Parsimonious Online Learning with Kernels via Sparse Projections in Function Space

Alec Koppel*, Garrett Warnell†, Ethan Stump†, and Alejandro Ribeiro* Contact E-mail: {akoppel, aribeiro}@seas.upenn.edu, {garrett.a.warnell.civ, ethan.a.stump2.civ}@mail.mil
* Department of Electrical and Systems Engineering, University of Pennsylvania, † Computational and Information Sciences Directorate, U.S. Army Research Laboratory

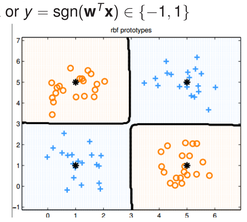
Penn — University of Pennsylvania

Parametric and Nonparametric Regression

- Supervised learning $\Rightarrow \mathbf{w}^* \in \mathbb{R}^p$ minimizing expected risk $L(\mathbf{w})$
- $\ell : \mathcal{W} \to \mathbb{R}$ $\Rightarrow$ convex loss ($\mathcal{W} \subset \mathbb{R}^p$), merit of statistical model
  - $\Rightarrow (\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$ is random pair $\Rightarrow$ training examples

$$\mathbf{w}^* := \operatorname*{argmin}_{\mathbf{w}} L(\mathbf{w}) := \operatorname*{argmin}_{\mathbf{w}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{w}; \mathbf{x}, \mathbf{y})]$$

- E.g., support vector machines, logistic regression, matrix completion
- Above formulation only works for *linear statistical models*
  - $\Rightarrow y = \mathbf{w}^T\mathbf{x} \in \mathbb{R}$ or $y = \text{sgn}(\mathbf{w}^T\mathbf{x}) \in \{-1, 1\}$



- Nonparametric regression: greater flexibility/smaller model bias
  - $\Rightarrow y = f(\mathbf{x})$ for some function $f \in \mathcal{H}$ $\Rightarrow$ nonlinear classifier

$$f^* = \operatorname*{argmin}_{f \in \mathcal{H}} R(f) := \operatorname*{argmin}_{f \in \mathcal{H}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)] + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$$

- Generally intractable $\Rightarrow$ need to restrict function class $\mathcal{H}$

Reproducing Kernels and Function Representation

- Reproducing kernels $\Rightarrow$ make function estimation possible!
- Equip $\mathcal{H}$ with a unique *kernel function*, $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, such that:

$$(i)\ \langle f, \kappa(\mathbf{x}, \cdot)\rangle_{\mathcal{H}} = f(\mathbf{x}) \quad \text{for all } \mathbf{x} \in \mathcal{X}\,,$$
$$(ii)\ \mathcal{H} = \overline{\text{span}\{\kappa(\mathbf{x}, \cdot)\}} \quad \text{for all } \mathbf{x} \in \mathcal{X}\,.$$

- Property (i) $\Rightarrow$ source of "kernel trick:"
  - $\Rightarrow$ define nonlinear map $\phi(\mathbf{x}) = \kappa(\mathbf{x}, \cdot)$ of feature vector $\mathbf{x}$
  - $\Rightarrow$ accessed only via inner products $\langle \phi(\mathbf{x}), \phi(\mathbf{x}')\rangle_{\mathcal{H}} = \kappa(\mathbf{x}, \mathbf{x}')$
- Property (ii) $\Rightarrow$ Representation Thms. from functional analysis
  - $\Rightarrow$ Functions $f \in \mathcal{H}$ are kernel expansions over data: $f(\mathbf{x}) = \sum_n w_n \kappa(\mathbf{x}_n, \mathbf{x})$
- Examples $\Rightarrow$ Gaussian/RBF $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|_2^2}{2\sigma^2}\right)$
  - $\Rightarrow$ Polynomial $\kappa(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T\mathbf{x}' + b)^c$
  - $\Rightarrow$ Sinc $\kappa(\mathbf{x}, \mathbf{x}') = \text{sinc}(\mathbf{x} - \mathbf{x}')$

