Incremental Feature Spaces Learning with Label Scarcity

SHILIN GU, The College of Liberal Arts and Science, National University of Defense Technology, China
YUHUA QIAN, The Institute of Big Data Science and Industry, Shanxi University, China
CHENPING HOU, The College of Liberal Arts and Science, National University of Defense Technology, China

Recently, learning and mining from data streams with incremental feature spaces have attracted extensive attention, where data may dynamically expand over time in both volume and feature dimensions. Existing approaches usually assume that the incoming instances can always receive true labels. However, in many real-world applications, e.g., environment monitoring, acquiring the true labels is costly due to the need of human effort in annotating the data. To tackle this problem, we propose a novel incremental Feature spaces Learning with Label Scarcity algorithm (FLLS), together with its two variants. When data streams arrive with augmented features, we first leverage the margin-based online active learning to select valuable instances to be labeled and thus build superior predictive models with minimal supervision. After receiving the labels, we combine the online passive-aggressive update rule and margin-maximum principle to jointly update the dynamic classifier in the shared and augmented feature space. Finally, we use the projected truncation technique to build a sparse but efficient model. We theoretically analyze the error bounds of FLLS and its two variants. Also, we conduct experiments on synthetic data and real-world applications to further validate the effectiveness of our proposed algorithms.

CCS Concepts: • **Computing methodologies** → **Online learning settings**; • **Theory of computation** → **Streaming models**.

Additional Key Words and Phrases: Incremental feature spaces, online learning, label scarcity

ACM Reference Format:
Shilin Gu, Yuhua Qian, and Chenping Hou. 2021. Incremental Feature Spaces Learning with Label Scarcity. 1, 1 (February 2021), 26 pages.

1 INTRODUCTION

In many real-world applications, data are usually accumulated over time and collected from open and dynamic environments[1], leading to the simultaneous increase of data volume and feature space. We refer this type of data as data streams with incremental feature spaces, or trapezoidal data streams[2]. There are a few online learning approaches that have been explored to learn from trapezoidal data streams[2–5]. Nonetheless, they all assume that the labels of data streams can always be received, which does not always hold in practice. For example, as shown in Fig. 1, to monitor environmental quality, different types of sensors are continuously installed, the volume and feature dimension of data simultaneously increase due to the continuously installed sensors, and each instance needs to be labeled by experts,

Chenping Hou and Yuhua Qian are the corresponding authors.
Authors' addresses: Shilin Gu, gslnudt@outlook.com, The College of Liberal Arts and Science, National University of Defense Technology, Changsha, Hunan, China, 410073; Yuhua Qian, jinchengqyh@126.com, The Institute of Big Data Science and Industry, Shanxi University, Taiyuan, Shanxi, China, 030006; Chenping Hou, hcpnudt@hotmail.com, The College of Liberal Arts and Science, National University of Defense Technology, Changsha, Hunan, China, 410073.

Manuscript submitted to ACM

which results in the expensive costs and scarcity of labels. Besides, it's not always informative or necessary to query every instance's label in many cases, e.g., if the model correctly predicts the instance's label with a high confidence[6].

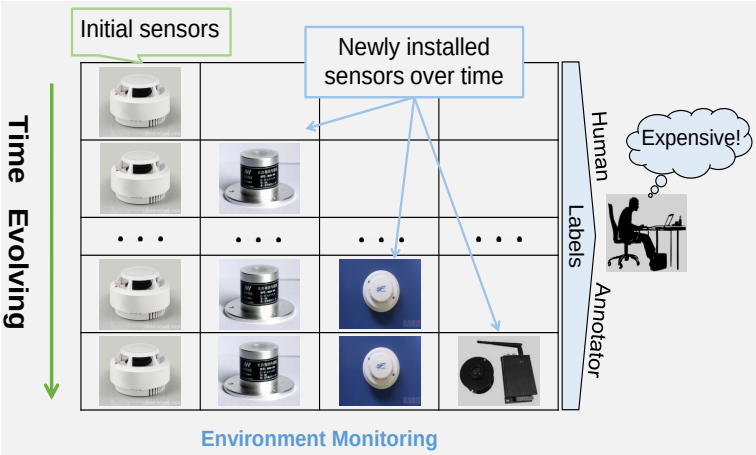

Fig. 1. Illustration of the setting arising from a real application. As time evolves, different types of sensors are installed to monitor environmental quality. Each row represents the instance composed of features that are collected by all current sensors. Therefore, the volume and feature dimension of data streams simultaneously increase over time. Besides, each instance needs to be labeled by experts, resulting in expensive costs and scarcity of labels.

