Sketched Follow-The-Regularized-Leader for Online Factorization Machine

Luo Luo
Shanghai Jiao Tong University
rickyluoluo@gmail.com

Wenpeng Zhang
Tsinghua University
zhangwenpeng0@gmail.com

Zhihua Zhang
Peking University / BIBDR, China
zhzhang@math.pku.edu.cn

Wenwu Zhu
Tsinghua University
wwzhu@tsinghua.edu.cn

Tong Zhang
Tencent AI Lab
tongzhang@tongzhang-ml.org

Jian Pei
JD.com / Simon Fraser University
jpei@cs.sfu.ca

ABSTRACT

Factorization Machine (FM) is a supervised machine learning model for feature engineering, which is widely used in many real-world applications. In this paper, we consider the case that the data samples arrive sequentially. The existing convex formulation for online FM has the strong theoretical guarantee and stable performance in practice, but the computational cost is typically expensive when the data is high-dimensional. To address this weakness, we devise a novel online learning algorithm called *Sketched Follow-The-Regularizer-Leader* (SFTRL). SFTRL presents the parameters of FM implicitly by maintaining low-rank matrices and updates the parameters via sketching. More specifically, we propose Generalized Frequent Directions to approximate indefinite symmetric matrices in a streaming way, making that the sum of historical gradients for FM could be estimated with tighter error bound efficiently. With mild assumptions, we prove that the regret bound of SFTRL is close to that of the standard FTRL. Experimental results show that SFTRL has better prediction quality than the state-of-the-art online FM algorithms in much lower time and space complexities.

CCS CONCEPTS

• **Mathematics of computing** → **Convex optimization**; • **Theory of computation** → **Online learning algorithms**; • **Computing methodologies** → **Linear algebra algorithms**;

KEYWORDS

Factorization Machine; Convex Online Learning; Follow-The-Regularized-Leader; Sketching

ACM Reference Format:
Luo Luo, Wenpeng Zhang, Zhihua Zhang, Wenwu Zhu, Tong Zhang, and Jian Pei. 2018. Sketched Follow-The-Regularized-Leader for Online Factorization Machine. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3220044

1 INTRODUCTION

The Factorization Machine (FM) [38] is a popular supervised approach for feature engineering. FM captures extra pairwise feature interaction and typically achieves better performance than linear models. It is widely used in many machine learning applications such as recommendation systems [26, 36, 39], computational advertising [23, 24], search ranking [20, 30], toxicogenomics prediction [44], etc. In many real-world scenarios, the training data arrives in a streaming fashion and the system should provide a response immediately. This motivates us to develop an efficient online learning method for factorization machine.

The interaction weight matrix is the core of training factorization machine. Maintaining interaction of all pairwise features results in overfitting and unacceptable computation cost. Hence, we usually suppose the interaction weight matrix to be low-rank. Many studies formulate FM by using the outer-product to present the interaction matrix [6, 8, 23, 29, 38], which allows the training algorithm efficiently. However, this formulation leads to a non-convex optimization problem, and the unexpected saddle points and bad local minima make prediction results heavily dependent on initialization.

The convex formulations of FM are typically more stable and effective in practice. Recently, one proposed online compact convexified factorization machine [28, 46], which imposes the low-rank property by the nuclear norm ball constraint and updates the parameters by online conditional gradient (OCG) [21]. The algorithm iterates with a linear optimization subroutine, which needs to compute the top singular vectors for each iteration. It also has sub-linear regret bound in theory. Some other studies also formulate FM by the nuclear norm regularization [5, 44], but only consider the off-online case.

In this paper we propose a novel online learning algorithm for factorization machine named as Sketched Follow-The-Regularized-Leader (SFTRL). SFTRL fully utilizes the matrix-type parameter of FM and compacts the interaction weight matrix via sketching, which leads to a good approximation to convex FM and avoids the expensive cost for optimizing nuclear norm constraint or regularization. Considering the specific loss function of FM and the update rule for Follow-The-Regularized-Leader (FTRL) [33], we design a new sketching strategy called Generalized Frequent Directions (GFD). The standard Frequent Directions technique [18, 27] is a deterministic sketching algorithm to approximate positive semi-definite matrices, and GFD extends it to deal with indefinite matrices that is important to online FM. Most of existing sketched online learning algorithms [7, 31, 32] focus on Newton-type algorithms with approximate Hessian, while SFTRL is a first order method which estimates the model parameter and its gradient. Theoretically, we prove that the regret bound of SFTRL is close to that of the standard FTRL given that gradients lie in an approximate low-rank space. The experimental results show that SFTRL achieves better prediction results than the state-of-the-art methods but with much lower computational cost.

The remainder of the paper is organized as follows. Section 2 presents preliminaries. Section 3 introduces our proposed SFTRL. Section 4 provides the theoretical analysis for the regret bound of SFTRL. Section 5 validates our algorithm empirically and shows its superiority. Section 6 introduces the related work. Finally, we conclude our work in Section 7.

2 PRELIMINARIES

In this section, we define the notation and introduce the background of the factorization machine, then provide a brief review of online convex optimization.

2.1 Notations

For a matrix $\mathbf{A} = [A_{ij}] \in \mathbb{R}^{n\times d}$ of rank r where $r \leq \min(n, d)$, let the condensed singular value decomposition (SVD) of $\mathbf{A}$ be $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{n\times r}$ and $\mathbf{V} \in \mathbb{R}^{d\times r}$ are column orthogonal and $\mathbf{\Sigma} = \operatorname{diag}(\sigma_{[1]}, \sigma_{[2]}, \dots, \sigma_{[r]})$ with $\sigma_{[1]} \geq \sigma_{[2]} \geq \dots, \geq \sigma_{[r]} > 0$ as the nonzero singular values. We let $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j} A_{ij}^2} = \sqrt{\sum_{i=1}^r \sigma_{[i]}^2}$ be the Frobenius norm, $\|\mathbf{A}\|_* = \sum_{i=1}^r \sigma_{[i]}$ be the nuclear norm, and $\|\mathbf{A}\|_2 = \sigma_{[1]}$ be the spectral norm. We define $\mathbf{A}_{[k]} = \mathbf{U}_{[k]}\mathbf{\Sigma}_{[k]}\mathbf{V}_{[k]}^\top$ for $k \leq r$, where $\mathbf{\Sigma}_{[k]} = \operatorname{diag}(\sigma_{[1]}, \sigma_{[2]}, \dots, \sigma_{[k]})$, $\mathbf{U}_{[k]} \in \mathbb{R}^{d\times k}$ and $\mathbf{V}_{[k]} \in \mathbb{R}^{n\times k}$ are respectively the first k columns of $\mathbf{U}$ and $\mathbf{V}$.

2.2 Factorization Machine

Factorization Machine (FM) [38] is a well-known supervised learning model which is powerful to feature engineering. FM captures the second-order pairwise feature interaction via a symmetric matrix. Compared with the linear model which only considering the first-order information of features, FM has much stronger prediction ability but requires more expensive computation costs.

For an input sample with features $\mathbf{a}_i \in \mathbb{R}^d$, FM predicts the result with the following formula:

$$z_i = z_i(w_0, \mathbf{w}, \mathbf{X}) = w_0 + \mathbf{w}^\top \mathbf{a}_i + \mathbf{a}_i^\top \mathbf{X}\mathbf{a}_i \in \mathbb{R}, \qquad (1)$$

where model parameters contain the bias $w_0 \in \mathbb{R}$, the linear coefficient vector $\mathbf{w} \in \mathbb{R}^d$ and the interaction weight matrix $\mathbf{X} \in \mathbb{R}^{d\times d}$. For the training sample $\mathbf{a}_i \in \mathbb{R}^d$ and corresponding label b_i, we define the loss function $f_i(w_0, \mathbf{w}, \mathbf{X}) = \phi_i(z_i)$, where $\phi_i : \mathbb{R} \to \mathbb{R}$. We assume ϕ_i is convex; that is,

$$\phi_i(u) \geq \phi_i(v) + \phi_i'(v)(u - v)$$

holds for any $u, v \in \mathbb{R}$. The concrete expression of ϕ_i is determined by specific problems. For binary classification, we can take the logistic loss $\phi_i(z_i) = \log(1 + \exp(-b_i z_i))$ or hinge loss $\phi_i(z_i) = \max(1 - b_i z_i, 0)$ with $b_i \in \{-1, 1\}$; and for regression problem, we can take the least square loss $\phi_i(z_i) = \frac{1}{2}(z_i - b_i)^2$ with $b_i \in \mathbb{R}$.

