Scalable Kernel Methods via Doubly Stochastic Gradients

Bo Dai[1], Bo Xie[1], Niao He[1], Yingyu Liang[2], Anant Raj, Maria-Florina Balcan[3], Le Song[1]

Georgia Institute of Technology[1], Princeton University[2], Carnegie Mellon University[3]

Motivation

Goal: scale kernel methods up with provable guarantee.

- Generality and simplicity. Applicable to many kernel methods.
- Efficient computation and low memory requirement.

	Training	Prediction
Computational Cost	$O(\frac{1}{2}dt^2)$	$O(dt)$
Memory Cost	$O(t)$	$O(t)$

- Nonparametric. Model complexity increases when data increases
- Theoretical guarantee. 1), Algorithm updates in infinite-dimension space. 2), converges to the optimal RKHS function in optimal rate $O(\frac{1}{t})$, same as stochastic gradient for strongly convex optimization.

Duality Between Kernels and Random Processes

Theorem (Bochner) *A continuous kernel $k(x,x') = k(x-x')$ on $\mathbb{R}^d$ is PD if and only if $k(x-x')$ is the Fourier transform of a non-negative measure $\mathbb{P}(\omega)$.*

$$k(x-x') = \int_{\mathbb{R}^d} e^{i\omega^\top(x-x')}\, d\mathbb{P}(\omega) = \mathbb{E}_\omega[\phi_\omega(x)\phi_\omega(x')].$$

Theorem *If $k(x,x') = \int_\Omega \phi_\omega(x)\phi_\omega(x')\, d\mathbb{P}(\omega)$ for a non-negative measure $\mathbb{P}(\omega)$ on Ω and $\phi_\omega(x) : \mathcal{X} \mapsto \mathbb{R}$ from $L_2(\Omega, \mathbb{P})$, then $k(x,x')$ is a PD kernel.*

Kernel	$k(x-x')$	$p(\omega)$
Gaussian	$\exp(-\frac{\lVert x-x'\rVert_2^2}{2})$	$2\pi^{-\frac{d}{2}}\exp(-\frac{\lVert\omega\rVert_2^2}{2})$
Laplacian	$\exp(-\lVert x-x'\rVert_1)$	$\prod_{i=1}^d \frac{1}{\pi(1+\omega_i^2)}$
Cauchy	$\prod_{i=1}^d \frac{2}{1+(x_i-x_i')^2}$	$\exp(-\lVert\omega\rVert_1)$

Doubly Stochastic Kernel Machines

Denote $\mathcal{H}$ as the RKHS associated with $k(\cdot,\cdot)$,

$$\text{Kernel Machines: } \operatorname*{argmin}_{f\in\mathcal{H}} R(f) := \mathbb{E}_{(x,y)\sim\mathbb{P}(x,y)}[l(f(x),y)] + \frac{\nu}{2}\lVert f\rVert_{\mathcal{H}}^2$$

The functional gradient $\nabla R(f)$ is defined as,

$$R(f+\epsilon g) = R(f) + \epsilon\,\langle \nabla R(f), g\rangle_{\mathcal{H}} + O(\epsilon^2).$$

Given $(x,y)\sim\mathbb{P}(x,y)$, the stochastic functional gradient of $R(f)$ is

$$g(\cdot) = l'(f(x),y)k(x,\cdot) + \nu f(\cdot) := \xi(\cdot) + \nu f(\cdot).$$

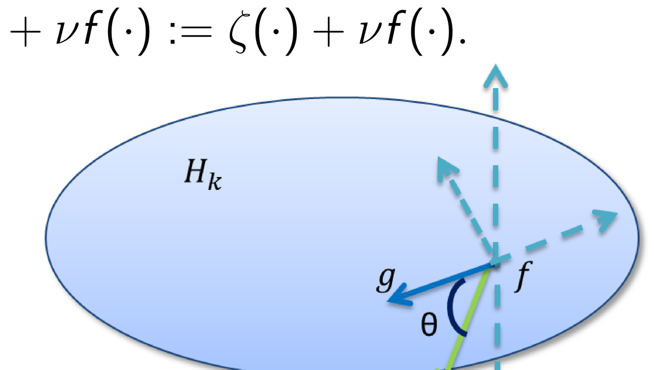
The stochastic functional gradient update is