Trapezoidal data streams learning with scarce labels faces much more difficulties than current online learning problems because the three important aspects of data: volume, features, and labels, are all different from traditional learning settings at the same time. There are two main challenges, one is how to lift the restriction that all instances must be labeled and how to effectively extract useful information from the very limited labeled instances to build a superior classifier; the other one is how to design a model update strategy with high dynamicity such that the model can adapt to the continuous expansion of the feature space and mine useful information. For the first challenge, a line of research, which we call online active learning and online semi-supervised learning, has been explored to learn from data streams with scarce labels[6–11]. For the second challenge, another line of research, which aims to learn data streams in dynamic feature spaces[12–16], has been explored to relax the constraint that the feature space of all arriving instances must be fixed. Unfortunately, the above two lines of research cannot be directly applied to handle trapezoidal data streams with label scarcity because their model update criteria are designed either in the case of scarce labels or in the case of dynamic feature spaces, without considering both at the same time. The goal of this paper is to fill this critical gap.

A simple and straightforward approach is to take advantage of the newly obtained labeled instance and learn a new model for classification. However, this approach may have two deficiencies. First, the newly coming data with the same feature space are usually scarce, which might be insufficient to build a superior predictive model. Second, the newly built model ignores the previously collected data, which can not fully exploit the information from historical data streams.

To solve the above two deficiencies, we propose a new *incremental Feature spaces Learning with Label Scarcity* algorithm (FLLS), together with its two variants FLLS-I and FLLS-II. We aim to effectively exploit the information of data streams with different feature dimensions and build a powerful predictive model with minimal supervision. Specifically,

when data streams arrive with augmented features, we first determine whether to obtain the current instance's label from an oracle based on the margin-based online active learning strategy, i.e., the probability of actively querying the label of current instance is inversely proportional to the predictive confidence of the current model. If the label is not queried, the model keeps unchanged; otherwise, the model jointly updates the dynamic classifier in shared and augmented feature space by combining the passive-aggressive update rule and margin-maximum principle. Finally, we use the projected truncation technique to build a sparse but efficient model. Theoretical and empirical studies validate the effectiveness of our proposed algorithms.

It is worthwhile to summarize the main contributions of the proposed approach as follows.

(1) We propose a new algorithm FLLS and its two variants to handle trapezoidal data streams with label scarcity, where the volume and feature dimension of data streams simultaneously increase, and the labels of data streams need to be actively acquired from an oracle. To the best of our knowledge, it may be the first work that is specially designed for this kind of data streams.
(2) We design a novel strategy to learn a highly dynamic classification model from trapezoidal data streams with minimal supervision. Besides, we theoretically analyze the error bounds of our proposed algorithms.
(3) Extensive experiments on both synthetic datasets and real-world applications validate the effectiveness of the proposed algorithms.

The remainder of this paper is organized as follows. Section II introduces related work. The proposed algorithms are presented in Section III. Section IV provides a corresponding theoretical analysis. The experimental results are reported in Section V. The conclusion is in Section VI.

2 RELATED WORK

In this paper, we aim to learn a classification model from trapezoidal data streams with scarce labels, which is mainly related to online learning from dynamic feature space and online learning with label scarcity. We mainly review the related work of this paper from the above two aspects.

2.1 Online Learning from dynamic Feature Space

The dynamic feature space means that the feature space of data streams keeps changing. The most relevant to our work is trapezoidal data streams learning, where the volume and feature dimension of data simultaneously increase. Zhang[2] is the first to deal with trapezoidal data streams. She proposed OL_{SF} with its two variants. Specifically, Zhang first divides the features of the current training instance into historical features and new features. Then a classifier updates historical features and new features by following different update rules. Recently, a few works have been proposed to handle feature evolvable data streams, where features would vanish or occur over time. For example, Hou[3] proposed the FESL algorithm, it first recovers historical features by a mapping function learned in the period where both historical features and new features exist, then it learns two models from features of the above two parts, respectively. Finally, ensemble learning is used to make the final prediction. Based on FESL, literature[1, 12] conducted in-depth exploration and expansion of FESL scenario and proposed EDM and PUFE algorithms respectively. EDM[1] handles data streams with evolving distribution and feature space by utilizing a discrepancy measure, and presents the generalization error analysis. PUFE[12] leverages an online matrix completion technique to deal with the case where historical features would vanish unpredictably. In addition, there are two methods OLVF[17] and GLSC[18] studying

more complex dynamic changes of feature space. They assume that the features of arriving instances can arbitrarily occur or vanish.

