Online Sparse Streaming Feature Selection with Uncertainty

Feilong Chen  
School of Computer Science and Technology,  
Chongqing University of Posts and Telecommunications, Chongqing 400715, China, chenfl@cqupt.edu.cn

Di Wu*  
College of Computer and Information Science, Southwest University, Chongqing 400715, China, wudi.cigit@gmail.com

Jie Yang  
School of Physics and Electronic Science, Zunyi Normal University, Zunyi 563002, China, yj30966074@foxmail.com

Yi He  
Department of Computer Science, Old Dominion University, Norfolk, VA 23529, USA, yihe@cs.odu.edu

Abstract—Online streaming feature selection (OSFS), which conducts feature selection in an online manner, plays an important role in dealing with high-dimensional data. In many real applications such as intelligent healthcare platform, streaming feature always has some missing data, which raises a crucial challenge in conducting OSFS, i.e., how to establish the uncertain relationship between sparse streaming features and labels. Unfortunately, existing OSFS algorithms never consider such uncertainty. To fill this gap, we in this paper propose an online sparse streaming feature selection with uncertainty (OS2FSU) algorithm. OS2FSU consists of two main parts: 1) latent factor analysis is utilized to pre-estimate the missing data in sparse streaming features before conducting feature selection, and 2) fuzzy logic and neighborhood rough set are employed to alleviate the uncertainty between estimated streaming features and labels during conducting feature selection. In the experiments, OS2FSU is compared with five state-of-the-art OSFS algorithms on six real datasets. The results demonstrate that OS2FSU outperforms its competitors when missing data are encountered in OSFS.

Keywords—Online Feature Selection, Latent Factor Analysis, Missing Data, Sparse Streaming Feature, Fuzzy Membership Function, Neighborhood Rough Set

I. INTRODUCTION

Feature selection is an important technique to reduce high-dimensional data [1]. It selects the best subset from all high-dimensional features as well as retains their essential characteristics. Traditional feature selection methods, such as mutual information-based [1], [2], neighborhood rough set-based [3]–[5], mRMR (minimal Redundancy and Maximal Relevance)-based [6], assume that the feature space is predefined and static. But in many real applications, it is impossible to observe all the feature space in advance as features are usually generated over time continually. Hence, the assumption of static feature space does not hold in real-world scenarios.

Considering dynamic feature space, online streaming feature selection (OSFS) methods were proposed [7]. OSFS performs feature selection on streaming features that arrive one by one over time with instance count remaining fixed. Recently, many OSFS algorithms have been proposed, including fast-OSFS [7], SAOLA [8], OSFASW [9], SFS_FI [10], etc. Notably, these OSFS algorithms have a common assumption that streaming features are complete without any missing data. While in real applications, it is impossible to collect dynamic features completely [11]. For example, the features describing patients’ symptoms involve many different healthcare service providers (hospitals, labs, etc.) and inspection equipment (respiratory sensors, pulse monitors, etc.). Since different patients are prone to have different symptoms, it is impossible that each patient can collect all the features. Therefore, a new problem arises: how to implement online sparse streaming feature selection (OS2FS)?

To address this problem, Wu et al. [12] proposed a latent-factor-analysis-based online sparse-streaming-feature selection algorithm (LOSSA). LOSSA adopts latent factor analysis (LFA) [13]–[22] to pre-estimate the missing data of sparse streaming features. However, since there are some inevitable errors between estimated data and real data, it is difficult to establish a certain relationship between sparse streaming features and labels. As a result, uncertainty pervades the process of OS2FS. Unfortunately, LOSSA never considers such uncertainty, which may cause inferior performance.

To fill this gap, we proposed an online sparse streaming feature selection with uncertainty (OS2FSU) algorithm. Significantly different from LOSSA, OS2FSU employs fuzzy logic and neighborhood rough set to alleviate the uncertainty between sparse streaming features and labels during implementing OS2FS, making it achieve better performance. The main contributions of this paper are summarized as follows:

a) Algorithm. We propose OS2FSU to address the two problems of missing data and uncertainty in OS2FS.

b) Theoretical analyses. We provide detailed technical principles, algorithm design, and time complexity analyses for the proposed OS2FSU.

c) Experiments. We compare the proposed OS2FSU with five algorithms on six real datasets. The results demonstrate that OS2FSU outperforms these comparison algorithms when missing data are encountered in OSFS.

