

A Latent Factor Analysis-Based Approach to Online Sparse Streaming Feature Selection

Di Wu^{ID}, *Member, IEEE*, Yi He^{ID}, Xin Luo^{ID}, *Senior Member, IEEE*, and MengChu Zhou^{ID}, *Fellow, IEEE*

Abstract—Online streaming feature selection (OSFS) has attracted extensive attention during the past decades. Current approaches commonly assume that the feature space of fixed data instances dynamically increases without any missing data. However, this assumption does not always hold in many real applications. Motivated by this observation, this study aims to implement online feature selection from sparse streaming features, i.e., features flow in one by one with missing data as instance count remains fixed. To do so, this study proposes a latent-factor-analysis-based online sparse-streaming-feature selection algorithm (LOSSA). Its main idea is to apply latent factor analysis to pre-estimate missing data in sparse streaming features before conducting feature selection, thereby addressing the missing data issue effectively and efficiently. Theoretical and empirical studies indicate that LOSSA can significantly improve the quality of OSFS when missing data are encountered in target instances.

Index Terms—Big data, computational intelligence, latent factor analysis (LFA), missing data, online algorithm, online feature selection, sparse streaming feature, streaming feature.

Manuscript received September 23, 2020; revised March 9, 2021 and May 20, 2021; accepted June 30, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61702475, Grant 61772493, Grant 62002337, and Grant 61902370; in part by the Natural Science Foundation of Chongqing, China, under Grant cstc2019jcyj-msxmX0578 and Grant cstc2019jcyjX0013; in part by the Chinese Academy of Sciences “Light of West China” Program; in part by the Technology Innovation and Application Development Project of Chongqing, China, under Grant cstc2018jszx-cydzX0041 and Grant cstc2019jcsx-fxydX0027; in part by CAAI-Huawei MindSpore Open Fund under Grant CAAIXSJLJJ-2020-004B; and in part by the Pioneer Hundred Talents Program of Chinese Academy of Sciences. This article was recommended by Associate Editor Z. Yu. (Di Wu and Xin Luo contributed equally to this work.) (Corresponding author: Xin Luo.)

Di Wu is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, and the Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China (e-mail: wudi@cigit.ac.cn).

Yi He is with the Department of Computer Science, Old Dominion University, Norfolk, VA 23462 USA (e-mail: yi.he.2021@outlook.com).

Xin Luo is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, and the Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China, and also with the Department of Big Data Analyses Techniques, Hengrui (Chongqing) Artificial Intelligence Research Center, Cloudwalk, Chongqing 401331, China (e-mail: luoxin21@cigit.ac.cn).

MengChu Zhou is with the Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA, and also with the Institute of Systems Engineering and Collaborative Laboratory for Intelligent Science and Systems, Macau University of Science and Technology, Macau 999078, China (e-mail: zhou@njit.edu).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TSMC.2021.3096065>.

Digital Object Identifier 10.1109/TSMC.2021.3096065

I. INTRODUCTION

IN THIS era of Information Explosion, people are inundated by big data [1]–[3]. For example, Google maintains more than 20-PB data and Flickr generates more than 3.6-TB data per day [4]. The global data sum is predicted to grow from 33 ZB in 2018 to 175 ZB by 2025 [5]. How to efficiently filter the valuable information out of mass data for our use has become a great challenge [1], [2], [6].

High dimensionality is one of the typical characteristics of big data from various areas like healthcare, social media, bioinformatics, electronic commerce, and online education [7]. It causes issues of computation costs and high storage, visualization and comprehension difficulties, and performance degradation on new data [8]. Hence, it is necessary to conduct feature selection on high-dimensional data [9]–[11]. As such, the original data with high dimensionality can be reduced into a low-dimensional feature space as well as keeping their essential characteristics to well discriminate each involved individual instance [12], [13].

Traditional feature selection approaches mostly assume that the feature space is predefined and static [14], [15]. But in real-world applications, such space usually keeps increasing dynamically [16], [17], making it impossible to observe the full feature space in advance [12]. To address this issue, many online streaming feature selection (OSFS) approaches are proposed [12], [18]–[25]. For instance, Perkins and Theiler [18] proposed the Grafting algorithm with a regularized framework. Zhou *et al.* [19] proposed an Alpha-investing algorithm by using stream-wise regression. Wu *et al.*, developed a Fast-OSFS algorithm [12] based on online relevance analysis. Besides, other representative algorithms include scalable and accurate online approach (SAOLA) [20] and online selection for features based on adaptive sliding-window (OSFASW) [21].

Despite their efficiency [12], existing OSFS approaches all assume that the input streaming features are complete without any missing data. Fig. 1(a) illustrates an example of this assumption. However, it does not always hold because we cannot completely collect dynamic features in many real applications [28], [29]. For instance, in an intelligent healthcare platform [26], [27], since the features that describe a patient's symptoms may generate from different inspection equipment (respiratory sensors, thermometers, pulse monitors, *etc.*) and healthcare service providers (hospitals, insurance companies, labs, *etc.*), it is impossible to collect all of them. Motivated by this phenomenon, we formulate such dynamic features

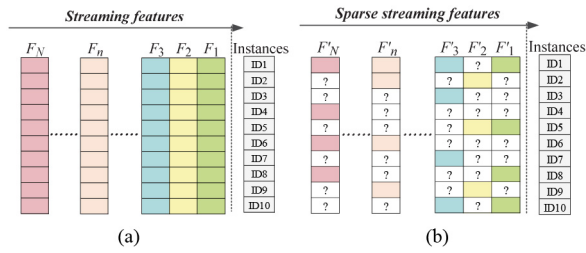


Fig. 1. Example of illustrating the different dynamic features scenarios. Multiple colors denote the observed features, symbols “?” denote the unobserved features (missing data). (a) Streaming features. (b) Sparse streaming features.

as *sparse streaming features*. As illustrated in Fig. 1(b), sparse streaming features flow in one by one with missing data while their number of data instances remains fixed. Thus, the problem of *Online Sparse Streaming Feature Selection* (OS²FS) rises, i.e., how to implement online feature selection from sparse streaming features without information loss?

Sparse data analysis and representation is a vital yet thorny issue in the area of big data analysis and knowledge discovery [30]–[37]. To date, a latent factor analysis (LFA)-based approach has proven to be highly efficient in addressing it [30]–[37]. For target sparse data, an LFA algorithm models them into a high-dimensional and sparse (HiDS) matrix or tensor and builds its low-rank approximation [32], [42]. Note that all entries of the achieved approximation are available, which can be considered as the representation to an HiDS matrix or tensor based on its known data only. Hence, this approximation precisely represents the known data while estimates the unknown ones of an HiDS matrix or tensor [40], [41]. From this point of view, will it be capable of pre-estimating the missing data of sparse streaming features, thereby establishing high-quality OS²FS algorithms based on existing OSFS ones?

Aiming at answering this question, we for the first time propose a latent-factor-analysis-based online sparse-streaming-feature selection algorithm (LOSSA). Its main idea is to adopt LFA [30]–[37] to pre-estimate missing data in sparse streaming features, and then adopt an OSFS algorithm on the achieved complete features to implement precise feature selection. We attempt to make the following contributions.

- 1) *Formulation of OS²FS Problem*: The problem of OSFS is generalized to include online sparse streaming feature selection (OS²FS), which is more frequently encountered in real applications.
- 2) *LOSSA Algorithm*: This algorithm is compatible with existing OSFS algorithms as well as helps them in implementing high-quality OS²FS without significant modifications, thereby establishing a new direction in performing OS²FS from the perspective of sparse data representation learning.
- 3) *Theoretical analysis of the proposed lossa algorithm*, including convergence analysis, algorithm design, and time complexity analysis. From these analyses, the performance of an LOSSA algorithm is theoretically guaranteed.

Empirical studies on 12 benchmark datasets from various big data-related applications are carefully conducted to evaluate LOSSA's performance. The results demonstrate that compared with existing OSFS algorithms, an LOSSA algorithm can effectively improve them to handle the OS²FS problem.

Section II introduces the preliminaries. Section III presents LOSSA. Section IV provides the experimental results. Section V discusses related studies. Finally, Section VI concludes this article.

II. PRELIMINARIES

A. Notations

Table I summarizes the adopted symbols of this article.

B. Online Feature Selection From Feature Stream

We first define streaming features and OSFS as follows.

Definition 1 (Streaming Features [12]): Supposing an instance set that has M instances and a feature set F that has N features are given. Let $F = \{F_1, F_2, \dots, F_N\}$ where $F_n = [f_{1,n}, f_{2,n}, \dots, f_{M,n}]^T$, $n \in \{1, 2, \dots, N\}$, is a vector that corresponds to M instances. Streaming features are encountered when F is presented sequentially. At each time point n , we only observe F_n as N is unknown.

Definition 2 (OSFS [12], [20]): Let $O_{n-1} = \{F_1, F_2, \dots, F_{n-1}\}$ be the observed streaming features set till time point $n-1$. Let $S_{n-1} \subseteq O_{n-1}$ be the selected streaming features set at time point $n-1$. OSFS is taken at a time point n to select a minimum subset S_n from $S_{n-1} \cup \{F_n\}$ to maximize a resultant model's performance.

C. Latent Factor Analysis

LFA [30]–[37] originates from decomposition-based matrix methods [38], [39]. It is widely adopted in many big data-related applications like a recommender system [40], [41]. Given a sparse matrix, an LFA model estimates its missing data by training latent factor matrices based on its known data only [32], [42]. We recall its definition [30]–[32], [43].

Definition 3 (LFA): Supposing a sparse matrix $Y^{W \times Z}$ is given, an LFA model is to obtain the rank- d approximation $\hat{Y}$ for Y . It trains two latent factor matrices $V^{Z \times d}$ and $U^{W \times d}$ on Y 's known entry set by minimizing the sum errors between $\hat{Y}$ and Y , where $\hat{Y} = UV^T$.

With Definition 3, we see that how to model an objective function to measure the sum errors between $\hat{Y}$ and Y is very crucial. Commonly, we adopt Euclidean distance to model objective function for LFA [30]–[32], [37], [43]

$$\varepsilon(U, V) = \frac{1}{2} \left\| \Omega \odot (Y - \hat{Y}) \right\|_F^2 = \frac{1}{2} \left\| \Omega \odot (Y - UV^T) \right\|_F^2 \quad (1)$$

where $\odot$ denotes the Hadamard product and Ω is a $W \times Z$ binary index matrix defined as follows:

$$\Omega_{w,z} = \begin{cases} 1, & \text{if } y_{w,z} \text{ is observed} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where $\Omega_{w,z}$ denotes the entry at the w th row and z th column of Ω . To avoid overfitting, L_2 -norm-based regularization is

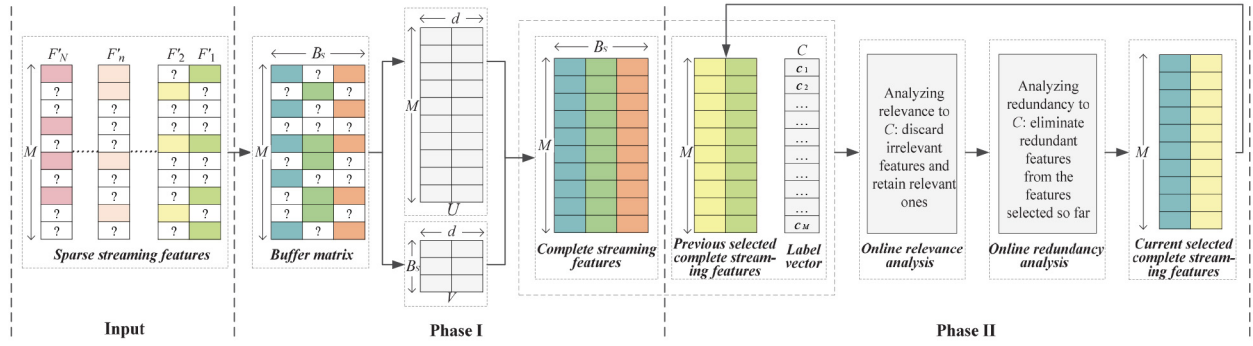


Fig. 2. Flowchart of LOSSA to achieve OS²FS.

TABLE I
SYMBOL ANNOTATIONS

Symbol	Explanation
M	Instance count.
F	Streaming features set, $F = \{F_1, F_2, \dots, F_N\}$.
n	A time point, $n \in \{1, 2, \dots, N\}$.
F'_n	F 's n -th streaming feature vector. It describes M instances.
$f'_{m,n}$	F'_n 's m -th element, $m \in \{1, 2, \dots, M\}$.
O_n	Observed streaming features set till n , $O_n = \{F'_1, F'_2, \dots, F'_n\}$.
S_n	Selected streaming features set at time point n , $S_n \subseteq O_n$.
F'	Sparse streaming features set, $F' = \{F'_1, F'_2, \dots, F'_N\}$.
F'_n	The n -th vector of F' .
$f'_{m,n}$	F'_n 's m -th element, $m \in \{1, 2, \dots, M\}$.
K_n	Known entry (observed entry) set of F'_n .
α_n	Missing data rate of F'_n .
α	Total missing data rate of F' .
O'_n	Observed sparse streaming features set till time point n , $O'_n = \{F'_1, F'_2, \dots, F'_n\}$.
S'_n	Selected sparse streaming features set at n , $S'_n \subseteq O'_n$.
C	Label vector of M instances, $C = [c_1, c_2, \dots, c_M]^T$.
Γ, Λ, ζ	Each one denotes a set.
B	A buffer matrix. It buffers the arrived F'_n from time point n to time point $n+B_S-1$.
B_S	Column number of B .
d	Latent factor dimension.
$\hat{B}$	The rank- d approximation for B achieved by using latent factor analysis based on B 's known entry set.
$\hat{F}'_n$	The completed streaming feature in $\hat{B}$ that corresponds to the prediction for F'_n .
$\hat{S}'_n$	Completed streaming features set that has been selected till time point n .
$M_B(C)_n$	Markov blanket of C at time point n .
R	A redundant completed streaming feature to C .
$P(\cdot \cdot)$	The conditional probability.
$Y^{W \times Z}$	A sparse matrix that has Z columns and W rows.
$y_{w,z}$	One element of Y at z -th column and w -th row, $z \in \{1, 2, \dots, Z\}$, $w \in \{1, 2, \dots, W\}$.
$U^{W \times d}$	Latent factor matrix. It corresponds to $Y^{W \times Z}$.
$u_{w,:}$	The w -th row-vector of U .
$V^{Z \times d}$	Latent factor matrix. It corresponds to $Y^{W \times Z}$.
$v_{:,z}$	The z -th row-vector of V .
$\hat{Y}$	The rank- d approximation for Y achieved by using latent factor analysis based on Y 's known entry set.
$\hat{y}_{w,z}$	One element of $\hat{Y}$ at z -th column and w -th row. It denotes prediction for $y_{w,z}$.
λ	A regularization parameter for LFA.
η	A learning rate parameter for LFA.
Ω	A $W \times Z$ binary index matrix.
$\odot$	Computing the Hadamard product.
$\ \cdot\ _F$	Computing the Frobenius norm of an enclosed matrix.

incorporated into (1) [30]–[32], [43], resulting in

$$\varepsilon(U, V) = \frac{1}{2} \|\Omega \odot (Y - UV^T)\|_F^2 + \frac{\lambda}{2} (\|U\|_F^2 + \|V\|_F^2) \quad (3)$$

where λ is the regularization parameter. With an optimization algorithm like stochastic gradient descent (SGD) to minimize (3), U and V can be achieved conveniently.

