中国科学院信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING,CAS

Approximate Manifold Regularization: Scalable Algorithm and Generalization Analysis

Jian Li, Yong Liu*, Rong Yin, and Weiping Wang

Institute of Information Engineering, Chinese Academy of Sciences

28th International Joint Conference on Artificial Intelligence (IJCAI 2019)

Contents

1. Introduction

2. Algorithm Design

3. Theoretical Analysis

4. Experiments

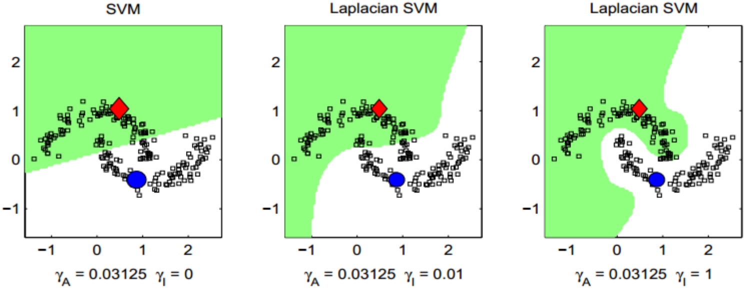
5. Conclusion

What is Manifold Regularization?

Consider common semi-supervised setting that a training dataset with n instances but only a few points m are labeled, where $m \ll n$.

$$\widehat{f_\lambda} = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{m} \ell(y_i, f(\mathbf{x}_i)) + \lambda_A \|f\|_{\mathcal{H}}^2 + \lambda_I \mathbf{f}^T \mathbf{L} \mathbf{f}.$$

where $\mathbf{L}$ is graph Laplacian by $\mathbf{L} = \mathbf{D} - \mathbf{W}$, $\mathbf{f} = [f(\mathbf{x}_1), \cdots, f(\mathbf{x}_n)]^T$, $\mathbf{W} \in \mathbb{R}^{n \times n}$ measures similarities between all points and $\mathbf{D}$ is a diagonal matrix $\mathbf{D}_{ii} = \sum_{j=1}^{n} W_{ij}$. [Belkin et al., 2006]



Scalability issues of LapRLS

1. Consider kernel ridge regression (KRR) with manifold regularization
   - Representer Theorem $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}_i, \mathbf{x})$
   - The squared loss $\ell(y_i, f(x_i)) = (y_i - f(x_i))^2$
   - Also called as Laplacian Regularized Least Squares (LapRLS)
2. LapRLS with closed-form solution

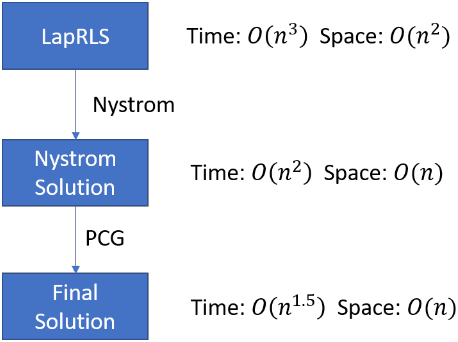
$$\widehat{\boldsymbol{\alpha}} = (\mathbf{JK} + \lambda_A \mathbf{I} + \lambda_I \mathbf{LK})^{-1} \mathbf{y}_n,$$

3. Scalability issues
   - Space complexity: $\mathcal{O}(n^2)$.
     e.g. Storing kernel matrix needs 18.6 GB when $n = 50,000$ while 74.5 GB when $n = 100,000$.
   - Time complexity: $\mathcal{O}(n^3)$.

Unfeasible to deal with large scale semi-supervised tasks!!!

Brief

1. Core Idea : LapRLS + Nyström + PCG.


LapRLS
Time: $O(n^3)$ Space: $O(n^2)$
Nystrom
Nystrom Solution
Time: $O(n^2)$ Space: $O(n)$
PCG
Final Solution
Time: $O(n^{1.5})$ Space: $O(n)$

2. Contributions:
   - Scalable Algorithm : $\mathcal{O}(n)$ space and $\mathcal{O}(n^{1.5})$ time.
   - Theoretical Guarantee: Excess risk bounds with convergence rate $\mathcal{O}(\frac{1}{\sqrt{m}})$.