II. RELATED WORK

OSFS is an important and hot research direction of feature selection. To date, many sophisticated OSFS algorithms were proposed [7], [23], [24].
A classical OSFS framework was proposed by Wu et al. [7]. They proposed to use the online relevance and online redundancy analyses to detect the irrelevant and redundant features during OSFS. Yu et al. [8] proposed SAOLA algorithm for large-scale OSFS. SAOLA employs a novel pairwise comparison technique to maintain a frugal model over training time. To reduce the number of selected features, OSFASW was proposed by You et al [9] based on a self-adaptive sliding-window sampling strategy. Later, Zhou et al. proposed OFS-Density [23] and OFS-A3M [24] algorithms based on the neighborhood rough set. Both of them do not consider domain information during training. OFS-Density uses the density information of the neighborhood instances, while OFS-A3M uses the gap information between neighbors of the target object, to select the proper final feature set automatically. Consequently, considering the feature interaction, SFS-FI [10] measures the interaction degree between features by interaction gain to guarantee that the selected features interact with each other. Notably, the above OSFS algorithms do not consider the missing data issue of OSFS. To address this problem, Wu et al. [12] proposed a LOSSA algorithm, adopts latent factor analysis (LFA) to pre-estimate the missing data of sparse streaming features.

Since there are some inevitable errors between estimated data and real data, uncertainty pervades the process of OSFS, which is not considered by LOSSA. Significantly different from LOSSA, OS2FSU employs fuzzy logic and neighborhood rough set to alleviate the adverse effects of such uncertainty, making it achieve much better performance than LOSSA.

### III. PRELIMINARIES

#### A. Latent Factor Analysis (LFA)

LFA [13]–[16], [19], [20], [25]–[37] is a decomposition-based matrix method that specializes in the high-dimensional and sparse matrix, aims to get a complement matrix by factorizing the observed elements into two low-rank matrices.

**Definition 1 (Latent Factor Analysis).** For a given sparse matrix $R^{M \times d}$, LFA factorizes into two low-rank matrices $P^{M \times r}$ and $Q^{r \times d}$. If the value of predicted matrix $R$ is similar to $R$ at the known position, then it can be considered that the value is also approximately in the predicted position, where $R = P \times Q'$. Loss function (1) used to measure the sum errors between $R$ and $R$, and add regularization to avoid overfitting.

$$L(P, Q) = \frac{1}{2} |\Omega| (R - PQ)^T (R - PQ) + \frac{\lambda}{2} (\|P\|_F^2 + \|Q\|_F^2)$$

(1)

Where $\lambda$ is the regularization parameter, $\Omega$ means Hadamard product and $\Omega_{mj}$ equals 1 if the data $r_{mj}$ at the $m$-th row and $j$-th column is observed, otherwise equals 0. $P$ and $Q$ can be acquired conveniently by stochastic gradient descent(SGD).

#### B. Fuzzy Logic

Fuzzy logic [38], [39] is a fuzzy method to deal with uncertain problems which can’t be solved by boolean logic. Whether the elements in the fuzzy set are discrete or continuous, the membership function (MF) [40] portrayed the ambiguity, and usually used to handle some part of model uncertainties. The common MFs graphs show in Figure 1.

#### C. Neighborhood Rough Set

For a neighbor rough set $NDS = <U, F, D>$ [3]–[5], where $U = \{x_1, x_2, \ldots, x_n\}$ represents a finite set of n samples, $F$ is the set of features. $D$ represents the decision attribute. Then, we have the following definitions for $NDS$.

