Dual Averaging Method for Online Graph-structured Sparsity

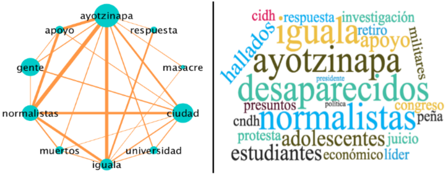
Baojian Zhou[1,2], Feng Chen[1], and Yiming Ying[2]

08/08/2019

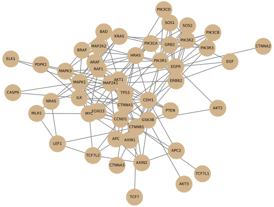
[1]Department of Computer Science,
[2]Department of Mathematics and Statistics,
University at Albany, NY, USA

Graph data is everywhere!

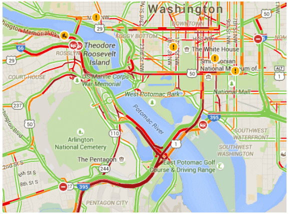
Social network



Keywords graph of social event



PPI network

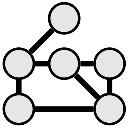


Transport network

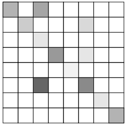
Motivation

We often encounter the following learning scenario:

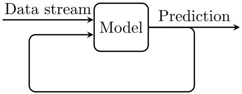
- Data samples $\{\boldsymbol{x}_t, y_t\}$ are available on the fly: at each round, the model makes a prediction based on current input sample.
- Data dimension is high, but only a small part of features is important. This small part of features is graph-structured (connectivity, density, etc) based on the graph information.



Graph-structured



Sparse



Online

Research Question

How do we learn such graph-structured sparse models under online setting ?

Problem Formulation

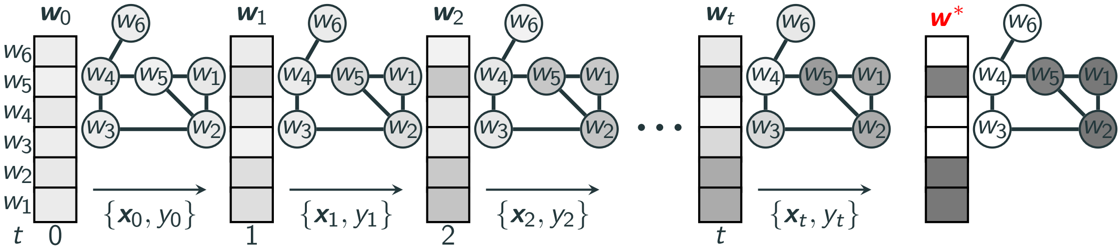
Under online learning setting, at each time t, the learner

1. receives question $\boldsymbol{x}_t \in \mathbb{R}^p$ and makes a prediction
2. receives a loss $f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\})$ after true label y_t is revealed
3. updates $\boldsymbol{w}_t$

Problem Formulation

Under online learning setting, at each time t, the learner

1. receives question $\boldsymbol{x}_t \in \mathbb{R}^p$ and makes a prediction
2. receives a loss $f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\})$ after true label y_t is revealed
3. updates $\boldsymbol{w}_t$



$\boldsymbol{w}^*$ is graph-structured (e.g. connectivity)!

Problem Formulation

Minimize the regret subject to a graph-structured sparsity constraint

$$R(T) := \sum_{t=1}^{T} \left\{ f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\}) \right\} - \min_{\boldsymbol{w} \in \mathcal{M}(\mathbb{M})} \sum_{t=1}^{T} \left\{ f_t(\boldsymbol{w}, \{\boldsymbol{x}_t, y_t\}) \right\}$$

- Regret
- Accumulated Loss
- Minimum Loss in hindsight
- Graph-structured sparsity set $\mathcal{M}(\mathbb{M})$

First Try: Online Projected Gradient Descent(PGD)

The online PGD algorithm updates $\boldsymbol{w}_t$ as the following

$$\boldsymbol{w}_{t+1} = \mathrm{P}(\boldsymbol{w}_t - \eta_t \nabla f_t\left(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\}\right), \mathcal{M}(\mathbb{M}))$$

First Try: Online Projected Gradient Descent(PGD)

The online PGD algorithm updates $\boldsymbol{w}_t$ as the following

$$\boldsymbol{w}_{t+1} = \mathrm{P}(\boldsymbol{w}_t - \eta_t \nabla f_t\left(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\}\right), \mathcal{M}(\mathbb{M}))$$

It hardly works due to **two main drawbacks**

1. It is unable to exploit the problem structure.
2. The new information is vanishing as steps $\eta_t \to 0$.

Challenge: Can we exploit the problem structure more effectively ?

