Online and Decentralized Statistical Learning

Alec Koppel
University of Pennsylvania, Philadelphia, PA

Phd Committee:
Alejandro Ribeiro (Advisor), Vijay Kumar (Chair),
Brian M. Sadler, Jonathan Fink

Phd Proposal

Philadelphia, PA, Dec. 1, 2016

Statistical Inference and Optimization

- Consider $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$ as random pair $\Rightarrow$ training examples
- Examples: classification $\mathbf{y} \in \{1, \ldots, c\}$ or regression $\mathbf{y} \in \mathbb{R}$

  $\Rightarrow$ Perform prediction $\Rightarrow$ optimize statistical inference accuracy

$$\hat{\mathbf{y}}^*(\mathbf{x}) := \underset{\hat{\mathbf{y}}(\mathbf{x})}{\operatorname{argmin}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\mathbb{I}\{\hat{\mathbf{y}}(\mathbf{x}) \neq \mathbf{y})\}]$$

- In general intractable $\Rightarrow$ distribution $\mathbb{P}(\mathbf{x}, \mathbf{y})$ is unknown

  $\Rightarrow$ but gives clear merit for choosing estimator $\hat{\mathbf{y}}$

  $\Rightarrow$ $\hat{\mathbf{y}}^*(\mathbf{x})$ achieves optimal Bayes Risk

Expected Risk Minimization

- Since optimizing statistical error rate directly is intractable
  - $\Rightarrow$ Replace $0-1$ loss $\Rightarrow$ convex loss $\ell : \mathcal{W} \to \mathbb{R}$, $\mathcal{W} \subset \mathbb{R}^p$,
  - $\Rightarrow$ Estimator $\hat{\mathbf{y}}(\mathbf{x}) = \hat{\mathbf{y}}(\mathbf{w}, \mathbf{x})$ depends on model parameters $\mathbf{w}$

$$\mathbf{w}^* := \operatorname*{argmin}_{\mathbf{w}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{w}; \mathbf{x}, \mathbf{y})] = \frac{1}{N} \sum_{n=1}^{N} \ell(\mathbf{w}; \mathbf{x}_n, \mathbf{y}_n)$$

- Can solve optimization problem, define $L(\mathbf{w}) = \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{w}; \mathbf{x}, \mathbf{y})]$
  - $\Rightarrow$ but unclear how to choose $\hat{\mathbf{y}}(\mathbf{w}, \mathbf{x})$ such that $\hat{\mathbf{y}}(\mathbf{w}^*, \mathbf{x}) \approx \hat{\mathbf{y}}^*$

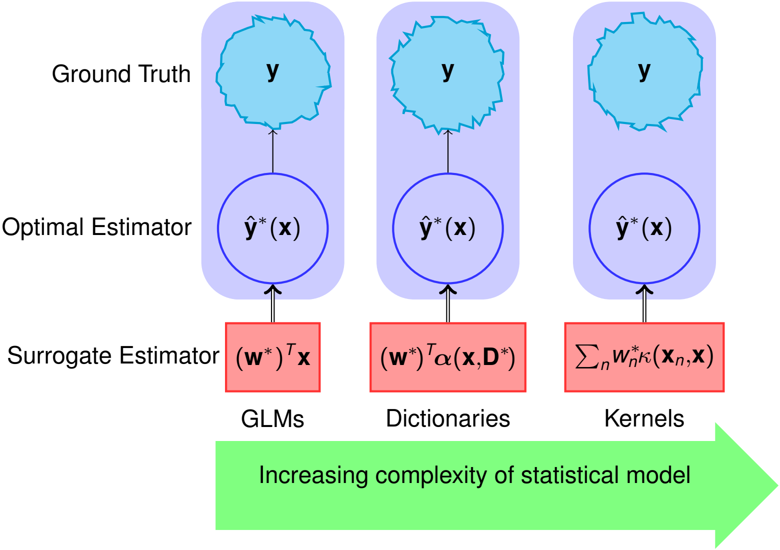
Expected Risk Minimization

- Since optimizing statistical error rate directly is intractable
  - $\Rightarrow$ Replace $0-1$ loss $\Rightarrow$ convex loss $\ell : \mathcal{W} \to \mathbb{R}$, $\mathcal{W} \subset \mathbb{R}^p$,
  - $\Rightarrow$ Estimator $\hat{\mathbf{y}}(\mathbf{x}) = \hat{\mathbf{y}}(\mathbf{w}, \mathbf{x})$ depends on model parameters $\mathbf{w}$

$$\mathbf{w}^* := \underset{\mathbf{w}}{\operatorname{argmin}}\, \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{w}; \mathbf{x}, \mathbf{y})] = \frac{1}{N} \sum_{n=1}^{N} \ell(\mathbf{w}; \mathbf{x}_n, \mathbf{y}_n)$$

- Can solve optimization problem, define $L(\mathbf{w}) = \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{w}; \mathbf{x}, \mathbf{y})]$
  - $\Rightarrow$ but unclear how to choose $\hat{\mathbf{y}}(\mathbf{w}, \mathbf{x})$ such that $\hat{\mathbf{y}}(\mathbf{w}^*, \mathbf{x}) \approx \hat{\mathbf{y}}^*$
- We'll present approaches to indirectly optimizing accuracy
  - $\Rightarrow$ Make use of surrogate losses for tractable model training
  - $\Rightarrow$ Higher complexity $\hat{\mathbf{y}}(\mathbf{w}^*, \mathbf{x})$ $\Rightarrow$ harder training, closer to $\hat{\mathbf{y}}^*$?
  - $\Rightarrow$ Therefore, progressively increase complexity of $\hat{\mathbf{y}}(\mathbf{w}, \mathbf{x})$
- Focus: streaming data settings, multi-agent systems

Statistical Estimators



Generalized Linear Models

Large Scale Learning with GLMs

- Recall ERM prob. $\Rightarrow$ loss ℓ, parameter vector $\mathbf{w} \in \mathbb{R}^p$

$$\mathbf{w}^* := \underset{\mathbf{w}}{\text{argmin}}\, \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{w};\mathbf{x},\mathbf{y})] = \underset{\mathbf{w}}{\text{argmin}} \sum_{n=1}^{N} [\ell(\mathbf{w};\mathbf{x}_n,\mathbf{y}_n)]$$

$\Rightarrow$ Alternative measure of performance is regret

$$\mathbf{Reg}_N = \sum_{n=1}^{N} \ell(\mathbf{w}_n;\mathbf{x}_n,\mathbf{y}_n) - \sum_{n=1}^{N} \ell(\mathbf{w}^*;\mathbf{x}_n,\mathbf{y}_n)$$

- Goal: asymptotic no-regret $\mathbf{Reg}_N/N \to 0$
  - $\Rightarrow$ benefit of this perspective is dropping i.i.d. assumption
  - $\Rightarrow$ $\mathbf{Reg}_N/N \approx$ time-average sub-optimality

Large Scale Learning with GLMs

- Recall ERM prob. $\Rightarrow$ loss ℓ, parameter vector $\mathbf{w} \in \mathbb{R}^p$
  $\Rightarrow$ Alternative measure of performance is regret

$$\mathbf{Reg}_N = \sum_{n=1}^{N} \ell(\mathbf{w}_n; \mathbf{x}_n, \mathbf{y}_n) - \sum_{n=1}^{N} \ell(\mathbf{w}^*; \mathbf{x}_n, \mathbf{y}_n)$$

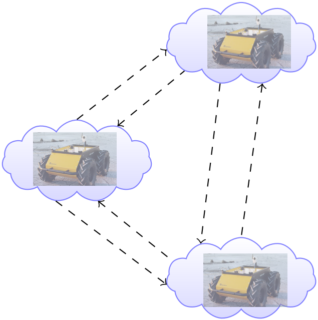
- Goal: asymptotic no-regret $\mathbf{Reg}_N / N \to 0$
  $\Rightarrow$ benefit of this perspective is dropping i.i.d. assumption
  $\Rightarrow$ $\mathbf{Reg}_N / N \approx$ time-average sub-optimality
- N large/data arrives online $\Rightarrow$ can't compute full gradient of $L(\mathbf{w})$
  $\Rightarrow$ Classically solved with stochastic (online) gradient

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla_{\mathbf{w}} \ell(\mathbf{w}_t; \mathbf{x}_t, \mathbf{y}_t) \ , \qquad \ell_t(\mathbf{w}) := \ell(\mathbf{w}; \mathbf{x}_t, \mathbf{y}_t)$$

- Descend w/ stoch. grad. rather than grad. $\Rightarrow$ one sample/time
- One can establish that $w_t \to \mathbf{w}^*$ a.s. and $\mathbf{Reg}_N / N \to 0$

Data Distributed Across a Network

- Network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
  $\Rightarrow |\mathcal{V}| = V$, $|\mathcal{E}| = E$
- Neighborhood of agent i
  $\Rightarrow n_i = \{j : (j, i) \in \mathcal{E}\}$
- Repeated game at agent i, time t
  $\Rightarrow$ params. $\tilde{\mathbf{w}}_{i,t}$ $\Rightarrow$ local loss $\ell_{i,t}$



