

Efficient Convex Algorithms for Universal Kernel Learning.

A. Talitckii, B. K. Colbert, M. M. Peet

Arizona State University

atalitck@asu.edu

bkcolbe1@asu.edu

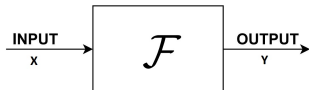
mpeet@asu.edu

NeurIPS 2024

November 24, 2024

Kernel Methods are sensitive to the choice of kernel.

Function Learning Problem



Find F minimizing $L(F)$.

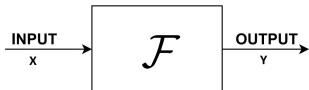
The loss function can be

$$L(F) = \sum_{i=1}^N \max \left\{ 0, \left| y_i - F(x_i) \right| - \varepsilon \right\},$$

where $\{(x_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathbb{R}$.

Kernel Methods are sensitive to the choice of kernel.

Function Learning Problem



Find F minimizing $L(F)$.

The loss function can be

$$L(F) = \sum_{i=1}^N \max \left\{ 0, \left| y_i - F(x_i) \right| - \varepsilon \right\},$$

where $\{(x_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathbb{R}$.

Kernel methods

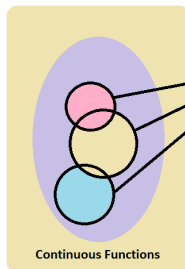
$$\min_{F \in \mathcal{H}_k} L(F)$$

$$\mathcal{H}_k = \left\{ \underbrace{\sum_i k(x, x_i) \alpha_i}_{\text{Reproducing Kernel Hilbert Space}} \mid \alpha_i \in \mathbb{R}, x_i \in \mathcal{X} \right\}$$

$k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is positive-definite kernel.

k is PD kernel if $K = K^T$ and $K \geq 0$

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_N) \\ \vdots & & \ddots \\ k(x_N, x_1) & \dots & k(x_N, x_N) \end{bmatrix}$$



Reproducing Kernel
Hilbert Spaces

$$k_1(x, y)$$

$$k_2(x, y)$$

$$k_3(x, y)$$

Function Space

Kernel Methods search a function for fixed k

How to **optimally** choose a kernel function?

Kernel Learning does not require the choice of Kernel.

Kernel Learning

$$\min_{k \in \mathcal{K}} \min_{F \in \mathcal{H}_k} L(F)$$

$$\mathcal{H}_k = \left\{ \sum_i k(x, x_i) \alpha_i \mid \alpha_i \in \mathbb{R}, x_i \in \mathcal{X} \right\}$$

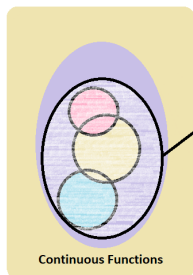
where $\mathcal{K}$ is a set of PD kernels

$$\mathcal{K} = \left\{ k = \int_{\mathcal{Z}} N^T(x, z) P N(y, z) dz \mid \overbrace{\substack{P \geq 0 \\ \text{tr}(P) = n_P}}^{\mathcal{P}} \right\},$$

where $N : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbb{R}^{n_P}$ – chosen functions,

$\mathcal{Z} = [a, b]^{n_Z}$ – integration interval.

$P \in \mathbb{R}^{n_P \times n_P}$ – Positive-Definite matrix.



$$F \in \mathcal{H} := \underbrace{\cup_{k \in \mathcal{K}} \mathcal{H}_k}_{\text{Set of RKHSs}}$$

Set of RKHSs

$$k \in \mathcal{K}$$

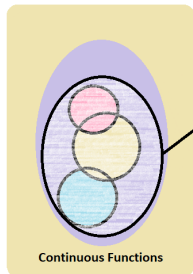
Function Space

Kernel Learning does not require the choice of Kernel.