III. PROPOSED ALGORITHM

A. Problem of OS²FS

Definition 4 (Sparse Streaming Features): Corresponding to streaming features set $F = \{F_1, F_2, \dots, F_N\}$, sparse streaming features are encountered when there are some missing data $\forall F_n = [f_{1,n}, f_{2,n}, \dots, f_{M,n}]^T$, ($n \in \{1, 2, \dots, N\}$). Let $F' = \{F'_1, F'_2, \dots, F'_N\}$ indicates a set of sparse streaming features. Then the missing data rate α_n of F'_n is computed by $\alpha_n = 1 - |K_n|/M$ as K_n be the known entries set of F'_n . At each time point n , we only observe F'_n as N is unknown.

Given F' , let $O'_{n-1} = \{F'_1, F'_2, \dots, F'_{n-1}\}$ be the observed sparse streaming features set till time point $n-1$. Let S'_{n-1} be the selected sparse streaming features set at time point $n-1$. Then, we have $S'_{n-1} \subseteq O'_{n-1}$. The challenge of OS²FS is to select a minimum subset S'_n from $S'_{n-1} \cup \{F'_n\}$ to maximize a resultant model's performance at each time point n . Assuming that the concerned model is a classifier with $C = [c_1, c_2, \dots, c_M]^T$ being a label vector for M instances, we formulate the problem of OS²FS as

$$S'_n = \arg \min_{\Gamma} \left\{ |\Gamma| : \Gamma = \arg \max_{\Lambda \subseteq S'_{n-1} \cup \{F'_n\}} P(C|\Lambda) \right\} \quad (4)$$

where Γ and Λ denote a set. Hence, the problem of OS²FS is to select a minimum subset S'_n from $S'_{n-1} \cup \{F'_n\}$ at time point n to maximize a classifier's accuracy on M instances with label $C = [c_1, c_2, \dots, c_M]^T$.

B. LOSSA

To make LOSSA as flexible as possible to improve existing OSFS algorithms to handle OS²FS, we design its processing flow, as shown in Fig. 2. LOSSA has two phases. Phase I preprocesses sparse streaming features to complete their missing data. Phase II performs feature selection from the completed features. Next, a case starting from a time point n is given to elaborate Phase I and Phase II of LOSSA.

1) *Phase I:* Let $B^{M \times B_S}$ buffer the arrived sparse streaming features. When it is well filled from time point n to $n+B_S-1$, it consists of $\{F'_n, F'_{n+1}, \dots, F'_{n+B_S-1}\}$. To address its incomplete data, we define a completed streaming feature matrix $\hat{B}$ next.

Definition 5 (Completed Streaming Feature Matrix): Supposing $B = \{F'_n, F'_{n+1}, \dots, F'_{n+B_S-1}\}$ is given. Its completed streaming feature matrix $\hat{B}$ is B 's rank- d approximation

achieved by training an LFA model on known entries set of B . Features in B denoted by $\{\hat{F}'_n, \hat{F}'_{n+1}, \dots, \hat{F}'_{n+B_S-1}\}$ are completed streaming features.

According to Definition 5, to obtain $\hat{B}$, it is necessary to apply LFA to B . However, since B contains numerous missing data, the objective function (3) needs to be reformulated to a single-entry-based form as [30], [32]

$$\begin{aligned} \forall m \in \{1, 2, \dots, M\} \quad \forall j \in \{n, n+1, \dots, n+B_S-1\}: \\ \varepsilon = \frac{1}{2} \sum_{f'_{m,j} \in K_j} \left(f'_{m,j} - \sum_{k=1}^d u_{m,k} v_{j,k} \right)^2 \\ + \frac{\lambda}{2} \sum_{f'_{m,j} \in K_j} \left(\sum_{k=1}^d u_{m,k}^2 + \sum_{k=1}^d v_{j,k}^2 \right) \end{aligned} \quad (5)$$

where K_j is the known entries set of F'_j , $f'_{m,j}$ is the m th element of F'_j , $u_{m,k}$ is the k th element of vector $u_{m,\cdot}$, and $v_{j,k}$ is the k th element of vector $v_{j,\cdot}$. Note that similar objective functions, e.g., the compound rank- k projections [68], are provided to address complete inputs effectively. However, (5) focuses on the LFs (i.e., projections) related to the known entries of B only. With such design, it efficiently and precisely represents buffered sparse streaming features as shown in Fig. 2.

Consider the instant loss $\varepsilon_{m,j}$ on a single entry $f'_{m,j}$ in (5), we have

$$\varepsilon_{m,j} = \frac{1}{2} \left(f'_{m,j} - \sum_{k=1}^d u_{m,k} v_{j,k} \right)^2 + \frac{\lambda}{2} \left(\sum_{k=1}^d u_{m,k}^2 + \sum_{k=1}^d v_{j,k}^2 \right). \quad (6)$$

As analyzed in [31], [32], and [44], SGD has the advantage of easy implementation and fast convergence rate. Hence, we adopt SGD to minimize (6)

$$\forall k \in \{1, 2, \dots, d\} : \begin{cases} u_{m,k} \leftarrow u_{m,k} - \eta \frac{\partial \varepsilon_{m,j}}{\partial u_{m,k}} \\ v_{j,k} \leftarrow v_{j,k} - \eta \frac{\partial \varepsilon_{m,j}}{\partial v_{j,k}} \end{cases} \quad (7)$$

where η is the learning rate. By combining (6) and (7), we have training rules as follows:

On $f'_{m,j}$, for $k = 1 \sim d$

$$\begin{cases} u_{m,k} \leftarrow u_{m,k} + \eta v_{j,k} \left(f'_{m,j} - \sum_{k=1}^d u_{m,k} v_{j,k} \right) - \lambda \eta u_{m,k} \\ v_{j,k} \leftarrow v_{j,k} + \eta u_{m,k} \left(f'_{m,j} - \sum_{k=1}^d u_{m,k} v_{j,k} \right) - \lambda \eta v_{j,k} \end{cases} \quad (8)$$

With (8), all known entries in B are sequentially adopted to train U and V in one iteration. As the model converges, U and V are utilized to build $\hat{B}$, i.e.,

$$\hat{B} = UV^T. \quad (9)$$

Note that the theoretical convergence analysis of Phase I is given in the Appendix.

2) *Phase II*: Let $\hat{S}'_{n-1}$ be the completed streaming features set in which features are selected from $\{\hat{F}'_1, \hat{F}'_2, \dots, \hat{F}'_{n-1}\}$ till time point $n-1$. Then, we present some definitions on $\hat{F}'_n$ and $\hat{S}'_{n-1}$ at time point n before introducing Phase II.

Definition 6 (Irrelevant Feature [45]): $\hat{F}'_n$ is an irrelevant feature to C , if

$$\forall \zeta \subseteq \hat{S}'_{n-1} \text{ s.t. } P(C|\zeta, \hat{F}'_n) = P(C|\zeta). \quad (10)$$

Definition 7 (Markov Blanket [12]): At time point $n-1$, a Markov blanket of C is a subset of $\hat{S}'_{n-1}$, represented as $M_B(C)_{n-1}$, $M_B(C)_{n-1} \subseteq \hat{S}'_{n-1}$, satisfying

$$\begin{aligned} \forall \zeta \subseteq \hat{S}'_{n-1} - M_B(C)_{n-1} \\ \text{s.t. } P(C|M_B(C)_{n-1}, \zeta) = P(C|M_B(C)_{n-1}) \end{aligned} \quad (11)$$

where $\hat{S}'_{n-1} - M_B(C)_{n-1}$ denotes the subset of $\hat{S}'_{n-1}$ excluding $M_B(C)_{n-1}$.

Definition 8 (Redundant Completed Streaming Features): At time point n , if $\hat{F}'_n$ is not an irrelevant feature to C , the redundant completed streaming features to C satisfy

$$\begin{aligned} \forall R \in M_B(C)_{n-1} \cup \{\hat{F}'_n\}, \exists \zeta \subseteq M_B(C)_{n-1} \cup \{\hat{F}'_n\} - \{R\} \\ \text{s.t. } P(C|R, \zeta) = P(C|\zeta) \end{aligned} \quad (12)$$

where R indicates a redundant completed streaming feature to C and $M_B(C)_{n-1} \cup \{\hat{F}'_n\} - \{R\}$ indicates the subset of $M_B(C)_{n-1} \cup \{\hat{F}'_n\}$ excluding $\{R\}$.

Although existing OSFS approaches work differently, their processing flows are similar and can be concluded into two main parts [12], [20]–[23]: 1) online relevance analysis and 2) online redundancy analysis. Next, we explain how to conduct online feature selection on B (obtained from Phase I) in Phase II following Definitions 6–8 with a small example starting from time point n to $n+B_S-1$.

- 1) *Initialization*: Initializing a cache set $CS = \hat{S}'_{n-1}$, a label vector C , and a Markov blanket $M_B(C)_{n-1} \subseteq CS$ that satisfies $\forall \zeta \subseteq CS - M_B(C)_{n-1} \text{ s.t. } P(C|M_B(C)_{n-1}, \zeta) = P(C|M_B(C)_{n-1})$.
- 2) *Online Relevance Analysis*: $\forall i \in \{0, 1, \dots, B_S-1\}$, a feature $\hat{F}'_{n+i}$ in B is sequentially analyzed with C following Definition 6. If a feature is irrelevant to C , i.e., $\forall \zeta \subseteq CS \text{ s.t. } P(C|\zeta, \hat{F}'_{n+i}) = P(C|\zeta)$, it is discarded; otherwise, it is added to CS .
- 3) *Online Redundancy Analysis*: Each feature Q in CS is sequentially identified with C following Definition 8. If a feature is redundant to C , i.e., if $\forall R \subseteq M_B(C)_{n-1} \cup \{Q\}, \exists \zeta \subseteq M_B(C)_{n-1} \cup \{Q\} - \{R\} \text{ s.t. } P(C|\zeta R, \hat{F}'_{n+i}) = P(C|\zeta)$, it is removed from CS .

After the above three steps, the remaining features in CS are the best-selected and completed streaming features at the current time point $n+B_S-1$. Please note that Phase II of LOSSA focuses on the task of streaming feature selection on the completed streaming feature matrix B , which does not involve convergence issues as discussed in [12] and [18]–[25].

C. Algorithm Design and Time Complexity Analysis

Algorithm Design: Based on the above analyses, we design *Algorithm LOSSA* to handle the problem of OS²FS. Its pseudo code is given in Table II, where Phase I consists of steps 4–21 and Phase II consists of steps 22–37.

- 1) *Phase I*: Steps 4–9 adopt a buffer matrix B (whose column count B_S is far less than the total number of

TABLE II
ALGORITHM LOSSA

1	initialize $\hat{S}=\emptyset, B=\emptyset, B_S, n=1, C$
2	while the feature stream keeps active
3	receive a sparse streaming feature F'_n
4	/*cache sparse streaming features*/
5	if $B_S \neq \emptyset$
6	$B = B \cup F'_n$
7	$B_S = B_S - 1$
8	continue
9	end if
10	/*conduct LFA on B to predict its missing data*/
11	initializing $d, \lambda, \eta, U, V, T_{\max}$ =maximum number of iteration, $t=1$
12	while $t \leq T_{\max}$ && not converge
13	for $j = n$ to $n+B_S-1$
14	for each $f'_{mj} \in K_j$
15	update $u_{m,k}$ according to (8)
16	update $v_{j,k}$ according to (8)
17	end for
18	end for
19	$t=t+1$
20	end while
21	$\hat{B} = UV^T$
22	/*analyze relationship between $\hat{F}'_{n+i}$ and C with Definition 6*/
23	for $i=0$ to B_S-1
24	fetch $\hat{F}'_{n+i}$ from $\hat{B}$
25	if $\text{Dep}(C, \hat{F}'_{n+i}, \emptyset)$
26	$\hat{S}' = \hat{S}' \cup \hat{F}'_{n+i}$
27	else continue
28	end if
29	/*identify redundant features in $\hat{S}'$ with Definitions 7 and 8*/
30	for each feature $Q \in \hat{S}'$
31	if $\exists \zeta \subseteq \hat{S}' - Q$ s.t. $\text{Ind}(C, Q \zeta)$
32	$\hat{S}' = \hat{S}' - Q$
33	end if
34	end for
35	initialize $B=\emptyset, B_S$
36	$n=n+1$
37	output selected completed streaming features set $\hat{S}'$
38	end while

sparse streaming features) to cache the received sparse streaming features. After B is built, LOSSA conducts an LFA process to predict its missing data of B as given in Steps 10–21. Thus, B is achieved and LOSSA goes to Phase II.