Yes: exploit the structure via the dual space

Inspired by dual avaraging [Xiao, 2010], our method updates $\boldsymbol{w}_t$ as the following

$$\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w} \in \mathcal{M}(\mathbb{M})} \left\{ \left\langle \frac{1}{t+1} \sum_{i=0}^{t} \boldsymbol{g}_i, \boldsymbol{w} \right\rangle + \frac{1}{2\sqrt{t}} \|\boldsymbol{w}\|^2 \right\},$$

- $\boldsymbol{g}_i \in \partial f_i(\boldsymbol{w}_i, \{\boldsymbol{x}_i, y_i\})$, each gradient is equivalently important
- $\frac{1}{t+1} \sum_{i=0}^{t} \boldsymbol{g}_i$ the average of the previous gradients — Dual Averaging
- tie-breaking rule: break ties arbitrarily

Yes: exploit the structure via the dual space

Inspired by dual avaraging [Xiao, 2010], our method updates $\boldsymbol{w}_t$ as the following

$$\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w} \in \mathcal{M}(\mathbb{M})} \left\{ \left\langle \frac{1}{t+1} \sum_{i=0}^{t} \boldsymbol{g}_i, \boldsymbol{w} \right\rangle + \frac{1}{2\sqrt{t}} \|\boldsymbol{w}\|^2 \right\},$$

- $\boldsymbol{g}_i \in \partial f_i(\boldsymbol{w}_i, \{\boldsymbol{x}_i, y_i\})$, each gradient is equivalently important
- $\frac{1}{t+1} \sum_{i=0}^{t} \boldsymbol{g}_i$ the average of the previous gradients — Dual Averaging
- tie-breaking rule: break ties arbitrarily

How to solve this minimization problem ?

Main Theorem

Our method updates $\boldsymbol{w}_t$ as the following

$$\boldsymbol{w}_{t+1} = \operatorname*{arg\,min}_{\boldsymbol{w}\in\mathcal{M}(\mathbb{M})} \left\{ \left\langle \frac{1}{t+1}\sum_{i=0}^{t} \boldsymbol{g}_i, \boldsymbol{w} \right\rangle + \frac{1}{2\sqrt{t}}\|\boldsymbol{w}\|^2 \right\}.$$

Denote the dual averaging $\bar{\boldsymbol{s}}_{t+1} = \frac{1}{t+1}\sum_{i=0}^{t} \boldsymbol{g}_i$, it can be expressed two equivalent problems:

$$\textbf{Problem 1}: \min_{S\in\mathbb{M}} \| -\sqrt{t}\bar{\boldsymbol{s}}_{t+1} - P(-\sqrt{t}\bar{\boldsymbol{s}}_{t+1}, S)\|^2.$$

$$\textbf{Problem 2}: \max_{S\in\mathbb{M}} \|P(-\sqrt{t}\bar{\boldsymbol{s}}_{t+1}, S)\|^2$$

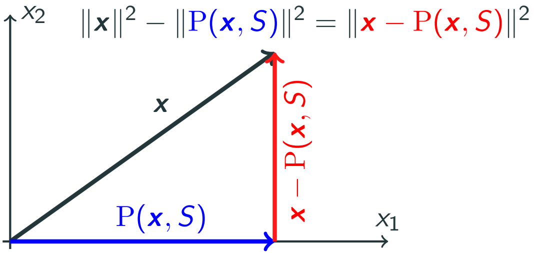
Theorem Insight: Problem 1

The original minimization problem can be equivalently expressed as

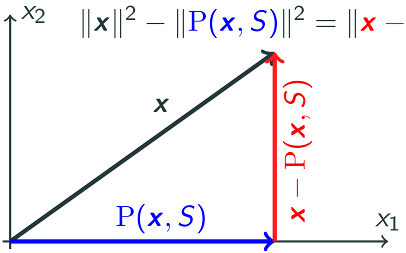
$$\begin{aligned}\boldsymbol{w}_{t+1} &= \arg\min_{\boldsymbol{w}\in\mathcal{M}(\mathbb{M})} \left\{ \langle \bar{\boldsymbol{s}}_{t+1}, \boldsymbol{w} \rangle + \frac{1}{2\sqrt{t}} \|\boldsymbol{w}\|^2 \right\} \\ &= \arg\min_{\boldsymbol{w}\in\mathcal{M}(\mathbb{M})} \left\| \boldsymbol{w} - \left(-\sqrt{t}\bar{\boldsymbol{s}}_{t+1} \right) \right\|^2,\end{aligned}$$

Each step is essentially a projection !

Theorem Insight: Problem 2



Theorem Insight: Problem 2



Step 1: Let $\boldsymbol{x} := -\sqrt{t}\bar{\boldsymbol{s}}_{t+1}$ and add min to both sides

$$\min_{S \in \mathbb{M}} \left\{ \|\boldsymbol{x}\|^2 - \|\mathrm{P}(\boldsymbol{x}, S)\|^2 \right\} = \min_{S \in \mathbb{M}} \|\boldsymbol{x} - \mathrm{P}(\boldsymbol{x}, S)\|^2.$$

Step 2: Move min into the negative term

$$\|\boldsymbol{x}\|^2 + \max_{S \in \mathbb{M}} \|\mathrm{P}(\boldsymbol{x}, S)\|^2 = \min_{S \in \mathbb{M}} \|\boldsymbol{x} - \mathrm{P}(\boldsymbol{x}, S)\|^2.$$