Kernel Learning

$$\min_{k \in \mathcal{K}} \min_{F \in \mathcal{H}_k} L(F)$$

$$\mathcal{H}_k = \left\{ \sum_i k(x, x_i) \alpha_i \mid \alpha_i \in \mathbb{R}, x_i \in \mathcal{X} \right\}$$



$$F \in \mathcal{H} := \underbrace{\bigcup_{k \in \mathcal{K}} \mathcal{H}_k}_{\text{Set of RKHSs}}$$

Set of RKHSs

$$k \in \mathcal{K}$$

Function Space

where $\mathcal{K}$ is a set of PD kernels

$$\mathcal{K} = \left\{ k = \int_{\mathcal{Z}} N^T(x, z) P N(y, z) dz \mid \overbrace{\substack{P \succeq 0 \\ \text{tr}(P) = n_P}}^{\mathcal{P}} \right\},$$

where $N : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbb{R}^{n_P}$ – chosen functions,

$\mathcal{Z} = [a, b]^{n_z}$ – integration interval.

$P \in \mathbb{R}^{n_P \times n_P}$ – Positive-Definite matrix.

Finite-Dimensional Optimization

$$\min_{P \in \mathcal{P}} \min_{\alpha \in \mathbb{R}^N} L(F_{\alpha, P})$$

$$F_{\alpha, P}(x) = \sum_{i=1}^N k_P(x, x_i) \alpha_i$$

$$k_P(x_1, x_2) = \int_{\mathcal{Z}} N^T(x, z) P N(y, z) dz$$

BUT, the objective is not convex in both variables!

$$L(F_{\alpha, P}) \sim \max \left\{ 0, \underbrace{\left| y_i - \sum_{j=1}^N k_P(x_i, x_j) \alpha_j \right|}_{\text{linear in } k \text{ and linear in } \alpha} - \varepsilon \right\}$$

Direct method to solve[*] $\implies$ Time complexity $O(N^4)$

Q: How to efficiently solve **Kernel Learning** Problem?

[*] G. Lanckriet and et. al. "Learning the kernel matrix with semidefinite programming." *JMLR*, 2004.

How to solve Kernel Learning Problem?

1. **Reformulate** as a saddle-point optimization problem.

$$\mathbf{1} = [1 \quad \dots \quad 1]^T$$

Primal Subproblem

$$\min_{P \in \mathcal{P}} \min_{\alpha \in \mathbb{R}^d} L(F_{\alpha,k})$$

$$F_{\alpha,k}(x) = \sum_{i=1}^N k_P(x, x_i) \alpha_i$$

Dual Subproblem

$$\min_{P \in \mathcal{P}} \max_{\alpha \in \mathcal{A}} L_D(F_{\alpha,P})$$

$$\mathcal{A} = \{\alpha \mid \mathbf{1}^T \alpha = 0, \alpha_i \in [-1, 1]\}$$

$$L_D(F_{\alpha,P}) = \underbrace{\sum_{i=1}^N y_i \alpha_i - \varepsilon \sum_{i=1}^N |\alpha_i| - \frac{1}{C} \sum_{i,j=1}^N k_P(x_i, x_j) \alpha_i \alpha_j}_{\text{concave in } \alpha \text{ and linear in } k \text{ (convex)}}$$

How to solve Kernel Learning Problem?

1. **Reformulate** as a saddle-point optimization problem.

$$\mathbf{1} = [1 \quad \dots \quad 1]^T$$

Primal Subproblem

$$\min_{P \in \mathcal{P}} \min_{\alpha \in \mathbb{R}^d} L(F_{\alpha, k})$$

$$F_{\alpha, k}(x) = \sum_{i=1}^N k_P(x, x_i) \alpha_i$$

Dual Subproblem

$$\min_{P \in \mathcal{P}} \max_{\alpha \in \mathcal{A}} L_D(F_{\alpha, P})$$

$$\mathcal{A} = \{\alpha \mid \mathbf{1}^T \alpha = 0, \alpha_i \in [-1, 1]\}$$

$$L_D(F_{\alpha, P}) = \underbrace{\sum_{i=1}^N y_i \alpha_i - \varepsilon \sum_{i=1}^N |\alpha_i| - \frac{1}{C} \sum_{i,j=1}^N k_P(x_i, x_j) \alpha_i \alpha_j}_{\text{concave in } \alpha \text{ and linear in } k \text{ (convex)}}$$

