Online Semi-Supervised Learning

Andrew B. Goldberg, Ming Li, Xiaojin Zhu*

jerryzhu@cs.wisc.edu
Computer Sciences
University of Wisconsin–Madison
Life-long learning

\[ x_1, x_2, \ldots, x_{1000}, \ldots, x_{1000000}, \ldots \]

\[ y_1 = 0, - , - , y_{1000} = 1, \ldots, y_{1000000} = 0, \ldots \]
This is how children learn, too

\[ x_1 \quad x_2 \quad \ldots \quad x_{1000} \quad \ldots \quad x_{1000000} \quad \ldots \]

\[ y_1 = 0 \quad \ldots \quad y_{1000} = 1 \quad \ldots \quad y_{1000000} = 0 \quad \ldots \]

Unlike standard supervised learning:

- \( n \to \infty \) examples arrive sequentially, cannot even store them all
- most examples unlabeled
- no iid assumption, \( p(x, y) \) can change over time
New paradigm: online semi-supervised learning

Main contribution: merging

1. *online learning*: learn non-iid sequentially, but fully labeled
2. *semi-supervised learning*: learn from labeled and unlabeled data, but in batch mode

1. At time $t$, adversary picks $x_t \in \mathcal{X}, y_t \in \mathcal{Y}$ not necessarily iid, shows $x_t$
2. Learner has classifier $f_t : \mathcal{X} \mapsto \mathbb{R}$, predicts $f_t(x_t)$
3. With small probability, adversary reveals $y_t$; otherwise it abstains (unlabeled)
4. Learner updates to $f_{t+1}$ based on $x_t$ and $y_t$ (if given). Repeat.
Review: batch manifold regularization

A form of graph-based semi-supervised learning [Belkin et al. JMLR06]:

- Graph on $x_1 \ldots x_n$
- Edge weights $w_{st}$ encode similarity between $x_s, x_t$, e.g., $k$NN
- Assumption: similar examples have similar labels

Manifold regularization minimizes risk:

$$J(f) = \frac{1}{l} \sum_{t=1}^{T} \delta(y_t)c(f(x_t), y_t) + \frac{\lambda_1}{2} \|f\|_K^2 + \frac{\lambda_2}{2T} \sum_{s,t=1}^{T} (f(x_s) - f(x_t))^2 w_{st}$$

$c(f(x), y)$ convex loss function, e.g., the hinge loss.
Solution $f^* = \text{arg min}_f J(f)$.
Generalizes graph mincut and label propagation.
From batch to online

**batch risk = average instantaneous risks**

\[ J(f) = \frac{1}{T} \sum_{t=1}^{T} J_t(f) \]

**Batch risk**

\[ J(f) = \frac{1}{l} \sum_{t=1}^{T} \delta(y_t) c(f(x_t), y_t) + \frac{\lambda_1}{2} \| f \|^2_K + \frac{\lambda_2}{2T} \sum_{s,t=1}^{T} (f(x_s) - f(x_t))^2 w_{st} \]

**Instantaneous risk**

\[ J_t(f) = \frac{T}{l} \delta(y_t) c(f(x_t), y_t) + \frac{\lambda_1}{2} \| f \|^2_K + \lambda_2 \sum_{i=1}^{t} (f(x_i) - f(x_t))^2 w_{it} \]

(includes graph edges between \( x_t \) and all previous examples)
Online convex programming

Instead of minimizing convex $J(f)$, reduce convex $J_t(f)$ at each step $t$.

\[ f_{t+1} = f_t - \eta_t \frac{\partial J_t(f)}{\partial f} \bigg|_{f_t} \]

Remarkable no regret guarantee against adversary:

- Accuracy can be arbitrarily bad if adversary flips target often
- If so, no batch learner in hindsight can do well either

\[ \text{regret} \equiv \frac{1}{T} \sum_{t=1}^{T} J_t(f_t) - J(f^*) \]

[Zinkevich ICML03] No regret: $\limsup_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} J_t(f_t) - J(f^*) \leq 0$.

If no adversary (iid), the average classifier $\bar{f} = 1/T \sum_{t=1}^{T} f_t$ is good: $J(\bar{f}) \to J(f^*)$. 