Contents

1. Introduction

2. Algorithm Design

3. Theoretical Analysis

4. Experiments

5. Conclusion

Nyström LapRLS

1. Uniform subsampling over the training set (n points → s Nyström centers)

$$\mathcal{H}_s = \{f \in \mathcal{H} | f = \sum_{i=1}^{s} \alpha_i K(\mathbf{x}_i, \cdot), \boldsymbol{\alpha} \in \mathbb{R}^s\},$$

2. Nyström LapRLS with a closed-form solution:

$$\widehat{f}_\lambda^s(\mathbf{x}) = \sum_{i=1}^{s} \alpha_i K(\mathbf{x}_i, \mathbf{x}), \quad \text{with}$$

$$\boldsymbol{\alpha} = (\underbrace{\mathbf{K}_{ms}^T \mathbf{K}_{ms} + \lambda_A \mathbf{K}_{ss} + \lambda_I \mathbf{K}_{ns}^T \mathbf{L} \mathbf{K}_{ns}}_{\mathbf{H}})^\dagger \underbrace{\mathbf{K}_{ms}^T \mathbf{y}}_{\mathbf{z}},$$

where $\mathbf{H}^\dagger$ denotes the Moore-Penorse pseudoinverse and $\mathbf{H} \in \mathbb{R}^{s \times s}$.

3. Computation of $\mathbf{H}$ needs $\mathcal{O}(ns^2)$. Consider iterative method: conjugate gradient (CG) to solve linear systems.

$$\mathbf{H}\boldsymbol{\alpha} = \mathbf{z}.$$

Preconditioned Conjugate Gradient (PCG)

1. Convergence properties of CG methods are determined by the condition number $\kappa(\mathbf{H})$: the larger $\kappa(\mathbf{H})$ is, the slower the improvement.

   $$\mathbf{H}\boldsymbol{\alpha} = \mathbf{z}.$$

   In most cases, $\kappa(\mathbf{H})$ is large (ill-conditioned), thus convergence is slow.

2. Preconditioning to reduce the condition number $\kappa(\mathbf{P}^{-1}\mathbf{H})$

   $$\mathbf{P}^{-1}\mathbf{H}\boldsymbol{\alpha} = \mathbf{P}^{-1}\mathbf{z}.$$

   The more similar $\mathbf{H}$ and $\mathbf{P}$ are, the smaller the condition number.
   We provide two preconditioners to approximate $\mathbf{H}$

   - $m \leq \sqrt{n}$

   $$\mathbf{P} = \mathbf{K}_{ms}^T\mathbf{K}_{ms} + \lambda_A\mathbf{K}_{ss} + \frac{\lambda_I n^2}{s^2}\mathbf{K}_{ss}\mathbf{L}_{ss}\mathbf{K}_{ss}.$$

   - $m > \sqrt{n}$

   $$\mathbf{P} = \frac{m}{s}\mathbf{K}_{ss}^T\mathbf{K}_{ss} + \lambda_A\mathbf{K}_{ss} + \frac{\lambda_I n^2}{s^2}\mathbf{K}_{ss}\mathbf{L}_{ss}\mathbf{K}_{ss}.$$

Better scalability

1. Avoid matrix-matrix multiplications
   For each iteration of PCG, we need to calculate

$$\mathbf{H}\mathbf{p}_t = (\mathbf{K}_{ms}^T\mathbf{K}_{ms} + \lambda_A\mathbf{K}_{ss} + \lambda_I\mathbf{K}_{ns}^T\mathbf{L}\mathbf{K}_{ns})\mathbf{p}_t$$

   where $\mathbf{p}_t \in \mathbb{R}^s$. If we figure out $\mathbf{H}$, it needs $\mathcal{O}(ns^2)$ times to perform matrix-matrix multiplications.
   While a series of matrix-vector multiplications only need $\mathcal{O}(ns)$ time

$$\mathbf{H}\mathbf{p}_t = \mathbf{K}_{ms}^T(\mathbf{K}_{ms}\mathbf{p}_t) + \lambda_A\mathbf{K}_{ss}\mathbf{p}_t + \lambda_I\mathbf{K}_{ns}^T(\mathbf{L}(\mathbf{K}_{ns}\mathbf{p}_t)).$$

2. Block matrix multiplications
   - Kernel matrix $\mathbf{K}_{ns}$: $\mathcal{O}(ns)$ space
   - Decompose into $s \times s$ size block matrix multiplications: $\mathcal{O}(s^2)$ space
3. Space complexity: $\mathcal{O}(s^2)$. Time complexity: $\mathcal{O}(nst + s^3t)$.
   $\mathcal{O}(s^3)$ is due to the computation of $\mathbf{P}^{-1}\mathbf{r}_t$ in each iteration, where $\mathbf{r}_t \in \mathbb{R}^s$.