- Minimize only local loss
  $\Rightarrow$ decoupled local learning
- Instead, each agent i aims to
  $\Rightarrow$ minimize global loss $\ell_t(\tilde{\mathbf{w}}_t) = \sum_{i=1}^{V} \ell_{i,t}(\tilde{\mathbf{w}}_t)$
  $\Rightarrow$ only observes local loss $\ell_{i,t}$ $\Rightarrow$ collaborate with other agents

Networked Regret

- **Local Regret** of node i of distributed online algorithm

$$\mathbf{Reg}_T^i = \sum_{t=1}^{T}\sum_{j=1}^{V} \ell_{j,t}(\tilde{\mathbf{w}}_{i,t}) - \sum_{t=1}^{T}\sum_{j=1}^{V} \ell_{j,t}(\mathbf{w}^*).$$

$\Rightarrow$ $\mathbf{w}^* = \text{argmin}_{\mathbf{x}} \sum_{t=1}^{T}\sum_{j=1}^{V} \ell_{j,t}(\mathbf{w})$ is the global batch solution

$\Rightarrow$ Quality of node i's prediction at others' losses

- **Global Networked Regret**

$$\mathbf{Reg}_T := \frac{1}{N}\sum_{i=1}^{V} \mathbf{Reg}_T^i = \frac{1}{N}\sum_{t=1}^{T}\sum_{i,j=1}^{V} \ell_{j,t}(\tilde{\mathbf{w}}_{i,t}) - \sum_{t=1}^{T}\sum_{j=1}^{N} \ell_{j,t}(\mathbf{w}^*),$$

- Networked online learning goal: $\mathbf{Reg}_T^i/T, \mathbf{Reg}_T/T \to 0$ as $T \uparrow$

$\Rightarrow$ Measures how well agents learn global information

Agent Agreement and Lagrangian Relaxation

- Convexity of $\ell_{i,t} \implies$ unique globally optimal offline strategy $\mathbf{w}^*$
  $\Rightarrow$ Node predictions should coincide at optimality.
- At each t we want to enforce $\mathbf{w}_{i,t} = \mathbf{w}_{j,t}$ for $j \in n_i$, or $\mathbf{C}\mathbf{w}_t = \mathbf{0}$.
  $\Rightarrow$ $\mathbf{C}$ is node-edge incidence matrix of $\mathcal{G}$.
  $\Rightarrow$ $\mathbf{w}_t$ is stacked version of $\mathbf{w}_{i,t}$.
- Constraint enforcement requires global coordination
  $\Rightarrow$ Lagrangian relaxation allows distributed computation
- Online Lagrangian for networked learning problem:

$$\mathcal{L}_t(\mathbf{w}_t, \boldsymbol{\lambda}_t) = \sum_{i=1}^{V} \ell_{i,t}(\mathbf{w}_{i,t}) + \boldsymbol{\lambda}_t^\mathsf{T} \mathbf{C}\mathbf{w}_t$$

- Convex/concave function in the primal/dual variables
- $\mathcal{L}_t(\mathbf{w}_t, \boldsymbol{\lambda}_t)$ is stoch. approx. of Lagrangian of prob. (under i.i.d.): $\min_{\mathbf{w}} \sum_{i=1}^{V} \mathbb{E}_{\mathbf{x}_i, \mathbf{y}_i}[\ell(\mathbf{w}_i; \mathbf{x}_i, \mathbf{y}_i)]$ such that $\mathbf{w}_i = \mathbf{w}_j$

Arrow-Hurwicz Saddle Point Method

- Arrow-Hurwicz saddle pt. to online Lagrangian $\mathcal{L}_t(\mathbf{w}_t, \boldsymbol{\lambda}_t)$
  - $\Rightarrow$ Primal Lagrangian subgradient descent
  - $\Rightarrow$ Dual Lagrangian subgradient ascent

$$\mathbf{w}_{t+1} = \mathcal{P}_{\mathcal{X}}[\mathbf{w}_t - \epsilon \nabla_{\mathbf{w}} \mathcal{L}_t(\mathbf{w}_t, \boldsymbol{\lambda}_t)]$$
$$\boldsymbol{\lambda}_{t+1} = \mathcal{P}_{\Lambda}[\boldsymbol{\lambda}_t + \epsilon \nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\mathbf{w}_t, \boldsymbol{\lambda}_t)]$$

- Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}$ for $\boldsymbol{\lambda}_t$ to remain in the image of $\mathbf{C}$
  - $\Rightarrow$ Required for bounded dual subgradients and constraint slacks
- Yields a decentralized algorithm:

$$\mathbf{w}_{i,t+1} = \mathcal{P}_{\mathcal{W}}\left[\mathbf{w}_{i,t} - \epsilon\left(\nabla_{\mathbf{w}_i} \ell_{i,t}(\mathbf{w}_{i,t}) + \sum_{j \in n_i} \lambda_{ij,t} - \lambda_{ji,t}\right)\right]$$

- Dual step at each edge (i,j) $\Rightarrow$ increase price of disagreement

$$\boldsymbol{\lambda}_{ij,t+1} = \mathcal{P}_{\Lambda_{ij}}\left[\boldsymbol{\lambda}_{ij,t} + \epsilon\left(\mathbf{w}_{i,t} - \mathbf{w}_{j,t}\right)\right]$$

Global Networked Regret Bound

Theorem

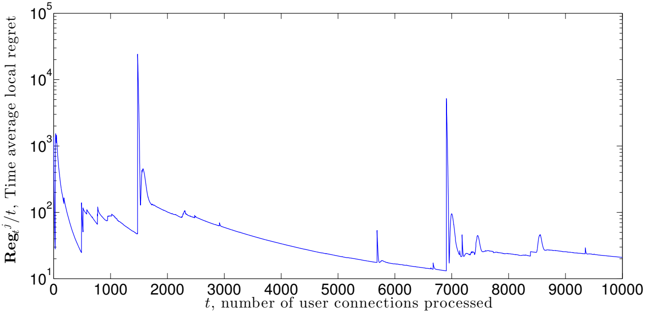
The saddle point alg. sequence with initialization $\boldsymbol{\lambda}_1 = \mathbf{0}$ *and constant step size* $\epsilon = 1/\sqrt{T}$ *achieves the Global networked regret bound*

$$\mathbf{Reg}_T \leq \frac{\sqrt{T}}{2}\left(\|\mathbf{w}_1 - \mathbf{w}^*\|^2 + EC_{\boldsymbol{\lambda}}^2 + G_{\mathbf{w}}^2 + G_{\boldsymbol{\lambda}}^2\right) = \mathcal{O}(\sqrt{T}).$$

- $\mathbf{Reg}_T/T \to 0$ with $T \uparrow$, learning constant depends on . . .
  $\Rightarrow$ Network size V and diameter D
  $\Rightarrow$ Initialization, Lipschitz constant K, gradient bounds $G_{\mathbf{w}}$, $G_{\boldsymbol{\lambda}}$
- $C_{\boldsymbol{\lambda}}$ must satisfy $C_{\boldsymbol{\lambda}} \geq DVK + 1$ $\Rightarrow$ dual set projection
- Comparable to centralized regret of online gradient descent

Detecting Attackers in Computer Networks

$\zeta = log(41)$, $\epsilon = 1/\sqrt{T}$, $N = 50$ cycle network, Arbitrarily chosen $j \in V$

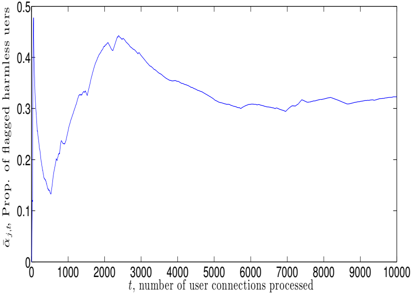


Local regret $\mathbf{Reg}_t^j/t$ vs. no. user connections t

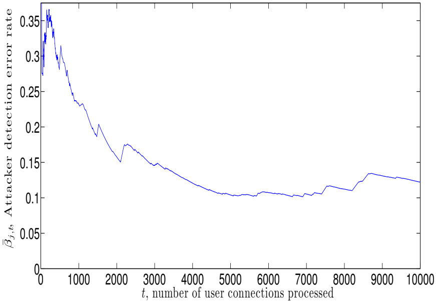
- Identify attackers in LAN
- KDDCup 99 data set:
  - $\Rightarrow VT \approx 5 \times 10^5$ data pts.
  - $\Rightarrow p = 41$ features
- $\mathbf{Reg}_T^j/T \to 0$
  - $\Rightarrow$ Spikes $\Rightarrow$ misclassifications
  - $\Rightarrow$ Recover quickly after mistakes
  - $\Rightarrow$ No raw info exchange

Detecting Attackers in Computer Networks

- Attacker detection protocol on fixed test set of size $T = 10^4$



(a) Avg. false alarm rate vs. no. of user connections t



(b) Avg. error rate vs. no. of user connections t

- $\bar{\alpha}_{j,t}$: False alarm rate
- Friendly users flagged

  $\Rightarrow$ within [.33, .30]

- $\bar{\beta}_{j,t}$: error rate
- Undetected attacks

  $\Rightarrow$ within [.13, .17]

