Distributed Learning in Wireless Sensor Networks

[J.B. Predd, S.R. Kulkarni, H.V. Poor]

Georg Piewald

Advanced Signal Processing Seminar, WS08

Distributed Learning in WSNs
Georg Piewald
Motivation
Classical Learning
The Supervised Learning Model
Kernel Methods
Distributed Learning in WSNs
Incremental Subgradient Methods
Alternating Projection Algorithm

Outline

1. Motivation
2. Classical Learning
   - The Supervised Learning Model
   - Kernel Methods
3. Distributed Learning in WSNs
   - Incremental Subgradient Methods
   - Alternating Projection Algorithm

Wireless Sensor Networks and Machine Learning

A strange combination?

- Sensor networks collect data.
- Machine learning extracts information from given data.
- In principle, two independent tasks.

Fundamental Problems

- Wireless sensor networks:
  - Tight energy constraints
  - Limited communication capabilities
- Machine learning:
  - Process of data acquisition is disregarded
  - Algorithms depend critically on instantaneous availability of training data

Machine learning

- Supervised learning
  - Generate a function that maps input to output data
  - Based on some training data
  - Kernel methods, boosting, nearest-neighbour rules...
- Unsupervised learning
  - No training data available
  - E.g. data clustering, principle component analysis...
- others

Distributed Learning in WSNs
Georg Piewald
Motivation
Classical Learning
The Supervised Learning Model
Kernel Methods
Distributed Learning in WSNs
Incremental Subgradient Methods
Alternating Projection Algorithm

Supervised Learning

Terminology

$\mathcal{X}$ feature, input or observation space
e.g. $\mathcal{X} \subseteq \mathbb{R}^3$, 2 space and 1 time dimension

$\mathcal{Y}$ label, output, target or parameter space
e.g. $\mathcal{Y} \subseteq \mathbb{R}$, measured temperature

Distinguish between:

- Classification (detection)
  e.g. $\mathcal{Y} = \{0, 1\}$
- Regression (estimation)
  e.g. $\mathcal{Y} \subseteq \mathbb{R}^d$

Loss Function

Find a function $g : \mathcal{X} \to \mathcal{Y}$ that minimizes

$$E\{l(g(X), Y)\} \qquad X \in \mathcal{X}, \quad Y \in \mathcal{Y}$$

Given a loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$

Typical loss functions:

- Regression: squared error

  $l(y, y') := |y - y'|^2$
- Classification: zero-one loss

  $l(y, y') := \begin{cases} 0 & \text{if} \quad y = y' \\ 1 & \text{if} \quad y \neq y' \end{cases}$

Distributed Learning in WSNs
Georg Piewald
Motivation
Classical Learning
The Supervised Learning Model
Kernel Methods
Distributed Learning in WSNs
Incremental Subgradient Methods
Alternating Projection Algorithm

Parametric vs. Nonparametric Models

Risk function:
$R(g) = E\{l(g(X), Y)\} = \sum_i l(g(X), Y) P_{XY}$

- Parametric setting:
  - Joint probability distribution P_{XY} known
  - Regression: $g(x) = E\{Y|X = x\}$
  - Classification: MAP decision rule
- Nonparametric setting:
  - P_{XY} unknown, but training data available
  - $S_n = \{(x_i, y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$
  - Empirical risk minimization

    $\tilde{R}_n(g) = \frac{1}{n} \sum_{i=1}^n l(g(x_i), y_i)$

Kernel Design

Design a *kernel* as a similarity measure for inputs:

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

Examples:

- Linear kernel: $K(x, x') := x^T x'$
- Naive kernel: $K(x, x') := \begin{cases} 1 & \text{if} \quad \|x - x'\| \leq r_n \\ 0 & \text{otherwise} \end{cases}$
- Gaussian kernel: $K_\sigma(x, x') = \exp(-\frac{1}{2\sigma^2}\|x - x'\|_2^2)$

Distributed Learning in WSNs
Georg Piewald
Motivation
Classical Learning
The Supervised Learning Model
Kernel Methods
Distributed Learning in WSNs
Incremental Subgradient Methods
Alternating Projection Algorithm

Construct an Estimator

Given a kernel, construct an estimator g_n:

$$g_n(X) = g_n(X, S_n) = \begin{cases} \frac{\sum_{i=1}^n K(X,X_i) Y_i}{\sum_{i=1}^n K(X,X_i)} & \text{if } \sum_{i=1}^n K(X, X_i) > 0 \\ 0 & \text{otherwise} \end{cases}$$

As $n \to \infty$, it can be proven that $\tilde{R}_n(g) \to R(g)$.