How many Nyström centers s and iterations t are needed?

Contents

1. Introduction

2. Algorithm Design

3. Theoretical Analysis

4. Experiments

5. Conclusion

Theoretical Analysis

Theorem (Simple version)

Under common assumptions and

$$s \geq \mathcal{O}(\sqrt{n}) \quad \textit{and} \quad t \geq \mathcal{O}(\log m)$$

then the following excess risk bound holds with high probability,

$$\mathcal{E}(\widehat{f}^{s}_{\lambda,t}) - \mathcal{E}(f_{\mathcal{H}}) \leq \mathcal{O}(\frac{1}{\sqrt{m}}).$$

Technical challenges:

- Multi-penalty regularization. [Rastogi and Sampath, 2017]
- Integral operator for Nyström methods. [Rudi et al., 2015]
- Convergence of PCG. [Rudi et al., 2017]

The complexity:

- Space complexity: $\mathcal{O}(s^2) = \mathcal{O}(n)$.
- Time complexity: $\mathcal{O}(nst + s^3 t) = \mathcal{O}(n\sqrt{n})$.

Contents

1. Introduction

2. Algorithm Design

3. Theoretical Analysis

4. Experiments

5. Conclusion

Compared methods

Estimators	Time	Space
RLS-Direct	$\mathcal{O}(m^3)$	$\mathcal{O}(m^2)$
LapRLS-Direct	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$
LapRLS-CG	$\mathcal{O}(n^{2.5})$	$\mathcal{O}(n^2)$
LapRLS-PCG	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
Nyström-Direct	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
Nyström-CG	$\mathcal{O}(n^{1.75})$	$\mathcal{O}(n)$
`Nyström-PCG`	$\mathcal{O}(n^{1.5})$	$\mathcal{O}(n)$

Table 1: Summary of time complexity and space complexity in terms of various methods. Here, we omit logarithmic terms.