Distributed Learning with Latent Correlation

- Before, we solved the problem

$$\min_{\mathbf{w}} \sum_{i=1}^{V} \mathbb{E}_{\mathbf{x}_i,\mathbf{y}_i}[\ell(\mathbf{w}_i, \mathbf{x}_i, \mathbf{y}_i)] \;\text{ such that }\; \mathbf{w}_i = \mathbf{w}_j$$

$\Rightarrow$ implicitly assumes nodes seek independent, common params

- If there is latent correlation among variables

$\Rightarrow$ equality constraint will harm estimation accuracy

- Introduce proximity function $h(\mathbf{w}_i, \mathbf{w}_j)$

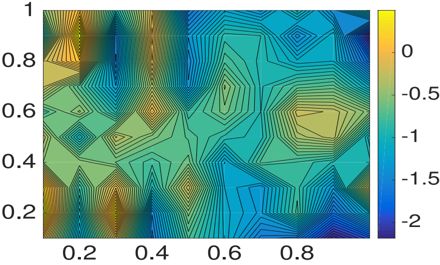
$\Rightarrow$ couples variables of i and j according to a prior γ_{ij} on $\rho(\mathbf{w}_i, \mathbf{w}_j)$

$$\min_{\mathbf{w}} \sum_{i=1}^{V} \mathbb{E}_{\mathbf{x}_i,\mathbf{y}_i}[\ell(\mathbf{w}_i, \mathbf{x}_i, \mathbf{y}_i)] \;\text{ such that }\; h(\mathbf{w}_i, \mathbf{w}_j) \leq \gamma_{ij}$$

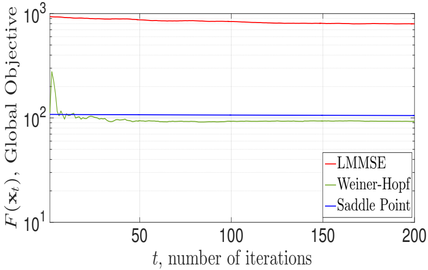
$\Rightarrow$ can solve this problem with comparable saddle pt. techniques

Random Field Estimation

- $N = 100$ grid sensor network
  - $\Rightarrow$ deployed in 200 sq. m. region
- Linear estimation w/ corr. obs.
  - $\Rightarrow$ distance corr. $\rho_{ij} = e^{-\|l_i - l_j\|}$
- Constant step-size $\epsilon = 10^{-2.75}$
  - $\Rightarrow$ Prox. func. $\|\mathbf{w}_i - \mathbf{w}_j\|^2 \leq \gamma_{ij}$
  - $\Rightarrow \gamma_{ij} \Rightarrow$ sample correlation
- Comparable performance to (recursive) Weiner-Hopf estimator
  - $\Rightarrow$ via proximity constraints



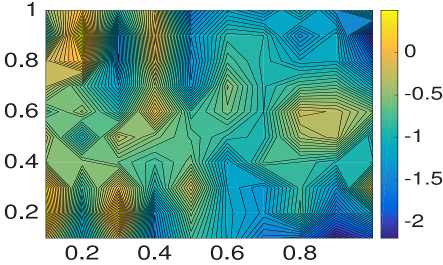
(a) Snapshot of random field



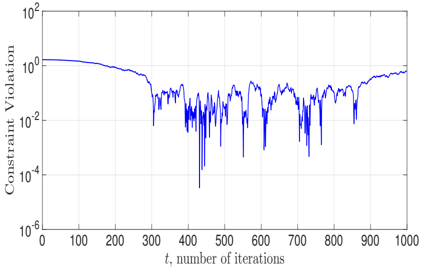
(b) Objective over iteration t

Random Field Estimation

- $N = 100$ grid sensor network
  - $\Rightarrow$ deployed in 200 sq. m. region
- Linear estimation w/ corr. obs.
  - $\Rightarrow$ distance corr. $\rho_{ij} = e^{-\|l_i - l_j\|}$
- Constant step-size $\epsilon = 10^{-2.75}$
  - $\Rightarrow$ Prox. func. $\|\mathbf{w}_i - \mathbf{w}_j\|^2 \leq \gamma_{ij}$
  - $\Rightarrow \gamma_{ij} \Rightarrow$ sample correlation
- Comparable performance to (recursive) Weiner-Hopf estimator
  - $\Rightarrow$ via proximity constraints



(a) Snapshot of random field



(b) Constraint Violation over iteration t

Dictionary Learning

Reducing Inference Error Rates

- In distributed convex setting $\Rightarrow$ we achieve optimum of ERM
  - $\Rightarrow$ even when data is scattered across a multi-agent network
  - $\Rightarrow$ mediocre accuracy $\Rightarrow$ complexity of relating $\mathbf{x}$ and $\mathbf{y}$
- Need better estimator $\Rightarrow$ alternative data representations
  - $\Rightarrow$ Signal encoding $\Rightarrow$ Fourier, wavelets, PCA, or data-driven
  - $\Rightarrow$ Task-driven: tailor dictionary to inference (Mairal '12)

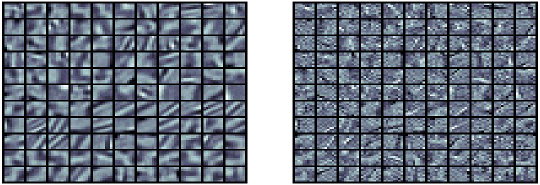


Figure: Initialized (left) and learned (right) dictionary for small image patches.

Dictionary Learning

- Represent signals $\mathbf{x}_t$ as combos. of k basis elements $\{\mathbf{d}_l\}_{l=1}^k$

  $\Rightarrow$ learn dictionary $\mathbf{D} \in \mathbb{R}^{m\times k}$ from data

  $\Rightarrow$ Denote the coding of $\mathbf{x}_t$ as $\boldsymbol{\alpha}_t \in \mathbb{R}^k$
- Representation loss $g(\boldsymbol{\alpha}_t, \mathbf{D}; \mathbf{x}_t)$ $\Rightarrow$ small if $\mathbf{D}\boldsymbol{\alpha}_t$ and $\mathbf{x}_t$ close

  $\Rightarrow$ $\mathbf{D}\boldsymbol{\alpha}_t$ is representation of $\mathbf{x}_t$ w.r.t dictionary $\mathbf{D}$
- Formulate the coding problem

$$\boldsymbol{\alpha}^*(\mathbf{D}; \mathbf{x}_t) := \underset{\boldsymbol{\alpha}_t \in \mathbb{R}^k}{\operatorname{argmin}}\, g(\boldsymbol{\alpha}_t, \mathbf{D}; \mathbf{x}_t)\ .$$

- Dictionary learning

  $\Rightarrow$ seek $\mathbf{D}$ such that signals $\mathbf{x}_t$ well-represented by $\mathbf{D}\boldsymbol{\alpha}^*(\mathbf{D}; \mathbf{x}_t)$

Discriminative Dictionary Learning

- Tailor dictionary to discriminative modeling task
- Use coding $\alpha^*(\mathbf{D}; \mathbf{x}_t)$ as representation of signal $\mathbf{x}_t$
- Decision variable $\mathbf{w}$ $\Rightarrow$ predict the label/vector $\mathbf{y}_t$ given $\alpha^*(\mathbf{D}; \mathbf{x}_t)$.
- Loss function $\ell(\mathbf{D}, \mathbf{w}; (\mathbf{x}_t, \mathbf{y}_t)) = \ell(\alpha^*(\mathbf{D}; \mathbf{x}_t), \mathbf{D}, \mathbf{w}; (\mathbf{x}_t, \mathbf{y}_t))$

  $\Rightarrow$ predictive quality of $\mathbf{w}$ for var. $\mathbf{y}_t$ given coding $\alpha^*(\mathbf{D}; \mathbf{x}_t)$
- Discriminative dictionary learning

$$(\mathbf{D}^*, \mathbf{w}^*) := \underset{\mathbf{D}\in\mathcal{D}, \mathbf{w}\in\mathcal{W}}{\operatorname{argmin}} \mathbb{E}_{\mathbf{x},\mathbf{y}}\Big[\ell\big(\mathbf{D}, \mathbf{w}; (\mathbf{x}, \mathbf{y})\big)\Big].$$

$\Rightarrow$ Learn jointly regression weights $\mathbf{w}$ and dictionary $\mathbf{D}$

$\Rightarrow$ Non-convex stochastic program

Application: Uncertain Robot Path-Planning

- One role for learning in robotics: learn model uncertainty
- Simplified physics models used for control due to complexity
  $\Rightarrow$ Models are available. Not perfect but not useless either
  $\Rightarrow$ Replace mechanical models with learned models
- Learn mismatch between model and reality when
  $\Rightarrow$ This mismatch has variability across different terrains
- Use sensory input to learn uncertainty in execution of controller

Formulating Disturbance

- Consider a discrete nonlinear state-space system of equations

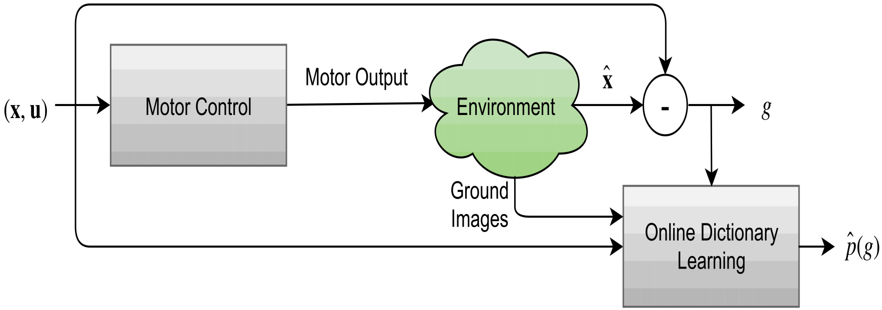
$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) + g(\mathbf{a}_k) = f(\mathbf{x}_k, \mathbf{u}_k) + g(\mathbf{u}_k, \mathbf{z}_k)$$