For ease of implementation and analysis, we present the parameters of FM in compact manner [28]

$$\mathbf{\Theta} = \begin{bmatrix} \mathbf{X} & \frac{1}{2}\mathbf{w} \\ \frac{1}{2}\mathbf{w}^\top & w_0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}.$$

With the argumented feature $\hat{\mathbf{a}}_i = [\mathbf{a}_i^\top, 1]^\top$, we have

$$\hat{\mathbf{a}}_i^\top \mathbf{\Theta} \hat{\mathbf{a}}_i = w_0 + \mathbf{w}^\top \mathbf{a}_i + \mathbf{a}_i^\top \mathbf{X}\mathbf{a}_i.$$

Hence, we can rewrite the expression of z_i as

$$z_i = z_i(\mathbf{\Theta}) = \hat{\mathbf{a}}_i^\top \mathbf{\Theta} \hat{\mathbf{a}}_i \in \mathbb{R},$$

which is equivalent to (1). Then the gradient of each loss function is a rank-one matrix as follows

$$\nabla f_i(\mathbf{\Theta}) = \phi_i'(z_i)\hat{\mathbf{a}}_i\hat{\mathbf{a}}_i^\top \in \mathbb{R}^{(d+1)\times(d+1)}. \qquad (2)$$

Additionally, we have that $f_i(\mathbf{\Theta})$ is convex due to the convexity of ϕ_i, that is,

$$f_i(\mathbf{\Theta}_1) \geq f_i(\mathbf{\Theta}_2) + \langle \nabla f_i(\mathbf{\Theta}_2), \mathbf{\Theta}_1 - \mathbf{\Theta}_2 \rangle \qquad (3)$$

for any $\mathbf{\Theta}_1, \mathbf{\Theta}_2 \in \mathbb{R}^{(d+1)\times(d+1)}$.

2.3 Online Convex Optimization

Online convex optimization [41, 47] is a framework for designing algorithms in the case that data samples arrive sequentially. At the t-th round, the adversary presents an example $(\mathbf{a}_t, b_t)$ and the learner updates the predictor $\mathbf{\Theta}_t$ by some convex loss function $\ell_t : \mathcal{K} \to \mathbb{R}$ with convex set $\mathcal{K}$, and suffers the loss $\ell_t(\mathbf{\Theta}_t)$. We want to minimize regret to the sequence of the predictor $\{\mathbf{\Theta}_t\}_{t=1}^T$, that is,

$$\text{Regret}_T(\mathbf{\Theta}_*) = \sum_{t=1}^T f_t(\mathbf{\Theta}_t) - \sum_{t=1}^T f_t(\mathbf{\Theta}_*),$$

where $\mathbf{\Theta}_* = \operatorname{argmin}_{\mathbf{\Theta}\in\mathcal{K}} \sum_{t=1}^T f_t(\mathbf{\Theta})$. There are many algorithms designed based on the online convex optimization framework [4, 15, 21, 31–33, 47]. Follow-The-Regularized-Leader (FTRL) [4] is a popular online learning algorithm, which has a regularization term to avoid the predictor varying wildly from one iteration to the next. We present the procedure of FTRL in Algorithm 1. The regret bound of FTRL is shown in the following theorem [21].

THEOREM 1. *The FTRL Algorithm 1 attains the following bound on regret*

$$\mathrm{Regret}_T(\mathbf{\Theta}_*) \le 2\eta \sum_{t=1}^{T} \|\nabla f_t(\mathbf{\Theta}_t)\|_F^2 + \frac{R(\mathbf{\Theta}_*) - R(\mathbf{\Theta}_1)}{\eta}.$$

If we know the upper bounds of the local gradient norms and the regularization term $\|\nabla f_t(\mathbf{\Theta}_t)\|_F^2 \le D_G$ and $|R(\mathbf{\Theta}_*) - R(\mathbf{\Theta}_1)| \le D_R$ the regret bound can be optimized by choosing appropriate η to obtain

$$\mathrm{Regret}_T(\mathbf{\Theta}_*) \le 2D_G D_R\sqrt{2T}.$$

We can directly use FTLR to solve the FM problem under the compact representation by taking $R(\mathbf{\Theta}) = \frac{1}{2}\|\mathbf{\Theta}\|_F^2$ and $\mathcal{K} = \mathbb{R}^{(d+1)\times(d+1)}$. Then we need $\mathcal{O}(d^2)$ space to store the parameter $\mathbf{\Theta}_t$ and the sum of historical gradients $\sum_{s=1}^{t} \nabla f_s(\mathbf{\Theta}_s)$. The running time of update rule (4) requires $\mathcal{O}(d^2)$ for each round and $\mathcal{O}(Td^2)$ totally. The cost is too expensive for large-scale and high-dimensional datasets.

Algorithm 1 Follow-The-Regularizer-Leader (FTRL)

```
1: Input: hyperparameter $\eta > 0$, regularization function $R$ and a convex set $\mathcal{K}$
2: Initialize $\mathbf{\Theta}_1 = \mathrm{argmin}_{\mathbf{\Theta}} R(\mathbf{\Theta})$
3: for $t = 1, \ldots, T$ do
4:     Predict by $\mathbf{\Theta}_t$
5:     Receive data $\mathbf{a}_t$ and loss $f_t(\mathbf{\Theta})$
6:     Update
           $\mathbf{\Theta}_{t+1} = \underset{\mathbf{\Theta}\in\mathcal{K}}{\mathrm{argmin}} \left\langle \sum_{s=1}^{t} \nabla f_s(\mathbf{\Theta}_s), \mathbf{\Theta} \right\rangle + \frac{1}{\eta} R(\mathbf{\Theta})$    (4)
7: end for
```

3 METHODOLOGY

The bottleneck of convex online FM algorithms is the update and the storage of the sum of historical gradients. If the sum of gradients can be computed and stored with low cost, all operations of the algorithm will be efficient. In this section, we first introduce a new sketching method name as Generalized Frequent Directions (GFD), which can be used to approximate the sum of gradients from FM efficiently. Then we proposed the Sketched-Follow-The-Regularized-Leader (SFTRL) to train the online FM model.

3.1 Generalized Frequent Directions

Sketching is a powerful tool to deal with large matrices [43]. Given a matrix, sketching algorithms compress it into a much smaller matrix with certain properties. Then the expensive computation can be avoided by the operation on the smaller matrices. There are many randomized sketching algorithms including column (row) sampling [11–14, 16, 37] and random projection [1–3, 9, 22, 25, 35, 40]. They produce accurate estimates with high probability by sufficient sketch size.

Frequent Directions (FD) technique [10, 17, 18, 27] is a deterministic sketching, which is numerically stable even when the sketch size is small. The original FD is only designed to approximate a positive semi-definite matrix in the streaming manner, while the gradient $\nabla f_i(\mathbf{\Theta})$ in FM may be negative definite as shown in (2) (because $\phi_i'(z_i)$ is not always positive). Hence, we propose Generalized Frequent Directions (GFD) to estimate an indefinite symmetric matrix, which can be applied to online FM cases. Suppose we have an indefinite symmetric matrix $\mathbf{G} = \sum_{t=1}^{T} s_t \mathbf{g}_t \mathbf{g}_t^\top$ where $\mathbf{g}_t \in \mathbb{R}^d$ and $s_t \neq 0$. Each component $s_t \mathbf{g}_t \mathbf{g}_t^\top$ comes in the streaming manner.

For a given sketch size $m \ll d$, GFD approximates $\mathbf{G}$ as

$$\mathbf{G} \approx (\mathbf{B}^+)^\top \mathbf{B}^+ - (\mathbf{B}^-)^\top \mathbf{B}^-, \qquad (5)$$

where $\mathbf{B}^+, \mathbf{B}^- \in \mathbb{R}^{(m-1)\times d}$. We describe the details of GFD in Algorithm 2. Similar to the standard FD [18], the cost of GFD is dominated by the SVD step in Lines 6 and 11. This operation needs $\mathcal{O}(m^2 d)$ running time in general, but can be reduced to $\mathcal{O}(md)$ by the Gu-Eisenstat procedure [19] due to the special forms of $\widehat{\mathbf{B}}^+$ and $\widehat{\mathbf{B}}^-$. The total running time of GFD is $\mathcal{O}(Tmd)$, and the space complexity is $\mathcal{O}(md)$.