2.2 Online Learning with Label Scarcity

Existing works of online learning with label scarcity can be divided into two groups[6]. The first group is online semi-supervised learning. The core is to fully exploit the continuously received unlabeled and labeled data for learning tasks. Manifold regularization[19] is a major framework in online semi-supervised learning (e.g.,Riemann manifolds[20–22]), it simultaneously minimizes the prediction loss on the labeled data and the prediction differences on the unlabeled data [6]. In other words, instances located in a neighborhood region tend to be classified into the same class, which leads to the propagating of label information.

The second group is online active learning and our approach belongs to this group. At each round, when a new unlabeled instance arrives, the online active learning algorithms first decide whether to query the instance's label by certain criteria; once the label is queried, the learning algorithms can update the current model by leveraging the labeled instance; otherwise, the model will not be updated[6]. The current literature on online active learning contains two mainstream solutions. The first solution is "selective sampling", which adapts current classical online learning approaches for active learning by drawing a Bernoulli random variable[23–26]. The second solution is "learning with experts' advices"[27–29], where the model contains a set of experts, and it queries the instance's label based on the degree of discrepancy advised by these experts[30]. Both of the above solutions reveal that the instance's label is only queried when it meets certain conditions, for example, the predictive confidence of the model is below some threshold.

However, all these above algorithms either assume that all the instances arrived are fully labeled, or assume that the feature space of entire data streams is fixed. Recently, only two works, AGDES[31] and SF^{2}EL[13], are developed to support both. They belong to online semi-supervised learning and both of them simultaneously handle data streams with variable feature spaces under incomplete supervision. Different from our setting where whether the current instance can receive a label is determined by the current instance and the classification model updated in the previous round, AGDES and SF^{2}EL assume that whether the current instance has a label is determined in advance. Thus, the technical challenges and solutions are different.

3 OUR PROPOSED APPROACH

3.1 Notations

We consider a task of binary classification on data streams with incremental feature spaces and scarce labels. Let $\{\mathbf{x}_t|t = 1, \ldots, T\}$ denote an input training sequence, where $\mathbf{x}_t \in \mathbb{R}^{d_t}$ represents the instance vector of d_t dimensions, $d_{t-1} \leq d_t$. $\mathbf{w}_t \in \mathbb{R}^{d_{t-1}}$ denotes the classifier built at round $t-1$. $\mathbf{w}_1 \in \mathbb{R}^{d_1}$ is a vector with all elements being zero. As illustrated in Fig. 2, at each round t, when an instance $\mathbf{x}_t$ carrying new features arrives, we divide the feature space into two groups: shared and augmented features. The shared features contain the same features as the instance $\mathbf{x}_{t-1}$ at round $t-1$, the augmented features are only contained by the current instance $\mathbf{x}_t$. We denote the projection of an instance at round t onto the shared feature space as $\mathbf{x}_t^s$ and augmented feature space as $\mathbf{x}_t^a$, i.e., $\mathbf{x}_t = [\mathbf{x}_t^s, \mathbf{x}_t^a]$. It also applies for the projections of other vectors such as classifier $[\mathbf{w}_t^s, \mathbf{w}_t^a]$.

To make clear description, we denote by $\Pi_{\mathbf{x}_t}\mathbf{x}_{t+1}$ a vector of d_t dimension, it consists of elements of $\mathbf{x}_{t+1}$ which are in the same feature space of $\mathbf{x}_t$. Similarly, we denote by $\Pi_{\neg\mathbf{x}_t}\mathbf{x}_{t+1}$ a vector of $d_{t+1} - d_t$ dimension, it consists

of elements of $\mathbf{x}_{t+1}$ which are not in the feature space of $\mathbf{x}_t$. Therefore, we can derive that $\mathbf{x}_t^s = \Pi_{\mathbf{x}_{t-1}}\mathbf{x}_t = \Pi_{\mathbf{w}_t}\mathbf{x}_t$, $\mathbf{x}_t^a = \Pi_{\neg\mathbf{x}_{t-1}}\mathbf{x}_t = \Pi_{\neg\mathbf{w}_t}\mathbf{x}_t$, $\mathbf{w}_{t+1}^s = \Pi_{\mathbf{w}_t}\mathbf{w}_{t+1}$, $\mathbf{w}_{t+1}^a = \Pi_{\neg\mathbf{w}_t}\mathbf{w}_{t+1}$.