- $\mathbf{x}_k \Rightarrow$ state vector, $\mathbf{u}_k \Rightarrow$ control input, $\mathbf{z}_k \Rightarrow$ sensory input
- Kinematic model $f(\mathbf{x}_k, \mathbf{u}_k)$ not exact $\Rightarrow$ add mismatch term $g(\mathbf{a}_k)$
  $\Rightarrow$ Want to learn $g(\mathbf{a}_k)$ to use as input to robust control block
- Measure estimate $\hat{\mathbf{x}}_k$ of state $\mathbf{x}_k$ (with on-board IMU, for instance)

$$\hat{g}(\mathbf{a}_{k-1}) = \hat{\mathbf{x}}_k - f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \ .$$

- Learning $\hat{g}(\mathbf{a}_{k-1})$ is challenging
  $\Rightarrow$ Captures difficult-to-model physics we typically ignore
  $\Rightarrow$ Dictionary leaning approach

Dictionary-Based Learning Architecture



- Platform's state **x**, control **u** intended by a kinematic planner
  $\Rightarrow$ differ from measured ground truth $\hat{\mathbf{x}}$ by disturbance g
- This difference, as well as state, control, and visual features
  $\Rightarrow$ Fed into dictionary learning method $\Rightarrow$ disturbance pred. $\hat{g}$
  $\Rightarrow$ Dictionary is a statistical model using sparse approximation

Parametric representation of disturbance

- Model disturbance $\hat{g}(\mathbf{a})$ as Gaussian conditional on features $\mathbf{a}$

$$\mathbb{P}[\hat{g}(\mathbf{a}) \mid \mathbf{a}] = \frac{1}{\sqrt{2\pi\sigma^2(\mathbf{a})}} \exp\left[-\frac{(\hat{g}(\mathbf{a}) - \mu(\mathbf{a}))^2}{2\sigma^2(\mathbf{a})}\right] .$$

- Distribution parameterized by unknown mean $\mu(\mathbf{a})$, var. $\sigma^2(\mathbf{a})$
  $\Rightarrow$ which depend on control $\mathbf{u}_k$ and sensory input $\mathbf{z}_k$

- Realizations of $(\mathbf{a}, \hat{g}(\mathbf{a}))$ available online
- Sequentially obtained while exploring feature space

- Utilize to learn parametric representation of the distribution

Disturbance Prediction as Dictionary Learning

- Learn online mean, variance $\Rightarrow$ introduce regressors $\mathbf{w}_1$, $\mathbf{w}_2$
  $\Rightarrow$ predict first, second-order stats. $\mu(\mathbf{a})$ and $\sigma^2(\mathbf{a})$, given $\mathbf{a}$

$$\hat{\mu}(\mathbf{a}) = \mathbf{w}_1^T \mathbf{a} \ , \quad \hat{\sigma}^2(\mathbf{a}) = \sigma^2_{\text{min}} + \left(\mathbf{w}_2{}^T \mathbf{a} + \sigma^2_{\text{init}}\right)^2$$

- Rather than use $\mathbf{a}$ directly, use a sparse code $\alpha^*(\mathbf{D}; \mathbf{a})$

$$\hat{\mu}(\mathbf{a}) = \mathbf{w}_1^T \alpha^*(\mathbf{D}; \mathbf{a}) \ , \quad \hat{\sigma}^2(\mathbf{a}) = \sigma^2_{\text{min}} + \left(\mathbf{w}_2{}^T \alpha^*(\mathbf{D}; \mathbf{a}) + \sigma^2_{\text{init}}\right)^2$$

  $\Rightarrow$ Regress on sparse approximation $\alpha^*(\mathbf{D}; \mathbf{a})$ w.r.t. dictionary $\mathbf{D}$

- Motivation for using sparse code $\alpha^*(\mathbf{D}; \mathbf{a})$, learning dictionary $\mathbf{D}$:
  $\Rightarrow$ $g(\cdot)$ relates robotic sensory perception, unexpected dynamics
  $\Rightarrow$ Relationship between $(\mathbf{a}, \hat{g}(\mathbf{a}))$ highly nonlinear
  $\Rightarrow$ Estimation accuracy $\Rightarrow$ boosted via alternative encoding

Task-Driven Dictionary Learning

- Dictionary $\mathbf{D} = \{\mathbf{d}_l\}_{l=1}^{m}$, $\mathbf{d}_l \in \mathbb{R}^k$ composed of m basis elements
- Estimate $\hat{\mathbf{a}}_k = \mathbf{D}\boldsymbol{\alpha}_k$ as linear combo. of dictionary elements
- Select coefficients that yield a sparse code (elastic net)

$$\boldsymbol{\alpha}^*(\mathbf{D}; \mathbf{a}_k) := \underset{\boldsymbol{\alpha}\in\mathbb{R}^k}{\operatorname{argmin}} \|\mathbf{a}_k - \mathbf{D}\boldsymbol{\alpha}\|^2 + \lambda\|\boldsymbol{\alpha}\|_1 + \nu\|\boldsymbol{\alpha}\|_2^2$$

- Jointly learn dictionary and regressors $\mathbf{w}_1$ and $\mathbf{w}_2$

$$(\mathbf{D}^*, \mathbf{w}_1^*, \mathbf{w}_2^*) := \underset{\mathbf{D}\in\mathcal{D}, \mathbf{w}_1, \mathbf{w}_2}{\operatorname{argmin}} \mathbb{E}_{\mathbf{a}, \hat{g}(\mathbf{a})}\Big(-\log \mathbb{P}[\hat{g}(\mathbf{a}) \,|\, \mathbf{a}, \mathbf{D}, \mathbf{w}_1, \mathbf{w}_2]\Big).$$

- Non-convex but convex w.r.t. $\mathbf{D}$ and $\mathbf{w}_1$ and $\mathbf{w}_2$ separately
- Objective is an expectation over dataset $\Rightarrow$ use stochastic grad.

Learning: SGD + Sparse Coding

- Observe signals $\mathbf{z}_k$, use past control $\mathbf{u}_k$ to compute coding

$$\boldsymbol{\alpha}_k^* := \underset{\boldsymbol{\alpha}_k \in \mathbb{R}^s}{\operatorname{argmin}} (1/2)\|\mathbf{a}_k - \mathbf{D}\boldsymbol{\alpha}_k\|_2^2 + \lambda\|\boldsymbol{\alpha}_k\|_1 + \nu\|\boldsymbol{\alpha}_k\|_2,$$

$\Rightarrow$ Update dictionary using stoch. grad. step w.r.t. dictionary

$$\mathbf{D}_{k+1} = \mathbf{D}_k - \epsilon_k \left(\nabla_{\mathbf{D}} \log \mathbb{P}[\hat{g}(\mathbf{a}_k) \,|\, \mathbf{a}_k, \mathbf{D}_k, \mathbf{w}_{1,k}, \mathbf{w}_{2,k}]\right)$$

- Update regressors along regressor gradient of loss function

$$\mathbf{w}_{1,k+1} = \mathbf{w}_{1,k} + \epsilon_k \left(\nabla_{\mathbf{w}_1} \log \mathbb{P}[\hat{g}(\mathbf{a}_k) \,|\, \mathbf{a}_k, \mathbf{D}_k, \mathbf{w}_{1,k}, \mathbf{w}_{2,k}]\right) ,$$
$$\mathbf{w}_{2,k+1} = \mathbf{w}_{2,k} + \epsilon_k \left(\nabla_{\mathbf{w}_2} \log \mathbb{P}[\hat{g}(\mathbf{a}_k) \,|\, \mathbf{a}_k, \mathbf{D}_k, \mathbf{w}_{1,k}, \mathbf{w}_{2,k}]\right) ,$$

- Converges to locally optimal dictionary and regressors

Implementation on a Ground Robot

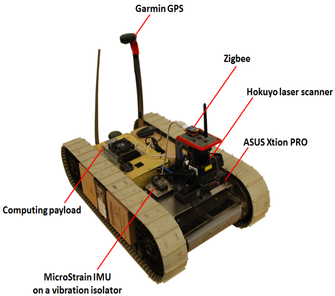


Figure: An iRobot *Packbot* was used in our experiments. It was additionally configured with a high-resolution camera.