Online Graph Dual Averaging Algorithm

GraphDA

Input: $\mathbb{M}$

$\bar{\boldsymbol{s}}_0 = \boldsymbol{0}, \boldsymbol{w}_0 = \boldsymbol{0}$

for $t = 0, 1, 2, \ldots$ **do**

  receive $\{\boldsymbol{x}_t, y_t\}$ and make prediction

  compute $\boldsymbol{g}_t = \nabla f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\})$

  $\bar{\boldsymbol{s}}_{t+1} = \bar{\boldsymbol{s}}_t + \boldsymbol{g}_t$

  $\boldsymbol{b}_{t+1} = \mathrm{P}(\bar{\boldsymbol{s}}_{t+1}, \mathbb{M})$

  $\boldsymbol{w}_{t+1} = \mathrm{P}(-\sqrt{t}\boldsymbol{b}_{t+1}, \mathbb{M})$

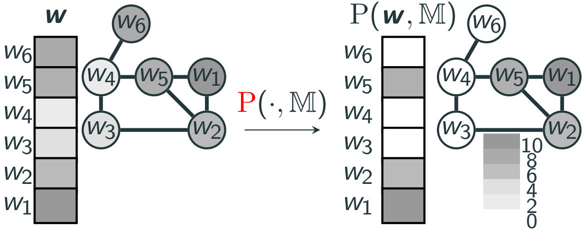
end for

Online Graph Dual Averaging Algorithm

GraphDA

> **Input**: $\mathbb{M}$
>
> $\bar{\boldsymbol{s}}_0 = \boldsymbol{0}, \boldsymbol{w}_0 = \boldsymbol{0}$
>
> **for** $t = 0, 1, 2, \ldots$ **do**
>
>   receive $\{\boldsymbol{x}_t, y_t\}$ and make prediction
>
>   compute $\boldsymbol{g}_t = \nabla f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\})$
>
> $\bar{\boldsymbol{s}}_{t+1} = \bar{\boldsymbol{s}}_t + \boldsymbol{g}_t$
>
> $\boldsymbol{b}_{t+1} = \mathrm{P}(\bar{\boldsymbol{s}}_{t+1}, \mathbb{M})$
>
> $\boldsymbol{w}_{t+1} = \mathrm{P}(-\sqrt{t}\boldsymbol{b}_{t+1}, \mathbb{M})$
>
> **end for**

Let $\mathbb{M} = \{S : |S| \leq 3, S \text{ is connected }\}$. Finding a connected subgraph up to 3 nodes.



Graph Projection Operator [Hegde et al., 2015]

What if the graph information is not available ?

DA-IHT

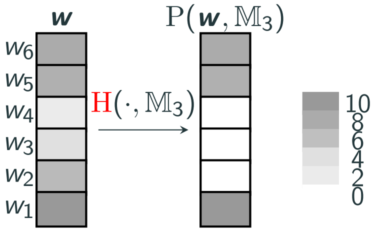
1: **Input**: $\mathbb{M}$
2: $\bar{\boldsymbol{s}}_0 = \mathbf{0}$, $\boldsymbol{w}_0 = \mathbf{0}$
3: **for** $t = 0, 1, 2, \ldots$ **do**
4: receive $\{\boldsymbol{x}_t, y_t\}$ and make a prediction
5: compute $\boldsymbol{g}_t = \nabla f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\})$
6: $\bar{\boldsymbol{s}}_{t+1} = \bar{\boldsymbol{s}}_t + \boldsymbol{g}_t$
7: $\boldsymbol{b}_{t+1} = \mathrm{H}(\bar{\boldsymbol{s}}_{t+1}, \mathbb{M})$
8: $\boldsymbol{w}_{t+1} = \mathrm{H}(-\sqrt{t}\boldsymbol{b}_{t+1}, \mathbb{M})$
9: **end for**

What if the graph information is not available ?

DA-IHT

Input: $\mathbb{M}$
$\bar{\boldsymbol{s}}_0 = \mathbf{0}, \boldsymbol{w}_0 = \mathbf{0}$
for $t = 0, 1, 2, \ldots$ **do**
  receive $\{\boldsymbol{x}_t, y_t\}$ and make a prediction
  compute $\boldsymbol{g}_t = \nabla f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\})$
  $\bar{\boldsymbol{s}}_{t+1} = \bar{\boldsymbol{s}}_t + \boldsymbol{g}_t$
  $\boldsymbol{b}_{t+1} = \mathrm{H}(\bar{\boldsymbol{s}}_{t+1}, \mathbb{M})$
  $\boldsymbol{w}_{t+1} = \mathrm{H}(-\sqrt{t}\boldsymbol{b}_{t+1}, \mathbb{M})$
end for

Let $\mathbb{M}_3 = \{S : |S| \leq 3\}$. Sorting the magnitudes of $\boldsymbol{w}$ and thresholding entries out of top s to zero.