We can also avoid complex operation of the Gu-Eisenstat procedure to obtain the same complexity conveniently by doubling the space, which is shown in Algorithm 3. This algorithm keeps the sketching matrix with $2m$ rows and does the standard SVD per m iterations. Then the running time is also $\mathcal{O}(Tmd)$.

Algorithm 2 Generalized Frequent Directions (GFD)

```
1: Input: $(\mathbf{g}_1, s_1), \ldots, (\mathbf{g}_T, s_T)$ and sketch size $m$
2: Initialize $\mathbf{B}_0^+ = \mathbf{B}_0^- = \mathbf{0}^{(m-1)\times d}$
3: for $t = 1, \ldots, T$ do
4:     if $s_t > 0$
5:         $\widehat{\mathbf{B}}_t^+ = \begin{bmatrix} \mathbf{B}_{t-1}^+ \\ \sqrt{s_t}\mathbf{g}_t^\top \end{bmatrix}$
6:         $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \mathrm{svd}(\widehat{\mathbf{B}}_t^+)$
7:         $\mathbf{B}_t^+ = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2 \mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
8:         $\mathbf{B}_t^- = \mathbf{B}_{t-1}^-$
9:     else
10:        $\widehat{\mathbf{B}}_t^- = \begin{bmatrix} \mathbf{B}_{t-1}^- \\ \sqrt{-s_t}\mathbf{g}_t^\top \end{bmatrix}$
11:        $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \mathrm{svd}(\widehat{\mathbf{B}}_t^-)$
12:        $\mathbf{B}_t^- = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2 \mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
13:        $\mathbf{B}_t^+ = \mathbf{B}_{t-1}^+$
14:    end if
15: end for
16: Output: $\mathbf{B}^+ = \mathbf{B}_T^+$ and $\mathbf{B}^- = \mathbf{B}_T^-$
```

Algorithm 3 Generalized Frequent Directions (GFD) with Doubling Space

```
1: Input: $(\mathbf{g}_1, s_1), \ldots, (\mathbf{g}_T, s_T)$ and sketch size $m$
2: Initialize $\mathbf{B}_0^+$ and $\mathbf{B}_0^-$ be empty
3: for $t = 1, \ldots, T$ do
4:   if $s_t > 0$
5:     $\widehat{\mathbf{B}}_t^+ = \begin{bmatrix} \mathbf{B}_{t-1}^+ \\ \sqrt{s_t}\mathbf{g}_t^\top \end{bmatrix}$
6:     if $\widehat{\mathbf{B}}_t^+$ has $2m$ rows
7:       $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \mathrm{svd}(\widehat{\mathbf{B}}_t^+)$
8:       $\mathbf{B}_t^+ = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2 \mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
9:     else
10:      $\mathbf{B}_t^+ = \mathbf{B}_{t-1}^+$
11:    end if
12:    $\mathbf{B}_t^- = \mathbf{B}_{t-1}^-$
13:  else
14:    $\widehat{\mathbf{B}}_t^- = \begin{bmatrix} \mathbf{B}_{t-1}^- \\ \sqrt{-s_t}\mathbf{g}_t^\top \end{bmatrix}$
15:    if $\widehat{\mathbf{B}}_t^-$ has $2m$ rows
16:      $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \mathrm{svd}(\widehat{\mathbf{B}}_t^-)$
17:      $\mathbf{B}_t^- = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2 \mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
18:    else
19:      $\mathbf{B}_t^- = \mathbf{B}_{t-1}^-$
20:    end if
21:    $\mathbf{B}_t^+ = \mathbf{B}_{t-1}^+$
22:  end if
23: end for
24: Output: $\mathbf{B}^+ = \mathbf{B}_T^+$ and $\mathbf{B}^- = \mathbf{B}_T^-$
```

GFD has the spectral norm error bound as Theorem 2 shown, which means the approximation error is small when the spectrums of $\mathbf{G}^+$ and $\mathbf{G}^-$ decay rapidly. Because the real-world data usually lies in an approximate low-rank space, the bound in the theorem is tight in practice.

THEOREM 2. *For any $k < m$ and using the notation of Algorithm 2, we have*

$$\left\| \mathbf{G} - \left((\mathbf{B}^+)^\top \mathbf{B}^+ - (\mathbf{B}^-)^\top \mathbf{B}^- \right) \right\|_2 \leq \frac{1}{m-k} \left(\left\| \mathbf{H}^+ - \mathbf{H}_{[k]}^+ \right\|_F^2 + \left\| \mathbf{H}^- - \mathbf{H}_{[k]}^- \right\|_F^2 \right),$$

where

$$\mathbf{G}^+ = \sum_{t:s_t>0} s_t \mathbf{g}_t \mathbf{g}_t^\top \text{ and } \mathbf{G}^- = \sum_{t:s_t<0} -s_t \mathbf{g}_t \mathbf{g}_t^\top,$$

and matrices $\mathbf{H}^+$ and $\mathbf{H}^-$ satisfy

$$\mathbf{G}^+ = (\mathbf{H}^+)^\top \mathbf{H}^+ \text{ and } \mathbf{G}^- = (\mathbf{H}^-)^\top \mathbf{H}^-.$$

PROOF. By the definitions of $\mathbf{G}^+$ snd $\mathbf{G}^-$, and the procedure of Algorithm 2, the matrices $(\mathbf{B}^+)^\top \mathbf{B}^+$ and $(\mathbf{B}^-)^\top \mathbf{B}^-$ can be regarded as the result of frequent directions sketching on $\mathbf{G}^+$ and $\mathbf{G}^-$, respectively. Based on the analysis in Section 2 of [18], we have

$$\begin{aligned} &\left\| \mathbf{G}^+ - (\mathbf{B}^+)^\top \mathbf{B}^+ \right\|_2 \\ &= \left\| \sum_{t:s_t>0} \left(s_t \mathbf{g}_t \mathbf{g}_t^\top - (\mathbf{B}_t^+)^\top \mathbf{B}_t^+ + (\mathbf{B}_{t-1}^+)^\top \mathbf{B}_{t-1}^+ \right) \right\|_2 \\ &\leq \sum_{t:s_t>0} \left\| s_t \mathbf{g}_t \mathbf{g}_t^\top - (\mathbf{B}_t^+)^\top \mathbf{B}_t^+ + (\mathbf{B}_{t-1}^+)^\top \mathbf{B}_{t-1}^+ \right\|_2 \\ &\leq \frac{1}{m-k} \left\| \mathbf{H}^+ - \mathbf{H}_{[k]}^+ \right\|_F^2 \end{aligned} \tag{6}$$

for any $k < m$.

The similar result also holds on the negative component, that is,

$$\left\| \mathbf{G}^- - (\mathbf{B}^-)^\top \mathbf{B}^- \right\|_2 \leq \frac{1}{m-k} \left\| \mathbf{H}^- - \mathbf{H}_{[k]}^- \right\|_F^2. \tag{7}$$

Then we obtain the final result

$$\begin{aligned} &\left\| \mathbf{G} - \left((\mathbf{B}^+)^\top \mathbf{B}^+ - (\mathbf{B}^-)^\top \mathbf{B}^- \right) \right\|_2 \\ &\leq \left\| \mathbf{G}^+ - (\mathbf{B}^+)^\top \mathbf{B}^+ \right\|_2 + \left\| \mathbf{G}^- - (\mathbf{B}^-)^\top \mathbf{B}^- \right\|_2 \\ &\leq \frac{1}{m-k} \left(\left\| \mathbf{H}^+ - \mathbf{H}_{[k]}^+ \right\|_F^2 + \left\| \mathbf{H}^- - \mathbf{H}_{[k]}^- \right\|_F^2 \right), \end{aligned}$$

where the first inequality is based on the fact $\mathbf{G} = \mathbf{G}^+ - \mathbf{G}^-$ and triangle inequality; the second step comes from (6) and (7). □

3.2 Factorization Machine by SFTRL

An efficient online learning algorithm for FM need to effectively compute the sum of gradient matrices which is usually indefinite. Naturally, we use the GFD to estimate the sum of historical gradients of FTRL. We derive the Sketched-Follow-The-Regularized-Leader (SFTRL) for FM and present its details in Algorithm 4.