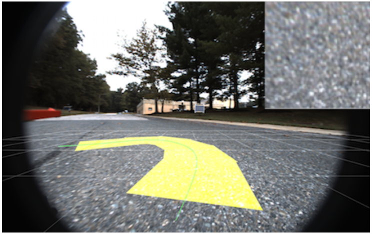
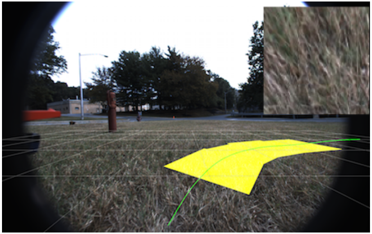
- We consider a differential drive model of a skid-steer robot

$$f(\mathbf{x}_k, \mathbf{u}_k) = \begin{bmatrix} \dot{x}_k \\ \dot{y}_k \\ \dot{\theta}_k \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\theta_k) & -\sin(\theta_k) & 0 \\ \sin(\theta_k) & \cos(\theta_k) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{A(\theta)} \begin{bmatrix} \nu_k \\ \omega_k \end{bmatrix}$$

- Disturbance $\Rightarrow$ commanded & actual angular velocity difference

Feature Construction

- Visual patch $\mathbf{z}_k$ $\Rightarrow$ associated with the portion of ground
  $\Rightarrow$ Collect images over time horizon of planned robot trajectory

- From the raw patch we construct statistical visual features $\mathbf{c}_k$
  $\Rightarrow$ mean, variance, skewness, kurtosis of RGB color channels
  $\Rightarrow$ Textures $\mathbf{h}_k$ via texton histogram (Leung '99)
- Concatenated with average linear, angular velocity in $[k, k+1]$

Empirical Performance Comparison

- Model fit of disturbance to task driven dictionary learnt dist.
- Compared to (windowed) recursive average of mean, variance

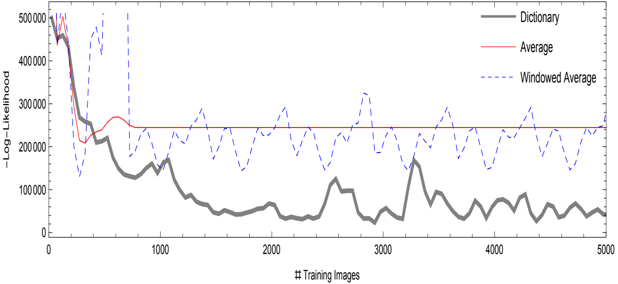


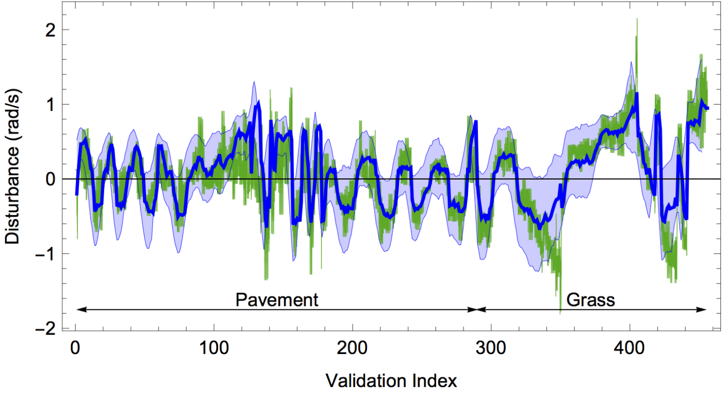
Figure: Comparison of dictionary learning vs. classical alternatives.

- Superior model fit to Gaussian approximation of disturbance
- Exploits sensory input to identify terrain type

  $\Rightarrow$ Pavement or grass essentially. But more granular than that

Predicting Future Uncertainty

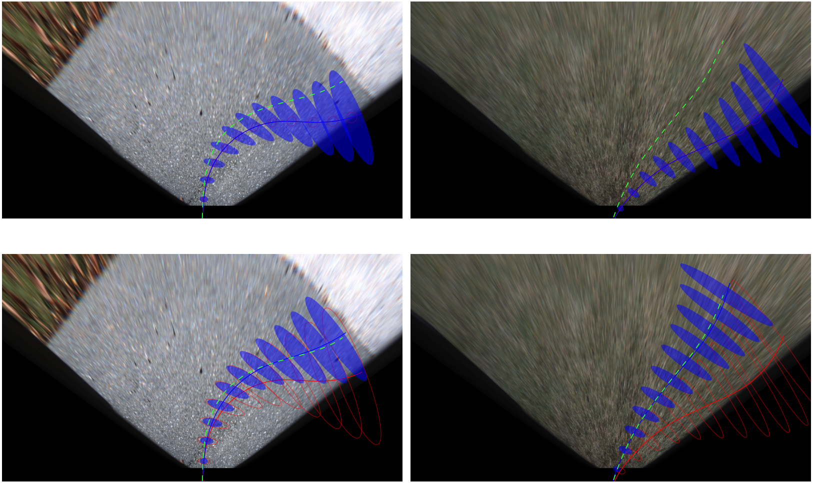
- Test trajectory with predicted & actual disturbance stats. overlaid
- Measured dist. (green) and predicted dist. (blue) for trajectory
  - $\Rightarrow$ Predicted mean and $\pm 2\sigma$ envelope shown



- Observations are mostly contained within confidence envelopes

Future Uncertainty Cones

- Actual trajectory not contained within cones for initial dictionary
  $\Rightarrow$ But contained within cone after dictionary is properly learnt



Decentralized Dynamic Discriminative Dict.

Multi-Agent Dictionary Learning

- Dictionary-based estimations better than GLM
  $\Rightarrow$ but what if data is scattered across a network?
- Incentivize agreement via constraint $\mathbf{D}_i = \mathbf{D}_j, \mathbf{w}_i = \mathbf{w}_j$ for all $j \in n_i$
- Decentralized task-driven dictionary learning problem

$$\{\mathbf{D}_i^*, \mathbf{w}_i^*\}_{i=1}^V := \underset{\mathbf{D}_i \in \mathcal{D}, \mathbf{w}_i \in \mathcal{W}}{\text{argmin}} \sum_{i=1}^V \mathbb{E}_{\mathbf{x}_i, \mathbf{y}_i} \left[\ell(\mathbf{D}_i, \mathbf{w}_i; (\mathbf{x}_i, \mathbf{y}_i)) \right].$$
$$\text{such that} \quad \mathbf{D}_i = \mathbf{D}_j, \mathbf{w}_i = \mathbf{w}_j \text{ for all } j \in n_i$$

- Similar to networked regret minimization
- Enforcing agreement constraint would require global coordination
  $\Rightarrow$ Define stochastic Lagrangian $\Rightarrow$ distributed alg.

$$\hat{\mathcal{L}}_t(\mathbf{D}, \mathbf{w}, \Lambda, \boldsymbol{\nu}) = \sum_{i=1}^V \left[\ell(\mathbf{D}_i, \mathbf{w}_i; (\mathbf{x}_{i,t}, \mathbf{y}_{i,t})) \right] + \text{tr}(\Lambda^T \mathbf{C}_D \mathbf{D}) + \boldsymbol{\nu}^T \mathbf{C}_w \mathbf{w}$$

Block-Stochastic Saddle Point Method

- Decentralized online dict. learning $\Rightarrow$ Block saddle point alg.
- Stochastic approximation: $\mathcal{L}(\mathbf{D}, \mathbf{w}, \boldsymbol{\Lambda}, \boldsymbol{\nu}) = \mathbb{E}_{\mathbf{x},\mathbf{y}}[\hat{\mathcal{L}}_t(\mathbf{D}, \mathbf{w}, \boldsymbol{\Lambda}, \boldsymbol{\nu})]$

  $\Rightarrow$ primal stochastic gradient descent

$$\mathbf{D}_{t+1} = \mathbf{D}_t - \epsilon_t \nabla_{\mathbf{D}} \hat{\mathcal{L}}_t(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t) ,$$
$$\mathbf{w}_{t+1} = \mathbf{w}_t - \epsilon_t \nabla_{\mathbf{w}} \hat{\mathcal{L}}_t(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t) .$$

  $\Rightarrow$ dual stochastic gradient ascent

$$\boldsymbol{\Lambda}_{t+1} = \boldsymbol{\Lambda}_t + \epsilon_t \nabla_{\boldsymbol{\Lambda}} \hat{\mathcal{L}}_t(\mathbf{D}_{t+1}, \mathbf{w}_{t+1}, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t) ,$$
$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \epsilon_t \nabla_{\boldsymbol{\nu}} \hat{\mathcal{L}}_t(\mathbf{D}_{t+1}, \mathbf{w}_{t+1}, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t) .$$

- $\nabla_{\mathbf{D}} \hat{\mathcal{L}}_t(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)$ $\Rightarrow$ Projected stoch. grad. w.r.t. $\mathbf{D}$

  $\Rightarrow$ gradient approximated with current signals $\{\mathbf{x}_{i,t}, \mathbf{y}_{i,t}\}_{i=1}^{V}$

Local View of Algorithm

- At agent i, time t, observe $(\mathbf{x}_{i,t}, \mathbf{y}_{i,t})$,
- Compute coding $\boldsymbol{\alpha}^{\star}_{i,t+1} = \text{argmin}_{\boldsymbol{\alpha} \in \mathbb{R}^k} g(\boldsymbol{\alpha}, \mathbf{D}_{i,t}; \mathbf{x}_{i,t})$