- 2) *Phase II*: Steps 22–28 are online relevance analysis, where function $\text{Dep}(C, \hat{F}'_{n+i}|\emptyset)$ denotes the conditional dependence test between $\hat{F}'_{n+i}$ and C given an empty set. Steps 29–34 are online redundant analysis, where function $\text{Ind}(C, Q|\zeta)$ denotes the conditional independence test between Q and C given a subset. The conditional dependence/independence tests can be implemented by adopting G^2 test [12], [69].
- 3) *Output*: After Phases I and II, $\hat{S}'$ is output as the optimal feature set at current time point n .
- 4) *Repeat*: When the feature stream keeps active, the above process is repeated to update the optimal feature set.

Time Complexity Analysis: In comparison with current OSFS approaches, such as Fast-OSFS [12], SAOLA [20], and OSFASW [21], *Algorithm LOSSA* costs extra time to estimate the missing data in sparse streaming features. As analyzed in [30]–[32] and [43], LFA's time complexity is mainly decided by the target sparse matrix's known entry count and latent

TABLE III
DETAILS OF SELECTED DATASETS

Mark	Dataset	#(Features)	#(Instances)	#(Class)
D1	Colon	2000	62	2
D2	Lung	3312	203	4
D3	Lungcancer	12533	181	2
D4	Lymphoma	4026	62	3
D5	Mart1	1024	500	2
D6	Prostate	6033	102	2
D7	Reged1	999	500	2
D8	SMK-CAN-187	19993	187	2
D9	COIL	241	1500	6
D10	Drive Face	6400	606	3
D11	Human-activity-recognition	561	10299	6
D12	HAPT	561	10929	12

factor dimension. If the arrived sparse streaming features have the size of M instances and N features, and their total missing data rate is α , then, LOSSA's time cost of Phase I is $\hat{F}'_{n+i}O(M \times N \times (1-\alpha) \times d)$. Note that such extra computational cost is acceptable in practice.

- 1) It is inversely proportional to α . In real applications, sparse streaming features have massive missing data, which enables the low computational cost of Phase I.
- 2) It can be significantly reduced through parallelization. According to [46] and [47], LFA can be parallelized with an alternating SGD algorithm. On this basis, we can also parallelize (8) for Phase I with it.

IV. EXPERIMENTS AND RESULTS

A. General Settings

Datasets: We select twelve benchmark datasets from UCI repositories [66], *NIPS* challenge [48], and studies in [49] and [50]. They all are collected from real applications as summarized in Table III.

Baselines: As analyzed in Section III, LOSSA is compatible with existing OSFS approaches. To validate its performance, we choose three state-of-the-art and representative OSFS algorithms to carry out the experiments. They are Fast-OSFS [12], SAOLA [20], and OSFASW [21]. Besides, random forest, support vector machine (SVM), and k -nearest neighbors (KNNs) are chosen as the base classifier to check the quality of selected features [51], [64], [65], [67]. All the parameters adopted by these classifiers and algorithms are summarized in Table IV. We, respectively, modify them to handle OS² FS by integrating them into LOSSA. For the sake of brevity, Fast-OSFS, SAOLA, and OSFASW and their modified versions are remarked as M1, M1 + LOSSA (M1's modified version), M2, M2 + LOSSA (M2's modified version), M3, and M3 + LOSSA (M3's modified version), respectively.

Experimental Designs: State-of-the-art OSFS algorithms and their modified versions boosted by LOSSA are carefully compared to validate the effectiveness of LOSSA. Meanwhile, LOSSA's sensitivity to hyperparameters, including missing rate α , regularization parameter λ , and size of buffer matrix B_S , are also carefully tested (default settings of hyperparameters

TABLE IV
ALL THE PARAMETERS USED IN THE EXPERIMENTS

Mark	Algorithm	Parameter
M1	Fast-OSFS	Z test, α is 0.05.
M2	SAOLA	Z test, α is 0.05.
M3	OSFASW	Z test, α is 0.05, sliding-window size is 1.
/	KNN	The neighbors are 3.
/	SVM	LIBSVM [52]: Default parameters values.
/	Random Forest	6 decision trees.

are $B_S = 10$, $\lambda = 0.01$, $\alpha = 0.1$, and $d = 10$). Its convergence ability in Phase I and computational efficiency are also revealed. A fivefold cross-validation strategy is adopted to evaluate the classification accuracy on each dataset. Besides, we repeat the whole experiment five times and report the average results with standard deviations. Note that the testing data have no missing data, while the training data have different situations of missing data that are explained in the corresponding sections. We run the experiments on a personal computer (Intel i7 CPU 3.4 GHz, RAM 16 GB).

B. Convergence Analysis of LOSSA in Phase I

To analyze the convergence of LOSSA in Phase I, i.e., preprocessing sparse streaming feature matrix B to build completed streaming feature matrix B , we test LFA's prediction accuracy for the missing data of B with different missing data rates α for validating its convergence ability. The root mean squared error (RMSE) [30]–[32] is adopted as the accuracy metric. Fig. 3 presents the training curve of LFA on all datasets. From it, we clearly see that LFA is convergent in predicting the missing data of B . It fast achieves a promising prediction accuracy on all the datasets. As α increases, the prediction accuracy decreases. For example, the lowest RMSEs are 0.0134, 0.0139, 0.0256, 0.0753, and 0.3311 when α is set as 0.1, 0.3, 0.5, 0.7, and 0.9, respectively. The reason is that as α increases, training data decrease, thus generating the LFA model with less information. However, to be shown next, even with less information, LOSSA can help an algorithm select high-quality features from sparse streaming feature data.

C. Selected Features Analysis

Situations of Selected Features: For LOSSA, the hyperparameters are fixed as $\alpha = 0.1$, $B_S = 10$, and $\lambda = 0.01$. The situations of selected features by original algorithms and their modified versions are recorded on Table V. We test original algorithms on streaming features that have no missing data, while testing their modified versions on sparse streaming features that have 10% missing data. Then, we have the following findings.

- 1) The number of selected features by original algorithms varies on the same dataset. This phenomenon indicates that original algorithms have different characteristics in feature selection.

- 2) With LOSSA, modified algorithms are likely to select the same features as their original versions on most datasets.

Hence, the above findings validate that LOSSA is compatible with state-of-the-art OSFS algorithms. Moreover, it efficiently improves them to handle OS²FS as well as keep the quality of selected features even with incomplete inputs.

Classification Accuracy of Selected Features: This paragraph analyzes the above-selected features' quality by employing them to train a classifier to evaluate its classification accuracy. In detail, we, respectively, adopt the features selected by the original algorithm and its modified version to train two different classifiers on each dataset. Then, we compare the accuracy of the two classifiers. Table VI records the comparison results. We highlight these results where modified algorithms have higher accuracy than their original versions. From Table VI, we have the following findings.

- 1) Modified algorithms entirely achieve lower accuracy than their corresponding original ones on D2. One possible reason is that modified versions have selected different and fewer features, as recorded in Table V.
- 2) With KNN and SVM as the base classifier, modified algorithms obtain higher accuracy than their corresponding original ones on most datasets.
- 3) With Random Forest as the base classifier, modified algorithms reach very close accuracy to their original ones.

D. Impacts of Missing Data Rate α

In the experiments, we increase the missing data rate α of sparse streaming features from 0.1 to 0.9, where $B_S = 10$.

Detailed Accuracy on D1: The detailed accuracies of different algorithms with different classifiers on D1 are recorded in Table VII, where we see that accuracy does not decrease as α increases. Especially, there are many cases that modified algorithms have better accuracy than their corresponding original ones (the highlighted cases). In general, when $\alpha = 0.2, 0.4, 0.5, 0.6, 0.7$, and 0.9 , the average accuracy on sparse streaming features is higher than that on streaming features. Moreover, to check whether there are significant differences between original algorithms and their corresponding modified ones, we conduct the pairwise comparison on the accuracies recorded in Tables VII with Wilcoxon signed-ranks test [53]. In detail, we compare accuracies on sparse streaming features one by one (from $\alpha = 0.1$ to $\alpha = 0.9$) with that on streaming features. The statistical results are presented in Table VIII, where $R+$ and $R-$, respectively, denote the achieved ranks by modified algorithms and their original version, and p -value denotes the probability that original algorithms perform better than their modified versions. From Table VIII, we observe that modified algorithms have a larger rank value when $\alpha = 0.2, 0.3, 0.4, 0.5, 0.6, 0.7$, and 0.9 . Only when $\alpha = 0.1$ and 0.8 , modified algorithms fail to have a larger rank value. However, no one p -value is smaller than 0.1 or larger than 0.9, which means that there are no significant differences between original algorithms and their modified ones.

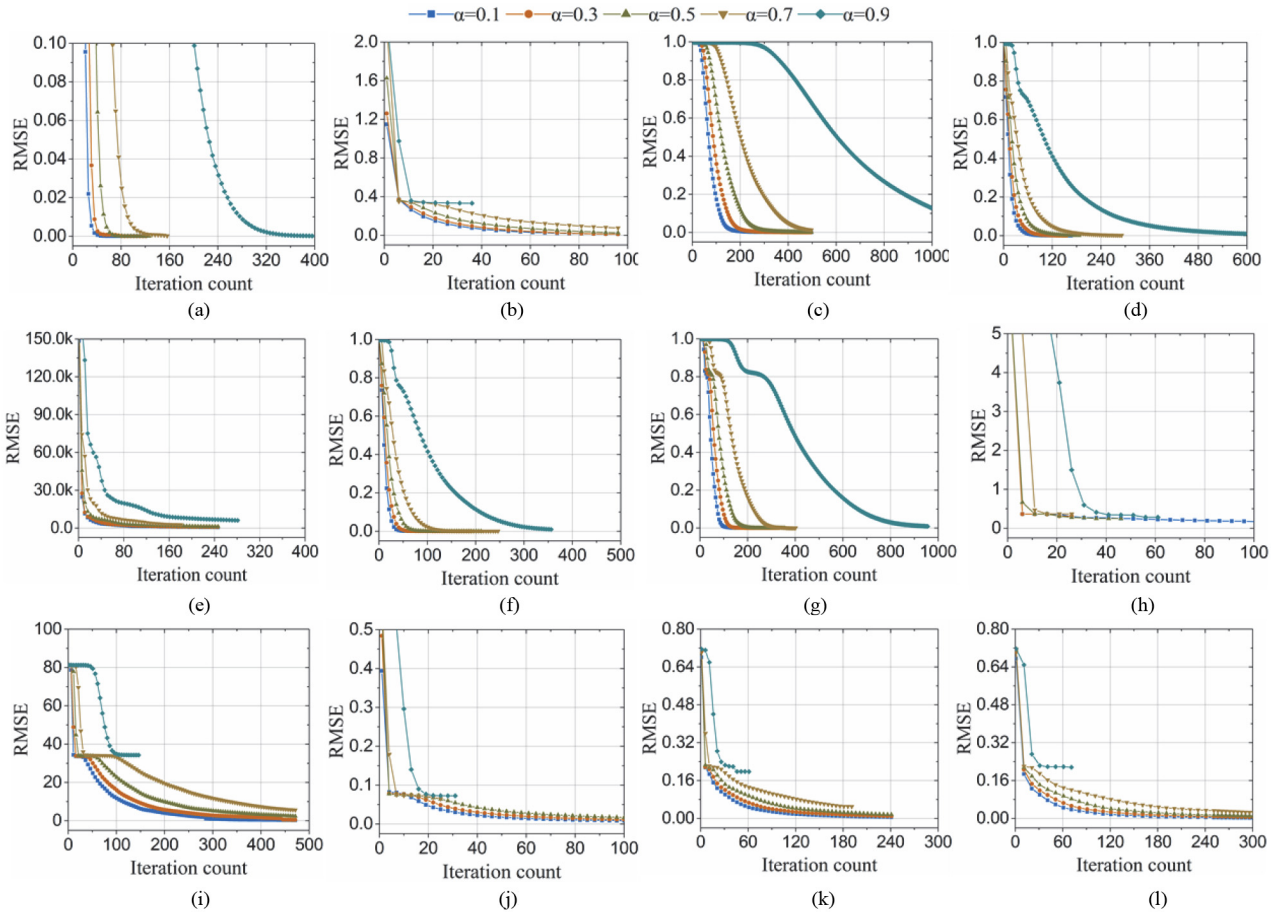


Fig. 3. Training process of LFA for predicting the missing data of B with different missing data rates α on all the datasets. (a) D1. (b) D2. (c) D3. (d) D4. (e) D5. (f) D6. (g) D7. (h) D8. (i) D9. (j) D10. (k) D11. (l) D12.

TABLE V
SITUATIONS OF SELECTED FEATURES BY ORIGINAL ALGORITHMS AND THEIR MODIFIED VERSIONS, $\alpha = 0.1$

Dataset	M1 vs. M1+LOSSA			M2 vs. M2+LOSSA			M3 vs. M3+LOSSA		
	M1 (without missing data)	M1+LOSSA (with 10% missing data)	#Same features	M2 (without missing data)	M2+LOSSA (with 10% missing data)	#Same features	M3 (without missing data)	M3+LOSSA (with 10% missing data)	#Same features
D1	3	3	3	4	4	4	2	2	2
D2	16	6	1	30	9	1	11	5	1
D3	8	9	8	39	40	37	4	4	4
D4	6	7	2	39	40	26	5	4	3
D5	1	1	1	1	1	1	1	1	1
D6	5	5	5	15	15	15	4	3	3
D7	13	13	13	16	16	16	13	13	13
D8	10	4	0	11	6	0	4	3	0
D9	6	6	6	3	3	3	6	6	6
D10	8	8	7	5	5	3	6	4	1
D11	17	16	13	12	9	8	12	9	5
D12	22	23	17	8	8	8	8	15	3

Average Accuracy on All the Datasets: The average accuracy of different algorithms with different classifiers on a specific dataset is recorded in Table IX. For example, the last row of Table VII denotes the average accuracy of different algorithms with different classifiers on D1, which is the same as the third row of Table IX. From it, we see the following.

