- Motivation
- **Background**
- State-of-the-art

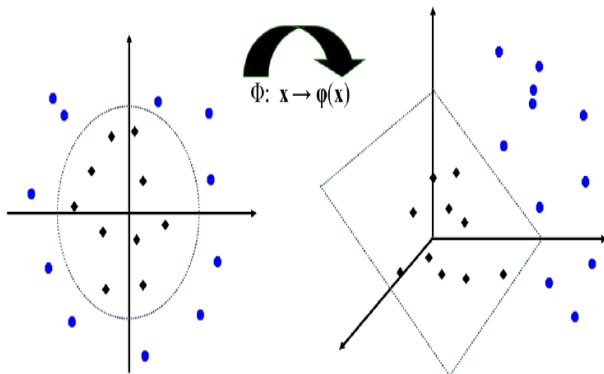
2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

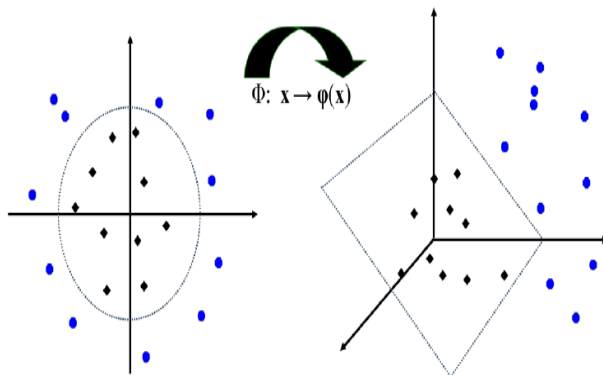
3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

Kernel



Kernel



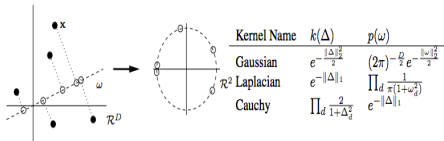
$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \quad (1)$$

Random Features for Kernel [Rahimi and Recht, NIPS '07]

$$K(x, y) = \langle \phi(x), \phi(y) \rangle = z(x)' z(y) \quad (2)$$

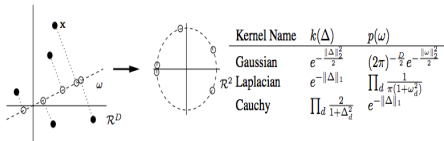
Random Features for Kernel [Rahimi and Recht, NIPS '07]

$$K(x, y) = \langle \phi(x), \phi(y) \rangle = z(x)' z(y) \quad (2)$$



Random Features for Kernel [Rahimi and Recht, NIPS '07]

$$K(x, y) = \langle \phi(x), \phi(y) \rangle = z(x)' z(y) \quad (2)$$



Algorithm 1 Random Fourier Features.

Require: A positive definite shift-invariant kernel $k(x, y) = k(x - y)$.

Ensure: A randomized feature map $z(x) : \mathcal{R}^d \rightarrow \mathcal{R}^D$ so that $z(x)' z(y) \approx k(x - y)$.

Compute the Fourier transform p of the kernel k : $p(\omega) = \frac{1}{2\pi} \int e^{-j\omega' \delta} k(\delta) d\Delta$.

Draw D iid samples $\omega_1, \dots, \omega_D \in \mathcal{R}^d$ from p and D iid samples $b_1, \dots, b_D \in \mathcal{R}$ from the uniform distribution on $[0, 2\pi]$.

Let $z(x) \equiv \sqrt{\frac{2}{D}} [\cos(\omega_1' x + b_1) \dots \cos(\omega_D' x + b_D)]'$.