SFTRL can be regarded as an approximation to convex online FM with inexact gradients. Considering that the gradient of each loss function in online FM is rank-one indefinite and comes sequentially, we approximate the sum by GFD with two sketching matrices $\mathbf{B}^+, \mathbf{B}^- \in \mathbb{R}^{(m-1)\times(d+1)}$,

$$\sum_{s=1}^{t} \nabla f_s(\mathbf{\Theta}_s) \approx (\mathbf{B}_t^+)^\top \mathbf{B}_t^+ - (\mathbf{B}_t^-)^\top \mathbf{B}_t^-, \tag{8}$$

where the sketch size m is much smaller than d. As the analysis in Section 3.1, maintaining $\mathbf{B}^+$ and $\mathbf{B}^-$ requires $\mathcal{O}(md)$ running time for each iteration. The update of $\mathbf{\Theta}_t$ (Line 17 of Algorithm 4) can be expressed as

$$\mathbf{\Theta}_{t+1} = -\eta\left((\mathbf{B}_t^+)^\top \mathbf{B}_t^+ - (\mathbf{B}_t^-)^\top \mathbf{B}_t^- \right). \tag{9}$$

Because both $\mathbf{B}_t^+$ and $\mathbf{B}_t^-$ have already been obtained by GFD, we can implicitly obtain $\mathbf{\Theta}_{t+1}$ based on relationship (9) without any additional operation or memory. Any quadratic complexity in the dimension is avoided by using the low-rank approximation of GFD.

The prediction for data $\hat{\mathbf{a}}$ relies on the formula

$$z = -\hat{\mathbf{a}}^\top \big((\mathbf{B}_t^+)^\top \mathbf{B}_t^+ - (\mathbf{B}_t^-)^\top \mathbf{B}_t^-\big)\hat{\mathbf{a}},$$

which also costs $\mathcal{O}(md)$ time that does not exceed the update of $\mathbf{B}^+$ and $\mathbf{B}^-$. In summary, the computation cost of SFTRL is $\mathcal{O}(Tmd)$ in total and it requires $\mathcal{O}(md)$ space.

Algorithm 4 Sketched Follow-The-Regularizer-Leader (S-FTRL)

```
1: Input: hyperparameter $\eta > 0$, sketch size $m$, regularization function $R(\mathbf{\Theta}) = \frac{1}{2}\|\mathbf{\Theta}\|_F^2$
2: Initialize $\mathbf{\Theta}_1 = \mathbf{0}^{(d+1)\times(d+1)}$, $\mathbf{B}_0^+ = \mathbf{B}_0^- = \mathbf{0}^{(m-1)\times(d+1)}$
3: for $t = 1, \ldots, T$ do
4:     Receive data $\mathbf{a}_t$ and loss $f_t(\mathbf{\Theta}) = \phi_t(\hat{\mathbf{a}}_t^\top \mathbf{\Theta}\hat{\mathbf{a}}_t)$
5:     Predict by $\mathbf{\Theta}_t$
6:     Let $q_t = \phi_t'(\hat{\mathbf{a}}_t^\top \mathbf{\Theta}_t\hat{\mathbf{a}}_t)$
7:     if $q_t > 0$
8:         $\widehat{\mathbf{B}}_t^+ = \begin{bmatrix}\mathbf{B}_{t-1}^+ \\ \sqrt{q_t}\hat{\mathbf{a}}_t^\top\end{bmatrix}$
9:         $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \mathrm{svd}(\widehat{\mathbf{B}}_+^t)$
10:        $\mathbf{B}_t^+ = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2\mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
11:        $\mathbf{B}_t^- = \mathbf{B}_{t-1}^-$
12:    else
13:        $\widehat{\mathbf{B}}_t^- = \begin{bmatrix}\mathbf{B}_{t-1}^- \\ \sqrt{-q_t}\hat{\mathbf{a}}_t^\top\end{bmatrix}$
14:        $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \mathrm{svd}(\widehat{\mathbf{B}}_-^t)$
15:        $\mathbf{B}_t^- = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2\mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
16:        $\mathbf{B}_t^+ = \mathbf{B}_{t-1}^+$
17:    end if
18:    Update:
           $\mathbf{\Theta}_{t+1} = \operatorname*{argmin}_{\mathbf{\Theta}} \big\langle (\mathbf{B}_t^+)^\top \mathbf{B}_t^+ - (\mathbf{B}_t^-)^\top \mathbf{B}_t^-, \mathbf{\Theta}\big\rangle + \frac{1}{\eta}R(\mathbf{\Theta})$
19: end for
```

4 THEORETICAL ANALYSIS

In this section, we analyze the regret bound of SFTRL and show our result is comparable to the standard FTRL.

We first present the following lemma that will be used in the analysis of the regret.

Lemma 1. *We define the matrices*

$$\begin{aligned}\tilde{\nabla}_t =&(\mathbf{B}_t^+)^\top \mathbf{B}_t^+ - (\mathbf{B}_{t-1}^+)^\top \mathbf{B}_{t-1}^+ \\ &- \big((\mathbf{B}_t^-)^\top \mathbf{B}_t^+ - (\mathbf{B}_{t-1}^-)^\top \mathbf{B}_{t-1}^-\big) - q_t\hat{\mathbf{a}}_t\hat{\mathbf{a}}_t^\top,\end{aligned} \tag{10}$$

and the functions

$$g_0(\mathbf{\Theta}) = \frac{1}{\eta}R(\mathbf{\Theta}) \text{ and } g_t(\mathbf{\Theta}) = \langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}\rangle.$$

Then we have

$$\sum_{t=0}^{T} g_t(\mathbf{\Phi}) \geq \sum_{t=0}^{T} g_t(\mathbf{\Theta}_{t+1})$$

for any $\mathbf{\Phi} \in \mathbb{R}^{(d+1)\times(d+1)}$.

Proof. We prove this result by induction on T.
Induction base: By definition of g_0 and $\mathbf{\Theta}_1$, we have that

$$\mathbf{\Theta}_1 = \operatorname*{argmin}_{\mathbf{\Theta}} R(\mathbf{\Theta}).$$

Hence for all $\mathbf{\Phi}$, we have

$$g_0(\mathbf{\Phi}) = R(\mathbf{\Phi}) \geq R(\mathbf{\Theta}_1) = g_0(\mathbf{\Theta}_1).$$

Induction step: Assume that for any $T' > 0$, we have

$$\sum_{t=0}^{T'} g_t(\mathbf{\Phi}) \geq \sum_{t=0}^{T'} g_t(\mathbf{\Theta}_{t+1}). \tag{11}$$

Then we prove the statement for $T' + 1$. The induction assumption implies

$$\mathbf{\Theta}_{T'+2} = \operatorname*{argmin}_{\mathbf{\Theta}} \sum_{t=0}^{T'+1} g_t(\mathbf{\Theta}). \tag{12}$$

Hence, it holds that

$$\begin{aligned}\sum_{t=0}^{T'+1} g_t(\mathbf{\Phi}) &\geq \sum_{t=0}^{T'+1} g_t(\mathbf{\Theta}_{T'+2}) \\ &= \sum_{t=0}^{T'} g_t(\mathbf{\Theta}_{T'+2}) + g_{T'+1}(\mathbf{\Theta}_{T'+2}) \\ &\geq \sum_{t=0}^{T'} g_t(\mathbf{\Theta}_{t+1}) + g_{T'+1}(\mathbf{\Theta}_{T'+2}) \\ &= \sum_{t=0}^{T'+1} g_t(\mathbf{\Theta}_{t+1}),\end{aligned}$$

where the inequalities is due to the result of induction as (12); and the equalities hold by the definition of g_t. □

Then we consider the regret bound of SFTRL. The following theorem display the connection between the SFTRL and standard FTRL in theoretical.