- 1) In general, accuracy does not evidently decrease or even has a slight improvement when α is smaller than 0.6.
- 2) There are many cases that a modified algorithm with LOSSA has higher accuracy than its corresponding

original version, such as on D1 with $\alpha = 0.2, 0.4, 0.5, 0.6, 0.7$, and 0.9 , on D3 with $\alpha = 0.1$ and 0.2 , on D4 with $\alpha = 0.5$, on D5 with $\alpha = 0.1, 0.4, 0.5$, etc.

- 3) There are opposite results on D2 and D12. A modified algorithm achieves better performance than its original version on any case of D12, while results are opposite on D2. The reasons can be found from Tables V and VI ($\alpha = 0.1$), where we see that with the incorporation of LOSSA, the modified algorithms (including

TABLE VI
USING THE SELECTED FEATURES (RECORDED IN TABLE V) TO TRAIN A CLASSIFIER FIRST AND THEN TESTING ITS ACCURACY (%), $\alpha = 0.1$

Models/Datasets	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	Average
M1	80.95 $\pm$ 2.63	88.77 $\pm$ 0.73	98.06 $\pm$ 0.54	91.59 $\pm$ 0.68	52.16 $\pm$ 1.24	86.70 $\pm$ 1.40	87.63 $\pm$ 0.26	76.81 $\pm$ 0.80	50.77 $\pm$ 1.68	90.20 $\pm$ 0.09	75.81 $\pm$ 0.17	81.48 $\pm$ 0.07	80.08 $\pm$ 1.69
M1+LOSSA	80.05 $\pm$ 3.03	84.74 $\pm$ 0.85	98.07 $\pm$ 0.39	90.69 $\pm$ 0.69	58.16 $\pm$ 18.03	87.63 $\pm$ 1.75	99.00 $\pm$ 0.14	76.46 $\pm$ 3.14	51.36 $\pm$ 1.51	90.33 $\pm$ 0.07	75.21 $\pm$ 0.1	81.06 $\pm$ 0.07	81.06 $\pm$ 2.48
M2	86.21 $\pm$ 1.95	90.50 $\pm$ 1.50	97.68 $\pm$ 0.23	87.05 $\pm$ 0.27	45.92 $\pm$ 10.68	91.39 $\pm$ 1.99	91.59 $\pm$ 0.17	73.44 $\pm$ 1.65	26.47 $\pm$ 2.87	92.34 $\pm$ 0.22	75.14 $\pm$ 0.05	70.69 $\pm$ 0.04	77.37 $\pm$ 1.8
M2+LOSSA	86.51 $\pm$ 1.87	84.18 $\pm$ 0.76	98.23 $\pm$ 0.23	85.79 $\pm$ 0.64	61.68 $\pm$ 15.84	91.59 $\pm$ 0.42	98.60 $\pm$ 0.00	76.36 $\pm$ 2.07	26.69 $\pm$ 1.38	92.14 $\pm$ 1.04	75.26 $\pm$ 0.08	67.59 $\pm$ 0.06	78.72 $\pm$ 2.03
M3	78.26 $\pm$ 0.82	89.80 $\pm$ 1.03	98.28 $\pm$ 0.17	92.18 $\pm$ 0.73	59.72 $\pm$ 12.07	88.45 $\pm$ 1.26	91.95 $\pm$ 0.23	76.41 $\pm$ 1.03	42.43 $\pm$ 2.66	90.99 $\pm$ 0.45	81.54 $\pm$ 0.10	50.84 $\pm$ 0.08	78.4 $\pm$ 1.72
M3+LOSSA	79.36 $\pm$ 2.36	80.55 $\pm$ 0.68	97.85 $\pm$ 0.66	93.23 $\pm$ 0.81	56.64 $\pm$ 15.75	91.95 $\pm$ 3.34	98.92 $\pm$ 0.23	76.23 $\pm$ 1.59	43.13 $\pm$ 1.58	92.41 $\pm$ 1.13	75.76 $\pm$ 0.07	79.92 $\pm$ 0.04	80.50 $\pm$ 2.35
M1	82.13 $\pm$ 2.59	88.23 $\pm$ 0.83	97.85 $\pm$ 0.18	99.67 $\pm$ 0.75	84.28 $\pm$ 0.58	95.71 $\pm$ 0.52	95.29 $\pm$ 0.26	75.30 $\pm$ 1.92	80.64 $\pm$ 0.20	94.75 $\pm$ 0.30	80.16 $\pm$ 0.24	82.52 $\pm$ 0.06	88.04 $\pm$ 0.70
M1+LOSSA	85.90 $\pm$ 2.19	85.13 $\pm$ 0.56	98.07 $\pm$ 0.40	100.0 $\pm$ 0.00	85.44 $\pm$ 0.55	95.29 $\pm$ 0.84	99.20 $\pm$ 0.14	75.51 $\pm$ 2.18	81.16 $\pm$ 0.47	95.51 $\pm$ 0.61	78.15 $\pm$ 0.13	81.96 $\pm$ 0.15	88.44 $\pm$ 0.69
M2	83.46 $\pm$ 1.46	87.33 $\pm$ 0.42	98.56 $\pm$ 0.29	99.33 $\pm$ 0.91	84.80 $\pm$ 0.62	93.14 $\pm$ 1.18	94.15 $\pm$ 0.26	70.39 $\pm$ 3.12	53.69 $\pm$ 0.40	94.49 $\pm$ 0.58	75.46 $\pm$ 0.32	69.59 $\pm$ 0.33	83.70 $\pm$ 0.82
M2+LOSSA	86.21 $\pm$ 3.35	83.90 $\pm$ 1.34	98.79 $\pm$ 0.23	98.41 $\pm$ 1.13	83.80 $\pm$ 0.58	94.15 $\pm$ 1.21	98.92 $\pm$ 0.11	71.58 $\pm$ 2.27	53.16 $\pm$ 0.91	94.22 $\pm$ 0.84	74.85 $\pm$ 0.06	67.59 $\pm$ 0.19	83.80 $\pm$ 1.02
M3	81.56 $\pm$ 1.77	88.11 $\pm$ 0.89	98.78 $\pm$ 0.23	100.0 $\pm$ 0.00	84.48 $\pm$ 0.67	94.50 $\pm$ 0.56	94.30 $\pm$ 0.22	72.31 $\pm$ 1.51	80.53 $\pm$ 0.43	94.49 $\pm$ 0.71	83.94 $\pm$ 0.27	52.13 $\pm$ 0.23	85.43 $\pm$ 0.63
M3+LOSSA	78.38 $\pm$ 2.70	77.73 $\pm$ 0.90	98.34 $\pm$ 0.02	97.67 $\pm$ 0.91	85.12 $\pm$ 0.99	94.30 $\pm$ 0.44	99.34 $\pm$ 0.28	70.03 $\pm$ 1.13	80.56 $\pm$ 0.54	94.62 $\pm$ 0.83	78.11 $\pm$ 0.17	79.01 $\pm$ 0.30	86.10 $\pm$ 0.77
M1	81.54 $\pm$ 1.45	87.30 $\pm$ 1.44	97.62 $\pm$ 0.74	95.21 $\pm$ 1.51	76.28 $\pm$ 1.03	92.75 $\pm$ 1.49	91.75 $\pm$ 0.50	71.47 $\pm$ 0.94	79.27 $\pm$ 0.54	93.86 $\pm$ 0.27	84.97 $\pm$ 0.37	88.15 $\pm$ 0.27	86.68 $\pm$ 0.88
M1+LOSSA	77.79 $\pm$ 4.73	83.74 $\pm$ 1.43	97.41 $\pm$ 0.78	91.46 $\pm$ 1.94	77.32 $\pm$ 0.67	91.75 $\pm$ 1.31	97.76 $\pm$ 0.61	72.81 $\pm$ 3.42	79.57 $\pm$ 1.12	93.86 $\pm$ 0.66	84.17 $\pm$ 0.32	88.51 $\pm$ 0.30	86.35 $\pm$ 1.44
M2	86.46 $\pm$ 2.45	86.42 $\pm$ 1.11	97.40 $\pm$ 0.91	92.38 $\pm$ 2.74	77.00 $\pm$ 1.18	91.99 $\pm$ 1.30	91.72 $\pm$ 0.62	71.01 $\pm$ 1.12	53.89 $\pm$ 1.02	93.93 $\pm$ 0.79	79.40 $\pm$ 0.23	77.88 $\pm$ 0.18	83.29 $\pm$ 1.14
M2+LOSSA	78.56 $\pm$ 1.97	83.35 $\pm$ 0.48	97.74 $\pm$ 0.70	94.79 $\pm$ 3.08	77.20 $\pm$ 0.14	91.72 $\pm$ 1.75	98.08 $\pm$ 0.30	70.07 $\pm$ 2.35	54.24 $\pm$ 1.04	94.39 $\pm$ 0.30	80.85 $\pm$ 0.24	75.53 $\pm$ 0.20	83.04 $\pm$ 1.04
M3	79.05 $\pm$ 2.05	85.80 $\pm$ 1.96	96.79 $\pm$ 0.63	95.51 $\pm$ 1.31	77.92 $\pm$ 1.09	92.34 $\pm$ 2.99	92.10 $\pm$ 0.26	72.19 $\pm$ 1.65	77.68 $\pm$ 0.17	94.02 $\pm$ 0.61	88.33 $\pm$ 0.21	58.89 $\pm$ 0.24	84.22 $\pm$ 1.1
M3+LOSSA	78.87 $\pm$ 2.43	78.25 $\pm$ 1.40	97.68 $\pm$ 0.63	96.44 $\pm$ 3.45	76.60 $\pm$ 1.50	92.10 $\pm$ 2.46	98.84 $\pm$ 0.30	71.45 $\pm$ 1.23	77.25 $\pm$ 0.60	94.09 $\pm$ 0.56	81.04 $\pm$ 0.29	87.33 $\pm$ 0.24	85.83 $\pm$ 1.26

TABLE VII
CLASSIFICATION ACCURACY (%) WHEN INCREASING α FROM 0.1 TO 0.9 ON D1

Classifier	Algorithm	*Streaming features	† Sparse streaming features with different α								
			0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
SVM	M1	80.95 $\pm$ 2.63	80.05 $\pm$ 3.03	86.85 $\pm$ 2.05	83.15 $\pm$ 4.38	82.62 $\pm$ 1.77	80.97 $\pm$ 3.24	85.54 $\pm$ 1.14	87.26 $\pm$ 3.00	82.21 $\pm$ 2.29	81.62 $\pm$ 0.54
	M2	86.21 $\pm$ 1.95	86.51 $\pm$ 1.87	85.41 $\pm$ 1.97	86.23 $\pm$ 0.81	87.15 $\pm$ 1.71	85.77 $\pm$ 0.86	85.87 $\pm$ 0.73	86.69 $\pm$ 0.74	85.31 $\pm$ 1.60	86.51 $\pm$ 2.86
	M3	78.26 $\pm$ 0.82	79.36 $\pm$ 2.36	78.08 $\pm$ 3.90	78.46 $\pm$ 0.88	79.28 $\pm$ 4.13	77.72 $\pm$ 0.85	78.00 $\pm$ 3.81	77.08 $\pm$ 4.02	79.03 $\pm$ 4.87	79.69 $\pm$ 4.93
KNN	M1	82.13 $\pm$ 2.59	85.90 $\pm$ 2.19	84.56 $\pm$ 2.22	83.72 $\pm$ 3.00	84.18 $\pm$ 5.14	83.87 $\pm$ 0.92	84.13 $\pm$ 3.03	87.51 $\pm$ 2.13	80.23 $\pm$ 1.27	82.67 $\pm$ 3.18
	M2	83.46 $\pm$ 1.46	86.21 $\pm$ 3.35	87.77 $\pm$ 2.10	85.44 $\pm$ 2.93	87.10 $\pm$ 2.78	86.33 $\pm$ 1.88	87.72 $\pm$ 1.76	86.28 $\pm$ 2.68	82.38 $\pm$ 0.90	87.10 $\pm$ 3.42
	M3	81.56 $\pm$ 1.77	78.38 $\pm$ 2.70	80.03 $\pm$ 2.62	80.00 $\pm$ 1.98	79.38 $\pm$ 2.42	83.51 $\pm$ 2.70	82.26 $\pm$ 2.37	78.15 $\pm$ 1.38	81.90 $\pm$ 2.46	79.54 $\pm$ 2.13
Random Forest	M1	81.54 $\pm$ 1.45	77.79 $\pm$ 4.73	82.67 $\pm$ 2.12	80.00 $\pm$ 2.04	80.23 $\pm$ 2.92	80.82 $\pm$ 3.46	78.90 $\pm$ 3.12	79.72 $\pm$ 5.28	81.62 $\pm$ 3.54	80.59 $\pm$ 4.25
	M2	86.46 $\pm$ 2.45	78.56 $\pm$ 1.97	82.95 $\pm$ 3.96	79.36 $\pm$ 4.45	80.28 $\pm$ 4.23	81.67 $\pm$ 3.93	83.82 $\pm$ 4.27	80.05 $\pm$ 3.33	84.13 $\pm$ 3.88	86.64 $\pm$ 4.43
	M3	79.05 $\pm$ 2.05	78.87 $\pm$ 2.43	78.08 $\pm$ 1.94	77.97 $\pm$ 1.71	79.59 $\pm$ 3.10	80.69 $\pm$ 4.08	80.95 $\pm$ 3.02	80.90 $\pm$ 2.27	80.90 $\pm$ 1.76	75.33 $\pm$ 1.14
Average		82.18 $\pm$ 1.91	81.29 $\pm$ 2.74	82.93 $\pm$ 2.54	81.59 $\pm$ 2.47	82.20 $\pm$ 3.13	82.37 $\pm$ 2.44	83.02 $\pm$ 2.58	82.63 $\pm$ 2.76	81.97 $\pm$ 2.51	82.19 $\pm$ 2.99

* The streaming features (without missing data) selected by original algorithms.