Curse of Kernelization and Need for Sparsification

- Consider empirical risk min. with sample size $N < \infty$ with Rep. Thm:

$$f^* = \operatorname*{argmin}_f \frac{1}{N}\sum_{n=1}^{N} \ell(f(\mathbf{x}_n), \mathbf{y}_n) \text{ takes the form } f(\mathbf{x}) = \sum_{n=1}^{N} w_n\, \kappa(\mathbf{x}_n, \mathbf{x})\,.$$

  - $\Rightarrow \mathbf{x}_n$ $\Rightarrow$ feature vectors, w_n $\Rightarrow$ scalar weight.
- Representer Thm. into ERM $\Rightarrow$ opt. over $\mathcal{H}$ reduces to $\mathbf{w} \in \mathbb{R}^N$

$$f^* = \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N} \ell(\sum_{m=1}^{N} w_m \kappa(\mathbf{x}_m, \mathbf{x}_n), y_n) + \frac{\lambda}{2}\|\sum_{n,m=1}^{N} w_n w_m \kappa(\mathbf{x}_m, \mathbf{x}_n)\|_{\mathcal{H}}^2$$
$$= \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N} \ell(\mathbf{w}^T\kappa_{\mathbf{X}}(\mathbf{x}_n), y_n) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{K}_{\mathbf{X},\mathbf{X}}\mathbf{w}$$

- As sample size $N \to \infty$, $N \times N$ kernel matrix $[\mathbf{K}_{\mathbf{X},\mathbf{X}}]_{m,n} := \kappa(\mathbf{x}_m, \mathbf{x}_n)$
  - $\Rightarrow \mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \cdots]$ $\Rightarrow$ kernel dictionary
  - $\Rightarrow \kappa_{\mathbf{X}}(\cdot) = [\kappa(\mathbf{x}_1, \cdot) \ldots \kappa(\mathbf{x}_N, \cdot)]^T$ $\Rightarrow$ empirical kernel map
- Model order $M (= N) \to \infty$ $\Rightarrow$ number of dictionary columns
- **Storage/representation issue** $\Rightarrow$ **"the curse of kernelization"**

Functional Stochastic Gradient Method

- Compute stochastic functional gradient (variation) of $L(f)$:

$$\nabla_f \ell(f(\mathbf{x}_t), y_t)(\cdot) = \frac{\partial \ell(f(\mathbf{x}_t), y_t)}{\partial f(\mathbf{x}_t)} \frac{\partial f(\mathbf{x}_t)}{\partial f}(\cdot)$$

  - $\Rightarrow$ where we apply the chain rule above
- Use reproducing property of kernel (i), differentiate both sides:

$$\frac{\partial f(\mathbf{x}_t)}{\partial f} = \frac{\partial \langle f, \kappa(\mathbf{x}_t, \cdot)\rangle_{\mathcal{H}}}{\partial f} = \kappa(\mathbf{x}_t, \cdot)$$

- SGD applied to R(f), given independent training example $(\mathbf{x}_t, \mathbf{y}_t)$:

$$f_{t+1} = (1 - \eta_t\lambda)f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)$$
$$= (1 - \eta_t\lambda)f_t - \eta_t \ell'(f(\mathbf{x}_t), y_t)\kappa(\mathbf{x}_t, \cdot)$$

- Newest feature vector $\mathbf{x}_t$ enters kernel dictionary $\mathbf{X}_t$
  - $\Rightarrow$ with associated weight $\ell'(f(\mathbf{x}_t), y_t) := \partial \ell(f_t(\mathbf{x}_t), y_t)/\partial f_t(\mathbf{x}_t)$
- FSGD: parametric updates on weights and dictionary

$$\mathbf{X}_{t+1} = [\mathbf{X}_t,\ \mathbf{x}_t], \quad \mathbf{w}_{t+1} = [(1 - \eta_t\lambda)\mathbf{w}_t,\ -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)]\,,$$