Kernelized algorithm

\[ f_t(\cdot) = \sum_{i=1}^{t-1} \alpha_i^{(t)} K(x_i, \cdot) \]

- **Init**:  \( t = 1, f_1 = 0 \)
- **Repeat**
  1. receive \( x_t \), predict \( f_t(x_t) = \sum_{i=1}^{t-1} \alpha_i^{(t)} K(x_i, x_t) \)
  2. occasionally receive \( y_t \)
  3. update \( f_t \) to \( f_{t+1} \) by
     \[
     \alpha_i^{(t+1)} = (1 - \eta_t \lambda_1) \alpha_i^{(t)} - 2\eta_t \lambda_2 (f_t(x_i) - f_t(x_t)) w_{it}, \quad i < t
     \]
     \[
     \alpha_t^{(t+1)} = 2\eta_t \lambda_2 \sum_{i=1}^{t} (f_t(x_i) - f_t(x_t)) w_{it} - \eta_t \frac{T}{l} \delta(y_t) c'(f(x_t), y_t)
     \]
  4. store \( x_t \), let \( t = t + 1 \)
Sparse approximation

The algorithm is impractical

- space $O(T)$: stores all previous examples
- time $O(T^2)$: each new example compared to all previous ones
- $T \rightarrow \infty$

Two ways to speed up:

- buffering, or
- random projection tree
Sparse approximation 1: buffering

Keep a size $\tau$ buffer

- approximate representers: $f_t = \sum_{i=t-\tau}^{t-1} \alpha_i^{(t)} K(x_i, \cdot)$
- approximate instantaneous risk

$$J_t(f) = \frac{T}{l} \delta(y_t) c(f(x_t), y_t) + \frac{\lambda_1}{2} \|f\|_K^2 + \lambda_2 \frac{t}{\tau} \sum_{i=t-\tau}^{t} (f(x_i) - f(x_t))^2 w_{it}$$

- dynamic graph on examples in the buffer
Sparse approximation 1: buffer update

- At each step, start with the current $\tau$ representers:

$$f_t = \sum_{i=t-\tau}^{t-1} \alpha_i^{(t)} K(x_i, \cdot) + 0K(x_t, \cdot)$$

- Gradient descent on $\tau + 1$ terms:

$$f' = \sum_{i=t-\tau}^{t} \alpha'_i K(x_i, \cdot)$$

- Reduce to $\tau$ representers $f_{t+1} = \sum_{i=t-\tau+1}^{t} \alpha_i^{(t+1)} K(x_i, \cdot)$ by

$$\min_{\alpha^{(t+1)}} \|f' - f_{t+1}\|^2$$

- Kernel matching pursuit
Sparse approximation 2: random projection tree

[Dasgupta and Freund, STOC08]

- Discretize data manifold by online clustering.
- When a cluster accumulates enough examples, split along random hyperplane.
- Extends $k$-d tree.
Sparse approximation 2: random projection tree

We use the clusters $\mathcal{N}(\mu_i, \Sigma_i)$ as representers:

$$f_t = \sum_{i=1}^{s} \beta_i^{(t)} K(\mu_i, \cdot)$$

“Cluster graph” edge weight between a cluster $\mu_i$ and example $x_t$ is

$$w_{\mu_i t} = \mathbb{E}_{x \sim \mathcal{N}(\mu_i, \Sigma_i)} \left[ \exp \left( -\frac{\|x - x_t\|^2}{2\sigma^2} \right) \right]$$

$$= \left(2\pi\right)^{-\frac{d}{2}} |\Sigma_i|^{-\frac{1}{2}} |\Sigma_0|^{-\frac{1}{2}} \left| \tilde{\Sigma} \right|^{-\frac{1}{2}}$$

$$\exp \left( -\frac{1}{2} \left( \mu_i^\top \Sigma_i^{-1} \mu_i + x_t^\top \Sigma_0^{-1} x_t - \tilde{\mu}^\top \tilde{\Sigma} \tilde{\mu} \right) \right)$$

A further approximation is

$$w_{\mu_i t} = e^{-\|\mu_i - x_t\|^2 / 2\sigma^2}$$

Update $f$ (i.e., $\beta$) and the RPtree, discard $x_t$. 
Experiment: runtime

Buffering and RPtree scales linearly, enabling life-long learning.

Spirals

MNIST 0 vs. 1

- Batch MR
- Online MR
- Online MR (buffer)
- Online RPtree
Experiment: risk

Online MR risk $J_{\text{air}}(T) \equiv \frac{1}{T} \sum_{t=1}^{T} J_t(f_t)$ approaches batch risk $J(f^*)$ as $T$ increases.

![Graph showing the relationship between risk and T for different methods: $J(f^*)$ Batch MR, $J_{\text{air}}(T)$ Online MR, $J_{\text{air}}(T)$ Online MR (buffer), and $J_{\text{air}}(T)$ Online RPtree.]
Experiment: generalization error of $\tilde{f}$ if iid

A variation of buffering as good as batch MR (preferentially keep labeled examples, but not their labels, in buffer).

(a) Spirals  
(b) Face  
(c) MNIST 0 vs. 1  
(d) MNIST 1 vs. 2
Experiment: adversarial concept drift

- Slowly rotating spirals, both $p(x)$ and $p(y|x)$ changing.
- Batch $f^*$ vs. online MR buffering $f_T$
- Test set drawn from the current $p(x,y)$ at time $T$. 

![Graph showing generalization error rate with time $T$.]
Conclusions

- Online semi-supervised learning framework
- Sparse approximations: buffering and RPtree
- Future work: new bounds, new algorithms (e.g., S3VM)