† The sparse streaming features (with missing data) selected by modified algorithms.

TABLE VIII
STATISTICAL RESULTS OF THE WILCOXON SIGNED-RANKS TEST ON THE ACCURACIES RECORDED IN TABLES VII (ON D1)

Modified algorithm vs. Original algorithm	*R+	*R-	† p-value
0.1	19	26	0.6738
0.2	27	18	0.3262
0.3	24	21	0.4551
0.4	25	20	0.4102
0.5	27	18	0.3262
0.6	29	16	0.2383
0.7	25	20	0.4102
0.8	19	26	0.6738
0.9	24	21	0.4551

* A larger value denotes a higher accuracy.

† No significant difference as $p\text{-value} \in [0.1, 0.9]$ at 0.1 significance level.

M1 + LOSSA, M2 + LOSSA, and M3 + LOSSA) select much fewer features than their original versions (i.e., M1, M2, and M3) on D2 as shown in Table V,

thus some useful features are dropped. Then, the modified algorithms obtain a lower accuracy. On the contrary, on D12, the modified algorithm M3 + LOSSA selects much more features than its original version M3, making M3 + LOSSA achieve much higher accuracy than M3 as shown in Table VI.

Besides, we also conduct a pairwise comparison with the Wilcoxon signed-ranks test for Table IX. The statistical results are recorded in Table X, where we see that there are no significant differences between original algorithms and their modified versions when $\alpha < 0.6$.

E. Impacts of Regularization Parameter λ

This set of experiments focus on λ 's effects in the feature selection results on all the datasets. The hyperparameters are set as $\alpha = 0.1$, $B_S = 10$, and λ increases from 0 to 1. Table XI records the average accuracy of different algorithms

TABLE IX
INCREASING α FROM 0.1 TO 0.9 TO TEST THE AVERAGE CLASSIFICATION ACCURACY (%) ON ALL THE DATASETS

Datasets	*Streaming features	[†] Sparse streaming features with different α								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
D1	82.18 $\pm$ 1.91	81.29 $\pm$ 2.74	82.93 $\pm$ 2.54	81.59 $\pm$ 2.47	82.20 $\pm$ 3.13	82.37 $\pm$ 2.44	83.02 $\pm$ 2.58	82.63 $\pm$ 2.76	81.97 $\pm$ 2.51	82.19 $\pm$ 2.99
D2	88.16 $\pm$ 1.10	82.40 $\pm$ 0.93	83.57 $\pm$ 1.13	82.81 $\pm$ 1.25	81.91 $\pm$ 1.30	82.02 $\pm$ 1.66	83.70 $\pm$ 1.47	81.68 $\pm$ 1.08	84.85 $\pm$ 1.41	83.49 $\pm$ 1.02
D3	97.89 $\pm$ 0.43	98.02 $\pm$ 0.45	98.19 $\pm$ 0.40	97.73 $\pm$ 0.41	97.76 $\pm$ 0.34	97.78 $\pm$ 0.32	95.31 $\pm$ 0.40	92.38 $\pm$ 0.39	92.24 $\pm$ 0.46	91.86 $\pm$ 0.47
D4	94.77 $\pm$ 0.99	94.28 $\pm$ 1.41	94.22 $\pm$ 1.23	94.50 $\pm$ 1.71	94.50 $\pm$ 1.49	95.10 $\pm$ 1.17	93.28 $\pm$ 1.32	92.37 $\pm$ 1.13	93.12 $\pm$ 1.49	89.74 $\pm$ 1.20
D5	71.40 $\pm$ 4.35	73.55 $\pm$ 6.01	70.19 $\pm$ 5.07	70.05 $\pm$ 4.23	71.85 $\pm$ 4.78	71.77 $\pm$ 4.94	70.92 $\pm$ 5.51	70.88 $\pm$ 3.81	70.00 $\pm$ 4.92	70.35 $\pm$ 6.52
D6	91.89 $\pm$ 1.41	92.28 $\pm$ 1.50	92.28 $\pm$ 1.24	92.27 $\pm$ 1.19	91.51 $\pm$ 1.18	91.55 $\pm$ 0.95	92.33 $\pm$ 1.37	91.22 $\pm$ 1.49	91.97 $\pm$ 1.31	91.05 $\pm$ 1.26
D7	98.73 $\pm$ 0.31	98.74 $\pm$ 0.23	98.81 $\pm$ 0.28	98.66 $\pm$ 0.35	98.72 $\pm$ 0.29	98.36 $\pm$ 0.32	98.00 $\pm$ 0.31	88.88 $\pm$ 0.55	85.02 $\pm$ 0.54	85.72 $\pm$ 0.61
D8	73.26 $\pm$ 1.53	73.39 $\pm$ 2.15	72.62 $\pm$ 1.84	74.24 $\pm$ 1.76	72.56 $\pm$ 1.69	72.11 $\pm$ 2.01	73.33 $\pm$ 1.87	68.60 $\pm$ 2.01	70.27 $\pm$ 2.03	71.43 $\pm$ 1.42
D9	60.60 $\pm$ 1.11	60.79 $\pm$ 1.02	60.41 $\pm$ 0.90	60.79 $\pm$ 0.76	60.77 $\pm$ 1.02	60.88 $\pm$ 0.92	60.74 $\pm$ 1.04	60.30 $\pm$ 1.01	61.04 $\pm$ 1.03	61.29 $\pm$ 0.97
D10	93.23 $\pm$ 0.45	93.51 $\pm$ 0.67	93.10 $\pm$ 0.60	93.14 $\pm$ 0.44	92.84 $\pm$ 0.56	92.88 $\pm$ 0.35	92.97 $\pm$ 0.37	92.92 $\pm$ 0.30	92.76 $\pm$ 0.45	92.71 $\pm$ 0.26
D11	80.53 $\pm$ 0.22	78.16 $\pm$ 0.16	79.51 $\pm$ 0.20	82.22 $\pm$ 0.21	79.92 $\pm$ 0.20	81.61 $\pm$ 0.16	79.86 $\pm$ 0.22	81.68 $\pm$ 0.16	77.99 $\pm$ 0.18	73.62 $\pm$ 0.16
D12	70.24 $\pm$ 0.17	78.72 $\pm$ 0.19	78.14 $\pm$ 0.16	76.40 $\pm$ 0.24	76.77 $\pm$ 0.21	77.21 $\pm$ 0.18	77.33 $\pm$ 0.21	74.65 $\pm$ 0.16	73.25 $\pm$ 0.22	76.68 $\pm$ 0.17
Average	83.57 $\pm$ 1.16	83.76 $\pm$ 1.46	83.66 $\pm$ 1.30	83.70 $\pm$ 1.25	83.44 $\pm$ 1.35	83.64 $\pm$ 1.28	83.40 $\pm$ 1.39	81.52 $\pm$ 1.24	81.21 $\pm$ 1.38	80.84 $\pm$ 1.42

* The streaming features (without missing data) are selected by the original algorithms.

[†] The sparse streaming features (with missing data) are selected by the modified algorithms.

TABLE X
STATISTICAL RESULTS OF THE WILCOXON SIGNED-RANKS TEST ON THE ACCURACIES RECORDED IN TABLE IX (ON ALL DATASETS)

Modified algorithm vs. Original algorithm	*R+	*R-	[†] p -value
0.1	42	36	0.4175
0.2	30	48	0.7651
0.3	40	38	0.4849
0.4	26	52	0.8494
0.5	37.5	40.5	0.5535
0.6	27	51	0.8303
0.7	17	61	0.9614
0.8	13	65	0.9829
0.9	14	64	0.9788

* A larger value denotes a higher accuracy.

[†] There is no significant difference when p -value $\in$ [0.1, 0.9] at the 0.1 significance level.

with different classifiers on a specific dataset with such settings. From Table XI, we find that λ has significant impacts on the feature selection results. First, we see that considering the models with $\lambda = 0$ and $\lambda \neq 0$, (i.e., situations without/with the regularization effects), the latter achieves better results on most cases, which indicates that regularization is vital for improving an LOSSA-incorporated algorithm's performance of feature selection because it can alleviate overfitting in Phase I. Moreover, as λ increases over its optimal value, the performance decreases evidently. These results demonstrate that the regularization of Phase I has significant impacts on LOSSA's performance.

Note that although the results of Table XI show that the optimal value of λ is data-dependent, a relatively small value of λ (smaller than 0.03) enables relatively high accuracy. Hence, we fix λ at 0.01 in the other experiments to practically evaluate the proposed LOSSA algorithm. However, it is highly desired and useful to make λ adaptive according to the characteristics of data, which is in our future research plan.

F. Impacts of Column Number of Buffer Matrix B_S

This set of experiments researches how the column count of buffer matrix B_S impacts the feature selection on all the

datasets. We fix α at 0.1 and increase B_S from 2 to 20. Table XII, respectively, records the average accuracy of different algorithms with different classifiers on a specific dataset. From Table XII, we have three findings. First, we find that B_S has different impacts on different datasets. The optimal value of B_S varies across all the datasets. For example, the optimal value of B_S is 14 for D1, 2 for D2, 18 for D3, 8 for D4, 10 for D5, 12 for D6, 6 for D7, 12 for D8, 14 for D9, 14 for D10, 4 for D11, and 6 for D12. The last row of Table XII is the statistic of the optimal value of B_S , which shows that the distribution of the optimal value of B_S is irregular on the different datasets. Second, we find that with the optimal value of B_S , LOSSA makes modified algorithms achieve higher classification accuracy than their original versions on each dataset. In particular, Table VI shows that modified algorithms do not perform well on D2. This problem, however, has been addressed when $B_S = 2$. Third, we find that larger B_S does not lead to better accuracy. One reason is that although more data can be used to conduct LFA with larger B_S , these data may contain some irrelevant and redundant features, which impairs the quality of estimations to missing data. Hence, B_S should also be chosen with care for ensuring the quality of feature selection from a sparse feature stream.

G. Computational Efficiency

As analyzed in Section III-C, LOSSA needs extra computation to modify original algorithms to solve. Hence, this section tests the computational efficiency of LOSSA. According to [46] and [47], LFA can be implemented in parallel. In our experiments, we use 16 CPU cores to implement Phase I for LOSSA. On each dataset, we measure the CPU running time that LOSSA costs in Phases I and II, respectively. Note that the CPU running cost in Phase I denotes the extra computation caused by LOSSA and that in Phase II denotes original algorithms' computational efficiency. α and B_S are set as 0.1 and 10, respectively. Fig. 4 records the results. From it, we observe the following.

- 1) LOSSA spends less CPU running time in Phase I than M3 costs in Phase II.

TABLE XI
AVERAGE CLASSIFICATION ACCURACY AS λ INCREASES FROM 0 TO 1 ($\alpha = 0.1, B_S = 10$)

Datasets	*Streaming features	† Sparse streaming features with different λ											
		0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1	1
D1	82.18±1.91	80.81±3.08	83.04±2.44	83.36±2.58	86.13±2.79	82.48±2.36	83.22±2.02	85.72 ±2.22	84.93±3.23	82.97±2.89	82.34±2.45	81.47±2.26	60.19±2.17
D2	88.16±1.1	88.42 ±1.09	88.15±1.07	82.50±1.45	82.55±1.43	81.57±1.58	82.82±1.07	82.83±1.33	82.97±1.34	80.32±1.26	78.93±1.22	79.04±1.47	78.80±1.4
D3	97.89±0.43	97.77±0.43	97.75±0.29	98.10 ±0.32	98.09±0.28	98.02±0.40	97.91±0.39	98.08±0.36	97.93±0.52	98.02±0.29	98.00±0.24	97.63±0.48	78.22±0.75
D4	94.77 ±0.99	93.54±0.82	93.72±0.89	93.99±1.52	94.20±1.53	93.63±1.08	91.59±1.17	91.92±1.32	92.42±1.16	91.90±1.50	91.70±1.31	92.50±0.74	85.73±0.94
D5	71.40±4.35	71.79±3.62	72.61±3.60	73.89 ±3.59	72.39±3.24	69.64±4.38	72.11±5.49	71.43±4.18	68.91±1.61	69.15±5.21	73.14±6.45	72.10±4.82	71.57±7.65
D6	91.89 ±1.41	91.67±1.11	91.80±1.47	91.11±1.03	91.31±1.42	90.23±1.32	90.60±1.61	89.55±1.62	89.52±1.75	90.14±1.37	88.97±1.24	87.15±1.46	69.96±3.18
D7	98.73±0.31	98.75±0.35	98.67±0.26	98.67±0.31	98.88 ±0.18	98.59±0.26	98.50±0.37	98.44±0.26	98.38±0.33	98.33±0.38	98.31±0.33	98.39±0.24	84.59±0.63
D8	73.26±1.53	70.59±1.61	71.82±1.99	69.62±1.52	72.70±1.80	73.05±1.37	72.97±1.67	74.78±2.01	74.91±1.48	71.58±1.62	70.37±1.58	76.38 ±1.33	58.58±2.31
D9	60.60±1.11	61.27±1.19	60.74±1.10	61.91±1.18	64.41 ±0.93	62.69±1.06	62.80±1.02	62.68±1.05	62.47±1.20	62.50±0.72	61.62±1.02	61.81±0.79	59.82±1.13
D10	93.23 ±0.45	93.22±0.34	92.89±0.25	92.86±0.23	92.69±0.25	92.44±0.39	91.59±0.30	91.53±0.28	91.63±0.39	91.94±0.27	90.87±0.27	91.65±0.27	92.21±0.34
D11	80.53 ±0.22	79.32±0.19	79.79±0.14	77.83±0.18	74.75±0.16	64.19±0.24	72.10±0.09	72.65±0.26	68.96±0.10	65.61±0.24	63.85±0.27	60.63±0.24	45.32±0.19
D12	70.24±0.17	77.08 ±0.19	75.14±0.16	73.93±0.13	74.99±0.16	76.21±0.23	74.71±0.17	71.56±0.12	74.93±0.14	68.60±0.18	66.78±0.17	59.66±0.18	36.22±0.23
Average	83.57±1.16	83.68±1.17	83.84 ±1.14	83.15±1.17	83.59±1.18	81.89±1.22	82.58±1.28	82.6±1.25	82.33±1.1	80.92±1.33	80.41±1.38	79.87±1.19	68.43±1.74

* The streaming features (without missing data) selected by original algorithms.

