Online learning of eigenvectors

A brief survey of recent advancements

Bhuvesh Kumar

May 2, 2018

Georgia Institute of Technology

Table of contents

1. Introduction
2. Matrix multiplicative weight update
3. Follow the Perturbed Leader
4. Follow the compressed leader

Introduction

Problem Formulation

We are considering the matrix generalization of the 2 player zero sum game where the loss matrix is a psd or nsd matrix

General framework

Algorithm 1 Online Learning: Matrix Loss

for $t = 1, 2, \ldots, T$ **do**
    Player plays unit vector $v_t \in \mathbb{S}^{d-1}$
    Adversary reveals a symmetric reward matrix A_t s.t $0 \preceq A_t \preceq 1$
    Player receives a reward $v_t^T A_t v_t = A_t \cdot v_t v_t^T$
end for

Regret

As in most online learning frameworks, the goal is to minimize regret, i.e. minimize the difference between the gain by the player and the gain by a posteriori best fixed strategy u

Regret minimzation objective

$$\begin{aligned}\text{minimize} \max_{u \in \mathbb{R}^d} \sum_{t=1}^{T} A_t \cdot \left(uu^t - v_t v_t^T\right) \\ = \lambda_{\max}\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} v_t^T A_t v_t\end{aligned}$$

It is evident the best fixed strategy is the eigenvector corresponding to $\lambda_{\max}\left(\sum_{t=1}^{T} A_t\right)$

Expected Regret

Since many online learning expert style algorithms use some kind of randomness i.e. v_t is selected from a distribution $\mathcal{D}$ over $\mathbb{S}^{n-1}$, we care about expected regret.

Expected regret minimzation

$$\begin{aligned} \text{minimize } & \lambda_{\max}\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} A_t \cdot \mathbb{E}\left[v_t v_t^T\right] \\ = & \lambda_{\max}\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} A_t \cdot P_t \end{aligned}$$

Where $P_t = \mathbb{E}\left[v^t v_t^T\right]$ is the density matrix. It is psd and has trace 1.

Relevance

Top eigenvector computaion of symmetric matrices is a primitive problem in machine learning theory and the online variant of the problem holds importance too. In particular, this problem finds application in:

- Efficient algorithms for semidefinite programming. [5]
- Online max-cut problem [8]
- Ramanujam Sparsifiers
- Derandomising expander graphs
- Density matrices crop up in Quantum computing

Matrix multiplicative weight update

Matrix multiplicative weight update

Matrix multiplicative weights update (MMWU) [5] [11] is the generalizing of the multiplicative weights algorithm [4].

Matrix Multiplicative weight update

Algorithm 2 Matrix Multiplicative weight update

Fix η

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

  $W_t \leftarrow \exp\left(\eta \sum_{i=1}^{t-1} A_i\right)$

  Density matrix $P_t \leftarrow \frac{W_t}{\mathrm{tr}(W_t)}$

  Either play $P_t = \sum_{j=1}^{d} p_j \cdot y_j y_j^T$ or play y_j with prob p_j

end for

Guarantees

Total Regret

The theoretical best choice for η in Matrix multiplicative weights update (MMWU) is $\eta = \sqrt{\log d / T}$ which gives a total expected regret of $O(\sqrt{T \log d})$[10].

Per Iteration cost

Since, we need to compute the full SVD for $\sum_{i=1}^{t-1} A_i$, per iteration running time is at least $O(d^\omega)$ and can also be up to d^3 if there are repeated eigenevalues.

Total time to get ϵ average regret

$\tilde{O}\left(\frac{d^\omega}{\epsilon^2}\right)$

More Comments

MMWU can also be derived from mirror descent where the Bregman divergence is the quantum relative entropy divergence. [11] [2]

It was also shown in [2] that MMWU can be recovered from from FTRL with the appropriate regulariser just like like MWU can be recovered from FTRL with negative entropy regularization [7].

Follow the Perturbed Leader

Follow the Perturbed Leader

FTPL [6] is also the generalization of FTPL [9] in the vector loss setting to the matrix loss setting.

Follow the Perturbed Leader

Algorithm 3 Follow the Perturbed Leader

Input $\mathcal{D}$ over $\mathbb{M}_d$
Sample $N \sim \mathcal{D}$
$v_1 =$ Top Eigenvector (N)
for $t = 1, 2, \ldots, T$ **do**
    Play v_t
    Observe A_t
    $v_{t+1} \leftarrow$ Top Eigenvector $\left(\sum_{i=1}^{t} A_i + N\right)$
end for

Here $\mathbb{M}_d$ is the linear space of all real symmetric $d \times d$ matrices.

Guarantees

Total Regret

If the noise N is sampled as $c \cdot xx^T$ where c is a parameter and entries of v are sampled i.i.d fron $\mathcal{N}(0,1)$, then the total expected regret of FTPL is $O(\sqrt{Td})$[6].

Per Iteration cost

Since we need to compute the top eigenvector upto some accuracy, it can be done in time $\tilde{O}(\min\left\{T^{3/4}d^{-1/4}\mathrm{nnz}(\sum_{i=1}^{t} A_i)), d^{\omega}\right\}$ which is better than MMWU.