$$f_{t+1}(\cdot) = f_t(\cdot) - \gamma_t g_t(\cdot) = f_t(\cdot) - \gamma_t[l'(f_t(x_{t+1}), y_{t+1})k(x_{t+1},\cdot) + \nu f_t(\cdot)], \quad (1)$$

$f_{t+1} = \sum_{i=1}^{t+1} a_i k(x_i, \cdot)$ needs all the data, which results $O(dt)$ memory cost.
Let $\omega\sim\mathbb{P}(\omega)$, the doubly stochastic functional gradient of $R(f)$ w.r.t. $f\in\mathcal{H}$ is

$$\tilde{g}(\cdot) = l'(f(x),y)\phi_\omega(x)\phi_\omega(\cdot) + \nu f(\cdot) := \zeta(\cdot) + \nu f(\cdot).$$

$$\xi(\cdot) = \mathbb{E}_\omega[\zeta(\cdot)] \Rightarrow$$
$$\nabla R(f) = \mathbb{E}_{(x,y)}[g(\cdot)] = \mathbb{E}_{(x,y)}\mathbb{E}_\omega[\tilde{g}(\cdot)]$$



Replace the $g(\cdot)$ with $\tilde{g}(\cdot)$ in update (1),

$$f_{t+1}(\cdot) = f_t(\cdot) - \gamma_t[l'(f_t(x_{t+1}), y_{t+1})\phi_{\omega_{t+1}}(x_{t+1})\phi_{\omega_{t+1}}(\cdot) + \nu f_t(\cdot)]$$
$$= (1-\gamma_t\nu)f_t - \gamma_t l'(f_t(x_{t+1}), y_{t+1})\phi_{\omega_{t+1}}(x_{t+1})\phi_{\omega_{t+1}}(\cdot).$$

We have $f_{t+1}(\cdot) = \sum_{i=1}^{t+1}[\beta_i\phi_{\omega_i}(x_i)]\phi_{\omega_i}(\cdot) := \sum_{i=1}^{t+1}\alpha_i\phi_{\omega_i}(\cdot)$ with update rule

$$\alpha_{t+1} = -\gamma_t l'(f(x_{t+1}), y_{t+1})\phi_{\omega_{t+1}}(x_{t+1}), \quad \alpha_j = (1-\gamma_t\nu)\alpha_j,\ j = 1,\ldots,t$$

We only need $O(t)$ memory by saving $\{\alpha_i\}_{i=1}^{t+1}$ and the seed for generating ω.

Algorithm: Doubly SGD

Train: $\{\alpha_i\}_{i=1}^t = \textbf{Train}(\mathbb{P}(x,y))$

Input: $\mathbb{P}(\omega)$, $\phi_\omega(x)$, $l(f(x),y)$, ν.

1: **for** $i = 1,\ldots,t$ **do**
2: Sample $(x_i, y_i)\sim\mathbb{P}(x,y)$.
3: Sample $\omega_i\sim\mathbb{P}(\omega)$ with seed i.
4: $f(x_i) = \textbf{Predict}(x_i, \{\alpha_j\}_{j=1}^{i-1})$.
5: $\alpha_i = -\gamma_i l'(f(x_i), y_i)\phi_{\omega_i}(x_i)$.
6: $\alpha_j = (1-\gamma_i\nu)\alpha_j$, $j = 1,\ldots,i-1$.
7: **end for**

Predict: $f(x) = \textbf{Predict}(x, \{\alpha_i\}_{i=1}^t)$

Input: $\mathbb{P}(\omega)$, $\phi_\omega(x)$.