Reproducing Kernel Method

Drawback of previous approach: arbitrarily slow convergence.

Alternative: *reproducing kernel method*

$$\min_{f \in \mathcal{H}_K} \left[\frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2 + \lambda \|f\|^2_{\mathcal{H}_K} \right]$$

- $\frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$ empirical risk
- $\|f\|^2_{\mathcal{H}_K}$ complexity control
- $\lambda \in \mathbb{R}_+$ tradeoff factor
- $\mathcal{H}_K$ reproducing kernel Hilbert space

Training Distributively using Incremental Subgradient Methods

$$\min_{f \in \mathcal{H}_K} \left[\frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2 + \lambda \|f\|^2_{\mathcal{H}_K} \right]$$

$$F(f) := \sum_{i=1}^{n} \left((f(X_i) - Y_i)^2 + \lambda_i \|f\|^2_{\mathcal{H}_K} \right)$$

Gradient descent algorithm:

$$\hat{f}^{(k+1)} = \hat{f}^{(k)} - \alpha_k \frac{\partial F}{\partial f} \left(\hat{f}^{(k)} \right)$$

Training Distributively using Incremental Subgradient Methods

$$\hat{f}^{(k+1)} = \hat{f}^{(k)} - \alpha_k \frac{\partial G_j}{\partial f}\left(\hat{f}^{(k)}\right)$$

$$G_j(f) := (f(X_j) - Y_j)^2 + \lambda_j \|f\|^2_{\mathcal{H}_K}$$

$$j = k \bmod m$$

Distributed Learning in WSNs

Georg Piewald

Motivation

Classical Learning

The Supervised Learning Model

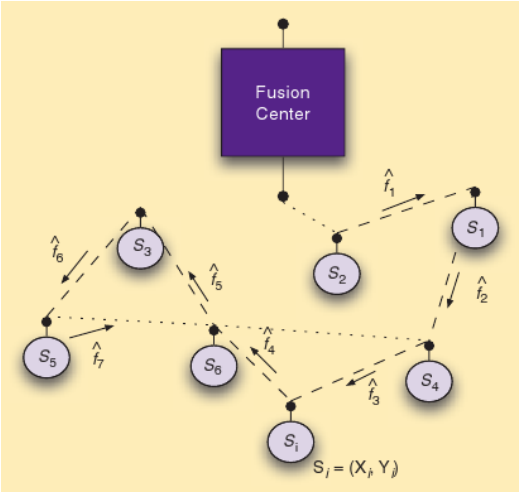
Kernel Methods

Distributed Learning in WSNs

Incremental Subgradient Methods

Alternating Projection Algorithm

1. Choose a path through the network
2. Initialize $\hat{f}^{(0)} = 0 \in \mathcal{H}_K$
3. Every node j along the path
   - receives $\hat{f}^{(k)}$
   - calculates $\hat{f}^{(k+1)}$ using (X_j, Y_j)
   - transmits $\hat{f}^{(k+1)}$ to the next node in the path



Training Distributively with Alternating Projections

Local estimation:

$$\min_{f \in \mathcal{H}_K} \left[\sum_{j \in N_i} (f(x_j) - y_j)^2 + \lambda_i \|f\|^2_{\mathcal{H}_K} \right]$$

N_i set of neighbours of sensor i

(x_j, y_j) training sample of neighbour j

$z_i \in \mathcal{Y}$ estimate of the field at X_i

Distributed Learning in WSNs
Georg Piewald
Motivation
Classical Learning
The Supervised Learning Model
Kernel Methods
Distributed Learning in WSNs
Incremental Subgradient Methods
Alternating Projection Algorithm

Algorithm outline:

1. each sensor initializes $z_i = y_i$
2. sensor i queries $\{(x_j, z_j)\}_{j \in N_i}$ from all neighbours
3. uses this as training data to compute $f_i \in \mathcal{H}_K$
4. calculates $z_j = f_i(x_j)$ for all $j \in N_i$
5. writes z_j back to the neighbours
6. repeat in multiple iterations

Properties:

- Local information propagates globally
- Sensors share data, not functions
- Parallelization possible

References I


J.B. Predd, S.R. Kulkarni, H.V. Poor
Distributed Learning in Wireless Sensor Networks.
IEEE Signal Processing Magazine, July 2006


M.G. Rabbat, R.D. Nowak
Quantized incremental algorithms for distributed optimization
IEEE J. Special Areas Commun., vol. 23, 2006


J.B. Predd, S.R. Kulkarni, H.V. Poor
Regression in Sensor Networks: Training distributively with alternating projections
Proc. SPIE Conf., July-Aug. 2005