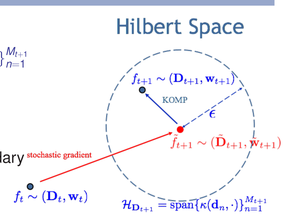
  - $\Rightarrow$ Note that model order $M_t = t - 1$ grows by one at each step

Controlling Model Order

- We want to learn f^* in such a way that $M_t << \infty$ for each f_t
- Accomplished by fixing a error nbhd. around SGD iterates
  - $\Rightarrow$ Remove maximal no. kernel dict. elements while inside nbhd.
- We propose using KOMP $\Rightarrow$ kernel orthogonal matching pursuit
  - $\Rightarrow$ a greedy compressive technique (Vincent & Bengio, 2002)
- Why greedy? $\Rightarrow$ need error nbhd. guarantee for convergence
- Lasso/convex methods require conditions on kernel matrix
  - $\Rightarrow$ mutual incoherence, restricted isometry, etc. for error bound
  - $\Rightarrow$ will not be satisfied for many practical cases

Greedily Orthogonal Subspace Projections

- Fix approximation error ϵ_t
- Define subspace $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_{t+1}}$
- Model points $\{\mathbf{d}_n\} \subset \{\mathbf{x}_u\}_{u \le t}$
  - $\Rightarrow$ subset of past training examples
- Sequentially remove kernel dictionary elements
  - $\Rightarrow$ Stop at Hilbert error nbhd. boundary
- Stopping criterion: $\|\tilde{f}_{t+1} - f_{t+1}\|_{\mathcal{H}} \le \epsilon_t$
- New model order: $M_{t+1} \le M_t + 1$



Kernel Orthogonal Matching Pursuit

Require: function $\tilde{f}$ via dict. $\tilde{\mathbf{D}} \in \mathbb{R}^{p \times \tilde{M}}$, coeffs. $\tilde{\mathbf{w}} \in \mathbb{R}^{\tilde{M}}$, approx. budget $\epsilon_t > 0$
Initialize $f = \tilde{f}$, dict. $\mathbf{D} = \tilde{\mathbf{D}}$ (indices $\mathcal{I}$), model order $M = \tilde{M}$, coeffs. $\mathbf{w} = \tilde{\mathbf{w}}$.
  while candidate dictionary is non-empty $\mathcal{I} \neq \emptyset$ **do**
    for $j = 1, \ldots, \tilde{M}$ **do**
      Find minimal approx. error with dict. element $\mathbf{d}_j$ removed

$$\gamma_j = \min_{\mathbf{w}_{\mathcal{I}\setminus\{j\}} \in \mathbb{R}^{M-1}} \|\tilde{f}(\cdot) - \sum_{k \in \mathcal{I}\setminus\{j\}} w_k \kappa(\mathbf{d}_k, \cdot)\|_{\mathcal{H}}\,.$$

    end for
    Find dictionary index minimizing approx. error: $j^* = \text{argmin}_{j \in \mathcal{I}} \gamma_j$
    if minimal approximation error exceeds threshold $\gamma_{j^*} > \epsilon_t$
      stop
    else
      Prune dictionary $\mathbf{D} \leftarrow \mathbf{D}_{\mathcal{I}\setminus\{j^*\}}$, revise $\mathcal{I} \leftarrow \mathcal{I} \setminus \{j^*\}$
      Revise model order $M \leftarrow M - 1$; compute $\mathbf{w}$ defined by $\mathbf{D}$

$$\mathbf{w} = \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^M} \|\tilde{f}(\cdot) - \mathbf{w}^T\kappa_{\mathbf{D}}(\cdot)\|_{\mathcal{H}}$$

    end
  end while
  return $f, \mathbf{D}, \mathbf{w}$ of model order $M \le \tilde{M}$ such that $\|f - \tilde{f}\|_{\mathcal{H}} \le \epsilon_t$