† The sparse streaming features (with missing data) selected by modified algorithms.

TABLE XII
AVERAGE CLASSIFICATION ACCURACY AS B_S INCREASES FROM 2 TO 20 ($\alpha = 0.1$)

Dataset	*Streaming features	† Sparse streaming features with different B_S									
		2	4	6	8	10	12	14	16	18	20
D1	82.18±1.91	74.09±2.38	82.62±2.24	83.17±2.63	82.58±2.86	81.99±2.74	81.58±2.15	87.60 ±2.45	81.87±2.75	83.46±2.45	81.24±1.92
D2	88.16±1.10	88.73 ±1.36	84.91±1.6	84.11±1.08	81.13±1.36	82.40±1.42	78.20±1.44	86.40±1.33	87.01±1.3	86.45±1.38	79.13±1.43
D3	97.89±0.43	97.93±0.32	98.06±0.1	98.04±0.30	97.84±0.15	98.00±0.32	98.12±0.27	97.71±0.23	97.97±0.27	98.16 ±0.15	97.91±0.20
D4	94.77 ±0.99	93.99±1.53	93.96±1.45	94.29±2.10	95.02 ±1.43	94.91±1.52	93.41±1.72	93.11±1.44	93.01±1.31	93.01±1.53	93.14±1.44
D5	71.40±4.35	69.58±3.53	69.6±4.82	72.26±6.03	72.46±6.29	74.16 ±5.01	71.56±5.72	69.39±3.79	71.84±5.78	73.77±5.30	68.52±4.03
D6	91.89±1.41	91.39±1.28	91.82±1.25	90.98±1.27	91.94±1.33	90.78±1.06	93.71 ±1.47	92.79±1.26	92.10±1.68	89.05±1.60	91.39±1.46
D7	98.63±0.31	98.52±0.25	98.20±0.28	98.68 ±0.34	97.76±0.30	97.92±0.29	98.04±0.25	98.32±0.30	98.20±0.25	97.52±0.37	97.64±0.30
D8	73.26±1.53	71.39±2.18	73.60±2.08	73.69±1.91	73.50±1.90	73.55±2.08	74.66 ±1.66	71.82±1.89	71.01±2.02	71.57±2.21	72.48±1.98
D9	60.60±1.11	60.51±0.87	60.73±0.69	60.95±0.75	60.67±1.20	60.29±0.74	62.90±1.12	74.29 ±0.85	69.72±0.95	60.60±1.38	68.08±0.89
D10	93.23±0.45	93.38±0.30	93.27±0.34	93.08±0.34	93.11±0.33	93.51±0.39	93.27±0.35	93.55 ±0.30	93.32±0.29	93.28±0.29	93.05±0.36
D11	80.53±0.22	80.08±0.17	82.23 ±0.15	81.11±0.18	81.71±0.17	78.36±0.23	75.89±0.22	79.48±0.21	78.07±0.19	78.10±0.20	76.24±0.19
D12	70.24±0.17	79.68±0.17	76.35±0.16	79.84 ±0.15	79.71±0.19	78.96±0.14	75.94±0.17	77.11±0.20	76.14±0.18	77.95±0.18	75.28±0.17
Counts of optimum	/	1	1	2	1	1	2	3	0	1	0

* The streaming features (without missing data) are selected by the original algorithms.

† The sparse streaming features (with missing data) are selected by the modified algorithms.

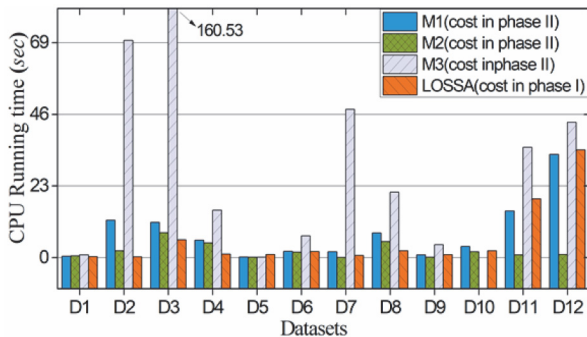


Fig. 4. CPU running time of LOSSA costs in phases I and II with M1, M2, and M3, respectively.

- LOSSA spends comparable CPU time in Phase I to that of M1 and M2 in Phase II. Besides, if we use more CPU cores to implement Phase I for LOSSA in parallel, its computational efficiency can be greatly improved. Therefore, this set of experiments demonstrates that

LOSSA has high computational efficiency in developing Fast-OSFS, SAOLA, and OSFASW to handle OS²FS.

H. Summary of Experiments

In the experiments, we conduct extensive comparisons between original algorithms (Fast-OSFS, SAOLA, and OSFASW) and their modified versions enhance with LOSSA. Based on the comparison results, we summarize LOSSA's advantages and disadvantages as follows.

Advantages:

- When the input streaming features are incomplete, existing algorithms fail to handle them due to their lack of missing data estimator, while an LOSSA algorithm can handle them smoothly with its incorporated LFA component in Phase I. From this point of view, existing algorithms target at OSFS (i.e., OSFS) only, while an LOSSA algorithm can address both issues of OS²FS (i.e., OS²FS) and OSFS. That is the most significant virtue of LOSSA when compared with state-of-the-art OSFS algorithms.

- 2) LOSSA is compatible with existing OSFS algorithms without changing their mechanisms or working schemes. Moreover, it helps them in implementing high-quality feature selection in the case of OS²FS. Especially, when the missing data rate α is smaller than 0.6, it can effectively improve existing OSFS algorithms to handle OS²FS with high accuracy.

Disadvantages:

- 1) It has to tune parameters (λ and B_S) to achieve the best performance. Currently, this issue can be addressed via conducting pretuning process on warming-up datasets. We plan to make this parameter self-adaptive as future work.
- 2) It needs extra computation to complete missing data of sparse streaming features in Phase I. However, such extra computational cost is acceptable in practice and can be significantly reduced through parallelization [46], [47].

V. RELATED WORK

The proposed LOSSA is closely related to LFA on sparse data. In it, LFA is adopted to complete the missing data of sparse streaming features before conducting feature selection owing to its high efficiency of storage and computation, as well as high representation learning ability on sparse data [60]. Besides LFA, other techniques like multiple imputations, expectation-maximization (EM), regression imputation, and matrix completion, can also address such a problem [60]–[63]. However, multiple imputations, EM, and regression imputations do not perform well when the missing rate is high [60]. Candès and Recht [61] and Keshavan *et al.* [62], respectively, proposed a matrix completion approach to address such a problem with nice accuracy. However, they are expensive in both computation and storage [63].

Meanwhile, the online feature selection problem can be further categorized into two branches [7], i.e., online feature selection from streaming data (OFSSD) and OSFS. OFSSD assumes that the feature size remains fixed while data instances increase over time [25], [56]–[59], while OSFS assumes that the data instance count is fixed and the feature dimension increases with time [12], [18]–[23]. In this article, we focus on the latter, i.e., OSFS.

To date, great efforts have been paid to handle OSFS issue [12], [18]–[25]. Perkins and Theiler [18] proposed the Grafting algorithm based on a regularized framework. Since Grafting needs to carefully tune its regularization parameter before determining which feature is most likely to be selected, it is ineffective to process the streaming features whose size is unknown [12]. Zhou *et al.* [19] an Alpha-investing algorithm by using stream-wise regression. Dhillon *et al.* [24] extended this algorithm to handle the problem of multiple feature classes. Although they can handle infinite streaming features, they are limited by their dependence on prior knowledge [23].

Wu *et al.* proposed an OSFS framework based on two parts, i.e., online relevance analysis and online redundancy analysis. Then, they develop the Fast-OSFS algorithm [12]. Based

on this framework, several algorithms have been proposed recently, including an OGFS algorithm [25] that can handle group streaming features with group structure information as prior knowledge, an SAOLA algorithm [20] that is designed for extremely high dimensional feature selection, an OSFASW algorithm [21] that has high prediction accuracy and reduces the selected features number by using self-adaptive sliding-window sampling, an OFS-Density algorithm [22] that does not need the domain information by using adaptive density neighborhood relation, an OFS-A3M algorithm [23] that does not also need the domain information and specify any parameter in advance by utilizing an adaptive neighborhood rough set, ROSFSMI algorithm [76] that employs mutual information in a streaming manner to evaluate the relevancy and redundancy of features, SFS-FI [77] algorithm that can select streaming features to interact with each other, and OGSFS-FI algorithm [78] that considers feature interaction within and between the streaming groups during feature selection.

The above algorithms are sophisticated OSFS algorithms with high efficiency. However, they can only process streaming features without missing data, while streaming features in most real-world applications have missing data. The proposed LOSSA, in comparison, makes significant progress in addressing the real-world feature selection with missing data. It fits industrial needs more appropriately than the prior methods when dealing with streaming features with missing data.

VI. CONCLUSION

This article proposes an LOSSA to handle OS²FS problem well. Its main idea is to adopt LFA to pre-estimate the missing data of sparse streaming features before selection, thereby implementing effective and efficient feature selection. It is compatible with the current OSFS approaches. In particular, three state-of-the-art OSFS algorithms are enhanced with LOSSA to conduct the experiments on twelve benchmark classification datasets. The experimental results well validate LOSSA's capability of effectively boosting an OSFS algorithm to address the issue of OS²FS precisely.

Note that some hyperparameters of LOSSA, including the size of a buffer matrix and regularization coefficient, require manual tuning that is time-consuming and tedious. It is vital to make them self-adaptive through evolutionary computation algorithms [54], [55] or other feasible frameworks to improve LOSSA's practicability. Meanwhile, representation learning is another promising way to process high-dimensional data, like neural networks-based one [74] and Bayesian-based one [75]. It is also highly interesting to extend LOSSA to be compatible with representation learning approaches and gain more applications [79], [80]. We plan to address these challenging issues in our future work.

APPENDIX

A. Theoretical Convergence Analysis of Phase I

This section theoretically analyzes the convergence of Phase I in preprocessing sparse streaming feature matrix B to completed streaming feature matrix $\hat{B}$. First, we, respectively, define L -smooth and strong convex function $f(x)$ [70].

Definition 9 [*L-Smooth Function* $f(x)$]: $f(x)$ is L -smooth, if $\forall x_1, x_2 \in \mathbb{R}^d$ s.t. $\|\nabla f(x_1) - \nabla f(x_2)\|_2 \leq L\|x_1 - x_2\|_2$. (13)

Definition 10 [*Strong Convex* $f(x)$]: $f(x)$ is strong convex if there exists a constant $\delta > 0$ satisfying

$$\forall x_1, x_2 \in \mathbb{R}^d \text{ s.t. } f(x_1) \geq f(x_2) + \nabla f(x_2)(x_1 - x_2)^T + \frac{1}{2}\delta\|x_1 - x_2\|_2^2. \quad (14)$$

Note that the learning objective (5) is nonconvex. Besides, it is the sum of the instant loss (6), i.e., $\varepsilon(U, V) = \sum_{(m,j) \in K_j} \varepsilon_{m,j}$. According to prior research [31], [32], [44], some relaxations should be made to analyze the final convergence as follows.

- 1) Instant loss $\varepsilon_{m,j}$ is considered instead of the sum loss $\varepsilon(U, V)$ because we adopt a single-shot SGD for each update, which corresponds to the single element $f'_{m,j}$.
- 2) One-half of the nonconvex term is fixed to make the instant loss $\varepsilon_{m,j}$ convex, i.e., $v_{j..}$ is treated as a constant to show the model convergence with the update of $u_{m..}$. Note that the update rule is symmetric for $u_{m..}$ and $v_{j..}$. Therefore, the convergence with the update of $v_{j..}$ can be achieved in the same way.

Lemma 1: The instant loss $\varepsilon_{m,j}$ is L -smooth when L is the maximum singular value for matrix $(v_{j..}^T v_{j..} + \lambda E_d)$ and E_d is a $d \times d$ identity matrix.

Proof: Assuming that $u_{\sigma..}$ and $u_{\phi..}$ are two arbitrary and independent row-vectors of latent factor matrix U , we have

$$\begin{aligned} \nabla \varepsilon_{m,j}(u_{\sigma..}) - \nabla \varepsilon_{m,j}(u_{\phi..}) &= -\left(f'_{m,j} - u_{\sigma..} v_{j..}\right) v_{j..} + \lambda u_{\sigma..} \\ &\quad + \left(f'_{m,j} - u_{\phi..} v_{j..}\right) v_{j..} - \lambda u_{\phi..} \\ &= (u_{\sigma..} - u_{\phi..}) \left(v_{j..}^T v_{j..} + \lambda E_d\right). \end{aligned} \quad (15)$$

With (15), we can achieve that

$$\begin{aligned} \|\nabla \varepsilon_{m,j}(u_{\sigma..}) - \nabla \varepsilon_{m,j}(u_{\phi..})\|_2 \\ = \|(u_{\sigma..} - u_{\phi..}) \left(v_{j..}^T v_{j..} + \lambda E_d\right)\|_2. \end{aligned} \quad (16)$$

According to the L_2 -norm properties of a matrix [71], we have the following inequality:

$$\begin{aligned} \|\nabla \varepsilon_{m,j}(u_{\sigma..}) - \nabla \varepsilon_{m,j}(u_{\phi..})\|_2 &\leq \left\| \left(v_{j..}^T v_{j..} + \lambda E_d\right) \right\|_2 \\ &\quad \times \|u_{\sigma..} - u_{\phi..}\|_2 \end{aligned} \quad (17)$$

where $\|(v_{j..}^T v_{j..} + \lambda E_d)\|_2$ denotes the largest singular value of $(v_{j..}^T v_{j..} + \lambda E_d)$. Based on the above inferences, we obtain $L = \|(v_{j..}^T v_{j..} + \lambda E_d)\|_2$. Hence, Lemma 1 holds. ■

Lemma 2: The instant loss $\varepsilon_{m,j}$ is of strong-convexity when δ is the minimum singular value for matrix $(v_{j..}^T v_{j..} + \lambda E_d)$.