The notations are listed in Table 1, we explain their meanings when they are first used.

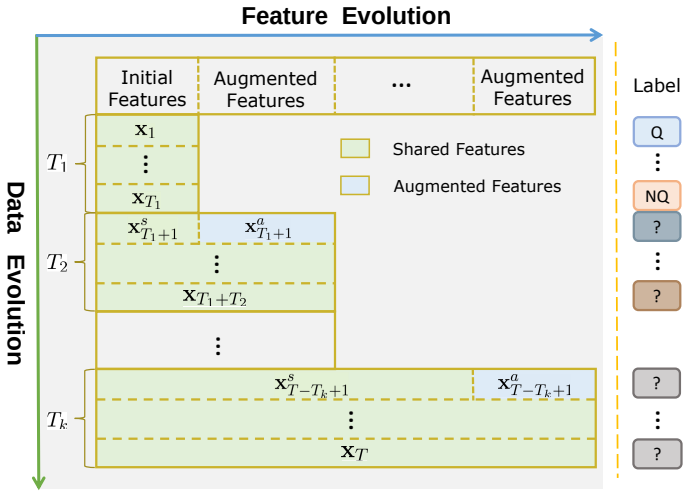

Fig. 2. FLLS notations. As time evolves, features and instances dynamically expand. The green and blue areas represent the shared features and the augmented features, respectively. $\mathbf{x}_t^s$ and $\mathbf{x}_t^a$ represents the projection of instance $\mathbf{x}_t$ in the shared and augmented feature space, respectively, $t = 1, \ldots, T$. "Q" means query the label, "NQ" means not query the label.

3.2 The Proposed Algorithm

We develop a new algorithm FLLS and its two variants in this section to handle data streams with incremental feature spaces and scarce labels. The difference between FLLS, FLLS-I, and FLLS-II is the model update strategies. Therefore, we first introduce the basic algorithm FLLS.

For instance $\mathbf{x}_t$ arrived at round t, we first compute its prediction margin by the current classifier $\mathbf{w}_t$. Since the feature space of data streams keeps expanding, the dimension of $\mathbf{x}_t$ and $\mathbf{w}_t$ may be different. Thus, we redefine the prediction margin as follows,

$$q_t = \mathbf{w}_t \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t = \mathbf{w}_t \cdot \mathbf{x}_t^s, \tag{1}$$

where q_t can be regarded as the predictive confidence. Whether to query the label of $\mathbf{x}_t$ is decided by $\delta_t \in \{0, 1\}$, which is a Bernoulli random variable, the probability of $\delta_t = 1$ is $\rho/(\rho + |q_t|)$, $\rho \geq 1$ is a smoothing parameter. The idea of the above strategy is inspired by margin-based active learning, we decide whether to query the label of an instance according to its importance in building a classification model. This importance is determined by the prediction margin of the current classifier, i.e., q_t. However, in online active learning, we may face the problem that the current classifier is unreliable, especially in the first several epochs. There are not enough instances to train a reliable classifier since the instances come in an online way. If we directly decide to query the label based on q_t, we may make a wrong decision. To alleviate the above problem, similar to the strategies in traditional works[6, 9, 24, 25], we also introduce the Bernoulli random variable $\delta_t \in \{0, 1\}$ into our model. It can shake our decision with a certain probability. Even when the current