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

- Heuristic rules to combine kernels
- Optimize structured compositions of kernels w.r.t an alignment metric [Elaborate]
- Jointly optimize kernel composition with empirical risk

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

- Create randomized features
- Solve an optimization problem to select a subset
- Train a model with the optimized features
- Learn lower dimensional models than original random-feature approach

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- **Formulation**
- Efficient solution

3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

Formulation

- For binary classification: given n data points $(x^i, y^i) \in R^d \times \{-1, 1\}$. Let $\phi : R^d \times \mathcal{W} \rightarrow [-1, 1]$ and Q be a probability measure on a space \mathcal{W} , a kernel can be defined as:

$$K_Q(x, x') := \int \phi(x, w)\phi(x', w)dQ(w) \quad (3)$$

Formulation

- For binary classification: given n data points $(x^i, y^i) \in R^d \times \{-1, 1\}$. Let $\phi : R^d \times \mathcal{W} \rightarrow [-1, 1]$ and Q be a probability measure on a space \mathcal{W} , a kernel can be defined as:

$$K_Q(x, x') := \int \phi(x, w)\phi(x', w)dQ(w) \quad (3)$$

- Find the "best" kernel K_Q over all distributions Q in some set \mathcal{P} of possible distributions on random features

$$\text{maximize}_{Q \in \mathcal{P}} \sum_{i,j} K_Q(x^i, x^j)y^i y^j \quad (4)$$

Formulation

- For binary classification: given n data points $(x^i, y^i) \in R^d \times \{-1, 1\}$. Let $\phi : R^d \times \mathcal{W} \rightarrow [-1, 1]$ and Q be a probability measure on a space \mathcal{W} , a kernel can be defined as:

$$K_Q(x, x') := \int \phi(x, w)\phi(x', w)dQ(w) \quad (3)$$

- Find the "best" kernel K_Q over all distributions Q in some set \mathcal{P} of possible distributions on random features

$$\text{maximize}_{Q \in \mathcal{P}} \sum_{i,j} K_Q(x^i, x^j) y^i y^j \quad (4)$$

- Given some base (user defined) distribution P_0 , Consider collections $\mathcal{P} := \{Q : D_f(Q||P_0) \leq \rho\}$, where $\rho > 0$ is a specified constant.

- Using randomized feature approach, approximate integral (3) as discrete sum over samples $W^i \sim P_0, i \in [N_w]$
- Approximate to $\mathcal{P} : \mathcal{P}_{N_w} := \{q : D_f(q||1/N_w) \leq \rho\}$
- Problem (4) becomes:

$$\underset{q \in \mathcal{P}_{N_w}}{\text{maximize}} \sum_{i,j} y^i y^j \sum_{m=1}^{N_w} q_m \phi(x^i, w^m) \phi(x^j, w^m) \quad (5)$$

Formulation contd.

- Given a solution \hat{q} , two ways to solve learning problem:
- Draw D samples $W^1, \dots, W^D \sim \hat{q}$ defining features $\phi^i = [\phi(x^i, w^1) \dots \phi(x^i, w^D)]^T$ and solve :

$$\hat{\theta} = \operatorname{argmin}_{\theta} \left\{ \sum_{i=1}^n c \left(\frac{1}{\sqrt{D}} \theta^T \phi^i, y^i \right) + r(\theta) \right\} \quad (6)$$

Formulation contd.

- Given a solution \hat{q} , two ways to solve learning problem:
- Draw D samples $W^1, \dots, W^D \sim \hat{q}$ defining features $\phi^i = [\phi(x^i, w^1) \dots \phi(x^i, w^D)]^T$ and solve :

$$\hat{\theta} = \operatorname{argmin}_{\theta} \left\{ \sum_{i=1}^n c \left(\frac{1}{\sqrt{D}} \theta^T \phi^i, y^i \right) + r(\theta) \right\} \quad (6)$$

- Set $\phi^i = [\phi(x^i, w^1) \dots \phi(x^i, w^{N_w})]^T$ (original random samples from P_0) and directly solve:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \left\{ \sum_{i=1}^n c \left(\theta^T \operatorname{diag}(\hat{q})^{1/2} \phi^i, y^i \right) + r(\theta) \right\} \quad (7)$$

Formulation contd.