Hard Thresholding Operator

Time Complexity and Regret

The time complexity of GraphDA mainly depends on two projections. If we use the weighted-graph model, the per-iteration time cost could be

- non-sparse graph: $\mathcal{O}(p + |\mathbb{E}| \log^3(p))$ **Edge-dependent**
- sparse graph: $\mathcal{O}(p + p \log^3(p))$ **Nearly-linear !**

Time Complexity and Regret

The time complexity of GraphDA mainly depends on two projections. If we use the weighted-graph model, the per-iteration time cost could be

- non-sparse graph: $\mathcal{O}(p + |\mathbb{E}| \log^3(p))$ **Edge-dependent**
- sparse graph: $\mathcal{O}(p + p \log^3(p))$ **Nearly-linear !**

If $\mathcal{M}(\mathbb{M})$ is a **convex set**, then

- The regret can be bounded as: $R(T, \mathcal{M}(\mathbb{M})) = C \cdot \mathcal{O}(\sqrt{T})$, where C is a constant.
- If we assume further that the loss is strongly convex, then $\|\boldsymbol{w}_T - \boldsymbol{w}^*\|_2^2 = \mathcal{O}(\frac{\ln T}{T})$.

If $\mathcal{M}(\mathbb{M})$ is a **nonconvex set**, can we still obtain a non-regret bound ?

Experimental setup

We compare GraphDA with baseline methods by using three datasets.

Method	Proposed in
Adam	Kingma and Ba [2014]
ℓ_1-RDA	Xiao [2010]
DA-GL	Yang et al. [2010]
DA-SGL	Yang et al. [2010]
AdaGrad	Duchi et al. [2011]
StoIHT	Nguyen et al. [2017]
GraphStoIHT	Zhou et al. [2019]
DA-IHT	**This paper**
GraphDA	**This paper**

Dataset	$\lvert\mathbb{V}\rvert$	$\lvert\mathbb{E}\rvert$
Benchmark	1,089	2,112
MNIST	786	1,516
KEGG	5,372	78,545

- non-sparse
- sparse: convex-based
- sparse: nonconvex-based

Two questions

Compared with baseline methods, we aim to answer the following two questions:

- **Q1:** Can GraphDA achieve better classification performance ?
- **Q2:** Can GraphDA learn a stronger interpretable model by capturing meaningful graph-structured features ?

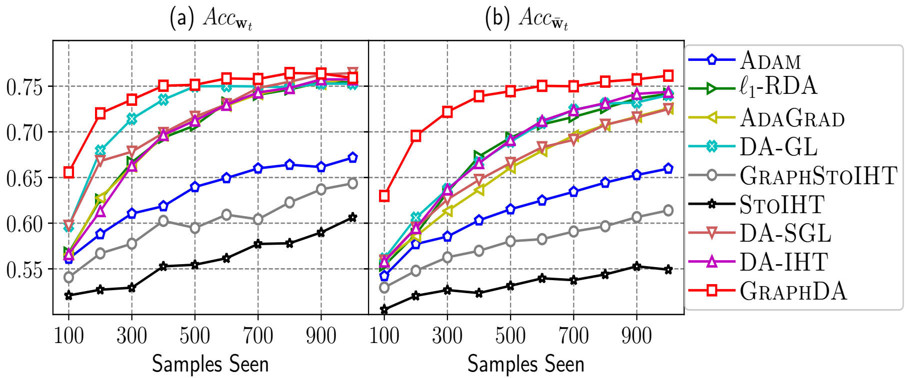
Application 1: event classification on Benchmark dataset

Given the training dataset $\{\boldsymbol{x}_i \in \mathbb{R}^p, y_i \in \{\pm 1\}\}_{i=1}^{t}$ on the fly

- $y_t = -1$: no event ("business-as-usual");
- $y_t = +1$: event: disease outbreak/computer virus etc.

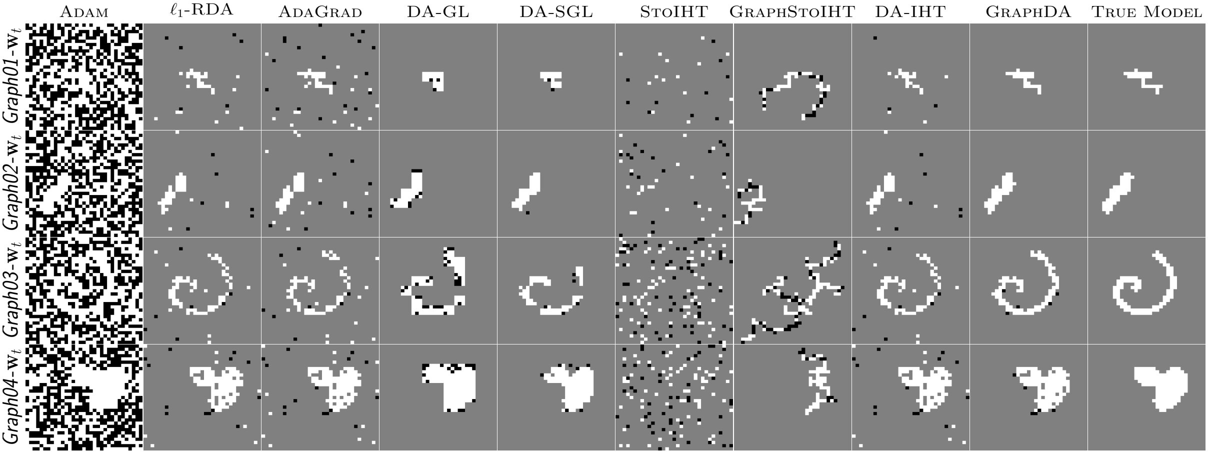
Task: To classify these samples online and at the same time to find the hidden structure on these events!