1: Set $f(x) = 0$.
2: **for** $i = 1,\ldots,t$ **do**
3: Sample $\omega_i\sim\mathbb{P}(\omega)$ with seed i.
4: $f(x) = f(x) + \alpha_i\phi_{\omega_i}(x)$.
5: **end for**

Theoretical Guarantees

Assumption

- There exists an optimal solution to the optimization, denoted as f_*.
- Loss function $\ell(u,y):\mathbb{R}\times\mathbb{R}\to\mathbb{R}$ is L-Lipschitz smooth in terms of u.
- $\forall\{(x_i,y_i)\}_{i=1}^t$, $\exists M>0$, such that $|\ell'(f_i(x_i), y_i)| \leqslant M$.
- $\exists\kappa,\phi>0$, $k(x,x')\leqslant\kappa$, $|\phi_\omega(x)\phi_\omega(x')|\leqslant\phi, \forall x,x'\in\mathcal{X}, \omega\in\Omega$.

Theorem *When $\gamma_t = \frac{\theta}{t}$ with $\theta>0$ such that $\theta\nu\in\mathbb{Z}_+$, $\forall x\in\mathcal{X}$, we have,*

$$\mathbb{E}_{\mathcal{D}^t,\omega^t}\left[|f_{t+1}(x) - f_*(x)|^2\right] \leqslant \frac{C_1}{t},$$

$$|f_{t+1}(x) - f_*(x)|^2 \leqslant \frac{C_2\ln(2t/\delta)\ln^2(t)}{t}, \text{ with probability } 1-3\delta,$$

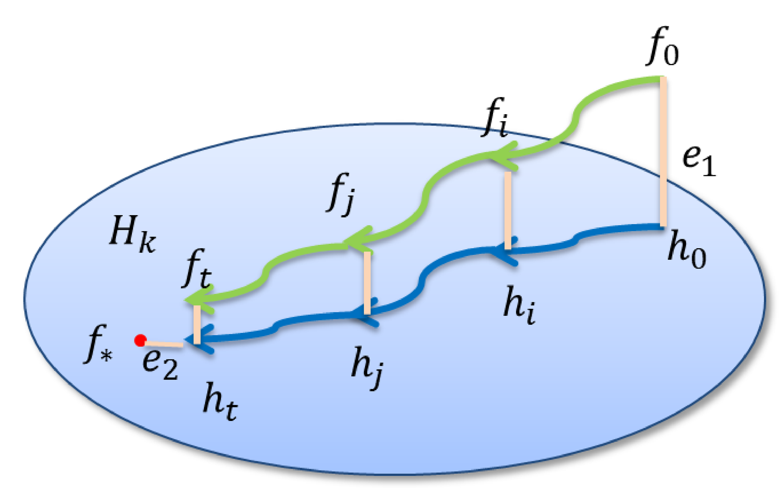
and the generalization error $R_{true}(f) = \mathbb{E}_{(x,y)}[l(f(x),y)]$,

$$R_{true}(f_{t+1}) - R_{true}(f_*) \leqslant \frac{C_3\sqrt{\ln(2t/\delta)}\ln(t)}{\sqrt{t}}, \text{ with probability } 1-3\delta,$$

where C_1, C_2 and C_3 only depend on $\theta,\nu,\kappa,\phi,L,M$,

Proof idea The main technical difficulty is that f_{t+1} may not be in the RKHS $\mathcal{H}$. The key of the proof is constructing an intermediate function $h_{t+1}\in\mathcal{H}$,

$$h_{t+1}(\cdot) = h_t(\cdot) - \gamma_t(\xi_t(\cdot) + \nu h_t(\cdot)) = \sum_{i=1}^t \beta_t^i\xi_i(\cdot), \quad \forall t>1 \quad \text{and } h_1(\cdot) = 0.$$