**Definition 2.** Given a nonempty metric space $< U, F, D >$, let $x \in U, h > 0$, $G \subseteq F, X \subseteq U$, so the point set:

$$\delta_{\partial}^h(x) = \{y | \delta(x, y) \leq h, y \in U\}$$

(2)

denotes the neighborhood of arbitrary $x$ on attribute subset $G$, centered at $x$ and $h$ is the distance radius. So the lower approximation and upper approximation of decision $D$ on $G$, in terms of a $\delta$ neighborhood relation are defined, respectively:

$$\bar{R}_{\delta}^G X = \{x | \delta_{\partial}^h(x) \subseteq X, x \in U\}$$

$$\tilde{R}_{\delta}^G X = \{x | \delta_{\partial}^h(x) \cap X \neq \emptyset, x \in U\}$$

(3)

**Definition 3.** Given $NDS = < U, F, D >$, $G \subseteq F$, the maximum nearest neighbor dependency degree of $G$ on the decision attribute $D$ is given as follows:

$$\gamma_c(D) = \frac{\text{Card}(R_cD)}{\text{Card}(U)}$$

(4)

### IV. OUR PROPOSED ALGORITHM

#### A. Problem statement

**Definition 4 (Sparse Streaming Features).** Let sparse streaming features set $F = \{F_1, F_2, \ldots, F_t\}$ where the feature vector $F_t = [f_{t1}, f_{t2}, \ldots, \text{NAN}, \ldots, f_{tm}]^T, t \in \{1, 2, \ldots, T\}$. $F$ is a feature set that has $M$ instances and $T$ features with missing data. $F_t$ means a sparse feature flowed at time point $t$, $\text{NAN}$ denotes missing data.

**Definition 5 (Online Sparse Streaming Feature Selection, OS2FSU [12]).** Let $O_{St} = \{F_1, F_2, \ldots, F_t\}$ denotes the observed streaming features set arrived at time point $t-1$. The selected sparse streaming feature subset $S_{t1} \subseteq O_{t1}$. At time point $t$, a new sparse feature $F_t$ arrives, OS2FSU accesses the feature relevance to label $C$, and decides it add to $S_{t1}$ or discard it, ensuring the model with maximum performance at time point $t$.

For OS2FSU, there is exist two challenges: 1) completing the missing data of the sparse streaming features, and 2) establishing the uncertain relationships between sparse streaming features and labels caused by complement.

#### B. Our Algorithm

Our proposed algorithm OS2FSU can be divided into two phases to handle the above two challenges. The first part is complete the missing value of the sparse streaming features and the second part is to select the best-classified prediction subset with fuzzy logic. The framework of our OS2FSU is shown in Fig.2.

**Phase 1:** For sparse streaming features, LFA predicts the missing elements based on the observed data. Let $R^{M \times Bs}$ denote the buffer matrix that received the streaming features. As the features flowed from time point $t$ to $t + Bs - 1$, i.e., $B = \{F_t, F_{t+1}, \ldots,$
TABLE I. ALGORITHM DESIGN OF OS2FSU