Proof: Given arbitrary vectors $u_{\sigma..}$ and $u_{\phi..}$, we expand the state of $\varepsilon_{m,j}$ at $u_{\phi..}$. Following the principle of Taylor-series:

$$\begin{aligned} \varepsilon_{m,j}(u_{\sigma..}) &\approx \varepsilon_{m,j}(u_{\phi..}) + \nabla \varepsilon_{m,j}(u_{\phi..})(u_{\sigma..} - u_{\phi..})^T \\ &\quad + \frac{1}{2}(u_{\sigma..} - u_{\phi..}) \nabla^2 \varepsilon_{m,j}(u_{\phi..})(u_{\sigma..} - u_{\phi..})^T \\ &\Rightarrow \varepsilon_{m,j}(u_{\sigma..}) - \varepsilon_{m,j}(u_{\phi..}) \\ &= \nabla \varepsilon_{m,j}(u_{\phi..})(u_{\sigma..} - u_{\phi..})^T \end{aligned}$$

$$+ \frac{1}{2}(u_{\sigma..} - u_{\phi..}) \nabla^2 \varepsilon_{m,j}(u_{\phi..})(u_{\sigma..} - u_{\phi..})^T. \quad (18)$$

As shown in Definition 10, if $\varepsilon_{m,j}$ is strong convex, we can achieve that

$$\begin{aligned} \varepsilon_{m,j}(u_{\sigma..}) - \varepsilon_{m,j}(u_{\phi..}) &\geq \nabla \varepsilon_{m,j}(u_{\phi..})(u_{\sigma..} - u_{\phi..})^T \\ &\quad + \frac{1}{2}\delta\|u_{\sigma..} - u_{\phi..}\|_2^2. \end{aligned} \quad (19)$$

Thus, Lemma 2 is equivalent to selecting δ to make the following inequality true:

$$(u_{\sigma..} - u_{\phi..}) \nabla^2 \varepsilon_{m,j}(u_{\phi..})(u_{\sigma..} - u_{\phi..})^T \geq \delta\|u_{\sigma..} - u_{\phi..}\|_2^2. \quad (20)$$

From the expression of $\varepsilon_{m,j}$, we can obtain that

$$\nabla^2 \varepsilon_{m,j}(u_{\phi..}) = v_{j..}^T v_{j..} + \lambda E_d. \quad (21)$$

By combining (20) and (21), we only need to prove that

$$(u_{\sigma..} - u_{\phi..}) \left(v_{j..}^T v_{j..} + \lambda E_d\right) (u_{\sigma..} - u_{\phi..})^T \geq \delta\|u_{\sigma..} - u_{\phi..}\|_2^2. \quad (22)$$

Furthermore, (22) is also equivalent to

$$(u_{\sigma..} - u_{\phi..}) \left(v_{j..}^T v_{j..} + \lambda E_d - \delta E_d\right) (u_{\sigma..} - u_{\phi..})^T \geq 0. \quad (23)$$

According to the properties of the matrix, (23) is equivalent to prove that $(v_{j..}^T v_{j..} + \lambda E_d - \delta E_d)$ is a positive semi-definite matrix. As unveiled by [71], $(v_{j..}^T v_{j..} + \lambda E_d - \delta E_d)$ is a positive semi-definite matrix when δ is the minimum singular value of matrix $(v_{j..}^T v_{j..} + \lambda E_d)$, and it satisfies positive semi-definiteness. Hence, Lemma 2 holds.

Considering the t th iteration of LFA in Phase I, from (8) we have the following update rule for $u_{m..}$ on a single entry $f'_{m,j}$

$$u_{m..}^\tau \leftarrow u_{m..}^{\tau-1} - \eta^{t-1} \cdot \nabla \varepsilon_{m,j}(u_{m..}^{\tau-1}) \quad (24)$$

where $u_{m..}^\tau$ and $u_{m..}^{\tau-1}$, respectively, denote the state of $u_{m..}$ updated by the τ th and $(\tau - 1)$ th entry in the t th iteration. Let $u_{m..}^*$ be the optimal state of $u_{m..}$, and we have

$$\begin{aligned} \|u_{m..}^\tau - u_{m..}^*\|_2^2 &= \|u_{m..}^{\tau-1} - \eta^{t-1} \nabla \varepsilon_{m,j}(u_{m..}^{\tau-1}) - u_{m..}^*\|_2^2 \\ &= \|u_{m..}^{\tau-1} - u_{m..}^*\|_2^2 - 2\eta^{t-1} \\ &\quad \cdot \nabla \varepsilon_{m,j}(u_{m..}^{\tau-1})(u_{m..}^{\tau-1} - u_{m..}^*)^T \\ &\quad + (\eta^{t-1}) \nabla \varepsilon_{m,j}(u_{m..}^{\tau-1}) \left\| (u_{m..}^{\tau-1} - u_{m..}^*) \right\|_2^2. \end{aligned} \quad (25)$$

Based on Lemma 2, we achieve that

$$\begin{aligned} \varepsilon_{m,j}(u_{m..}^*) - \varepsilon_{m,j}(u_{m..}^{\tau-1}) &\geq \nabla \varepsilon_{m,j}(u_{m..}^{\tau-1})(u_{m..}^* - u_{m..}^{\tau-1})^T \\ &\quad + \frac{1}{2}\delta\|u_{m..}^* - u_{m..}^{\tau-1}\|_2^2. \end{aligned} \quad (26)$$

Owing to $u_{m..}^*$ is the optimal state of $u_{m..}$, we have that

$$\begin{cases} \nabla \varepsilon_{m,j}(u_{m..}^*) = 0 \\ \varepsilon_{m,j}(u_{m..}^*) < \varepsilon_{m,j}(u_{m..}^{\tau-1}). \end{cases} \quad (27)$$

By substituting (27) into (26), we see that

$$\nabla \varepsilon_{m,j} \left(u_{m,\cdot}^{\tau-1} \right) \left(u_{m,\cdot}^{\tau-1} - u_{m,\cdot}^* \right)^T \geq \frac{1}{2} \delta \left\| u_{m,\cdot}^{\tau-1} - u_{m,\cdot}^* \right\|_2^2. \quad (28)$$

Thus, based on (28), (25) is equivalent to

$$\left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2 \leq \left(1 - \eta^{t-1} \delta \right) \left\| u_{m,\cdot}^{\tau-1} - u_{m,\cdot}^* \right\|_2^2 + \left(\eta^{t-1} \right)^2 \left\| \nabla \varepsilon_{m,j} \left(u_{m,\cdot}^{\tau-1} \right) \right\|_2^2. \quad (29)$$

By taking the expectation of (29), we have

$$E \left[\left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2 \right] \leq \left(1 - \eta^{t-1} \delta \right) E \left[\left\| u_{m,\cdot}^{\tau-1} - u_{m,\cdot}^* \right\|_2^2 \right] + \left(\eta^{t-1} \right)^2 E \left[\left\| \nabla \varepsilon_{m,j} \left(u_{m,\cdot}^{\tau-1} \right) \right\|_2^2 \right]. \quad (30)$$

Following [72], assume that there is a positive number z such that

$$E \left[\left\| \nabla \varepsilon_{m,j} \left(u_{m,\cdot}^{\tau-1} \right) \right\|_2^2 \right] \leq z^2. \quad (31)$$

Thus, based on (31), (30) is equivalent to

$$E \left[\left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2 \right] \leq \left(1 - \eta^{t-1} \delta \right) E \left[\left\| u_{m,\cdot}^{\tau-1} - u_{m,\cdot}^* \right\|_2^2 \right] + \left(\eta^{t-1} \right)^2 z^2. \quad (32)$$

Let us take the learning rate $\eta^{t-1} = \beta / (\delta t)$ with $\beta > 1$, (32) can be reformulated as

$$E \left[\left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2 \right] \leq \left(1 - \frac{\beta}{t} \right) E \left[\left\| u_{m,\cdot}^{\tau-1} - u_{m,\cdot}^* \right\|_2^2 \right] + \frac{1}{t^2} \left(\frac{\beta z}{\delta} \right)^2. \quad (33)$$

By expanding the expression of (33) by induction, we obtain a bound

$$E \left[\left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2 \right] \leq \frac{1}{t} \max \left\{ \left\| u_{m,\cdot}^1 - u_{m,\cdot}^* \right\|_2^2, \frac{\beta^2 z^2}{\delta \beta - 1} \right\} \quad (34)$$

where $u_{m,\cdot}^1$ denotes the initial state of $u_{m,\cdot}$ at the t th iteration.

According to Lemma 1, $\varepsilon_{m,j}$ is L -smooth and we can achieve that

$$\varepsilon_{m,j} \left(u_{m,\cdot}^{\tau} \right) - \varepsilon_{m,j} \left(u_{m,\cdot}^* \right) \leq \frac{L}{2} \left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2. \quad (35)$$

By taking the expectation of (35), we can obtain that

$$E \left[\varepsilon_{m,j} \left(u_{m,\cdot}^{\tau} \right) - \varepsilon_{m,j} \left(u_{m,\cdot}^* \right) \right] \leq \frac{L}{2} E \left[\left\| u_{m,\cdot}^{\tau} - u_{m,\cdot}^* \right\|_2^2 \right]. \quad (36)$$

By substituting (34) into (36), we have the following deduction:

$$E \left[\varepsilon_{m,j} \left(u_{m,\cdot}^{\tau} \right) - \varepsilon_{m,j} \left(u_{m,\cdot}^* \right) \right] \leq \frac{L}{2t} \Theta(\beta) \quad (37)$$

where we have

$$\Theta(\beta) = \max \left\{ \left\| u_{m,\cdot}^1 - u_{m,\cdot}^* \right\|_2^2, \frac{\beta^2 z^2}{\delta \beta - 1} \right\}. \quad (38)$$

Then, we expand (37) on all the known entries of K_j

$$E \left[\sum_{(m,j) \in K_j} \left(\varepsilon_{m,j} \left(u_{m,\cdot}^{\tau} \right) - \varepsilon_{m,j} \left(u_{m,\cdot}^* \right) \right) \right] \leq |K_j| \frac{L}{2t} \Theta(\beta) \quad (39)$$

where $t \rightarrow \infty$, we have $|K_j| (L/2t) \Theta(\beta) \rightarrow 0$.

We can encounter the same situation when $u_{m,\cdot}$ is treated as a constant. Although the learning objective (5) is nonconvex, $u_{m,\cdot}$ and $v_{j,\cdot}$ can be updated alternatively by SGD. Moreover, as unveiled by [73], SGD requires the learning rate $\eta \leq 1/\delta t$ in the t th iteration. Thus, following Lemma 2, the learning rate in the t th iteration satisfies $\eta^{t-1} \leq 1/\delta t$, where δ is the minimum singular value of the matrix $(v_{j,\cdot}^T v_{j,\cdot} + \lambda E_d)$. Besides, we see that the regularization does not affect the convergence. Hence, the convergence analysis of Phase I is completed. ■

REFERENCES

- [1] S. Ramírez-Gallego *et al.*, "An information theory-based feature selection framework for big data under Apache Spark," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 9, pp. 1441–1453, Sep. 2018.
- [2] R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A study on big knowledge and its engineering issues," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 9, pp. 1630–1644, Sep. 2019.
- [3] S. Zhang, L. Yao, and A. Sun, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surveys*, vol. 51, no. 1, pp. 1–35, 2019.
- [4] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018.
- [5] A. Patrizio, *IDC: Expect 175 Zettabytes of Data Worldwide by 2025*, NetworkWorld IDC, Needham, MA, USA, 2018. [Online]. Available: <https://www.networkworld.com/article/3325397/idc-expect-175-zettabytes-of-data-worldwide-by-2025.html>
- [6] S. Athey, "Beyond prediction: Using big data for policy problems," *Science*, vol. 355, no. 6324, pp. 483–485, 2017.
- [7] X. Hu, P. Zhou, P. Li, J. Wang, and X. Wu, "A survey on online feature selection with streaming features," *Front. Comput. Sci.*, vol. 12, no. 3, pp. 479–493, 2018.
- [8] J. Li *et al.*, "Feature selection: A data perspective," *ACM Comput. Surveys*, vol. 50, no. 6, p. 94, 2018.
- [9] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review," in *Data Clustering*. Boca Raton, FL, USA: Chapman Hall/CRC, 2018, pp. 29–60.
- [10] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, 2014.
- [11] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," in *Data Classification: Algorithms and Applications*. Boca Raton, FL, USA: Taylor Francis, 2014, pp. 37–64.
- [12] X. Wu, K. Yu, W. Ding, H. Wang, and X. Zhu, "Online feature selection with streaming features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 5, pp. 1178–1192, May 2013.
- [13] E. Beyazit, J. Alagurajah, and X. Wu, "Online learning from data streams with varying feature spaces," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 3232–3239.
- [14] S. Loscalzo, L. Yu, and C. Ding, "Consensus group stable feature selection," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2009, pp. 567–576.
- [15] I. Rodríguez-Lujan, R. Huerta, C. Elkan, and C. S. Cruz, "Quadratic programming feature selection," *J. Mach. Learn. Res.*, vol. 11, no. 49, pp. 1491–1516, 2010.
- [16] J. Ni, H. Fei, W. Fan, and X. Zhang, "Automated medical diagnosis by ranking clusters across the symptom-disease network," in *Proc. IEEE Int. Conf. Data Min.*, 2017, pp. 1009–1014.
- [17] Y. Shen, C. Wu, C. Liu, Y. Wu, and N. Xiong, "Oriented feature selection SVM applied to cancer prediction in precision medicine," *IEEE Access*, vol. 6, pp. 48510–48521, 2018.
- [18] S. Perkins and J. Theiler, "Online feature selection using grafting," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 592–599.
- [19] J. Zhou, D. P. Foster, R. A. Stine, and L. H. Ungar, "Streamwise feature selection," *J. Mach. Learn. Res.*, vol. 7, no. 67, pp. 1861–1885, 2006.
- [20] K. Yu, X. Wu, W. Ding, and J. Pei, "Scalable and accurate online feature selection for big data," *ACM Trans. Knowl. Disc. Data*, vol. 11, no. 2, p. 16, 2016.
- [21] D. You *et al.*, "Online feature selection for streaming features using self-adaption sliding-window sampling," *IEEE Access*, vol. 7, pp. 16088–16100, 2019.