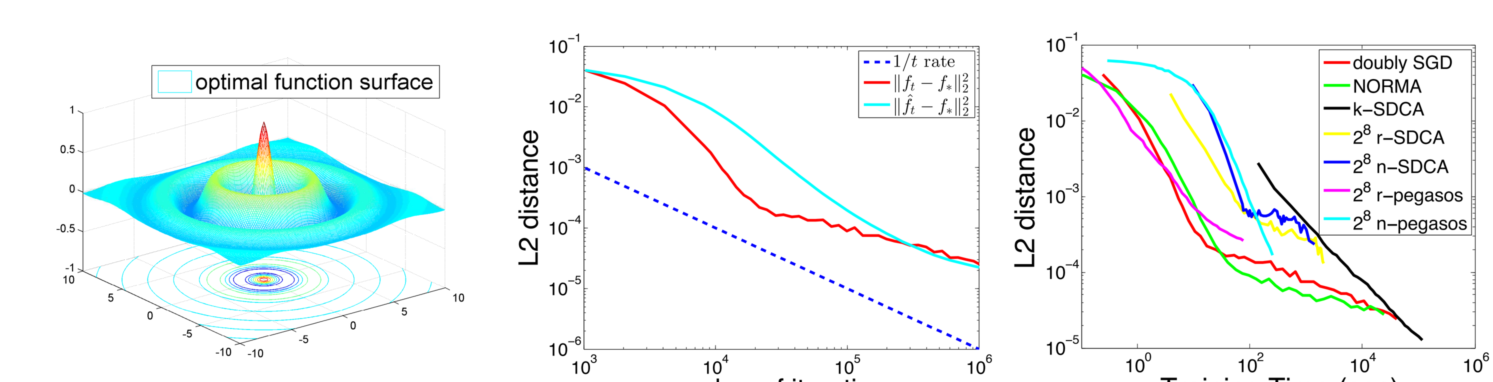


The error can be decomposed as two terms

$$|f_{t+1}(x) - f_*(x)|^2 \leqslant 2\underbrace{|f_{t+1}(x) - h_{t+1}(x)|^2}_{e_1 = \text{ error due to random functions}} + 2\kappa\underbrace{\lVert h_{t+1} - f_*\rVert_{\mathcal{H}}^2}_{e_2 = \text{error due to random data}}$$

Verification for Convergence Rate

We verify the rate of convergence on a synthetic dataset with 2^{20} data using kernel ridge regression.



(1) Synthetic Dataset (2) Convergence Rate (3) Accuracy vs. Time

Experimental results for kernel ridge regression on synthetic dataset.

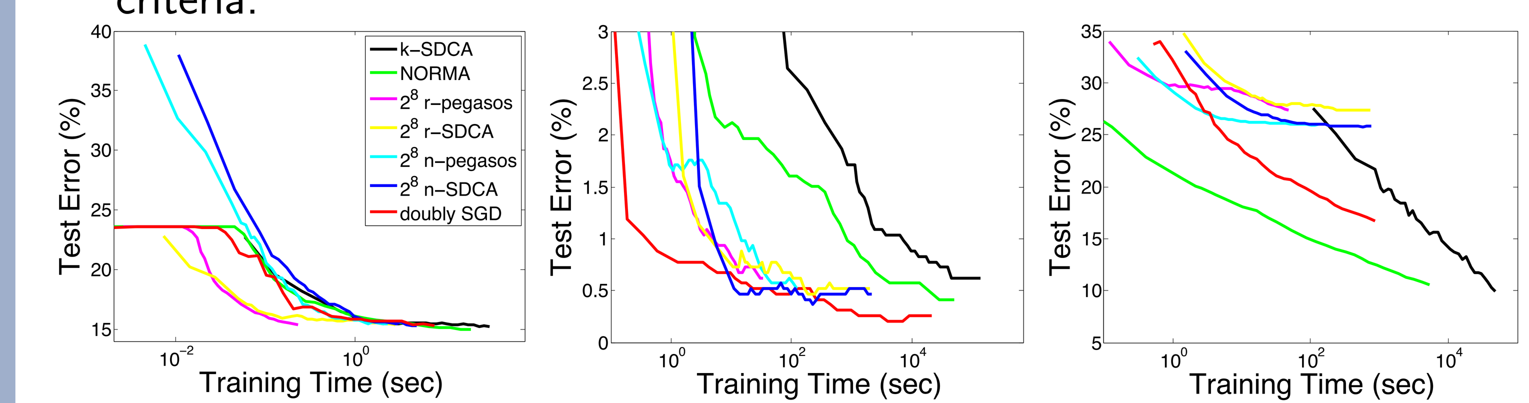
Computation, Memory and Statistics Trade-off