- Given a solution \hat{q} , two ways to solve learning problem:
- Draw D samples $W^1, \dots, W^D \sim \hat{q}$ defining features $\phi^i = [\phi(x^i, w^1) \dots \phi(x^i, w^D)]^T$ and solve :

$$\hat{\theta} = \operatorname{argmin}_{\theta} \left\{ \sum_{i=1}^n c \left(\frac{1}{\sqrt{D}} \theta^T \phi^i, y^i \right) + r(\theta) \right\} \quad (6)$$

- Set $\phi^i = [\phi(x^i, w^1) \dots \phi(x^i, w^{N_w})]^T$ (original random samples from P_0) and directly solve:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \left\{ \sum_{i=1}^n c \left(\theta^T \operatorname{diag}(\hat{q})^{1/2} \phi^i, y^i \right) + r(\theta) \right\} \quad (7)$$

- This is a two step approach

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- Formulation
- **Efficient solution**

3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

Efficient solution

- Re-write the optimization problem in (5) as:

$$q^T((\phi y) \circ (\phi y)) \quad (8)$$

Efficient solution

- Re-write the optimization problem in (5) as:

$$q^T((\phi y) \circ (\phi y)) \quad (8)$$

- Also solve (5) via bisection over dual variable λ . Using $\lambda \geq 0$ for constraint $D_f(Q||P_0) \leq \rho$, partial Lagrangian is:

$$\mathcal{L}(q, \lambda) = q^T((\phi y) \circ (\phi y)) - \lambda(D_f(q||1/N_w) - \rho) \quad (9)$$

Efficient solution

- Re-write the optimization problem in (5) as:

$$q^T((\phi y) \circ (\phi y)) \quad (8)$$

- Also solve (5) via bisection over dual variable λ . Using $\lambda \geq 0$ for constraint $D_f(Q||P_0) \leq \rho$, partial Lagrangian is:

$$\mathcal{L}(q, \lambda) = q^T((\phi y) \circ (\phi y)) - \lambda(D_f(q||1/N_w) - \rho) \quad (9)$$

- **Consistency:** Solution to problem (5) approaches a population optimum as data and random sampling increases.
- **Generalization:** Class of estimators used has strong performance guarantees.

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

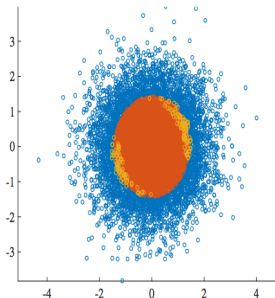
2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

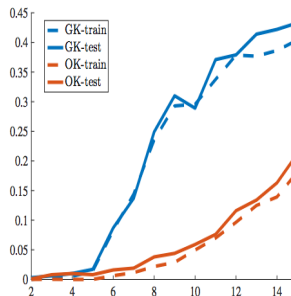
3 Evaluation

- Learning a kernel
- Feature Selection
- Benchmark Datasets

Learning a kernel



(a) Training data & optimized features for $d=2$



(b) Error vs. d

Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

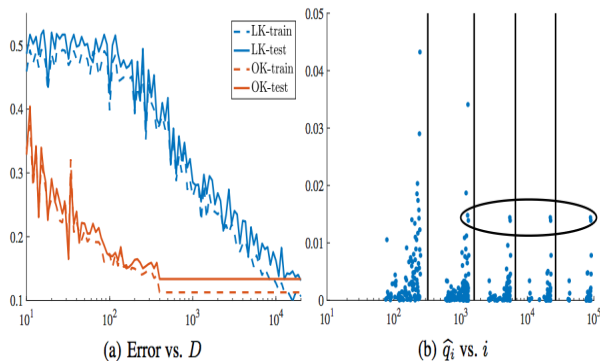
2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

3 Evaluation

- Learning a kernel
- **Feature Selection**
- Benchmark Datasets

Feature Selection



Outline

1 Introduction

- Motivation
- Background
- State-of-the-art

2 Proposed Approach

- Work-flow
- Formulation
- Efficient solution

3 Evaluation

- Learning a kernel
- Feature Selection
- **Benchmark Datasets**

Benchmark Datasets

Table 1: Best test results over benchmark datasets

Dataset	n , n_{test}	d	Model	Our error (%), time(s)		Random error (%), time(s)	
adult	32561, 16281	123	Logistic	15.54,	3.6	15.44,	43.1
reuters	23149, 781265	47236	Ridge	9.27,	0.8	9.36,	295.9
buzz	105530, 35177	77	Ridge	4.92,	2.0	4.58,	11.9

Summary

- Learn a kernel in a supervised manner using random features.
- Demonstrate consistency and generalization of the method.
- Attain competitive results on benchmark datasets with a fraction of training time.
- Future Direction
 - Usefulness of simple optimization methods on random features in speeding up traditionally expensive learning problems.