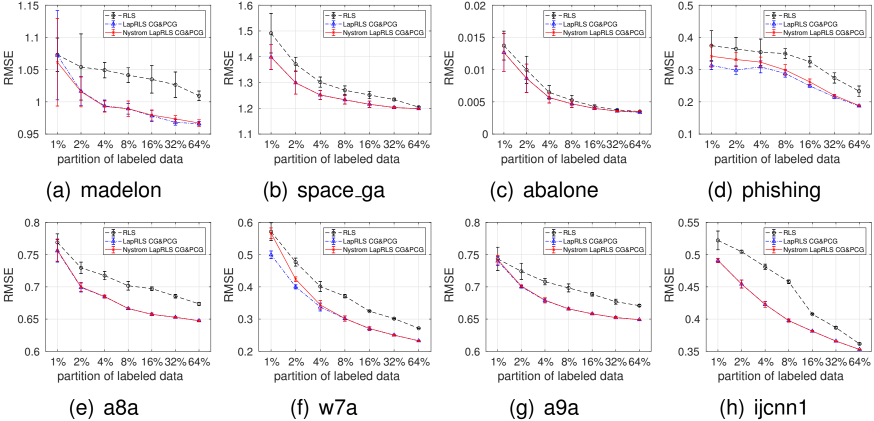
RMSE, Iteration and Runing time

dataset	sample size	RLS-CG	LapRLS-CG	LapRLS-PCG	Nyström-CG	Nyström-PCG
madelon	2000	1.036±0.009	**0.990±0.007**	**0.990±0.007**	0.991±0.009	0.991±0.009
space_ga	3107	1.251±0.004	**1.210±0.004**	**1.210±0.004**	**1.210±0.004**	**1.210±0.004**
abalone	4177	4.55±0.2×10^3	**4.17±0.1×10^3**	**4.17±0.1×10^3**	4.18±0.1×10^3	4.18±0.1×10^3
phishing	11055	0.426±0.049	0.294±0.005	**0.273±0.007**	0.295±0.005	0.275±0.008
a8a	22696	0.702±0.002	**0.664±0.002**	**0.664±0.002**	0.664±0.002	0.664±0.002
w7a	24692	0.291±0.002	**0.283±0.002**	**0.283±0.002**	0.284±0.002	0.284±0.002
a9a	32561	0.698±0.005	**0.664±0.000**	**0.664±0.002**	**0.664±0.000**	**0.664±0.002**
ijcnn1	49990	0.434±0.005	**0.389±0.002**	**0.389±0.002**	0.393±0.001	0.463±0.001
cod-rna	59535	0.686±0.002	/	/	0.614±0.001	0.614±0.001
connect-4	67757	0.781±0.015	/	/	**0.739±0.002**	**0.739±0.002**
skin_nonskin	245057	3.119±0.023	/	/	**2.620±0.043**	**2.620±0.043**
YearPrediction	463715	0.198±0.001	/	/	**0.187±0.001**	**0.187±0.001**

	RLS-CG		LapRLS-CG		LapRLS-PCG		Nyström-CG		Nyström-PCG	
	iter	time	iter	time	iter	time	iter	time	iter	time
madelon	32	0.003	13	0.029	6	0.032	12	0.043	**1**	**0.006**
space_ga	11	0.004	23	1.220	5	0.569	23	0.113	**2**	**0.016**
abalone	64	0.053	98	26.50	4	0.903	94	0.363	**2**	**0.067**
phishing	74	0.031	300	24.20	56	8.210	300	2.470	**3**	**0.045**
a8a	100	0.068	50	189.1	3	20.98	50	44.71	**1**	**4.370**
w7a	13	0.072	32	143.2	2	9.683	213	107.7	**1**	**2.252**
a9a	300	0.529	64	1699	3	30.30	65	70.40	**1**	**4.034**
ijcnn1	242	8.204	57	2154	9	72.41	53	108.8	**5**	**4.186**
cod-rna	96	7.178	/	/	/	/	55	134.6	**7**	**8.154**
connect-4	103	11.07	/	/	/	/	154	186.5	**10**	**4.220**
skin_nonskin	43	91.39	/	/	/	/	65	1490	**3**	**40.05**
YearPrediction	37	236.5	/	/	/	/	94	2479	**2**	**116.1**

We randomly select 10% samples ($m = 0.1n$) as labeled data and 10% samples ($s = 0.1n$) as Nyström centers.

Average RMSE for different labeled data proportion



(a) madelon

(b) space_ga

(c) abalone

(d) phishing

(e) a8a

(f) w7a

(g) a9a

(h) ijcnn1

Figure 1: Average RMSE for different labeled data proportion.

Contents

1. Introduction

2. Algorithm Design

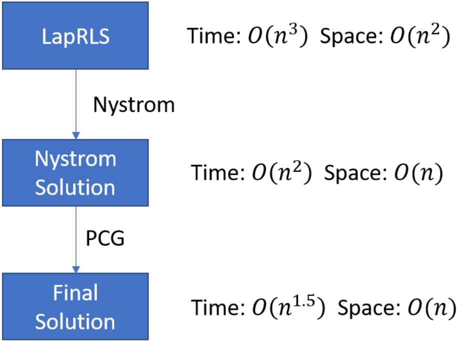
3. Theoretical Analysis

4. Experiments

5. Conclusion

Conclusion

1. Core Idea : LapRLS + Nyström + PCG.



LapRLS

Time: $O(n^3)$ Space: $O(n^2)$

Nystrom

Nystrom Solution

Time: $O(n^2)$ Space: $O(n)$

PCG

Final Solution

Time: $O(n^{1.5})$ Space: $O(n)$

2. Contributions:
    - Scalable Algorithm : $\mathcal{O}(n)$ space and $\mathcal{O}(n^{1.5})$ time.
    - Theoretical Guarantee: Excess risk bounds with convergence rate $\mathcal{O}(\frac{1}{\sqrt{m}})$.

References

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7(Nov):2399–2434, 2006.

Abhishake Rastogi and Sivananthan Sampath. Manifold regularization based on nystr {\" o} m type subsampling. *arXiv preprint arXiv:1710.04872*, 2017.

Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 1657–1665, 2015.

Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. Falkon: An optimal large scale kernel method. In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 3888–3898, 2017.

Any Questions?