Parsimonious Online Learning with Kernels

- Define un-projected/unsparsified iterate at step $t + 1$

$$\tilde{f}_{t+1} = (1 - \eta_t\lambda)f_t - \eta_t \nabla_f \ell(f_t; \mathbf{x}_t, \mathbf{y}_t).$$

  - $\Rightarrow$ parameterized by dictionary and coefficients

$$\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t,\ \mathbf{x}_t], \qquad \tilde{\mathbf{w}}_{t+1} = [(1 - \eta_t\lambda)\mathbf{w}_t,\ -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)]\,.$$

- Our method: $(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{KOMP}(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$
- This amounts to a certain orthogonal subspace projection

$$f_{t+1} = \operatorname*{argmin}_{f \in \mathcal{H}_{\mathbf{D}_{t+1}}} \left\| f - \left((1 - \eta_t\lambda)f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)\right)\right\|_{\mathcal{H}}^2$$
$$:= \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\left[(1 - \eta_t\lambda)f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)\right].$$

- Recall the Hilbert subspace $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_{t+1}}$
  - $\Rightarrow \mathbf{d}_n$ are model points $\Rightarrow$ subset of past feature vectors $\{\mathbf{x}_u\}_{u \le t}$

Convergence Results

Theorem: *The POLK sequence f_t run with diminishing step-size $\sum_{t=1}^{\infty} \eta_t = \infty$, $\sum_{t=1}^{\infty} \eta_t^2 < \infty$, $\epsilon_t = \eta_t^2$, $\eta_0 < \lambda$ converges almost surely to the optimum f^**

$$\lim_{t \to \infty} \|f_t - f^*\|_{\mathcal{H}}^2 = 0 \qquad a.s.$$

- Compression budget goes to null, model order might grow arbitrarily large

Theorem: *The POLK sequence f_t with $f_0 = 0$, using regularizer $\lambda > 0$, constant algorithm step-size such that $\eta < 1/\lambda$, and constant compression budget $\epsilon = K\eta^{3/2} = \mathcal{O}(\eta^{3/2})$, $K > 0$ is a scaler, converges to a neighborhood*

$$\liminf_{t \to \infty} \|f_t - f^*\|_{\mathcal{H}} \le \frac{\sqrt{\eta}}{\lambda}\left(K + \sqrt{K^2 + \lambda\sigma^2}\right) = \mathcal{O}(\sqrt{\eta}) \qquad a.s.$$

- Convergence to a nbhd. with constant compression budget, learning rate

Theorem: *The POLK sequence f_t with constant step-size $\eta_t = \eta < 1/\lambda$, approx. budget $\epsilon = K\eta^{3/2}$ has model order M_t that is always finite, i.e., $\max M_t \le M^{\infty} < \infty$*

- Model order of limiting function $f^{\infty} = \lim_t f_t$ is always finite
- M^{∞} depends on $(K\sqrt{\eta})/(C)$
  - $\Rightarrow$ KOMP criterion, step-size η, constant K, Lipschitz mod. of ℓ

Online Multi-class Kernel Support Vector Machines

- Given feature vectors $\mathbf{x}$ with labels $y \in \{1, \ldots, C\}$
  - $\Rightarrow$ Classifier $\Rightarrow$ optimize a geometric criterion of separation
- Multi-class SVM $\Rightarrow$ for each class c a class-specific function $f_c : \mathcal{X} \to \mathbb{R}$.
  - $\Rightarrow$ Classify $\mathbf{x}$ $\Rightarrow$ max class-conditional prob. $y = \max_{y'} f_{y'}(\mathbf{x})$.
- Define vectorized function $\mathbf{f} = [f_1, \ldots, f_C] \in \mathcal{H}^C$.
  - $\Rightarrow$ Want to minimize λ-regularized multi-class hinge loss