Table 1. Notations

Notations	Descriptions	Notations	Descriptions
$\mathbf{w}_t$	$\mathbf{w}_t \in \mathbb{R}^{d_{t-1}}$, $\mathbf{w}_1 \in \mathbb{R}^{d_1}$, classifier built at round $t-1$	$\mathbf{w}_{t+1}^s$	A vector consisting of elements which are owned by both $\mathbf{w}_{t+1}$ and $\mathbf{w}_t$
C	$C > 0$, penalty cost parameter, tradeoff in the optimization problem of FLLS-I and FLLS-II	$\mathbf{w}_{t+1}^a$	A vector consisting of elements of $\mathbf{w}_{t+1}$ that are beyond the feature space of $\mathbf{w}_t$
ℓ_t^*	hinge loss on instance $(\mathbf{x}_t, y_t)$ based on the classifier $\mathbf{u} \in \mathbb{R}^{d_T}$	$\bar{\mathbf{w}}_{t+1}$	intermediate variable of $\mathbf{w}_{t+1}$ after the update operation
$\mathbf{x}_t^s$	A vector consisting of elements which are owned by both $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$	$\hat{\mathbf{w}}_{t+1}$	intermediate variable of $\mathbf{w}_{t+1}$ on the L_1 ball without truncation
$\mathbf{x}_t^a$	A vector consisting of elements of $\mathbf{x}_t$ that are beyond the feature space of $\mathbf{x}_{t-1}$	$\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}$	A vector which consists of elements of $\mathbf{u}$ that are in the feature space of $\mathbf{w}_{t+1}$ but not $\mathbf{w}_t$
τ_t	learning rate variable	$\Pi_{\mathbf{w}_t}\mathbf{w}_{t+1}$	Equivalent to the definition of $\mathbf{w}_{t+1}^s$
T	$T \in N^+$, total number of obtained instances	$\Pi_{\neg\mathbf{w}_t}\mathbf{w}_{t+1}$	Equivalent to the definition of $\mathbf{w}_{t+1}^a$
ℓ_t	hinge loss on instance $(\mathbf{x}_t, y_t)$	$\mathbf{x}_t$	$\mathbf{x}_t \in \mathbb{R}^{d_t}$, the instance obtained on round t
$\mathbf{u}$	$\mathbf{u} \in \mathbb{R}^{d_T}$, arbitrary vector in $\mathbb{R}^{d_T}$	d_t	$d_t \le d_{t+1}$, dimension of instance $\mathbf{x}_t$
δ_t	$\delta_t \in \{0, 1\}$, Bernoulli random variable	$d_{\mathbf{w}_t}$	dimension of $\mathbf{w}_t$
ξ	slack variable	y_t	$y_t \in \{-1, +1\}$, true label of $\mathbf{x}_t$
B	$B \in (0, 1]$, ratio of reserved features	$\hat{y}_t$	$\hat{y}_t \in \{-1, +1\}$, predicted label of $\mathbf{x}_t$

classifier makes a wrong prediction, we may still make the right choice due to the randomness. Moreover, using the Bernoulli random variable has been proven effective and reliable in many real-world applications, such as personalized recommendation, medical diagnosis, and malicious URL detection in literatures [6, 9, 24, 25].

There are two cases for δ_t, if $\delta_t = 0$, the algorithm FLLS will not ask oracle for querying the instance label and thus the model keeps unchanged; if $\delta_t = 1$, the label of $\mathbf{x}_t$ will be revealed by an oracle and FLLS will suffer an instantaneous prediction loss $\ell_t(\mathbf{w}_t, (\mathbf{x}_t, y_t))$, where $y_t \in \{-1, +1\}$. Then it's able to exploit the potential of the newly achieved instance-label pair $(\mathbf{x}_t, y_t)$ to update the classification model.

We choose the hinge loss as the loss function, i.e., $\ell(\mathbf{w}, (\mathbf{x}_t, y_t)) = \max\{0, 1 - y_t(\mathbf{w} \cdot \mathbf{x}_t)\}$, where the dimension of $\mathbf{w}$ and $\mathbf{x}_t$ are the same. The main reasons why we chose the hinge loss are as follows. First, hinge loss is widely used in many literatures. It is specifically designed to solve the binary classification problem, which fits well with our binary classification setting. Second, in optimization, the application of hinge loss makes the updating of our models simple and effective. Finally, hinge loss can make the proposed algorithms have nice theoretical properties, which will be stated in detail in Section 4. Particularly, the instantaneous prediction loss $\ell_t(\mathbf{w}_t, (\mathbf{x}_t, y_t))$ is defined as

$$\ell_t = \ell_t(\mathbf{w}_t, (\mathbf{x}_t, y_t)) = \max\{0, 1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t^s)\}. \tag{2}$$