| No. | Operation                                      |
|-----|------------------------------------------------|
| 1   | Parameters: maximum number of iteration \(\text{iter}\), the number of latent factor \(\delta\), the fixed column \(\text{B}_k\) |
| 2   | Initialize: \(SF=\emptyset\), \(FSF=\emptyset\), \(B_0=\emptyset\), \(B_3\) |
| 3   | Repeat                                       |
|     | A new sparse streaming feature \(f_i\) arrives at the time \(t\); |
|     | 4   | If \(|B|>B_3\) |
|     | 5   | \(B = B \cup f_i\) |
|     | 6   | continue |
|     | 7   | initializing \(d, \eta, \eta, P, Q, \text{Imax}\), \(\text{iter} = 1\) |
|     | 8   | While \(\text{iter} < \text{Imax}\) & \& not converge |
|     | 9   | for \(j = t\) to \(t+B_3\) |
|     | 10  | for each \(f_m \in K_j\), \(K_j\) is the known entry set of \(F_i\) |
|     | 11  | update \(p_m, q\) according to (1) |
|     | 12  | end for |
|     | 13  | end for |
|     | 14  | end while |
| 15  | \(B = PQ\) |
| 16  | /* online relevance analysis */ |
| 17  | for \(i = t\) to \(t+B_3\) |
| 18  | fetch \(f_i\) from \(B\) |
| 19  | If \(\text{re-Dep}(C, \hat{F}_i) \neq \emptyset\) |
| 20  | /* online redundancy analysis I */ |
| 21  | If \(S \subseteq SF \cap \hat{F}_i\) |
| 22  | discard \(f_i\) and continue |
| 23  | end if |
| 24  | \(SF = SF \cup \hat{F}_i\) |
| 25  | /* online redundancy analysis II */ |
| 26  | for each feature \(Q \in SF \cap \hat{F}_i\) |
| 27  | if \(S \subseteq SF \cup Q\) s.t. \(\text{re-Ind}(C, \hat{F}_i, S)\) |
| 28  | \(SF = SF \cup Q\) |
| 29  | end if |
| 30  | end if |
| 31  | else if not \(\text{re-Ind}(C, \hat{F}_i) \neq \emptyset\) |
| 32  | /* online fuzzy correlation analysis */ |
| 33  | \(FSF = FSP \cap \hat{F}_i\) |
| 34  | Calculated \(\gamma_i(C)\) by (4) and sorted in \(FSF\) |
| 35  | end if |
| 36  | end for |
| 37  | Add the top \(|SF|/2\) of \(FSF\) to \(SF\) |
| 38  | \(B = B \cup f_i\), \(t = t+1\) |
| 39  | Until no features are available |

Output selected completed streaming features subset \(SF\).

After \(B\) is filled, LFA will be applied to complete the missing value in \(B\), and let \(B = \{F_i, F_{i+1}, \ldots, F_{t+B_3}\}\) denotes the corresponding completed matrix. According to definition 1, we reformulate the objective function for every single entry in \(B\) as follows:

\[
E_{m,j} = \frac{1}{2}\left(f_{m,j} - \sum_{i=1}^{d} p_{m,i} q_{j,i}\right)^2 + \frac{\lambda}{2} \left(\sum_{i=1}^{d} p_{m,i}^2 + \sum_{i=1}^{d} q_{j,i}^2\right).
\]  (5)

Where \(d\) is the rank of \(B\), \(f_{m,j}\) is the observed element of \(m\)-th row of \(F_i\). As SGD has an excellent iterative effect, we minimize \(E_{m,j}\) by SGD. In one iteration, all observed entries in \(B\) are used to train \(P\) and \(Q\) unceasingly. Then \(\hat{B} = PQ^T\).

**Phase II:** This phase conducts feature selection for all the features in \(\hat{B} = \{\hat{F}_i, \hat{F}_{i+1}, \ldots, \hat{F}_{t+B_3}\}\), and contains three main operations, i.e., online relevance analysis, online redundancy analysis and online fuzzy correlation analysis.

**Online relevance analysis:** At time point \(t\), we fetch \(\hat{F}_i\) from \(\hat{B}\) and evaluate its relevance to label \(C\). For continuous data, Fisher’s-z test [7] is used to calculate the relevance. Let \(p\) denotes the \(p\)-value return by Fisher’s-z test. Significance level \(\alpha\) is used to measure the probability of rejecting the null hypothesis, if \(p > \alpha\), the null hypothesis is accepted, \(\hat{F}_i\) and \(C\) are conditionally independent within given \(S\), called \(\text{Ind}(C, \hat{F}_i | S)\). If \(p \leq \alpha\), the null hypothesis is rejected, defined \(\text{Dep}(C, \hat{F}_i | S)\) to describe the conditionally dependent between \(\hat{F}_i\) and \(C\) within given \(S\). For discrete data, \(G^2\) test will be implemented.

**Online redundancy analysis:** If \(\hat{F}_i\) is relevant to \(C\), then we analyze its redundancy. If \(S \subseteq MB(C_i, t)\), s.t. \(P(C | \hat{F}_i, S) = P(C | S)\), \(\hat{F}_i\) is the redundant feature and discarded. Furthermore, we also evaluate the features exited in \(MB(C_i, t)\) if become redundant as \(\hat{F}_i\) added, where \(MB(C)\) is the Markov blanket for \(C\).