We fix $\lVert f - f_*\rVert_2^2 \leqslant \epsilon$, and assume that the number of samples, $n = O(1/\epsilon)$. The number of random features/ranks r will be $O(n) = O(1/\epsilon)$.

Algorithms	Preprocessing	Computation Cost		Memory Cost	
	Computation	Training	Prediction	Training	Prediction
Doubly SGD	$O(1)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(1/\epsilon)$	$O(1/\epsilon)$
NORMA	$O(1)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(d/\epsilon)$	$O(d/\epsilon)$
k-SDCA	$O(1)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(d/\epsilon)$	$O(d/\epsilon)$
r-Pegasos	$O(1)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(1/\epsilon)$	$O(1/\epsilon)$
r-SDCA	$O(1)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(1/\epsilon)$	$O(1/\epsilon)$
n-Pegasos	$O(1/\epsilon^3)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(1/\epsilon)$	$O(1/\epsilon)$
n-SDCA	$O(1/\epsilon^3)$	$O(d/\epsilon^2)$	$O(d/\epsilon)$	$O(1/\epsilon)$	$O(1/\epsilon)$

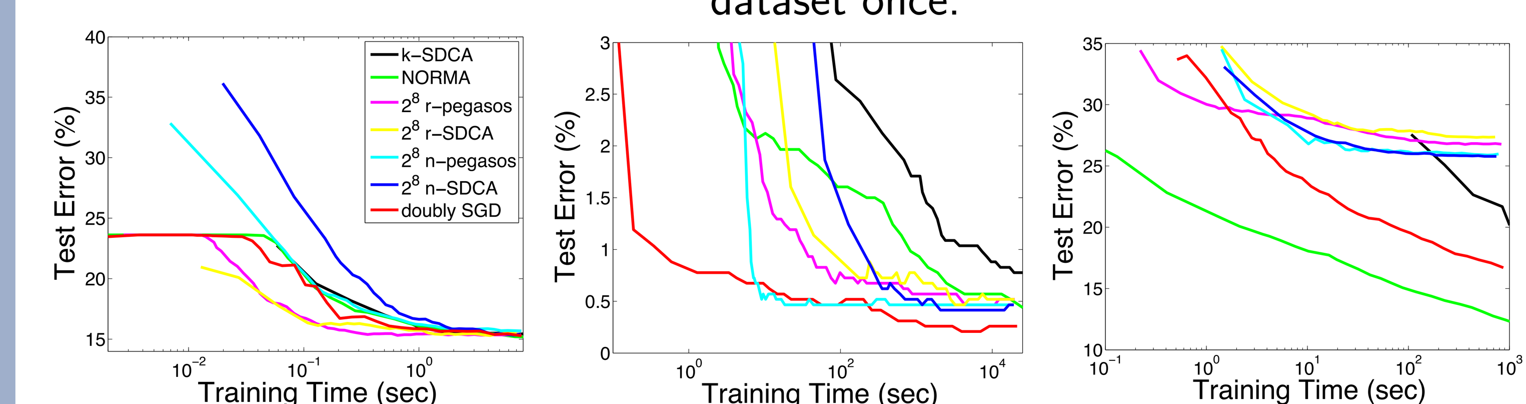
Experiments on Real Datasets

- Comparison with kernel SVM solvers on classification datasets with two criteria.