To update $\mathbf{w}_t$ with $(\mathbf{x}_t, y_t)$, we adopt the online passive-aggressive learning idea proposed in [32] and [2]. Specifically, at round t, the newly updated classifier $\mathbf{w}_{t+1} \in \mathbb{R}^{d_t}$ will be composed of two parts, which can be written as $\mathbf{w}_{t+1} = [\mathbf{w}_{t+1}^s, \mathbf{w}_{t+1}^a]$, where $\mathbf{w}_{t+1}^s \in \mathbb{R}^{d_{t-1}}$ and $\mathbf{w}_{t+1}^a \in \mathbb{R}^{d_t - d_{t-1}}$. The role of $\mathbf{w}_{t+1}^s$ is to update shared features and inherit information from $\mathbf{w}_t$, and the role of $\mathbf{w}_{t+1}^a$ is to update augmented features. Therefore, the update of $\mathbf{w}_t$ can be transformed into the update of $\mathbf{w}_{t+1}^s$ and $\mathbf{w}_{t+1}^a$ by optimizing the problem in (3)

$$\begin{aligned} \mathbf{w}_{t+1} = & \underset{\mathbf{w}=[\mathbf{w}^s, \mathbf{w}^a]}{\arg\min} \frac{1}{2}\left\|\mathbf{w}^s - \mathbf{w}_t\right\|^2 + \frac{1}{2}\left\|\mathbf{w}^a\right\|^2, \\ s.t. \quad & \ell(\mathbf{w}, (\mathbf{x}_t, y_t)) = 0. \end{aligned} \tag{3}$$

where $\mathbf{w} \in \mathbb{R}^{d_t}$, $\ell(\mathbf{w}, (\mathbf{x}_t, y_t))$ is the prediction loss of $\mathbf{w}$ on $\mathbf{x}_t$, which is,

$$\ell(\mathbf{w}, (\mathbf{x}_t, y_t)) = \max\{0, 1 - y_t(\mathbf{w}^s \cdot \mathbf{x}_t^s) - y_t(\mathbf{w}^a \cdot \mathbf{x}_t^a)\}. \tag{4}$$

From Eq. (3) and (4) we can derive that the solution to the above problem is the projection of $\mathbf{w}$ onto the set of all weight vectors that obtain a loss of zero. On one hand, if the label prediction of existing classifier $\mathbf{w}_t$ for the current instance $\mathbf{x}_t$ is correct, i.e., $\ell_t = \max\{0, 1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t^s)\} = 0$, the resulting algorithm is *passive*, that is, $\mathbf{w}^s = \mathbf{w}_t$, $\mathbf{w}^a = (0, \ldots, 0)$, and $\mathbf{w}_{t+1} = [\mathbf{w}_t, 0, \ldots, 0]$.

On the other hand, if the label prediction of $\mathbf{w}_t$ for the instance $\mathbf{x}_t$ is incorrect, i.e., $\ell_t = \max\{0, 1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t^s)\} > 0$. The algorithm *aggressively* updates classifier to meet the constraint in Eq. (3). It attempts to keep $\mathbf{w}_{t+1}^s$ as close to $\mathbf{w}_t$ as possible to acquire information from $\mathbf{w}_t$ and let the norm of $\mathbf{w}_{t+1}^a$ be as small as possible to avoid over-fitting. The Lagrangian of (3) is

$$\mathcal{L}(\mathbf{w}, \tau) = \frac{1}{2}\left\|\mathbf{w}^s - \mathbf{w}_t\right\|^2 + \frac{1}{2}\left\|\mathbf{w}^a\right\|^2 + \tau(1 - y_t(\mathbf{w}^s \cdot \mathbf{x}_t^s) - y_t(\mathbf{w}^a \cdot \mathbf{x}_t^a)). \tag{5}$$

where $\tau \geq 0$ is a Lagrange multiplier. If we take the derivative of $\mathcal{L}$ with respect to $\mathbf{w}^s$ and $\mathbf{w}^a$, and set them to zero, we can achieve the following results

$$\begin{aligned} &\nabla_{\mathbf{w}^s}(\mathcal{L}) = \mathbf{w}^s - \mathbf{w}_t - \tau y_t \mathbf{x}_t^s = 0 \quad \Rightarrow \quad \mathbf{w}^s = \mathbf{w}_t + \tau y_t \mathbf{x}_t^s, \\ &\nabla_{\mathbf{w}^a}(\mathcal{L}) = \mathbf{w}^a - \tau y_t \mathbf{x}_t^a = 0 \quad \Rightarrow \quad \mathbf{w}^a = \tau y_t \mathbf{x}_t^a. \end{aligned} \tag{6}$$