Theorem 3. *By using the notation of Lemma 1, the regret of Algorithm 4 satisfies*

$$\begin{aligned}\mathrm{Regret}_T(\mathbf{\Theta}_*) \leq& 2\eta\sum_{t=1}^{T} \|\nabla f_t(\mathbf{\Theta}_t)\|_F^2 + \frac{R(\mathbf{\Theta}_*) - R(\mathbf{\Theta}_1)}{\eta} \\ &+ \sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_*\rangle,\end{aligned}$$

where $\mathbf{\Theta}_* = \mathrm{argmin}_{\mathbf{\Theta}\in\mathbb{R}^{(d+1)\times(d+1)}} \sum_{t=1}^{T} f_t(\mathbf{\Theta})$.

Proof. Firstly, we can bound the regret as follows

$$\begin{aligned}&\mathrm{Regret}_T(\mathbf{\Theta}_*) \\ =&\sum_{t=1}^{T} \Big(f_t(\mathbf{\Theta}_t) - f_t(\mathbf{\Theta}_*)\Big)\end{aligned}$$

$$\begin{aligned}
&\le \sum_{t=1}^{T} \langle \nabla f_t(\mathbf{\Theta}_t), \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle \\
&= \sum_{t=1}^{T} \langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle + \sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle \\
&= \sum_{t=1}^{T} \Big[g_t(\mathbf{\Theta}_t) - g_t(\mathbf{\Theta}_*) \Big] + \sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle \\
&\le \sum_{t=1}^{T} \Big[g_t(\mathbf{\Theta}_t) - g_t(\mathbf{\Theta}_{t+1}) \Big] + \Big[g_0(\mathbf{\Theta}_1) - g_0(\mathbf{\Theta}_*) \Big] \\
&\quad + \sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle \\
&= \sum_{t=1}^{T} \langle \nabla f(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle + \frac{R(\mathbf{\Theta}_1) - R(\mathbf{\Theta}_*)}{\eta} \\
&\quad + \sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle. \qquad (13)
\end{aligned}$$

The first inequality is obtained by the convex property as (3) and the second inequality comes from Lemma 1; and all the equalities are due to the definition of g_t.

Then we define the functions

$$h_t(\mathbf{\Theta}) = \eta \sum_{s=1}^{t} \langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta} \rangle + R(\mathbf{\Theta}),$$

for $t = 1, \ldots T$. The definition of (10) implies

$$(\mathbf{B}_t^+)^\top \mathbf{B}_t^+ - (\mathbf{B}_t^-)^\top \mathbf{B}_t^- = \sum_{s=1}^{t} \nabla f_s(\mathbf{\Theta}_s) - \sum_{s=1}^{t} \tilde{\nabla}_s t.$$

The update rule of Line in of Algorithm 4 means that $\mathbf{\Theta}_{t+1}$ is the minimizer of $h_t(\mathbf{\Theta})$. Hence, we have $\nabla h_t(\mathbf{\Theta}_{t+1}) = \mathbf{0}$ and $h_t(\mathbf{\Theta}_t)$ can be bounded as follows.

$$\begin{aligned}
&h_t(\mathbf{\Theta}_t) \\
=&h_t(\mathbf{\Theta}_{t+1}) + \langle \nabla h_t(\mathbf{\Theta}_{t+1}), \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle + \frac{1}{2} \|\mathbf{\Theta}_t - \mathbf{\Theta}_{t+1}\|_F^2 \\
\ge&h_t(\mathbf{\Theta}_{t+1}) + \frac{1}{2} \|\mathbf{\Theta}_t - \mathbf{\Theta}_{t+1}\|_F^2. \qquad (14)
\end{aligned}$$

For the Frobenius norm term in (14), we have that

$$\begin{aligned}
&\frac{1}{2} \|\mathbf{\Theta}_t - \mathbf{\Theta}_{t+1}\|_F^2 \\
\le&h_t(\mathbf{\Theta}_t) - h_t(\mathbf{\Theta}_{t+1}) \\
=&h_{t-1}(\mathbf{\Theta}_t) - h_{t-1}(\mathbf{\Theta}_{t+1}) + \eta \langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle \\
\le&\eta \langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle, \qquad (15)
\end{aligned}$$

where the first inequality comes from (14) and the second inequality is because $\mathbf{\Theta}_t$ is the minimizer of h_{t-1}.

Combing with the Cauchy-Schwarz inequality and the result of (15), we have

$$\begin{aligned}
&\langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle \\
\le&\|\nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t\|_F \|\mathbf{\Theta}_t - \mathbf{\Theta}_{t+1}\|_F \\
\le&\|\nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t\|_F \sqrt{2\eta \langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle}.
\end{aligned}$$

By rearranging the above inequality, we obtain

$$\begin{aligned}
&\langle \nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_{t+1} \rangle \\
\le&2\eta \|\nabla f_t(\mathbf{\Theta}_t) - \tilde{\nabla}_t\|_F^2 \\
\le&2\eta \|\nabla f_t(\mathbf{\Theta}_t)\|_F^2. \qquad (16)
\end{aligned}$$

Finally, we achieve the result of this theorem by concentrating (13) and (16). ☐

Compared with the regret bound of FTRL in Theorem 1, the one SFTRL has an additional term

$$\sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle.$$

We bound the difference in the following lemma.

Lemma 2. *Using the notation of the above lemmas and theorems, we have*

$$\begin{aligned}
&\sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle \\
\le&\frac{\max_t \|\mathbf{\Theta}_t - \mathbf{\Theta}_*\|_*}{m-k} \Big(\|\mathbf{M}^+ - \mathbf{M}_{[k]}^+\|_F^2 + \|\mathbf{M}^- - \mathbf{M}_{[k]}^-\|_F^2 \Big),
\end{aligned}$$

where $\mathbf{M}^+$ *and* $\mathbf{M}^-$ *satisfy*

$$\sum_{t:q_t>0} \nabla f_t(\mathbf{\Theta}_t) = (\mathbf{M}^+)^\top \mathbf{M}^+ \text{ and } \sum_{t:q_t>0} \nabla f_t(\mathbf{\Theta}_t) = -(\mathbf{M}^-)^\top \mathbf{M}^-,$$

and $k < m$.

Proof. We bound the difference term as follows

$$\begin{aligned}
&\sum_{t=1}^{T} \langle \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \rangle \\
=&\Big\langle \sum_{t=1}^{T} \tilde{\nabla}_t, \mathbf{\Theta}_t - \mathbf{\Theta}_* \Big\rangle \\
\le&\sum_{t=1}^{T} \|\tilde{\nabla}_t\|_2 \|\mathbf{\Theta}_t - \mathbf{\Theta}_*\|_* \\
\le&\max_t \|\mathbf{\Theta}_t - \mathbf{\Theta}_*\|_* \sum_{t=1}^{T} \|\tilde{\nabla}_t\|_2 \\
\le&\frac{\max_t \|\mathbf{\Theta}_t - \mathbf{\Theta}_*\|_*}{m-k} \Big(\|\mathbf{M}^+ - \mathbf{M}_{[k]}^+\|_F^2 + \|\mathbf{M}^- - \mathbf{M}_{[k]}^-\|_F^2 \Big),
\end{aligned}$$

where the first inequality is based on the Hölder inequality since the spectral and nuclear norms are dual to each other [42], and the last inequality comes from the result (6) and (7) in the proof of Theorem 2. ☐

Concentrating the results of Theorem 3 and Lemma 2, we can derive the regret bound of SFTRL.