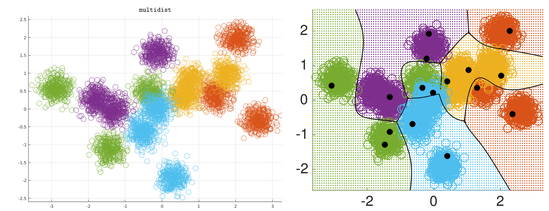
$$\mathbf{f}^* = \operatorname*{argmin}_{\mathbf{f}} \frac{1}{N}\sum_{n=1}^{N} \ell(\mathbf{f}(\mathbf{x}_n), y_n) + \lambda \sum_{c'=1}^{C} \|f_{c'}\|_{\mathcal{H}}^2\,,$$

  - $\Rightarrow \ell(\mathbf{f}(\mathbf{x}), y) = \max(0, 1 + f_r(\mathbf{x}) - f_y(\mathbf{x}))$, $r = \text{argmax}_{c' \neq y} f_{c'}(\mathbf{x})$.
  - $\Rightarrow$ Like normal multi-class SVM, uses $f_r(\mathbf{x})$ rather than inner prod. $\mathbf{w}_r^T\mathbf{x}$

Parameter selection:
- $\Rightarrow$ Gaussian kernel with bandwidth $\tilde{\sigma}^2 = 0.6$
- $\Rightarrow$ regularizer $\lambda = 10^{-6}$
- $\Rightarrow$ constant learning rate $\eta = 6.0$
- $\Rightarrow$ approximation budget $\epsilon = K\eta^{3/2}$
- $\Rightarrow$ parsimony constant $K = 0.04$
- $\Rightarrow$ Initialize kernel classifier as null, i.e., $f_0 = 0$

Gaussian Mixtures: Efficient Learning Class Conditional Densities

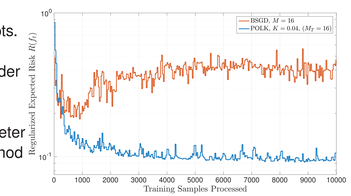
- Case where training examples for a fixed class
  - $\Rightarrow$ drawn from a distinct Gaussian mixture
- 3 Gaussians per mixture, $C = 5$ classes total for this experiment
  - $\Rightarrow$ 15 total Gaussians generate data



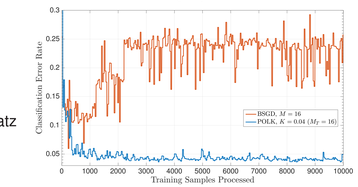
- Grid colors $\Rightarrow$ decision, bold black dots $\Rightarrow$ kernel dict. elements
- $\sim$ 97% accuracy

Empirical Convergence

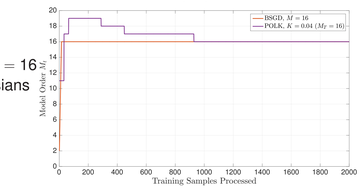
- $N = 5000$ training, 2500 test pts.
- Train for many epochs
  - $\Rightarrow$ maintain small model order
- Algorithm run with
  - $\Rightarrow$ constant step-size
  - $\Rightarrow$ constant sparsity parameter
- Comparison to SVM-only method
  - $\Rightarrow$ fixes model order
- POLK exhibits convergence



- POLK $\Rightarrow$ high test accuracy
  - $\Rightarrow$ error rate around 2.6%
- Competitor diverges
- Similar acc. on MNIST, Brodatz



- POLK *learns* model order $M_T = 16$
  - $\Rightarrow$ true data $\Rightarrow$ 15 Gaussians



Conclusions

- We focus on nonparametric regression problems
  - $\Rightarrow$ discuss how they require infinite memory for streaming data
- We propose a new algorithm based on functional stochastic gradient
  - $\Rightarrow$ operating in tandem with greedy compression via matching pursuit
- We establish exact convergence w.p.1
  - $\Rightarrow$ with diminishing learning rate, approximation budget
- We establish convergence w.p.1 to a neighborhood
  - $\Rightarrow$ with constant learning rate, specific approx. budget
- Numerical examples yield promising pattern recognition results
  - $\Rightarrow$ interpretable decision surfaces $\Rightarrow$ class-conditional densities