GraphDA has higher classification accuracy



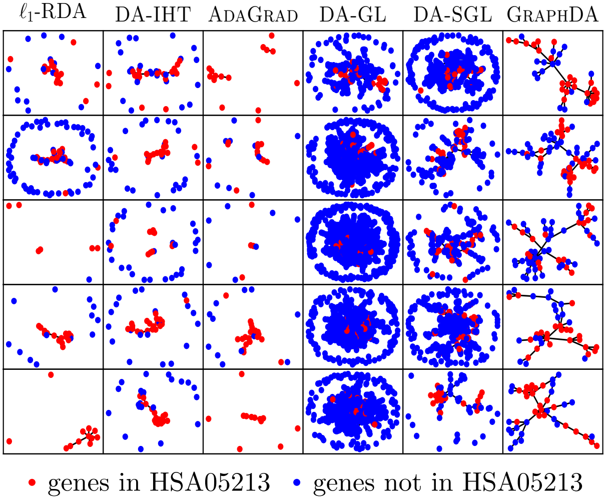
- Online PGD-based: StoIHT and GraphStoIHT **do not work!**
- Online DA-based: ℓ_1-RDA,DA-GL,DA-SGL and DA-IHT work well.
- GraphDA outperforms other DA-based with the help of graph priors.

GraphDA learns more interpretable models

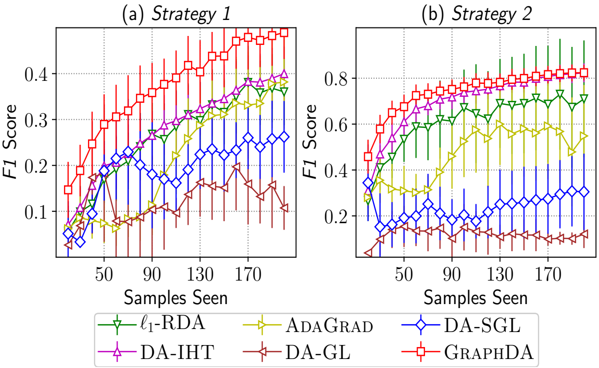


Black dots: $(\boldsymbol{w}_t)_i < 0$; Gray $(\boldsymbol{w}_t)_i = 0$; White dots $(\boldsymbol{w}_t)_i > 0$

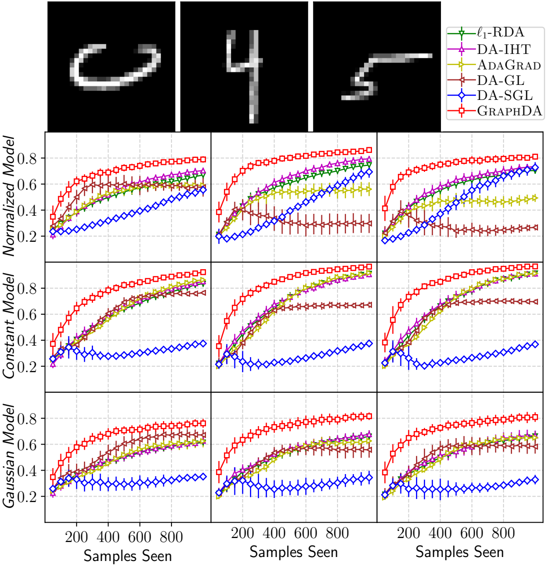
Application 2: gene identification on KEGG dataset



$$F1_{\boldsymbol{w}_t} = \frac{2|\mathrm{supp}(\boldsymbol{w}^*) \cap \mathrm{supp}(\boldsymbol{w}_t)|}{|\mathrm{supp}(\boldsymbol{w}^*)| + |\mathrm{supp}(\boldsymbol{w}_t)|}$$



GraphDA learns more cancer-related genes and more structures (edges).



Use the least square loss $f_t(\boldsymbol{w}_t, \{\boldsymbol{x}_t, y_t\}) = (y_t - \langle \boldsymbol{w}_t, \boldsymbol{x}_t \rangle)^2$.

Samples are generated by using the following linear relation: $y_t = \langle \boldsymbol{x}_t, \boldsymbol{w}^* \rangle$, where $\boldsymbol{x}_t \in \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. We use three different strategies to obtain $\boldsymbol{w}^*$.

Task: To learn the structure of $\boldsymbol{w}^*$!

Summary

Conclusion

- We propose a dual averaging-based method, GraphDA, for online graph-structured sparsity constraint problems.
- We prove that the minimization problem in the dual averaging step can be formulated as two equivalent optimization problems.
- GraphDA achieves better classification performance and stronger interpretability.

Summary

Conclusion

- We propose a dual averaging-based method, GraphDA, for online graph-structured sparsity constraint problems.
- We prove that the minimization problem in the dual averaging step can be formulated as two equivalent optimization problems.
- GraphDA achieves better classification performance and stronger interpretability.

Future work

- Does GraphDA have non-regret bound under some proper assumption ?
- What if true structure of features are time evolving ?

Thank you!