Theorem 4. *The SFTRL Algorithm 4 attains the following bound on regret:*

$$\begin{aligned}
&\mathrm{Regret}_T(\mathbf{\Theta}_*) \\
\le&2\eta \sum_{t=1}^{T} \|\nabla f_t(\mathbf{\Theta}_t)\|_F^2 + \frac{R(\mathbf{\Theta}_*) - R(\mathbf{\Theta}_1)}{\eta} \qquad (17) \\
&+\frac{\max_t \|\mathbf{\Theta}_t - \mathbf{\Theta}_*\|_*}{m-k} \Big(\|\mathbf{M}^+ - (\mathbf{M}^+)_k\|_F^2 + \|\mathbf{M}^- - (\mathbf{M}^-)_k\|_F^2 \Big).
\end{aligned}$$

We further impose the local assumptions near $\mathbf{\Theta}_*$ such that $|R(\mathbf{\Theta}_1) - R(\mathbf{\Theta}_*)| \leq D_R$, $\|\mathbf{\Theta}_t - \mathbf{\Theta}_*\|_F \leq D_{\mathbf{\Theta}}$ and $\|\nabla f_t(\mathbf{\Theta}_t)\|_F^2 \leq D_G$. Then we can optimize over η on the bound (17) to obtain

$$\begin{aligned}&\text{Regret}_T(\mathbf{\Theta}_*)\\ \leq& 2D_G D_R\sqrt{2T} + \frac{D_{\mathbf{\Theta}}}{m-k}\Big(\|\mathbf{M}^+ - (\mathbf{M}^+)_k\|_F^2 + \|\mathbf{M}^- - (\mathbf{M}^-)_k\|_F^2\Big).\end{aligned}$$

In real-world applications, dataset usually lies in an approximate low-rank space, so the spectrums of $\mathbf{M}^+$ and $\mathbf{M}^-$ typically decay rapidly which leads to that the terms $\|\mathbf{M}^+ - (\mathbf{M}^+)_k\|_F^2$ and $\|\mathbf{M}^- - (\mathbf{M}^-)_k\|_F^2$ are very small and the regret bound of SFTRL is closed to the one of standard FTRL.

5 EXPERIMENTS

In this section, we evaluate the performance of the proposed Sketched Follow-The-Regularizer-Leader (SFTRL) for online FM on two popular machine learning tasks: online rating prediction for recommendation and online binary classification[1].

5.1 Experimental Setup

We compare SFTRL with various online learning algorithms based on compact convexified FM [28] including Online Gradient Descent (OGD); Follow-The-Regularized-Leader (FTRL) and Online Conditional Gradient (OCG) [28]. For the online recommendation tasks, we use the Movielens datasets[2], including Movielens-100K (MVL-100K), Movielens-1M (MVL-1M) and Movielens-10M (MVL-10M); for the online binary classification tasks, we use datasets from LibSVM[3]. including IJCNN, Spam and Epsilon. The details of the datasets are summarized in Table 1. To evaluate the performances on both tasks properly, we report the Root Mean Square Error (RMSE) for the rating prediction; and the Area Under Curve (AUC) for the binary classification.

Table 1: Statistical details of the datasets

Datasets	#Features	#Samples	Label
MLV-100K	2,625	100,000	Numerical
MLV-1M	9,940	1,000,209	Numerical
MLV-10M	82,248	10,000,000	Numerical
IJCNN	22	141,691	Binary
Spam	252	350,000	Binary
Epsilon	2,000	100,000	Binary

5.2 Online Rating Prediction

For online recommendation task, we receive user-item pairs sequentially and then predict the corresponding value of the incoming rating. We denote the sample arriving at round t as (u_t, i_t, b_t), where u_t is the user ID, i_t is the item ID, and b_t is the rating provided by user u_t to item i_t. Let $\mathcal{U}$ and $\mathcal{I}$ be the set of user IDs and item IDs respectively.

[1]The code is available in https://github.com/bmdy/SFTRL
[2]https://grouplens.org/datasets/movielens/
[3]https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Algorithm 5 Sketched-Follow-The-Regularizer-Leader (S-FTRL) with Doubling Space

```
1: Input: hyperparameter $\eta > 0$, sketch size $m$, regularization function $R(\mathbf{\Theta}) = \frac{1}{2}\|\mathbf{\Theta}\|_F^2$
2: Initialize $\mathbf{\Theta}_1 = \mathbf{0}^{(d+1)\times(d+1)}$ and $\mathbf{B}_0^+$, $\mathbf{B}_0^-$ be empty
3: for $t = 1, \ldots, T$ do
4:   Receive data $\mathbf{a}_t$ and loss $f_t(\mathbf{\Theta}) = \phi_t(\hat{\mathbf{a}}_t^\top \mathbf{\Theta}\hat{\mathbf{a}}_t)$
5:   Predict by $\mathbf{\Theta}_t$
6:   Let $q_t = \phi_t'(\hat{\mathbf{a}}_t^\top \mathbf{\Theta}\hat{\mathbf{a}}_t)$
7:   if $q_t > 0$
8:     $\widehat{\mathbf{B}}_t^+ = \begin{bmatrix}\mathbf{B}_{t-1}^+\\ \sqrt{q_t}\hat{\mathbf{a}}_t^\top\end{bmatrix}$
9:     if $\widehat{\mathbf{B}}_t^+$ has $2m$ rows
10:      $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{svd}(\widehat{\mathbf{B}}_+^t)$
11:      $\mathbf{B}_t^+ = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2\mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
12:    else
13:      $\mathbf{B}_t^+ = \mathbf{B}_{t-1}^+$
14:    end if
15:    $\mathbf{B}_t^- = \mathbf{B}_{t-1}^-$
16:  else
17:    $\widehat{\mathbf{B}}_t^- = \begin{bmatrix}\mathbf{B}_{t-1}^-\\ \sqrt{-q_t}\hat{\mathbf{a}}_t^\top\end{bmatrix}$
18:    if $\widehat{\mathbf{B}}_t^-$ has $2m$ rows
19:      $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{svd}(\widehat{\mathbf{B}}_-^t)$
20:      $\mathbf{B}_t^- = \sqrt{\mathbf{\Sigma}_{[m-1]}^2 - \sigma_{[m]}^2\mathbf{I}} \cdot \mathbf{V}_{[m-1]}^\top$
21:    else
22:      $\mathbf{B}_t^- = \mathbf{B}_{t-1}^-$
23:    end if
24:    $\mathbf{B}_t^+ = \mathbf{B}_{t-1}^+$
25:  end if
26:  Update:
       $\mathbf{\Theta}_{t+1} = \underset{\mathbf{\Theta}}{\text{argmin}} \left\langle (\mathbf{B}_t^+)^\top\mathbf{B}_t^+ - (\mathbf{B}_t^-)^\top\mathbf{B}_t^-, \mathbf{\Theta}\right\rangle + \frac{1}{\eta}R(\mathbf{\Theta})$
27: end for
```

We construct the input feature vector $\mathbf{a}_t$ with $|\mathcal{U}| + |\mathcal{I}|$ binary codes as follows

$$\mathbf{a}_t = \big[\overbrace{0,0,\ldots\underbrace{,1,}_{u_t\text{-th user}}\ldots 0,0}^{|\mathcal{U}|\text{ binary codes}},\overbrace{0,0,\ldots\underbrace{,1,}_{i_t\text{-th item}}\ldots 0,0}^{|\mathcal{I}|\text{ binary codes}}\big]^\top,$$

where $|\mathcal{U}|$ and $|\mathcal{I}|$ are the number of users and items. We take the least square loss functions

$$f_t(\mathbf{\Theta}) = \frac{1}{2}(\hat{\mathbf{a}}_t^\top \mathbf{\Theta}\hat{\mathbf{a}}_t - b_t)^2,$$

where $\hat{\mathbf{a}}_t = [\mathbf{a}_t^\top, 1]^\top$. The model predict the rating with $\hat{\mathbf{a}}_t^\top \mathbf{\Theta}_t \hat{\mathbf{a}}_t$ at t-th round with $\hat{\mathbf{a}}_t = [\mathbf{a}_t^\top, 1]^\top$ for input feature $\mathbf{a}_t \in \mathbb{R}^d$.

We list the RMSE of SFTRL and other compared algorithms in Table 2. From our observation, SFTRL achieves higher prediction accuracy than the other online learning algorithms. We measure the efficiency of OGD, FTRL, OCG and SFTRL on different datasets and show how fast the average losses decrease with the running time in Figure 1 and report the detailed running time for all these algorithms in Table 3. We clearly observe that SFTRL runs significantly faster than OGD, FTRL and OCG, which illustrates usefulness of using sketching technique in our algorithm. To investigate the influence of the sketch size m for the performance, we run a series of experiments using different m of SFTRL on Movielens100K. The result is Table 4 We find that larger m will lead to decreased training loss and increased running time.