**Fuzzy correlation analysis:** Traditionally, significance level \(\alpha\) is \(0.01\) or \(0.05\). But in OSFS, due to the uncertain error between real value and predicted value, a fixed \(\alpha\) is no longer acceptable. So our algorithm utilizes trapezoidal MF to regulate \(\alpha\) fluctuating between \([0.01, 0.1]\), then we define the fuzzy relevance feature and redefine \(\text{Ind}(C, \hat{F}_i | S)\) and \(\text{Dep}(C, \hat{F}_i | S)\).

**Definition 6 (Fuzzy Relevance Feature).** Let \(\mu\) denote fuzzy significant threshold. The \(p\)-value of feature \(\hat{F}_i\) is calculated by relevance analysis, if \(\mu \leq p \leq 0.1\), we call \(\hat{F}_i\) is fuzzy relevance feature. And will be calculated its dependency degree \(\gamma_i(C)\) by (4) and sorted.

**Definition 7:** If \(p > 0.1\), we accept the null hypothesis that \(\hat{F}_i\) and label are conditionally independent within given \(S\), called \(\text{re-Ind}(C, \hat{F}_i | S)\). If \(p < 0.1\), reject the null hypothesis, \(\hat{F}_i\) is the correlation with \(C\) within given \(S\), defined \(\text{re-Dep}(C, \hat{F}_i | S)\).

**Algorithm Design.** Based on the theorem analysis, the pseudocode of OSFS is given in Table I. Step 5-8 uses
buffer matrix $B$ to cache the received features. When $B$ is filled, steps 9-18 implement LFA model to predict the missing data, and get the completed matrix $\hat{B}$. For each feature $\hat{F}_j$ in $\hat{B}$, Steps 19-22 utilize re-Det$(C, F_j)$ to evaluate the correlation between $\hat{F}_j$ and $C$. If $\hat{F}_j$ is relevant, it will go to step 23 to online redundancy analysis. In steps 23-34, redundancy analysis I evaluate the redundancy of the new relevant feature $\hat{F}_j$. If $\hat{F}_j$ is redundant, it will be discarded. Otherwise, $\hat{F}_j$ will be added to the currently selected feature set $SF$. Then redundancy analysis II is triggered. The original feature exiting in $SF$ will be checked to whether become redundant as the new feature $\hat{F}_j$ added. If $\hat{F}_j$ is neither relevance feature nor irrelevance feature, the dependency degree $\gamma$ of $\hat{F}_j$ will be calculated and sorted in steps 35-40, the top $|SF|/2$ of $SF$ will be added into $SF$.

C. Time complexity Analysis.

The time complexity of our algorithm mainly depends on the missing data completion. Supposing there are $M$ instances and $T$ features, the rate of total missing is $\theta$, the time complexity of Phase I is $O(M \times T \times (1-\theta) \times d)$. For Phase II, let $RF$ denotes all relevant features in the total feature set, $RF_j$ is the performed redundant features in $RF$. The time complexity is $O(|RF-RF_j| \gamma(\omega_0) + |RF_j| \gamma(\omega_0))$, where $\omega$ considers the same subset of $SF$. Thus the total time cost is $O(M \times T \times (1-\theta) \times d + |RF-RF_j| \gamma(\omega_0) + |RF_j| \gamma(\omega_0))$.

V. EXPERIMENTS AND RESULTS

A. Experimental Settings

1) Dataset. We apply our algorithm on six datasets from ASU, openML, UCI. They are summarized in Table II.

| Mark | Datasets | Features Instances Classes |
|------|----------|---------------------------|
| D1   | mfeat-factors | 217 2000 2          |
| D2   | SMK-CAN-187   | 19993 187 2        |
| D3   | lung        | 3312 203 4          |
| D4   | USPS        | 256 9298 10        |
| D5   | COIL        | 241 1500 6         |
| D6   | Isolet      | 617 1560 26        |

2) Baselines. To validate our algorithm is excellent, we compare OS²FSU with five algorithms, i.e., LOSSA[12], fast-OSFS[7], SAOLA[8], OSFASW[9], and SFS_M[10].