Q & A

Code and datasets: https://github.com/baojianzhou/graph-da

Install GraphDA: **pip install sparse-learn**

Roadmap

- Motivation
- Problem Formulation
- Proposed Algorithm
- Experimental Results
- Conclusion

Implementation Details

Algorithm 1 Head/Tail Projection ($\mathrm{P}(\boldsymbol{w}, \mathbb{M})$) Hegde et al. [2015]

Input: $\boldsymbol{w}, max\_iter, \mathbb{M} = (\mathbb{G}(\mathbb{V}, \mathbb{E}, \boldsymbol{c}), s_l, s_h, g)$
$\boldsymbol{\pi} = \boldsymbol{w} \cdot \boldsymbol{w}$ // vector dot product, i.e., $\pi_i = w_i * w_i$
$\lambda_l = 0, \lambda_h = \max\{\pi_1, \pi_2, \ldots, \pi_p\}$, $\lambda_m = 0$, $t = 0$
repeat
    $\lambda_m = (\lambda_l + \lambda_h)/2$; $\boldsymbol{c}_m = \lambda_m \cdot \boldsymbol{c}$ // scale dot product, i.e., $(\boldsymbol{c}_m)_i = \lambda_m * c_i$
    $\mathcal{F} = \mathrm{PCST}(\mathbb{G}(\mathbb{V}, \mathbb{E}, \boldsymbol{c}_m), \boldsymbol{\pi}, g)$
    if $s_l < |\mathcal{F}| < s_h$ **then** **return** $\boldsymbol{w}_{\mathcal{F}}$;
    if $|\mathcal{F}| > s_h$ **then** $\lambda_l = \lambda_m$ **else** $\lambda_h = \lambda_m$;
    $t = t + 1$
until $t >$ max_iter
$\boldsymbol{c}_h = \lambda_h \cdot \boldsymbol{c}$; $\mathcal{F} = \mathrm{PCST}(\mathbb{G}(\mathbb{V}, \mathbb{E}, \boldsymbol{c}_h), \boldsymbol{\pi}, g)$;
return $\boldsymbol{w}_{\mathcal{F}}$

The code is written in C language with the standard C11.

How to choose the sparsity

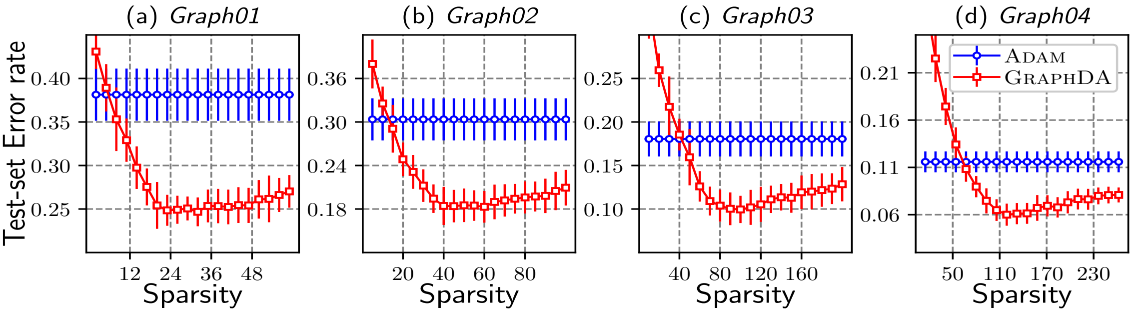


Figure: Test dataset error rates as a function of sparsity s

Parameter Tuning

- ℓ_1-RDA
  - regularization: $\lambda \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5, 1, 3, 5, 10\}$.
  - learning rate (implicit): $\gamma \in \{1, 5, 1e1, 5e1, 1e2, 5e2, 1e3, 5e3, 1e4\}$
  - sparsity-enhancing:
    $\rho \in \{0.0, 0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$
- Adam
  - $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$
  - $\alpha \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$.
- DA-GL/SGL
  - $\lambda \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5, 1, 3, 5, 1e1\}$
  - $\gamma \in \{1, 5, 1e1, 5e1, 1e2, 5e2, 1e3, 5e3, 1e4\}$
  - 3×3 grids as groups for Benchmark dataset.
  - 2×2 grids for MNIST dataset.

Parameter Tuning

- AdaGrad
  - $\lambda \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5, 1, 3, 5, 1e1\}$
  - $\eta \in \{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 1e1, 5e1, 1e2, 5e2, 1e3, 5e3\}$.
- StoIHT
  - sparsity $s \in \{5, 10, \ldots, 150\}$
  - $\gamma \in \{1, 5, 1e1, 5e1, 1e2, 5e2, 1e3, 5e3, 1e4\}$
- GraphStoIHT/GraphDA
  - sparsity $s \in \{5, 10, \ldots, 150\}$
  - $\gamma \in \{1, 5, 1e1, 5e1, 1e2, 5e2, 1e3, 5e3, 1e4\}$