Table 2: RMSE of online rating prediction

Algorithms	RMSE on Datasets		
	MVL-100K	MVL-1M	MVL-10M
OGD	1.2721	1.0645	1.0291
FTRL	1.0873	1.0441	0.9725
OCG	1.0359	0.9702	0.9441
SFTRL	**0.9624**	**0.9336**	**0.8811**

Table 3: Running time (s) comparison in online rating prediction (the sketch sizes of SFTRL are 10, 10 and 50 for MVL-100K, MVL-1M and MVL-10M respectively)

Datasets	SFTRL	OCG	Other
MVL-100K	167.39	$> 10^4$	$> 10^5$
MVL-1M	557.73	$> 10^5$	$> 10^6$
MVL-10M	3248.71	$> 10^5$	$> 10^6$

Table 4: Average Training Loss of SFTRL under varying sketch sizes on dataset Movielens 100K

m	Average Training Loss	Time(s)
10	0.988384	532.20
20	0.985037	1084.93
30	0.984426	1824.53
40	0.983962	2609.74
50	0.983917	3248.71

5.3 Online Binary Classification

For online binary classification problems, the examples are denoted as $(\mathbf{a}_t, b_t)$, where $\mathbf{a}_t \in \mathbb{R}^d$ is the input feature and $b_t \in \{-1, 1\}$ is the label. We take the logistic loss function that

$$f_t(\mathbf{\Theta}) = \log\left(1 + \exp(-b_t(\hat{\mathbf{a}}_t^\top \mathbf{\Theta} \hat{\mathbf{a}}_t)\right)$$

with $\hat{\mathbf{a}}_t = [\mathbf{a}_t^\top, 1]^\top$. For each round, we predict the label by $\text{sign}(\hat{\mathbf{a}}_t^\top \mathbf{\Theta}_t \hat{\mathbf{a}}_t)$ with the argument input vector $\hat{\mathbf{a}}_t \in \mathbb{R}^{d+1}$.

We list the AUC of SFTRL and other compared algorithms in Table 5. SFTRL achieves higher classification accuracy than all the other online learning baselines. We investigate the running time similar to previous task. The results are displayed in Figure 2 and Table 6-7. As we expected, SFTRL is still significantly faster than baselines on classification tasks.

Table 5: AUC in online binary classification tasks

Algorithms	IJCNN	Spam	Epsilon
OGD	0.9192	0.9239	0.8277
FTRL	0.9757	0.9398	0.8668
OCG	0.9859	0.9414	0.8737
SFTRL	**0.9926**	**0.9866**	**0.9508**

Table 6: Running time(s) comparison in online binary classification on dataset Spam, IJCNN and Epsilon (the sketch sizes of SFTRL are 100, 50 and 5 for Spam, IJCNN and Epsilon respectively)

Datasets	SFTRL	OCG	Other
Spam	447.19	$> 10^3$	$> 10^4$
IJCNN	184.31	$> 10^3$	$> 10^4$
Epsilon	292.69	$> 10^4$	$> 10^5$

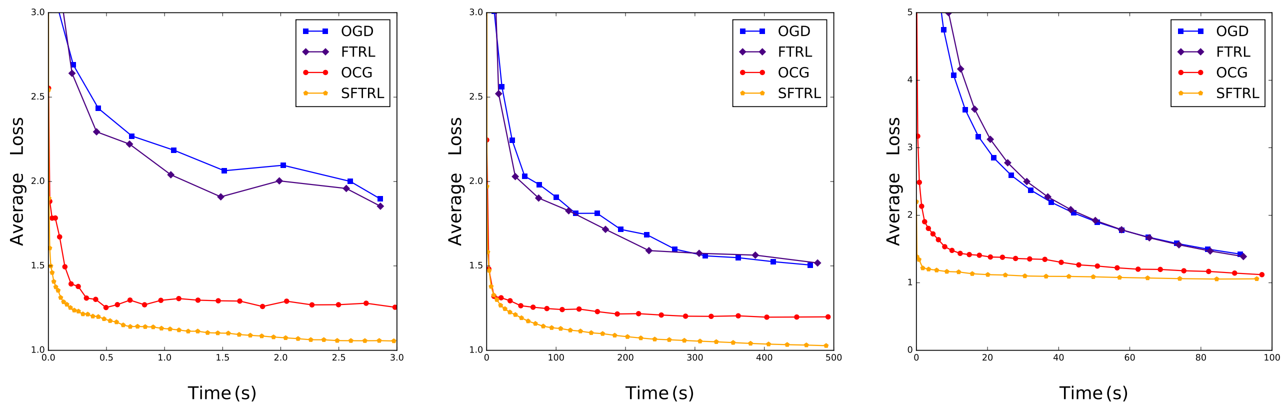
Table 7: Average Training Loss of SFTRL under varying sketch sizes on dataset Epsilon

m	Average Training Loss	Time(s)
5	0.150675	292.69
10	0.150138	539.08
15	0.150062	618.15
20	0.150125	823.27
25	0.149838	1129.68

6 RELATED WORK

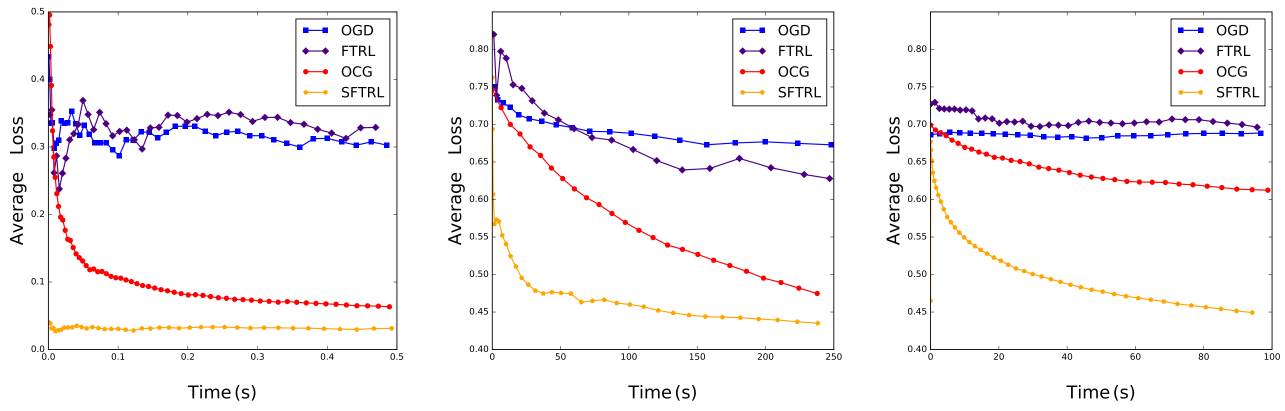
The existing optimization algorithms for FM can be classified into two categories. One category is non-convex formulation [6, 8, 23, 26, 29, 38], which defines the interaction (or compact) matrix as $\mathbf{X} = \mathbf{Q}\mathbf{Q}^\top$, and optimizes on thin matrix $\mathbf{Q}$. The bad local minima or saddle point may affect the performance heavily. Moreover, it is difficult to study how to select a good initial point in theoretical.

The other one is convex formulation [5, 28, 44] with nuclear norm regularization (or constraint), but these methods are



(a) MVL-100K (b) MVL-1M (c) MVL-10M

Figure 1: Comparison of the efficiency for online rating prediction on Movielens 100K, 1M and 10M



(a) IJCNN (b) Spam (c) Epsilon

Figure 2: Comparison of the efficiency of OGD, FTRL, OCG and SFTRL on IJCNN, Spam and Epsilon

not vey efficient for online learning. For example, the update of online conditional gradient [28] needs to compute the leading singular vector of a $d \times d$ matrix, which requires a few inner iterations by $\mathcal{O}(d^2)$ running time. The low-rank structure of the interaction matrix is only benefit to the prediction step. Compared with the above methods, our SFTRL simultaneously enjoys the efficient update as non-convex formulation and achieves the stable performance like convex models.

Most of existing efficient online learning algorithms by sketching [31, 32] focus on approximating the Hessian matrix that is positive semi-definite. Recently, one has studied to use the idea of FD to estimate the products of two matrices in general cases [34, 45]. Applying these methods to online FM achieves comparable regret bound with different expression to our algorithm, but gives rise to more computational cost than using GFD because they ignore the symmetric property of the matrix of gradients.