3) Experimental Designs. For OS²FSU and LOSSA, we default set $B_0=15$, $\lambda=0.01$, and $\eta=0.00001$. Note that fast-OSFS, SAOLA, OSFASW, and SFS_FI are used to deal with completed dataset, so we fill the missing data by zero or means. If filled zero we renamed by OSFS_Z, SAOLA_Z, OSFASW_Z, SFS_FI_Z. If filled means, we renamed by OSFS_M, SAOLA_M, OSFASW_M, SFS_FI_M.

To evaluate the classification accuracy of the selected features, we employ three classifiers KNN, SVM, and Random Forest, then take their average value as the comparison result. To obtain objective results, 5-fold cross-validation was performed. Note the testing data was not sparse, while the training data have different missing rates from 0.1 to 0.9. We conduct all experiments on a PC with Intel(R) i7-9700 3.00GHz CPU, and 16G RAM.

B. Results and Discussions of Comparison Experiments

1) Details of selected features. We evaluate the number of selected features of all the algorithms on each dataset at $\theta=0.1$. Table III presents the comparison results, where we see that OS²FSU tends to select a moderate number of features.

2) Comparison results on all datasets with $\theta=0.1$. Focus on $\theta=0.1$, the detail results of ten algorithms execution are recorded in Table IV. OS²FSU outperforms other algorithms in 6/6 datasets. Comparing the average classification accuracy of six datasets, OS²FSU is 10.29% higher than the algorithm LOSSA, and beyond 19.81% than the algorithm SAOLA_M which ranked last.

3) Comparison results on D1 with different $\theta$. We evaluate the classification accuracy on D1 when $\theta$ increases from 0.1 to 0.9. Table V shows the detail average classification accuracy of three classifiers comparison results of OS²FSU with other algorithms on D1. On all test cases, OS²FSU achieves better classification accuracy than the other algorithms.

4) Comparison results on all datasets with different $\theta$. We evaluate each algorithm’s average accuracy when $\theta$ increases from 0.1 to 0.9 on each dataset and present in Fig 3. We observe that OS²FSU achieves higher accuracy in most cases. And when $\theta$ is smaller than 0.6, the classification accuracy does not evidently decrease or even has a slight improvement, while when $\theta$ is greater than 0.6, the classification accuracy decreases rapidly, which maybe because the errors between estimated and real data become significant.

5) Statistical analysis. To effectively explore the statistical significance of our algorithms, the Wilcoxon signed-ranks test[41] will be used. Based on each algorithm’s average accuracy of three classifiers, we compare the six results of OS²FSU with other algorithms one by one by the Wilcoxon signed-ranks. Table VI presents the statistical results of $R^+$ and $R^-$, where we see that in almost all cases the values of $R$ are smaller than 2. The not significant cases are signed by $\dagger$, almost all of them are in the case of missing rate $\theta=0.9$, even so, OS²FSU still achieves a much larger $R^+$ than $R^-$, which shows that it performs slightly better than other algorithms.

VI. CONCLUSION

In this paper, we handle the uncertainty in OS²FS by incorporating fuzzy membership function and neighborhood rough set into online sparse streaming feature selection. Then, we propose an online sparse streaming feature selection with uncertainty (OS²FSU) algorithm. In OS²FSU, LFA is utilized to pre-estimate the missing data in sparse streaming features at first, then fuzzy logic and neighborhood rough set are employed to alleviate the uncertainty between estimated streaming features and labels during conducting feature selection. Compared with other state-of-the-art algorithms on six datasets, we achieve high prediction accuracy. Notably, when the missing data ratio is larger than 0.7, the performance of all the algorithms dramatically decrease. Next, we will consider the distribution of observed data to fill missing values, then further improve the OS²FS performance.
If the smaller of the two average values is greater than 2, the null-hypothesis will be accepted.