  $\Rightarrow$ In practice chosen as *sparse coding* via lasso or elastic-net
- Update primal variables at agent i

$$\mathbf{D}_{i,t+1} = \mathbf{D}_{i,t} - \epsilon_t \Big(\nabla_{\mathbf{D}_i} \ell_i(\mathbf{D}_{i,t}, \mathbf{w}_{i,t}; (\mathbf{x}_{i,t}, \mathbf{y}_{i,t})) + \sum_{j \in n_i} (\boldsymbol{\Lambda}_{ij,t} - \boldsymbol{\Lambda}_{ji,t}) \Big) \ ,$$

$$\mathbf{w}_{i,t+1} = \mathbf{w}_{i,t} - \epsilon_t \Big(\nabla_{\mathbf{w}_i} \ell_i(\mathbf{D}_{i,t}, \mathbf{w}_{i,t}; (\mathbf{x}_{i,t}, \mathbf{y}_{i,t})) + \sum_{j \in n_i} (\boldsymbol{\nu}_{ij,t} - \boldsymbol{\nu}_{ji,t}) \Big) \ ,$$

- Update dual variables at network communication link (i, j)

$$\boldsymbol{\Lambda}_{ij,t+1} = \boldsymbol{\Lambda}_{ij,t} + \epsilon_t \left(\mathbf{D}_{i,t} - \mathbf{D}_{j,t} \right)$$

$$\boldsymbol{\nu}_{ij,t+1} = \boldsymbol{\nu}_{ij,t} + \epsilon_t \left(\mathbf{w}_{i,t} - \mathbf{w}_{j,t} \right)$$

Convergence Result

Theorem

Saddle pt. seq. $(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)$ *converges to stationarity in expectation:*

$$\lim_{t\to\infty} \mathbb{E}[\|\nabla_{\mathbf{D}}\mathcal{L}(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)\|] = 0 \ ,$$

$$\lim_{t\to\infty} \mathbb{E}[\|\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)\|] = 0$$

Asymptotic feasibility condition achieved in expectation:

$$\lim_{t\to\infty} \mathbb{E}[\|\nabla_{\boldsymbol{\Lambda}}\mathcal{L}(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)\|] = 0$$

$$\lim_{t\to\infty} \mathbb{E}[\|\nabla_{\boldsymbol{\nu}}\mathcal{L}(\mathbf{D}_t, \mathbf{w}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)\|] = 0$$

- Performance guarentee for D4L
  - $\Rightarrow$ convergence in non-convex stochastic opt.
  - $\Rightarrow$ sensitive to data distribution, step-size, network structure

Image Processing Experiments

- Texture database classification problem $\Rightarrow$ Brodatz textures
  - $\Rightarrow$ Insight into dynamic image processing problems
  - $\Rightarrow$ Toy model of real-time navigability analysis in robotic teams
- Real-time image data $\Rightarrow$ train multi-class logistic reg. weights
- Decentralized dynamic texture classification
  - $\Rightarrow$ Subset of textures: {grass, bark, straw, herringbone_weave}

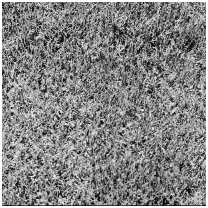


Figure: Sample images from Brodatz textures.

Incomplete Sampling

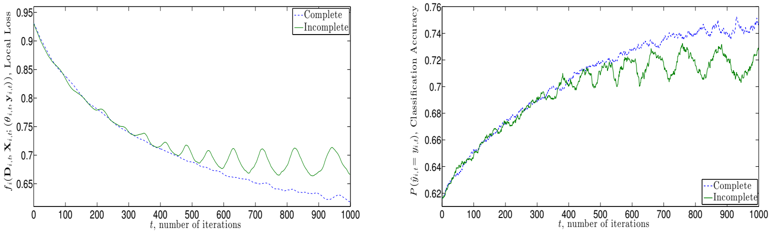


Figure: Local loss (left) and classification accuracy (right) versus iteration t.

- $V = 10$ node random network, results shown for random $j \in \mathcal{V}$
- Agents observe random incomplete subsets of feature space
- Still learn global information and reach consensus
- Moderate classifier performance

  $\Rightarrow$ due to small step-size required for convergence

  $\Rightarrow$ Small step-sizes required for convergence

Robotic Field Setting

- Each robot in the network sequentially observes images
  $\Rightarrow$ partitions them into small patches
  $\Rightarrow$ classify patches with multi-class logistic regression.
- Terrain classification has been used as a layer in robust control
  $\Rightarrow$ Classes are terrains of varying traversability
- Data from Lejeune Robotics Test Facility $\Rightarrow$ Thanks to ARL!

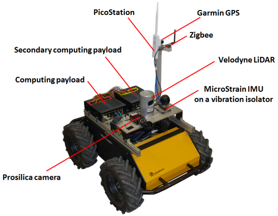
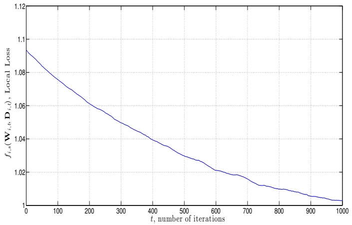



Figure: Sample image (left) from a $N = 3$ robot network of Huskies (right).

Results on Robotic Network



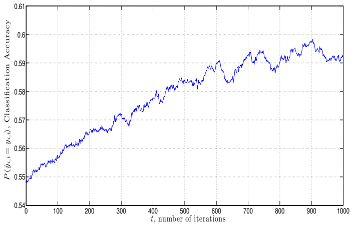


Figure: Local loss (left) and classification accuracy (right) versus iteration t. $V = 3$ complete graph w/ complete sampling.

- Main takeaways: training is slow, but convergent
  - $\Rightarrow$ due to necessity of small step-sizes
  - $\Rightarrow$ good for centralized learning, slow in distributed case
- Non-convexity makes training/optimization challenging
  - $\Rightarrow$ nonlinear classifiers + convex training? $\Rightarrow$ flexible dictionary

Nonparametric Regression

Large-Scale Function Estimation

- Learning nonlinear statistical models $\Rightarrow$ function estimation
- Want to find $f^* \in \mathcal{H}$ to minimize regularized expected risk $R(f)$

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

$\Rightarrow$ Loss $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ penalize deviations between $f(\mathbf{x})$, $\mathbf{y}$

- Generally intractable $\Rightarrow$ infinite dimensional data-dependent opt.

Large-Scale Function Estimation

- Learning nonlinear statistical models $\Rightarrow$ function estimation
- Want to find $f^* \in \mathcal{H}$ to minimize regularized expected risk $R(f)$

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

$\Rightarrow$ Loss $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ penalize deviations between $f(\mathbf{x})$, $\mathbf{y}$

- Generally intractable $\Rightarrow$ infinite dimensional data-dependent opt.
- Reproducing kernels $\Rightarrow$ framework to make this task possible!

$\Rightarrow$ $\mathcal{H}$ is equipped with unique *kernel function*, $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$

$$f^* := \operatorname*{argmin}_{f \in \mathcal{H}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\sum_{n \in \mathcal{I}} w_n \kappa(\mathbf{x}_n, \mathbf{x}), y)] + \frac{\lambda}{2}\|\sum_{n,m \in \mathcal{I}} w_n w_m \kappa(\mathbf{x}_n, \mathbf{x}_m)\|^2_{\mathcal{H}}$$

$\Rightarrow$ via use of Representer Theorem, dating back to Riesz Thm.

Nonlinear Inferences: Dictionaries vs. Kernels

- Kernel methods learn function $f(\mathbf{x}) = \sum_{n \in \mathcal{I}} w_n \kappa(\mathbf{x}_n, \mathbf{x})$
  - $\Rightarrow$ comes from Representer Theorem, dating back to Riesz Thm.
  - $\Rightarrow$ $\mathcal{I}$ is a countably infinite indexing set
- Maintain convexity while learning nonlinear statistical model
  - $\Rightarrow$ complicated representation: $|\mathcal{I}| = \infty$ vs. k basis elements
  - $\Rightarrow$ flexible dictionary, model conditional density of inference
- Train with functional stochastic gradient descent?