3. Propose an algorithm.

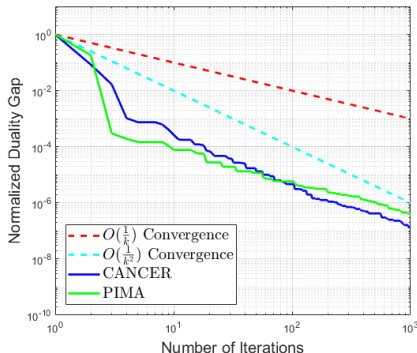
```

Initialize  $P_0 = I$ 
For  $i = 1$  to  $m$  do
    •  $\alpha_i = \arg \max_{\alpha \in \mathcal{A}} L_D(\alpha, P_i)$ 
    •  $S_i = \arg \min_{P \in \mathcal{P}} L_D(\alpha_i, P)$ 
    •  $\gamma_i = \arg \min_{\gamma \in [0, 1]} L_D(\alpha_i, P_i + \gamma(S_i - P_i))$ 
    •  $P_{i+1} = P_i + \gamma_i(S_i - P_i)$ 
enddo
Return  $\alpha_m, P_m$ .
```

$\leftarrow$ SV regression $O(N^{2.3})$
 $\leftarrow$ SDP (using analytic solution)
 $\leftarrow$ line-search for P updates
vs SDP $O(N^4)$
 Time complexity is $\sim O(mN^{2.3})$
 m is number of iterations
 N is size of data set.

Numerical Examples

Convergence of Algorithm



Min-Max Formulation

$$\min_{P \in \mathcal{P}} \max_{\alpha \in \mathcal{A}} L_D(\alpha, P),$$

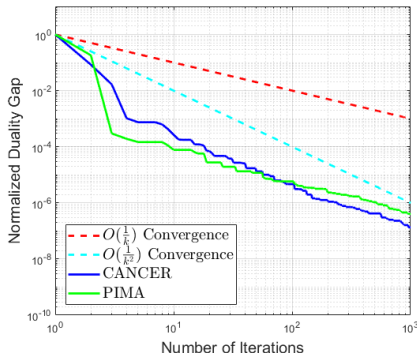
where $L_D(\alpha, P)$ is convex in α
and concave in P .

$$\text{Duality Gap} = \left| \min_{P \in \mathcal{P}} L_D(\alpha, P) - \max_{\alpha \in \mathcal{A}} L_D(\alpha, P) \right|$$

- Just few iterations enough for convergence

Numerical Examples

Convergence of Algorithm



Min-Max Formulation

$$\min_{P \in \mathcal{P}} \max_{\alpha \in \mathcal{A}} L_D(\alpha, P),$$

where $L_D(\alpha, P)$ is **convex** in α
and **concave** in P .

$$\text{Duality Gap} = \left| \min_{P \in \mathcal{P}} L_D(\alpha, P) - \max_{\alpha \in \mathcal{A}} L_D(\alpha, P) \right|$$

- Just few iterations enough for convergence

Kernel Learning for Regression

Data set	Method	Error	Time (s)
Gas $d = 11$ $N = 30000$	Kernel Learning	0.23 ± 0.01	13580 ± 2060
	NNet	0.27 ± 0.03	1172 ± 100
	RF	0.38 ± 0.02	16.44 ± 0.57
	XGBoost	0.33 ± 0.005	49.46 ± 1.93

N – number of data points
 d – dimension of the data set

$$\text{Error} = \sum_{i=1} \|y_i - F(x_i)\|^2$$

Using multiple data sets $\Rightarrow$ KL outperforms other algorithms by **23.6%**

Conclusion Remarks

Conclusion

- **Parameterize kernels** using positive-definite matrices.
- Formulate KL problem as **the saddle point problem**
- Propose the **algorithm for solving Kernel Learning**

More details in the paper.

- **The convergence proof.**
- The time and memory **complexity** of the algorithm
- More numerical experiments
- **Alternative algorithm** for solving Kernel Learning

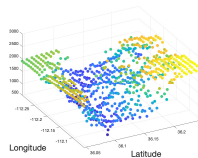
Future Plans

- **Scalability** of Kernel Learning
- New class of kernel function
- Improved algorithm for Kernel Learning

Learning for the Grand Canyon



Data



Fitting