Total time to get ϵ average regret

Since we have an extra $\sqrt{d}$ in the regret, we need $\tilde{O}\left(\frac{d^{\mathbf{1.5}} nnz(\Sigma)}{\epsilon^{\mathbf{3.5}}}\right)$ total time

Follow the compressed leader

Follow the compressed Leader

FTCL[1] is a compression of MMWU to a constant dimension of 3.

Follow the Compressed Leader

Algorithm 4 Follow the Compressed Leader

1: **Input** η, q
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Sample u_1, u_2, u_3, each index iid from $\mathcal{N}(0, 1)$.
4: $U_t \leftarrow \frac{1}{3}\left(u_1 u_1^t + u_2 u_2^T + u_3 u_3^T\right)$
5: $\Sigma_t \leftarrow \sum_{i=1}^{t-1} A_t$
6: Define $X_t = (c_t I - \eta\Sigma_t)^{-q}$ s.t. $Tr(X_t U_t) = 1$ and $c_t I - \eta\Sigma_t \succ 0$
7: Compute $X_t^{1/2} U_t X_t^{1/2} = \sum_{j=1}^{3} p_j y_j y_j^T$
8: Play y_j with prob p_j
9: **end for**

Discussion

The compression requires minimum 3 dimensions because the regret analysis is dependent on expected mean of $1/\left|U_i i\right|$ and we know that if $x_1, \cdots_{x\ k}$ are sampled iid from $\mathcal{N}(0,1)$ then $\frac{1}{\sum_{i=1}^{k} x_i^2}$ is an inverse chi-squared distribution and the expectation of this variable if bounded for $k \geq 3$.

Implementation overview

The theoretical choice of q is $\Theta\left(\log\left(dT\right)\right)$ and for η is $\frac{\log^{-\mathbf{3}}(dT)}{\sqrt{\lambda_{\max}(\Sigma_T)}}$, but *eta* is mostly fine tuned.

The major steps in the algorithm are

- **Finding $\mathbf{c_t}$** c_t : is calculated using a binary search
- **Calculating** $\left(c_t I - \eta\Sigma_t\right)^{-q/2} u_j$ **for** $j \in [3]$**:** This is needed so that the SVD $X_t^{1/2} U_t X_t^{1/2} = \sum_{j=1}^{3} p_j y_j y_j^T$ can be done in $O(d)$. This can actually be solved by using a convex optimization routine like gradient descent or Nesterov's acceleration or by SVRG [3] where the time taken is similar to solving a linear system of equations, $q/2$ times, here $q/2$ only adds a poly log factor.

Guarantees

Total Regret

Setting appropriate value of η and q, the total expected regret of FTCL is $\tilde{O}\left(\sqrt{T}\right)$ [1].

Per Iteration cost

We need to solve a linear system of equations poly log times which can be done in $\tilde{O}\left(\min\left\{\min\left\{T^{1/4}, d\right\} nnz(\Sigma_T), d^{\omega}\right\}\right)$ which is better than MMWU and not worse than FTPL.

Total time to get ϵ average regret

$\tilde{O}\left(\frac{nnz(\Sigma)}{\epsilon^{\mathbf{2.5}}}\right)$

Questions?

Z. Allen-Zhu and Y. Li.
Follow the compressed leader: Faster online learning of eigenvectors and faster mmwu.
In *International Conference on Machine Learning*, pages 116–125, 2017.

Z. Allen-Zhu, Z. Liao, and L. Orecchia.
Spectral sparsification and regret minimization beyond matrix multiplicative updates.
In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 237–245. ACM, 2015.

Z. Allen-Zhu and Y. Yuan.
Improved svrg for non-strongly-convex or sum-of-non-convex objectives.
In *International conference on machine learning*, pages 1080–1089, 2016.

S. Arora, E. Hazan, and S. Kale.
The multiplicative weights update method: a meta-algorithm and applications.
Theory of Computing, 8(1):121–164, 2012.

S. Arora and S. Kale.
A combinatorial, primal-dual approach to semidefinite programs.
In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 227–236. ACM, 2007.

D. Garber, E. Hazan, and T. Ma.
Online learning of eigenvectors.
In *ICML*, pages 560–568, 2015.

E. Hazan.
10 the convex optimization approach to regret minimization.
Optimization for machine learning, page 287, 2012.

E. Hazan, S. Kale, and S. Shalev-Shwartz.
Near-optimal algorithms for online matrix prediction.
In *Conference on Learning Theory*, pages 38–1, 2012.

A. Kalai and S. Vempala.
Efficient algorithms for online decision problems.
Journal of Computer and System Sciences, 71(3):291–307, 2005.

L. Orecchia.
Fast approximation algorithms for graph partitioning using spectral and semidefinite-programming techniques.
University of California, Berkeley, 2011.

K. Tsuda, G. Rätsch, and M. K. Warmuth.
Matrix exponentiated gradient updates for on-line learning and bregman projection.
Journal of Machine Learning Research, 6(Jun):995–1018, 2005.