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

- Problem: training complexity with SGD is cubic in iteration index
  - $\Rightarrow$ Sparsify the solution? Need to sparsify training

Online Multi-Class Kernel SVM

- We implement functional generalization of SGD

$$\tilde{f}_{t+1}(\cdot) = (1-\eta_t\lambda)f_t - \eta_t \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot)$$

$$(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{KOMP}(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$$

$\Rightarrow$ operating in tandem with projection step onto subspaces of $\mathcal{H}$

$\Rightarrow$ subspaces greedily constructed via matching pursuit

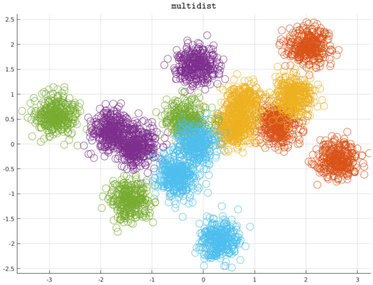
$$f_{t+1} = \underset{f \in \mathcal{H}_{\mathbf{D}_{t+1}}}{\operatorname{argmin}} \left\| f - \Big((1-\eta_t\lambda)f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big) \right\|_{\mathcal{H}}^2$$

- Proposed work: online training of sparsified kernel classifiers

$\Rightarrow$ Main problem: sparsification bias may ruin stochastic descent

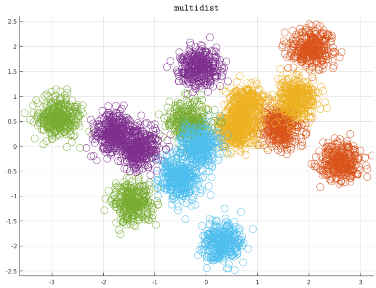
Online Multi-Class Kernel SVM

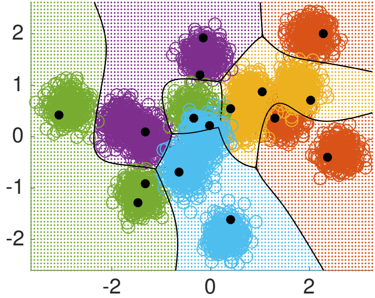
- Case where training examples for a fixed class
  $\Rightarrow$ drawn from a distinct Gaussian mixture
- 3 Gaussians per mixture, $C = 5$ classes total for this experiment
  $\Rightarrow$ 15 total Gaussians generate data



Online Multi-Class Kernel SVM

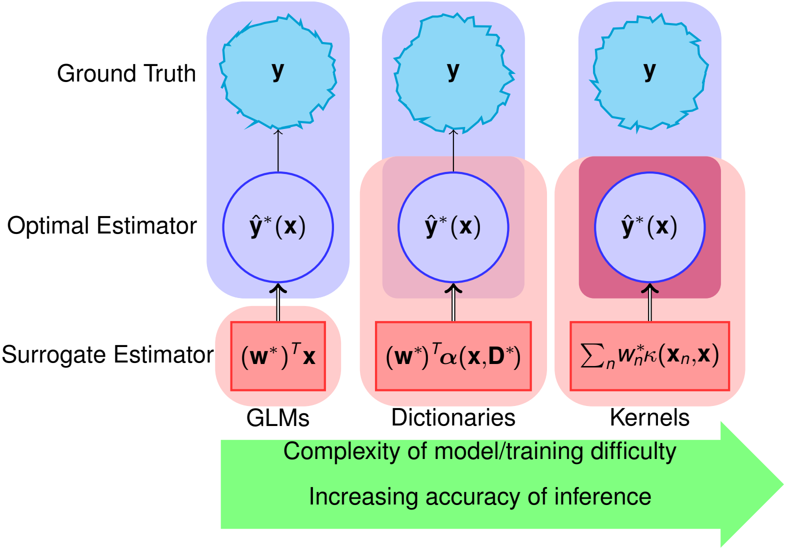
- Case where training examples for a fixed class
  $\Rightarrow$ drawn from a distinct Gaussian mixture
- 3 Gaussians per mixture, $C = 5$ classes total for this experiment
  $\Rightarrow$ 15 total Gaussians generate data





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

Conclusion



Ground Truth
Optimal Estimator
Surrogate Estimator
y
y
y
$\hat{\mathbf{y}}^*(\mathbf{x})$
$\hat{\mathbf{y}}^*(\mathbf{x})$
$\hat{\mathbf{y}}^*(\mathbf{x})$
$(\mathbf{w}^*)^T\mathbf{x}$
$(\mathbf{w}^*)^T\alpha(\mathbf{x},\mathbf{D}^*)$
$\sum_n w_n^*\kappa(\mathbf{x}_n,\mathbf{x})$
GLMs
Dictionaries
Kernels
Complexity of model/training difficulty
Increasing accuracy of inference

Conclusion

- Can establish strong convergence guarantees for GLMs
  - $\Rightarrow$ In centralized and distributed online cases
  - $\Rightarrow$ But we are solving the wrong problem $\Rightarrow$ too far from $\hat{y}^*(\mathbf{x})$
- Dictionary methods $\Rightarrow$ richer estimators, nonlinear classifiers
  - $\Rightarrow$ promising results on robotics application
  - $\Rightarrow$ non-convexity $\Rightarrow$ challenge to train in distributed setting
- Kernel methods allow learning nonlinear classifiers
  - $\Rightarrow$ preserve convexity, directly model conditional density
  - $\Rightarrow$ but training has prohibitive complexity in online setting
- Proposal: learn kernel classifiers online with low complexity

References

Journals

- A. Koppel, F. Jakubeic, and A. Ribeiro, “A saddle point algorithm for networked online convex optimization,” IEEE Trans. Signal Process., vol.PP, no.99, June. 2015.
- A. Koppel, B. Sadler, and A. Ribeiro, ”Proximity without Consensus in Online Multi-Agent Optimization,” in IEEE Trans. Signal Proc. (submitted), June 2016.
- A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, “D4L: Decentralized Dynamic Discriminative Dictionary Learning,” in IEEE Trans. Signal Info. Process over Networks (submitted)., June. 2016.

Conferences

- A. Koppel, F. Jakubeic and A. Ribeiro, “Regret Bounds of a distributed saddle point algorithm,” in Proc. Int. Conf. Accoustics Speech Signal Process., Brisbane, Australia, Apr 19-24 2015.
- A. Koppel, F. Y. Jakubiec, and A. Ribeiro, “A saddle point algorithm for networked online convex optimization.” in 39th Proc. Int. Conf. Acoust. Speech Signal Process., Florence, Italy, May 4-9 2014, pp. 8292 - 8296.
- A. Koppel, B. M. Sadler and A. Ribeiro, ”Proximity without consensus in online multi-agent optimization,” in Proc. Int. Conf. Accoustics Speech Signal Process., Shanghai, China, Mar. 20-25 2016.
- A. Koppel, B. M. Sadler, and A. Ribeiro, “Decentralized Online Optimization with Heterogeneous Data Sources”, IEEE Global Conference on Signal and Information Processing (to appear), Washington, DC, Dec. 7-9, 2016.
- A. Koppel, J. Fink, G. Warnell, E. Stump, and A. Ribeiro, “Online Learning for Characterizing Unknown Environments in Ground Robotic Vehicle Models,” in Proc. Int. Conf. Intelligent Robotics and Systems, Daejeon, Korea, Oct9-Oct14 2016
- A. Koppel, G. Warnell, and E. Stump. “Task-Driven Dictionary Learning in Distrubted Online Settings.” in Proc. Asilomar Conf. on Signals Systems Computers, Pacific Grove, CA, November 8-11 2015.
- A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, “D4L: Decentralized Dynamic Discriminative Dictionary Learning,” in Proc. Int. Conf. Intelligent Robotics and Systems, Hamburg, Germany, Sep 28-Oct2 2015.

Appendix

DSPA: Technical Assumptions

- The network $\mathcal{G}$ is symmetric and connected with diameter D.
- Loss function gradients for any $\mathbf{w}$ bounded by constant G, i.e.

$$\|\nabla \ell_t(\mathbf{w})\|_2 \leq G.$$

- Losses $\ell_{i,t}(\mathbf{x})$ are Lipschitz continuous with modulus $K_{i,t} \leq K$

$$\|\ell_{i,t}(\mathbf{w}) - \ell_{i,t}(\mathbf{w})\|_2 \leq K_{i,t}\|\mathbf{w} - \mathbf{v}\|_2 \leq K\|\mathbf{w} - \mathbf{v}\|_2$$

D4L: Technical Assumptions

- Network $\mathcal{G} \Rightarrow$ symmetric and connected with diameter D.
- Diminishing step-size rules: $\sum_{t=0}^{\infty} \epsilon_t = \infty$ and $\sum_{t=0}^{\infty} \epsilon_t^2 < \infty$
- Mean and variance conditions of Lagrangian stochastic gradients

$$\mathbb{E}[\|\boldsymbol{\delta}_{\mathbf{D},t}\| \mid \mathcal{F}_t] \leq A\epsilon_t \ ,$$
$$\mathbb{E}[\|\nabla_{\mathbf{D}}\hat{\mathcal{L}}_t(\mathbf{D}_t, \mathbf{x}_t, \boldsymbol{\Lambda}_t, \boldsymbol{\nu}_t)\|^2 \mid \mathcal{F}_t] \leq \sigma^2.$$

- Feasible dictionary set is those with unit column-norms

$$\mathcal{D} = \{\mathbf{D} \in \mathbb{R}^{m \times k} : \|\mathbf{d}_j\| \leq 1, j = 1 \ldots k\}.$$

Decentralized Online SVM

- Instantaneous loss $\Rightarrow$ hinge loss at current data point

$$\ell_{i,t}(\mathbf{w}) = \frac{\zeta}{2}\|\mathbf{w}\|_2^2 + \max\left(0, 1 - y_{i,t}\mathbf{w}^T\mathbf{x}_{i,t}\right)$$

- $\mathbf{Reg}_T$ measures price of distributed causal classifier training
- Algorithm Formulation

$$\mathbf{w}_{i,t+1} = \mathcal{P}_{\mathcal{W}}\left[\mathbf{w}_{i,t} - \epsilon\left(\zeta\mathbf{w}_{i,t} - y_{i,t}\mathbf{x}_{i,t}\mathbb{I}\left(y_{i,t}\mathbf{w}_{i,t}^T\mathbf{x}_{i,t} < 1\right) + \sum_{j\in n_i}(\boldsymbol{\lambda}_{ij,t} - \boldsymbol{\lambda}_{ji,t})\right)\right],$$

$$\boldsymbol{\lambda}_{ij,t+1} = \mathcal{P}_{\Lambda_{ij}}\left[\boldsymbol{\lambda}_{ij,t} + \epsilon\left(\mathbf{w}_{i,t} - \mathbf{w}_{j,t}\right)\right].$$