7 CONCLUSION

In this paper, we have proposed an online FM algorithms called Sketched Follow-The-Regularizer-Leader. SFTRL utilizes the structure of FM model and approximates the parameter in a low-rank space via sketching. It achieves better prediction performance than the state-of-the-art convex formulations. We have proved that SFTRL has comparable regret bound to the standard FTRL and the experimental results on recommendation and binary classification tasks have shown that SFTRL is better than the state-of-art online learning algorithms.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (No. 61572017 and 11771002), National Program on Key Basic Research Project (No. 2015CB352300), National Natural Science Foundation of China Major Project

(No. U1611461), the NSERC Discovery Grant program, the Canada Research Chair program and the NSERC Strategic Grant program. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

We thank Xiao Lin for his help in conducting the experiments.

REFERENCES

[1] Nir Ailon and Bernard Chazelle. 2006. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *STOC*.

[2] Nir Ailon and Edo Liberty. 2009. Fast dimension reduction using Rademacher series on dual BCH codes. *Discrete & Computational Geometry* 42, 4 (2009), 615.

[3] Nir Ailon and Edo Liberty. 2013. An almost optimal unrestricted fast Johnson-Lindenstrauss transform. *ACM Transactions on Algorithms (TALG)* 9, 3 (2013), 21.

[4] Peter L. Bartlett, Elad Hazan, and Alexander Rakhlin. 2007. Adaptive Online Gradient Descent. In *NIPS*.

[5] Mathieu Blondel, Akinori Fujino, and Naonori Ueda. 2015. Convex Factorization Machines. In *ECML/PKDD*.

[6] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. 2016. Higher-Order Factorization Machines. In *NIPS*.

[7] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. 2017. Second-Order Kernel Online Convex Optimization with Adaptive Sketching. In *ICML*.

[8] Chen Cheng, Fen Xia, Tong Zhang, Irwin King, and Michael R. Lyu. 2014. Gradient boosting factorization machines. In *RecSys*.

[9] Kenneth L Clarkson and David P Woodruff. 2013. Low rank approximation and regression in input sparsity time. In *STOC*.

[10] Amey Desai, Mina Ghashami, and Jeff M Phillips. 2016. Improved practical matrix sketching with guarantees. *IEEE Transactions on Knowledge and Data Engineering* 28, 7 (2016), 1678–1690.

[11] Petros Drineas, Ravi Kannan, and Michael W Mahoney. 2006. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM J. Comput.* 36, 1 (2006), 132–157.

[12] Petros Drineas, Ravi Kannan, and Michael W Mahoney. 2006. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on computing* 36, 1 (2006), 158–183.

[13] Petros Drineas, Malik Magdon-Ismail, Michael W Mahoney, and David P Woodruff. 2012. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research* 13, Dec (2012), 3475–3506.

[14] Petros Drineas, Michael W Mahoney, and S Muthukrishnan. 2008. Relative-error CUR matrix decompositions. *SIAM J. Matrix Anal. Appl.* 30, 2 (2008), 844–881.

[15] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[16] Alan Frieze, Ravi Kannan, and Santosh Vempala. 2004. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)* 51, 6 (2004), 1025–1041.

[17] Mina Ghashami, Edo Liberty, and Jeff M Phillips. 2016. Efficient frequent directions algorithm for sparse matrices. In *SIGKDD*.

[18] Mina Ghashami, Edo Liberty, Jeff M Phillips, and David P Woodruff. 2016. Frequent directions: Simple and deterministic matrix sketching. *SIAM J. Comput.* 45, 5 (2016), 1762–1792.

[19] Ming Gu and Stanley C Eisenstat. 1993. A stable and fast algorithm for updating the singular value decomposition. *Technical Report YALEU/DCS/RR-966* (1993).

[20] Weiyu Guo, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Personalized ranking with pairwise Factorization Machines. *Neurocomputing* 214 (2016), 191–200.

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

[22] William B Johnson and Joram Lindenstrauss. 1984. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics* 26, 189-206 (1984), 1.

[23] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. 2017. Field-aware factorization machines in a real-world online advertising system. In *WWW Companion*.

[24] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *RecSys*.

[25] Daniel M Kane and Jelani Nelson. 2014. Sparser johnson-lindenstrauss transforms. *Journal of the ACM (JACM)* 61, 1 (2014), 4.

[26] Takuya Kitazawa. 2016. Incremental Factorization Machines for Persistently Cold-starting Online Item Recommendation. *arXiv preprint arXiv:1607.02858* (2016). https://arxiv.org/abs/1607.02858

[27] Edo Liberty. 2013. Simple and deterministic matrix sketching. In *SIGKDD*.

[28] Xiao Lin, Wenpeng Zhang, Min Zhang, Wenwu Zhu, Jian Pei, Peilin Zhao, and Junzhou Huang. 2018. Online Compact Convexified Factorization Machine. In *WWW*.

[29] Chun-Ta Lu, Lifang He, Weixiang Shao, Bokai Cao, and Philip S. Yu. 2017. Multilinear Factorization Machines for Multi-Task Multi-View Learning. In *WSDM*.

[30] Chun-Ta Lu, Lifang He, Weixiang Shao, Bokai Cao, and Philip S Yu. 2017. Multilinear factorization machines for multi-task multi-view learning. In *WSDM*.

[31] Haipeng Luo, Alekh Agarwal, Nicolò Cesa-Bianchi, and John Langford. 2016. Efficient second order online learning by sketching. In *NIPS*.

[32] Luo Luo, Cheng Chen, Zhihua Zhang, Wu-Jun Li, and Tong Zhang. 2017. Robust Frequent Directions with Application in Online Learning. *arXiv preprint arXiv:1705.05067* (2017). https://arxiv.org/abs/1705.05067

[33] H. Brendan McMahan. 2011. Follow-the-Regularized-Leader and Mirror Descent: Equivalence Theorems and L1 Regularization. In *AISTATS*.

[34] Youssef Mroueh, Etienne Marcheret, and Vaibhava Goel. 2017. Co-Occurring Directions Sketching for Approximate Matrix Multiply. In *AISTATS*.

[35] Jelani Nelson and Huy L Nguyên. 2013. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *FOCS*.

[36] Trung V Nguyen, Alexandros Karatzoglou, and Linas Baltrunas. 2014. Gaussian process factorization machines for context-aware recommendations. In *SIGIR*.

[37] Dimitris Papailiopoulos, Anastasios Kyrillidis, and Christos Boutsidis. 2014. Provable deterministic leverage score sampling. In *SIGKDD*.

[38] Steffen Rendle. 2010. Factorization machines. In *ICDM*.

[39] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *SIGIR*.

[40] Tamas Sarlos. 2006. Improved approximation algorithms for large matrices via random projections. In *FOCS*.

[41] Shai Shalev-Shwartz. 2011. Online learning and online convex optimization. *Foundations and Trends in Machine Learning* 4, 2 (2011), 107–194.

[42] Lloyd N Trefethen and David Bau III. 1997. *Numerical linear algebra*. Vol. 50. Society for Industrial and Applied Mathematics.

[43] David P. Woodruff. 2014. Sketching as a Tool for Numerical Linear Algebra. *Foundations and Trends in Theoretical Computer Science* 10, 1-2 (2014), 1–157.

[44] Makoto Yamada, Wenzhao Lian, Amit Goyal, Jianhui Chen, Kishan Wimalawarne, Suleiman A Khan, Samuel Kaski, Hiroshi Mamitsuka, and Yi Chang. 2017. Convex factorization machine for toxicogenomics prediction. In *SIGKDD*.

[45] Qiaomin Ye, Luo Luo, and Zhihua Zhang. 2016. Frequent Direction Algorithms for Approximate Matrix Multiplication with Applications in CCA. In *IJCAI*.

[46] Wenpeng Zhang, Xiao Lin, Tong Zhang, Peilin Zhao, Wenwu Zhu, Min Zhang, and Jian Pei. 2018. Compact convexified factorization machine: formulation and online algorithms. *arXiv preprint arXiv:1802.01379* (2018). https://arxiv.org/abs/1802.01379

[47] Martin Zinkevich. 2003. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *ICML*.