**REFERENCES**

[1] S. Lall, D. Sinha, A. Ghosh, D. Sengupta, and S. Bandyopadhyay, “Stable feature selection using copula based mutual information,” *Pattern Recognition*, vol. 112, p. e1402, 2021.

[2] X. Song, Y. Zhang, D. Gong, and X. Sun, “Feature selection using bare-bones particle swarm optimization with mutual information,” *Pattern Recognition*, vol. 112, p. 107804, 2021.

[3] W. Ji *et al.*, “Fuzzy rough sets and fuzzy rough neural networks for feature selection: A review,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 3, p. e1402, 2021.

[4] H. Zhao, P. Wang, Q. Hu, and P. Zhu, “Fuzzy Rough Set Based Feature
X. Luo, Y. Zhou, Z. Liu, L. Hu, and M. Zhou, “Generalized Nesterov’s
X. Luo, Y. Yuan, M. Zhou, Z. Liu, and M. Shang, “Non-Negative Latent
X. Luo, Y. Yuan, S. Chen, N. Zeng, and Z. Wang, “Position-Transitional
IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 1, 2021.
doi: 10.1109/TKDE.2020.3041360.
IEEE Transactions on Knowledge Discovery from Data (TKDD), vol. 11, no. 2, pp. 1–39, 2016.
IEEE Access, vol. 7, pp. 16088–16100, 2019, doi: 10.1109/ACCESS.2019.2849412.
IEEE Transactions on Services Computing, vol. 15, no. 3, pp. 1334–1344, 2022, doi: 10.1109/TSC.2020.2988760.
IEEE Transactions on Services Computing, vol. 15, no. 5, pp. 1922–1935, 2021, doi: 10.1109/TCYB.2019.2903736.
IEEE Transactions on Systems, Man, Cybernetics, Systems, vol. 16, no. 6, pp. 1922–1935, 2019, doi: 10.1109/TCB.2018.2844256.
IEEE Transactions on Cybernetics, vol. 52, no. 3, 2019, doi: 10.1109/TNNLS.2020.2990990.
IEEE Transactions on Cybernetics, vol. 11, no. 2, pp. 1–39, 2016.
IEEE Transactions on Cybernetics, vol. 16, no. 1, 2021.
IEEE Transactions on Cybernetics, vol. 11, no. 2, pp. 16088–16100, 2019, doi: 10.1109/ACCESS.2019.2849412.
IEEE Transactions on Information Forensics and Security, vol. 14, no. 10, pp. 2772–2782, 2019, doi: 10.1109/TIFS.2018.2876528.
IEEE Transactions on Services Computing, vol. 15, no. 2, pp. 793–805, 2022, doi: 10.1109/TSC.2020.2981393.
IEEE Transactions on Services Computing, vol. 16, no. 6, pp. 1922–1935, 2021, doi: 10.1109/TCYB.2019.2903736.
Advancing Non-Negative Latent Factorization of Tensors With Diversified Regularization Schemes,” IEEE Transactions on Services Computing, vol. 15, no. 3, pp. 1334–1344, 2022, doi: 10.1109/TSC.2020.2988760.
IEEE Transactions on Services Computing, vol. 15, no. 2, pp. 793–805, 2022, doi: 10.1109/TSC.2019.2961895.
A Deep Latent Factor Model for High-Dimensional and Sparse Matrices in Recommender Systems,” IEEE Transactions on System, Man, and Cybernetics: Systems, vol. 51, no. 7, pp. 4285–4296, 2021, doi: 10.1109/TCSMC.2019.2931393.
A novel streamlining feature selection method,” Pattern Recognition, vol. 86, pp. 48–61, Feb. 2019, doi: 10.1016/j.patcog.2018.08.009.
P. Zhou, X. Hu, P. Li, and X. Wu, “OF5-Density: a novel online streaming
feature selection method,” Pattern Recognition, vol. 86, no. 10, pp. 1891–1903, Oct. 2019, doi: 10.1109/TFUZZ.2019.2892349.