- Limit classifier complexity to set $W = \{\mathbf{w}_i \in \mathbb{R}^p : \|\mathbf{w}_i\|_2 \leq \zeta\}$
- $\mathbb{I}\left(y_{i,t}\mathbf{w}^T\mathbf{x}_{i,t} < 1\right) = 1$ if $y_{i,t}\mathbf{w}^T\mathbf{x}_{i,t} < 1$, $\mathbb{I}\left(y_{i,t}\mathbf{w}^T\mathbf{x}_{i,t} < 1\right) = 0$ else
- Projected Perceptron with dual correction (neighbor info)

Sparse Multi-class texture classification

- Multi-class logistic reg. prob. $\Rightarrow$ Agent i receives signals $\mathbf{x}_{i,t}$
  $\Rightarrow$ output a decision variable $\mathbf{y}_{i,t} \in \{0,1\}^C$ $\Rightarrow$ C no. of classes
- $[\mathbf{y}_{i,t}]_c$ $\Rightarrow$ binary indicator of whether signal falls in class c.
- Local loss ℓ_i $\Rightarrow$ negative log-likelihood of prob. model

$$\ell_i(\mathbf{D}_i, \mathbf{W}_i; (\boldsymbol{\theta}_i, \mathbf{y}_i)) = \log\left(\sum_{c=1}^{C} e^{\mathbf{w}_{i,c}^T \boldsymbol{\alpha}_i^* + w_{i,c}^0}\right) - \sum_{c=1}^{C}\left(y_{i,c}\mathbf{w}_{i,c}^T\boldsymbol{\alpha}_i^* + w_{i,c}^0\right) + \xi\|\mathbf{W}_i\|_F^2 ,$$

- $\boldsymbol{\alpha}_i^*$ $\Rightarrow$ sparse coding via elastic-net min. prob.
- $g_c(\boldsymbol{\alpha}_i^*) = e^{\mathbf{w}_{i,c}^T \boldsymbol{\alpha}_i^* + w_{i,c}^0}$ is activation function;
  $\Rightarrow$ $g_c(\mathbf{z}_i)/\sum_{c'} g_{c'}(\mathbf{z}_i)$ $\Rightarrow$ prob. $\mathbf{z}_i$ in class c
  $\Rightarrow$ $\mathbf{z}_i$ $\Rightarrow$ average of image sub-patches
- Classification decision $\Rightarrow$ maximum likelihood class label
  $\Rightarrow$ $\tilde{c} = \text{argmax}_c\, g_c(\mathbf{z}_i)/\sum_{c'} g_{c'}(\mathbf{z}_i)$; $[\mathbf{y}_{i,t}]_c = 0$ for $c \neq \tilde{c}$

Large-Scale Function Estimation

- 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
- Kernel examples:
  - $\Rightarrow$ Gaussian/RBF $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|_2^2}{2c^2}\right\}$
  - $\Rightarrow$ polynomial $\kappa(\mathbf{x}, \mathbf{x}') = \left(\mathbf{x}^T\mathbf{x}' + b\right)^c$

Function Representation

- Consider empirical risk minimization case: sample size $N < \infty$
- Representer Theorem:

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

$\Rightarrow$ $\mathbf{x}_n$ are feature vectors, and w_n is a scalar weight.

$\Rightarrow$ f is a kernel expansion over training set

Function Representation

- Consider empirical risk minimization case: sample size $N < \infty$
- Representer Theorem:

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

$\Rightarrow$ $\mathbf{x}_n$ are feature vectors, and w_n is a scalar weight.

$\Rightarrow$ f is a kernel expansion over training set

- Representer Thm. into ERM $\Rightarrow$ opt. over $\mathcal{H}$ reduces to $\mathbf{w} \in \mathbb{R}^N$

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

$$= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N} \ell(\mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n), y_n) + \frac{\lambda}{2}\mathbf{w}^T \mathbf{K}_{\mathbf{X},\mathbf{X}}\mathbf{w}$$

Function Representation

- Consider empirical risk minimization case: sample size $N < \infty$
- Representer Theorem:

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

$\Rightarrow$ $\mathbf{x}_n$ are feature vectors, and w_n is a scalar weight.

$\Rightarrow$ f is a kernel expansion over training set

- Example: kernel logistic regression $\mathbb{P}\,(y = 0 \mid \mathbf{x}) = \frac{\exp\{f(\mathbf{x})\}}{1+\exp\{f(\mathbf{x})\}}$.

$$= \underset{f \in \mathcal{H}}{\text{argmin}} \frac{1}{N} \sum_{n=1}^{N} \left[\log\left(1 + \exp\{f(\mathbf{x}_n)\}\right) - \mathbb{I}(y_n = 1) - f(\mathbf{x}_n)\mathbb{I}(y_n = 0) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \right]$$

$$= \underset{\mathbf{w} \in \mathbb{R}^N}{\text{argmin}} \frac{1}{N} \sum_{n=1}^{N} \left[\log\left(1 + \exp\{\mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n)\}\right) - \mathbb{I}(y_n = 1) - \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n)\mathbb{I}(y_n = 0) + \frac{\lambda}{2}\mathbf{w}^T \mathbf{K}_{\mathbf{X},\mathbf{X}}\mathbf{w} \right]$$

Function Representation

- Consider empirical risk minimization case: sample size $N < \infty$
- Representer Theorem:

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

$\Rightarrow$ $\mathbf{x}_n$ are feature vectors, and w_n is a scalar weight.

$\Rightarrow$ f is a kernel expansion over training set

- Unfortunately, as sample size $N \to \infty$

$\Rightarrow$ kernel matrix $[\mathbf{K}_{\mathbf{X},\mathbf{X}}]_{m,n} := \kappa(\mathbf{x}_m, \mathbf{x}_n)$ becomes infinite too!

$\Rightarrow$ $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \cdots]$ $\Rightarrow$ kernel dictionary

$\Rightarrow$ $\boldsymbol{\kappa}_{\mathbf{X}}(\cdot) = [\kappa(\mathbf{x}_1, \cdot) \ldots \kappa(\mathbf{x}_N, \cdot)]^T$ $\Rightarrow$ empirical kernel map

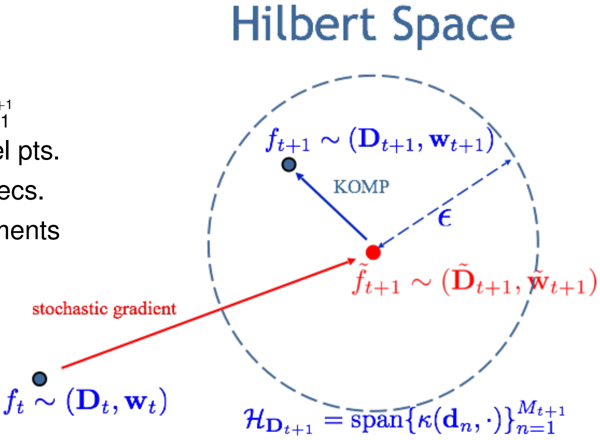
$\Rightarrow$ model order $M(= N) \to \infty$ $\Rightarrow$ number of dictionary columns

- Storage/representation issue $\Rightarrow$ “the curse of kernelization”

Kernel Matching Pursuit

- Fix approx. error ϵ_t
- Define subspace $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_{t+1}}$
- $\{\mathbf{d}_n\} \subset \{\mathbf{x}_u\}_{u \leq t} \Rightarrow$ model pts.

  $\Rightarrow$ subset of past feat. vecs.
- Remove kernel dict. elements
- Stopping criterion: $\|\tilde{f}_{t+1} - f_{t+1}\|_{\mathcal{H}} \leq \epsilon_t$
- New model order: $M_{t+1} \leq M_t + 1$



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

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

- 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 $\{\mathbf{x}_u\}_{u \leq t}$

Kernel Orthogonal Matching Pursuit (KOMP)

Require: function $\tilde{f}$ defined by 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 $\mathbf{D}$

$$\mathbf{w} = \underset{\mathbf{w}\in\mathbb{R}^M}{\text{argmin}} \|\tilde{f}(\cdot) - \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}}$$

end

end while

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

Online Kernel Multi-Class SVM

- Given feature vectors $\mathbf{x}$ with labels $y \in \{1, \ldots, C\}$

  $\Rightarrow$ Classifier $\Rightarrow$ optimize a geometric criterion of separation
- Kernel multi-class SVM $\Rightarrow$ for class c, 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

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

$\Rightarrow$ $\ell(\mathbf{f}(\mathbf{x}), y) = \max(0, 1 + f_r(\mathbf{x}) - f_y(\mathbf{x}))$, $r = \operatorname{argmax}_{c' \neq y} f_{c'}(\mathbf{x})$.

Online Kernel Multi-Class SVM

- Parameter selection:
  - $\Rightarrow$ Gaussian kernel with bandwidth $\tilde{\sigma}^2 = 0.6$
  - $\Rightarrow$ regularizer $\lambda = 10^{-6}$, 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., $f_0 = 0$