Adult MNIST 8M 8 vs. 6 Forest

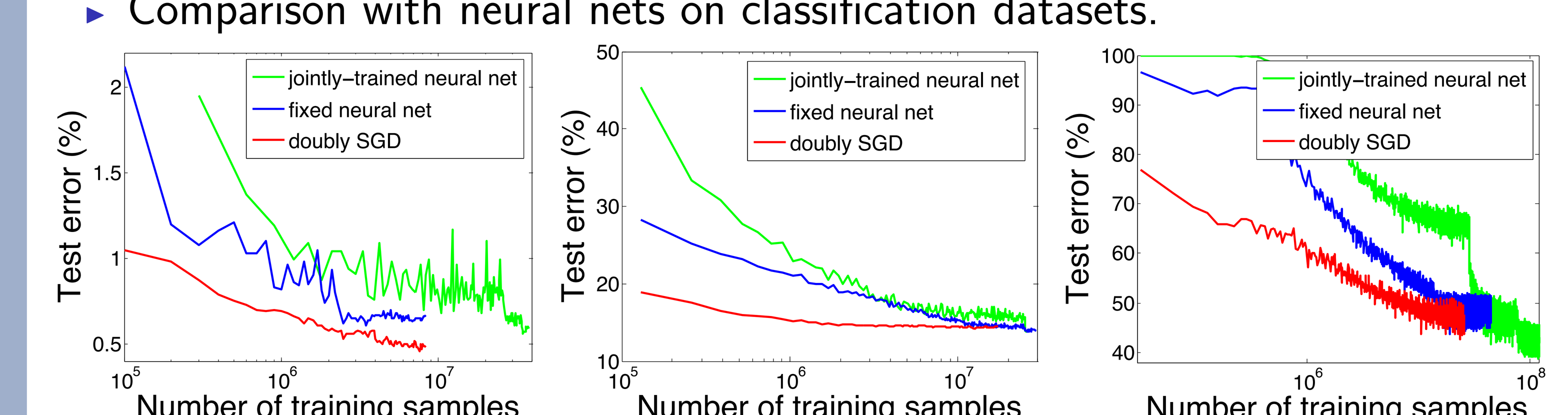
Stopping Criterion 1: Stop algorithms when they pass through the entire dataset once.



Adult MNIST 8M 8 vs. 6 Forest.

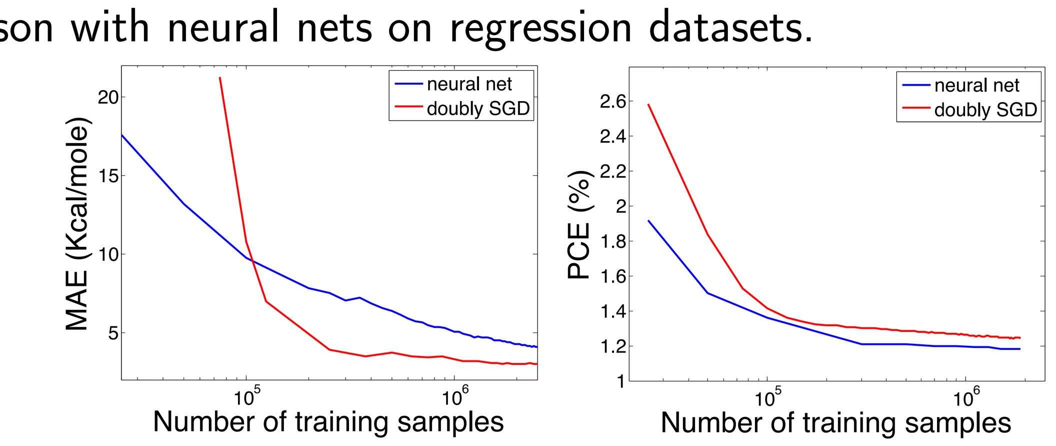
Stopping Criterion 2: Stop algorithms within the same time budget.

- Comparison with neural nets on classification datasets.



MNIST 8M CIFAR 10 ImageNet

- Comparison with neural nets on regression datasets.



QuantumMachine MolecularSpace.