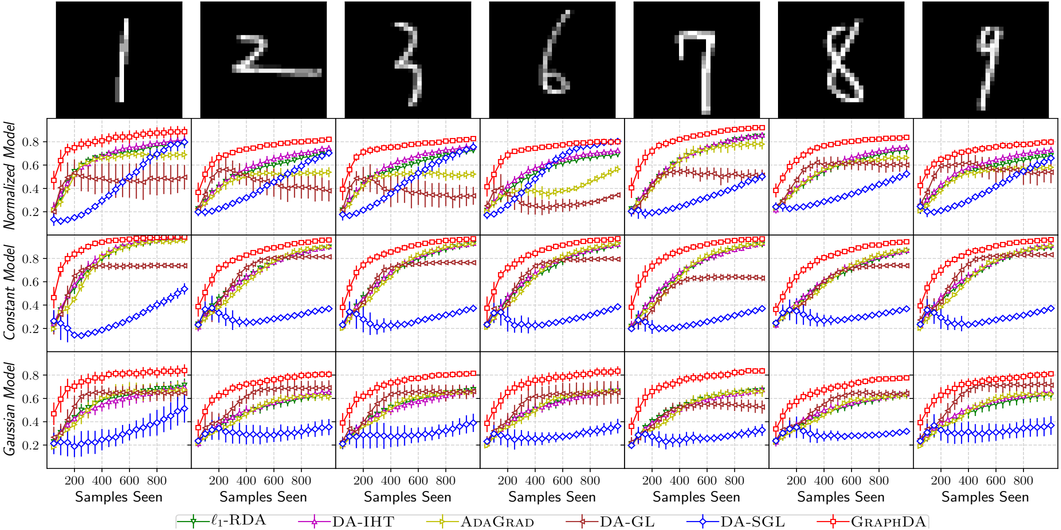
Classification Performance

$$Pre_{\boldsymbol{w}_t} = \frac{|\text{supp}(\boldsymbol{w}^*) \cap \text{supp}(\boldsymbol{w}_t)|}{|\text{supp}(\boldsymbol{w}_t)|},\ Rec_{\boldsymbol{w}_t} = \frac{|\text{supp}(\boldsymbol{w}^*) \cap \text{supp}(\boldsymbol{w}_t)|}{|\text{supp}(\boldsymbol{w}^*)|} F1_{\boldsymbol{w}_t} = \frac{2|\text{supp}(\boldsymbol{w}^*) \cap \text{supp}(\boldsymbol{w}_t)|}{|\text{supp}(\boldsymbol{w}^*)| + |\text{supp}(\boldsymbol{w}_t)|},\ \text{NR}_{\boldsymbol{w}} = \frac{|\text{supp}(\boldsymbol{w})|}{p}.$$

Method	$Pre_{\boldsymbol{w}_t}$	$Rec_{\boldsymbol{w}_t}$	$F1_{\boldsymbol{w}_t}$	$AUC_{\boldsymbol{w}_t, \bar{\boldsymbol{w}}_t}$	$Acc_{\boldsymbol{w}_t, \bar{\boldsymbol{w}}_t}$	$Miss_{\boldsymbol{w}_t, \bar{\boldsymbol{w}}_t}$	$NR_{\boldsymbol{w}_t, \bar{\boldsymbol{w}}_t}$
ADAM	0.024	**1.000**	0.047	(0.618, 0.603)	(0.619, 0.603)	(166.35, 173.10)	(100.0%, 100.0%)
ℓ_1-RDA	0.267	0.863	0.389	(0.693, 0.672)	(0.694, 0.673)	(155.30, 166.05)	(11.58%, 83.60%)
ADAGRAD	0.256	0.877	0.379	(0.696, 0.636)	(0.696, 0.637)	(156.00, 166.00)	(11.33%, 100.0%)
DA-GL	0.176	0.967	0.283	(0.735, 0.666)	(0.735, 0.667)	(142.90, 162.20)	(15.99%, 100.0%)
DA-SGL	0.523	0.854	0.506	(0.699, 0.647)	(0.699, 0.647)	(151.00, 165.50)	(25.54%, 100.0%)
STOIHT	0.057	0.150	0.072	(0.552, 0.523)	(0.553, 0.523)	(194.55, 195.25)	(7.79%, 40.62%)
GRAPHSTOIHT	0.151	0.356	0.194	(0.603, 0.570)	(0.602, 0.570)	(174.65, 181.40)	(7.84%, **22.06**%)
DA-IHT	0.507	0.744	0.566	(0.697, 0.666)	(0.697, 0.666)	(155.65, 162.85)	(4.35%, 39.50%)
GRAPHDA	**0.869**	0.906	**0.880**	(**0.749**, **0.739**)	(**0.749**, **0.739**)	(**133.45**, **136.20**)	(**2.56**%, 32.12%)

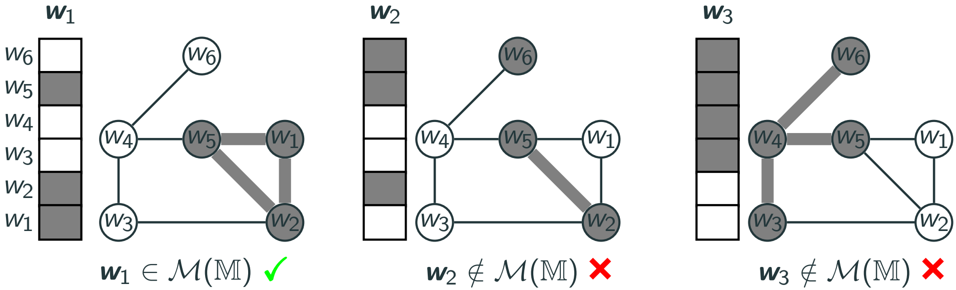
Online Graph Sparse Linear Regression

F1 score as a function of samples seen (2nd to 4th row) on seven handwritten digits.