- [22] P. Zhou, X. Hu, P. Li, and X. Wu, "OFS-density: A novel online streaming feature selection method," *Pattern Recognit.*, vol. 86, pp. 48–61, Feb. 2019.
- [23] P. Zhou, X. Hu, P. Li, and X. Wu, "Online streaming feature selection using adapted neighborhood rough set," *Inf. Sci.*, vol. 481, pp. 258–279, May 2019.
- [24] P. Dhillon, D. Foster, and L. Ungar, "Feature selection using multiple streams," in *Proc. 30th Int. Conf. Artif. Intell. Stat.*, 2010, pp. 153–160.
- [25] J. Wang *et al.*, "Online feature selection with group structure analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 3029–3041, Nov. 2015.
- [26] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of Things for smart healthcare: Technologies, challenges, and opportunities," *IEEE Access*, vol. 5, pp. 26521–26544, 2017.
- [27] G. Manogaran, R. Varatharajan, D. Lopez, P. M. Kumar, R. Sundarasekar, and C. Thota, "A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system," *Future Gener. Comput. Syst.*, vol. 82, pp. 375–387, May 2018.
- [28] J. Ignatius, A. Hatami-Marbini, A. Rahman, L. Dhamotharan, and P. Khoshnevis, "A fuzzy decision support system for credit scoring," *Neural Comput. Appl.*, vol. 29, no. 10, pp. 921–937, 2018.
- [29] C. Bai, B. Shi, F. Liu, and J. Sarkis, "Banking credit worthiness: Evaluating the complex relationships," *Omega*, vol. 83, pp. 26–38, Mar. 2019.
- [30] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and X. Wu, "A data-characteristic-aware latent factor model for Web service QoS prediction," *IEEE Trans. Knowl. Data Eng.*, early access, Aug. 5, 2020, doi: [10.1109/TKDE.2020.3014302](https://doi.org/10.1109/TKDE.2020.3014302).
- [31] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
- [32] X. Luo, Z. Wang, and M. Shang, "An instance-frequency-weighted regularization scheme for non-negative latent factor analysis on high dimensional and sparse data," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 6, pp. 3522–3532, Jun. 2021.
- [33] C. Wang *et al.*, "Confidence-aware matrix factorization for recommender systems," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 434–442.
- [34] H.-J. Xue, X.-Y. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 3203–3209.
- [35] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, "Aspect-aware latent factor model: Rating prediction with ratings and reviews," in *Proc. Int. World Wide Web Conf.*, 2018, pp. 639–648.
- [36] L. Qiu, S. Gao, W. Cheng, and J. Guo, "Aspect-based latent factor model by integrating ratings and reviews for recommender system," *Knowl. Based Syst.*, vol. 110, pp. 233–243, Oct. 2016.
- [37] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and M. Zhou, "A deep latent factor model for high-dimensional and sparse matrices in recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 7, pp. 4285–4296, Jul. 2021.
- [38] M. Gong, X. Jiang, H. Li, and K. C. Tan, "Multiobjective sparse non-negative matrix factorization," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 2941–2954, Aug. 2019.
- [39] J. Wang, F. Tian, H. Yu, C. H. Liu, K. Zhan, and X. Wang, "Diverse non-negative matrix factorization for multiview data representation," *IEEE Trans. Cybern.*, vol. 48, no. 9, pp. 2620–2632, Sep. 2018.
- [40] P. Melville and V. Sindhwani, "Recommender systems," in *Encyclopedia of Machine Learning and Data Mining*. New York, NY, USA: Springer, 2017, pp. 1056–1066.
- [41] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: Introduction and challenges," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2015, pp. 1–34.
- [42] X. Luo, Z. Liu, S. Li, M. Shang, and Z. Wang, "A fast non-negative latent factor model based on generalized momentum method," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 610–620, Jan. 2021.
- [43] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Comput. Surveys*, vol. 47, no. 1, pp. 1–45, 2014.
- [44] D. Wu, Q. He, X. Luo, M. Shang, Y. He, and G. Wang, "A posterior-neighborhood-regularized latent factor model for highly accurate Web service QoS prediction," *IEEE Trans. Services Comput.*, early access, Dec. 24, 2019, doi: [10.1109/TSC.2019.2961895](https://doi.org/10.1109/TSC.2019.2961895).
- [45] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artif. Intell.*, vol. 97, nos. 1–2, pp. 273–324, 1997.
- [46] H. Li, K. Li, J. An, and K. Li, "MSGD: A novel matrix factorization approach for large-scale collaborative filtering recommender systems on GPUs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 7, pp. 1530–1544, Jul. 2018.
- [47] X. Luo, H. Liu, G. Gou, Y. Xia, and Q. Zhu, "A parallel matrix factorization based recommender by alternating stochastic gradient descent," *Eng. Appl. Artif. Intell.*, vol. 25, no. 7, pp. 1403–1412, 2012.
- [48] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the NIPS 2003 feature selection challenge," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Assoc., 2005, pp. 545–552.
- [49] A. Rosenwald *et al.*, "The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma," *New Eng. J. Med.*, vol. 346, no. 25, pp. 1937–1947, 2002.
- [50] O. Chapelle, B. Scholkopf, and A. Zien, "Semi-supervised learning (Chapelle, O. *et al.*, Eds.; 2006) [book reviews]," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, p. 542, Mar. 2009.
- [51] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation and prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.
- [52] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
- [53] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
- [54] W. Dong and M. Zhou, "Gaussian classifier-based evolutionary strategy for multimodal optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 6, pp. 1200–1216, Jun. 2014.
- [55] H. Lin, B. Zhao, D. Liu, and C. Alippi, "Data-based fault tolerant control for affine nonlinear systems through particle swarm optimized neural networks," in *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 4, pp. 954–964, Jul. 2020.
- [56] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time object tracking via online discriminative feature selection," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 4664–4677, Dec. 2013.
- [57] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1631–1643, Oct. 2005.
- [58] V. R. Carvalho and W. W. Cohen, "Single-pass online learning: Performance, voting schemes and online feature selection," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2006, pp. 548–553.
- [59] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, May 2017.
- [60] J. Scheffer, "Dealing with missing data," in *Research Letters in the Information and Mathematical Sciences*, vol. 3. Auckland, New Zealand: Inst. Inf. Math. Sci., 2002, pp. 153–160.
- [61] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, no. 6, pp. 717–772, 2009.
- [62] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from a few entries," *IEEE Trans. Inf. Theory*, vol. 56, no. 6, pp. 2980–2998, Jun. 2010.
- [63] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. 45th Annu. ACM Symp. Theory Comput.*, 2013, pp. 665–674.
- [64] H. Liu, M. Zhou, and Q. Liu, "An embedded feature selection method for imbalanced data classification," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 3, pp. 703–715, May 2019.
- [65] X. S. Lu, M. Zhou, L. Qi, and H. Liu, "Clustering algorithm-based analysis of rare event evolution via social media data," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 2, pp. 301–310, Apr. 2019.
- [66] D. Dua and C. Graff, (2019). *UCI Machine Learning Repository*. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [67] P. Zhang, S. Shu, and M. Zhou, "An online fault detection method based on SVM-grid for cloud computing systems," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 2, pp. 445–456, Mar. 2018.
- [68] X. Chang, F. Nie, S. Wang, Y. Yang, X. Zhou, and C. Zhang, "Compound rank-k projections for bilinear analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 7, pp. 1502–1513, Jul. 2016.
- [69] R. Neapolitan, *Learning Bayesian Networks*. Harlow, U.K.: Prentice Hall, 2003.
- [70] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
- [71] X. D. Zhang, *Matrix Analysis and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

- [72] A. S. Nemirovski, A. Juditsky, G. Lan, and A. A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM J. Optim.*, vol. 19, no. 4, pp. 1574–1609, Jan. 2009.
- [73] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 1571–1578.
- [74] E. P. Frady, S. J. Kent, B. A. Olshausen, and F. T. Sommer, "Resonator networks, 1: An efficient solution for factoring high-dimensional, Distributed representations of data structures," *Neural Comput.*, vol. 32, no. 12, pp. 2311–2331, 2020.
- [75] S. Park and M. Thorpe, "Representing and learning high dimensional data with the optimal transport map from a probabilistic viewpoint," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7864–7872.
- [76] M. Rahmaninia and P. Moradi, "OSFSMI: Online stream feature selection method based on mutual information," *Appl. Soft Comput.*, vol. 68, pp. 733–746, Jul. 2018.
- [77] P. Zhou, P. Li, S. Zhao, and X. Wu, "Feature interaction for streaming feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 6, 2020, doi: [10.1109/TNNLS.2020.3025922](https://doi.org/10.1109/TNNLS.2020.3025922).
- [78] P. Zhou, N. Wang, and S. Zhao, "Online group streaming feature selection considering feature interaction," *Knowl. Based Syst.*, vol. 226, pp. 1–11, Aug. 2021.
- [79] L. Chen, L. Wang, Z. Han, J. Zhao, and W. Wang, "Variational inference based kernel dynamic Bayesian networks for construction of prediction intervals for industrial time series with incomplete input," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 5, pp. 1437–1445, Sep. 2020.
- [80] J. Bi, H. Yuan, and M. Zhou, "Temporal prediction of multiapplication consolidated workloads in distributed clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1763–1773, Oct. 2019.



Di Wu (Member, IEEE) received the Ph.D. degree in computer application technology from the Chongqing Institute of Green and Intelligent Technology (CIGIT), Chinese Academy of Sciences (CAS), Chongqing, China, in 2019.

He is currently an Associate Professor with the CIGIT, CAS. He was a Visiting Scholar with the University of Louisiana at Lafayette, Lafayette, LA, USA, from April 2018 to April 2019. He has published over 40 papers, including IEEE

TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, ICDM, WWW, ECAI, IJCAI, and PAKDD. His research interests include machine learning and data mining.



Yi He received the B.E. degree in transportation engineering from the Harbin Institute of Technology, Harbin, China, in 2013, and the Ph.D. degree in computer science from the Center for Advanced Computer Studies, University of Louisiana at Lafayette, Lafayette, LA, USA, in 2020.

He is an Assistant Professor with the Department of Computer Science, Old Dominion University, Norfolk, VA, USA. His research interests lie broadly in data mining, artificial intelligence, and

optimization theory and specifically in online machine learning, real-time data analytics, representation learning, graph mining, and XAI. His works have been published in top-tier venues—AAAI, IJCAI, WWW, ICDM, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.



Xin Luo (Senior Member, IEEE) received the B.S. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2011.

In 2016, he joined the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, as a Professor of Computer Science and Engineering. His current research interests include big data analysis and intel-

ligent control. He has published over 100 papers (including over 60 IEEE transactions papers) in the above areas.

Prof. Luo has received the Outstanding Associate Editor Reward from IEEE/CAA JOURNAL OF AUTOMATICA SINICA in 2020, and from IEEE ACCESS in 2018. He was a recipient of the Hong Kong Scholar Program jointly by the Society of Hong Kong Scholars and China Post-Doctoral Science Foundation in 2014, the Pioneer Hundred Talents Program of Chinese Academy of Sciences in 2016, and the Advanced Support of the Pioneer Hundred Talents Program of Chinese Academy of Sciences in 2018. He is currently serving as an Associate Editor for the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, IEEE ACCESS, and *Neurocomputing*.



MengChu Zhou (Fellow, IEEE) received the B.S. degree in control engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1983, the M.S. degree in automatic control from the Beijing Institute of Technology, Beijing, China, in 1986, and the Ph.D. degree in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990.

He joined New Jersey Institute of Technology (NJIT), Newark, NJ, USA, in 1990, where he is currently a Distinguished

Professor of Electrical and Computer Engineering. He has over 900 publications, including 12 books, over 600 journal papers (over 500 in IEEE transactions), 29 patents, and 29 book-chapters. His research interests are in Petri nets, intelligent automation, Internet of Things, big data, Web services, and intelligent transportation.

Dr. Zhou is a recipient of the Humboldt Research Award for U.S. Senior Scientists from Alexander von Humboldt Foundation, the Franklin V. Taylor Memorial Award and the Norbert Wiener Award from IEEE Systems, Man and Cybernetics Society, Excellence in Research Prize and Medal from NJIT, and the Edison Patent Award from the Research and Development Council of New Jersey. He is the Founding Editor of IEEE Press Book Series on Systems Science and Engineering, the Editor-in-Chief of IEEE/CAA JOURNAL OF AUTOMATICA SINICA, and an Associate Editor of IEEE INTERNET OF THINGS JOURNAL, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, and IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS. He is a Life Member of Chinese Association for Science and Technology-USA and served as its President in 1999. He is a Fellow of International Federation of Automatic Control, American Association for the Advancement of Science, Chinese Association of Automation, and National Academy of Inventors.