X. Luo, W. Wang, D. Ying, Y. Qian, and J. Xu, “Feature Selection Using Fuzzy Neighborhood Entropy-Based Uncertainty Measures for Fuzzy Neighborhood Multigranulation Rough Sets,” IEEE Trans. Fuzzy Syst., vol. 23, no. 1, pp. 19–33, Jan. 2015, doi: 10.1109/TFUZZ.2014.2890998.
X. Yan and M. Jia, “Intelligent fault diagnosis of rotating machinery using improved multiscale dispersion entropy and mKMR feature selection,” Knowledge-Based Systems, vol. 163, pp. 450–471, 2019.
D. Wu, K. Yu, W. Ding, H. Wang, and X. Zhu, “Online Feature Selection with Streaming Features,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 5, pp. 1178–1192, 2013, doi: 10.1109/TPAMI.2012.197.
X. Li, Y. Wang, and R. Ruiz, “A Survey on Sparse Learning Models for Feature Selection,” IEEE Transactions on Cybernetics, vol. 52, no. 3, pp. 1642–1660, 2022, doi: 10.1109/TCYB.202982445.
D. Wu, Y. He, X. Luo, and M. Zhou, “A Latent Factor Analysis-Based Approach to Online Sparse Streaming Feature Selection,” IEEE Trans. Syst., Man Cybern., Syst., pp. 1–15, 2021, doi: 10.1109/TSMC.2021.3096065.
D. Wu, Q. He, X. Luo, and M. Shang, “An L₁-and-L₂-Norm-Oriented Neighborhood-Regularized Latent Factor Model for Highly Accurate Web Service QoS Prediction,” IEEE Transactions on Services Computing, pp. 1–1, 2022, doi: 10.1109/TSC.2022.3178543.
D. Wu, M. Shang, X. Luo, and Z. Wang, “An L-and-L-Norm-Oriented Latent Factor Model for Recommender Systems,” IEEE Trans. Neural Netw. Learning Syst., pp. 1–14, 2021, doi: 10.1109/TNNLS.2020.3071392.
IEEE Transactions on Cybernetics, vol. 52, no. 3, 2019, doi: 10.1109/TNNLS.2020.2990990.
IEEE Transactions on Service Computing, vol. 51, no. 6, pp. 3522–3532, 2019.
IEEE Transactions on Systems, Man, Cybernetics - Systems, vol. 51, no. 6, pp. 3522–3532, 2019.
IEEE Access, vol. 8, no. 8, pp. 4612–4623, 2021, doi: 10.1109/ACCESS.2020.2931468.
IEEE Transactions on Big Data, vol. 8, no. 2, pp. 420–431, 2022, doi: 10.1109/TBDATA.2020.2973141.
IEEE Transactions on Cybernetics, vol. 34, no. 8, pp. 3958–3970, 2022, doi: 10.1109/TCYB.2020.3033324.
IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 8, pp. 3958–3970, 2022, doi: 10.1109/TKDE.2020.3033324.
IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 10, pp. 1891–1903, Oct. 2019, doi: 10.1109/TFUZZ.2019.2892349.
IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 5, pp. 1178–1192, 2013, doi: 10.1109/TPAMI.2012.197.
IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 5, pp. 1178–1192, 2013, doi: 10.1109/TPAMI.2012.197.
IEEE Transactions on Services Computing, vol. 15, no. 3, pp. 1642–1660, 2022, doi: 10.1109/TCYB.202982445.
IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 5, pp. 1178–1192, 2013, doi: 10.1109/TPAMI.2012.197.
IEEE Transactions on Cybernetics, vol. 52, no. 3, 2019, doi: 10.1109/TNNLS.2020.2990990.
IEEE Transactions on Cybernetics, vol. 34, no. 6, pp. 2525–2538, 2022, doi: 10.1109/TKDE.2020.3014302.
IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 6, pp. 2525–2538, 2022, doi: 10.1109/TKDE.2020.3014302.
IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 6, pp. 2525–2538, 2022, doi: 10.1109/TKDE.2020.3014302.