$\mathcal{M}(\mathbb{M})$— A toy example

$\mathcal{M}(\mathbb{M}) := \{\boldsymbol{w} | \text{supp}(\boldsymbol{w}) \in \mathbb{M}\}$ where $\mathbb{M} := \{S | \mathbb{G}(S, \mathbb{E}') \text{ is connected subgraph up to size 3.}\}$



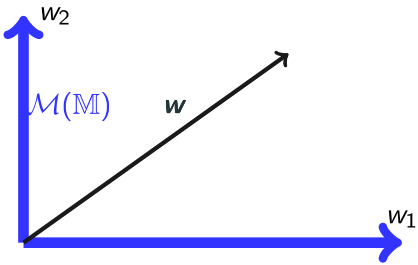
At each time t, we need $\mathbb{G}(\text{supp}(\boldsymbol{w}_t), \mathbb{E}')$ is connected and $|\text{supp}(\boldsymbol{w}_t)| \leq 3$.

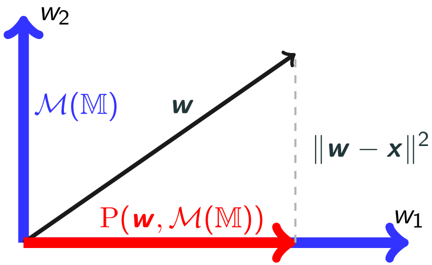
Projection explanation

How can we make sure $\boldsymbol{w}_t$ is in $\mathcal{M}(\mathbb{M})$? **Projection!**

The projection onto $\mathcal{M}(\mathbb{M})$ is defined as

$$\mathrm{P}(\boldsymbol{w}, \mathcal{M}(\mathbb{M})) = \arg\min_{\boldsymbol{x} \in \mathcal{M}(\mathbb{M})} \|\boldsymbol{w} - \boldsymbol{x}\|^2.$$





Online Mirror Descent

Algorithm 2 OMD(Online Mirror Descent [Hazan et al., 2016])

Input: η, $R(\boldsymbol{x}), B_R(\boldsymbol{x}, \boldsymbol{y}), \mathcal{K}$
let $\boldsymbol{y}_1$ be such that $\nabla R(\boldsymbol{y}_1) = 0$ and $\boldsymbol{x}_1 = \arg\min_{\boldsymbol{x}\in\mathcal{K}} B_R(\boldsymbol{x}, \boldsymbol{y}_1)$
for $t = 0, 1, 2, \ldots$ **do**
    update $\boldsymbol{y}_t$ by rule $\nabla R(\boldsymbol{y}_{t+1}) = \nabla R(\boldsymbol{y}_t) - \eta \nabla f_t(\boldsymbol{x}_t)$
    projection step $\boldsymbol{x}_{t+1} = \arg\min_{\in\mathcal{K}} B_R(\boldsymbol{x}, \boldsymbol{y}_{t+1})$
end for

- $R(\boldsymbol{x})$ is a Legendre Function (1. strictly convex; 2. has continuous first order derivatives; and 3. $\lim_{\boldsymbol{x}\to\bar{\mathcal{K}}\setminus\mathcal{K}} \|\nabla R(\boldsymbol{x}) = +\infty$).
- $B_R(\boldsymbol{x}, \boldsymbol{y}) = R(\boldsymbol{x}) - R(\boldsymbol{y}) - \langle \boldsymbol{x} - \boldsymbol{y}, \nabla F(\boldsymbol{y})\rangle$
- Online Mirror Descent is a generalized version of Online PGD.
- When $R(\boldsymbol{x}) = \frac{1}{2}\langle \boldsymbol{x}, \boldsymbol{x}\rangle$, OMD is exactly the same as Online PGD.

Regret (Continue)

Open problems: Provided the GraphDA algorithm,

- Can we obtain a non-regret bound under some condition ?
- What are the conditions ?

References

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12(Jul):2121–2159, 2011.

Elad Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Chinmay Hegde, Piotr Indyk, and Ludwig Schmidt. A nearly-linear time framework for graph-structured sparsity. In *ICML*, pages 928–937. PMLR, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Nam Nguyen, Deanna Needell, and Tina Woolf. Linear convergence of stochastic iterative greedy algorithms with sparse constraints. *IEEE Transactions on Information Theory*, 63(11):6869–6895, 2017.

Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *JMLR*, 11(Oct):2543–2596, 2010.

Haiqin Yang, Zenglin Xu, Irwin King, and Michael R Lyu. Online learning for group lasso. In *ICML*, pages 1191–1198. PMLR, 2010.

Baojian Zhou, Feng Chen, and Yiming Ying. Stochastic iterative hard thresholding for graph-structured sparsity optimization. In *International Conference on Machine Learning*, pages 7563–7573, 2019.

The social network image is from the following article: imagine a social network like facebook with no facebook