To get the value of τ, we introduce the KKT conditions[33]. Since $\mathbf{w}^s = \mathbf{w}_t + \tau y_t \mathbf{x}_t^s$, $\mathbf{w}^a = \tau y_t \mathbf{x}_t^a$, we plug these two equations into Eq. (5) and take the derivative of $\mathcal{L}(\tau)$ with respect to τ and set it to zero, we can achieve

$$\begin{aligned} &\mathcal{L}(\tau) = -\frac{1}{2}\tau^2\left\|\mathbf{x}_t^s\right\|^2 - \frac{1}{2}\tau^2\left\|\mathbf{x}_t^a\right\|^2 + \tau - \tau y_t(\mathbf{w}_t \cdot \mathbf{x}_t^s), \\ &\nabla_\tau(\mathcal{L}) = 0 \quad \Rightarrow \quad \tau_t = \frac{1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t^s)}{\left\|\mathbf{x}_t^s\right\|^2 + \left\|\mathbf{x}_t^a\right\|^2} = \frac{\ell_t}{\|\mathbf{x}_t\|^2}. \end{aligned} \tag{7}$$

Thus, the general update strategy of FLLS is $\mathbf{w}_{t+1} = [\mathbf{w}_t + \tau_t y_t \mathbf{x}_t^s, \tau_t y_t \mathbf{x}_t^a]$, where $\tau_t = \ell_t / \|\mathbf{x}_t\|^2$.

Eq. (3) needs the newly updated model to correctly predict the label of the current instance, which makes the model of FLLS rigorous and sensitive to noise[34]. To solve this shortage, we adopt the soft-margin strategy and introduce a slack variable ξ into Eq. (3), and then propose two variants of the FLLS algorithm, i.e., FLLS-I, FLLS-II. The objective function of FLLS-I is,

$$\begin{aligned} &\mathbf{w}_{t+1} = \underset{\mathbf{w}=[\mathbf{w}^s, \mathbf{w}^a]}{\arg\min} \frac{1}{2}\left\|\mathbf{w}^s - \mathbf{w}_t\right\|^2 + \frac{1}{2}\left\|\mathbf{w}^a\right\|^2 + C\xi, \\ &s.t. \quad \ell(\mathbf{w}, (\mathbf{x}_t, y_t)) \leq \xi, \xi \geq 0. \end{aligned} \tag{8}$$

where $C > 0$ is a tradeoff between rigidness and slackness. The update step will become more rigid as the value of C increases. The objective function of FLLS-II scales quadratically with ξ,

$$\begin{aligned} &\mathbf{w}_{t+1} = \underset{\mathbf{w}=[\mathbf{w}^s, \mathbf{w}^a]}{\arg\min} \frac{1}{2}\left\|\mathbf{w}^s - \mathbf{w}_t\right\|^2 + \frac{1}{2}\left\|\mathbf{w}^a\right\|^2 + C\xi^2, \\ &s.t. \quad \ell(\mathbf{w}, (\mathbf{x}_t, y_t)) \leq \xi, \xi \geq 0. \end{aligned} \tag{9}$$

Due to the space constraints, we omit the details of the optimization of FLLS-I and FLLS-II since they are similar to FLLS. FLLS-I and FLLS-II have the same closed-form of update strategy as FLLS, i.e., $\mathbf{w}_{t+1} = [\mathbf{w}_t + \tau_t y_t \mathbf{x}_t^s, \tau_t y_t \mathbf{x}_t^a]$. In

general, the step size τ_t of the three proposed methods are:

$$\tau_t = \begin{cases} \ell_t / \|\mathbf{x}_t\|^2 & \text{(FLLS)} \\ \min\left(C, \ell_t / \|\mathbf{x}_t\|^2\right) & \text{(FLLS - I)} \\ \ell_t / \left(\|\mathbf{x}_t\|^2 + 1/(2C)\right). & \text{(FLLS - II)} \end{cases} \tag{10}$$

3.3 The Sparsity Strategy

As the feature space of data keeps expanding, the dimension of $\mathbf{w}_t$ will increase rapidly. To control the maximum dimension and improve the memory usage and running efficiency, once the model is updated, we further conduct projection and truncation on this model based on $B \in [0, 1]$ to filter out redundant features. Here we stipulate that only at most a ratio of B elements of classifier $\mathbf{w}_t \in \mathbb{R}^{d_t}$ are nonzero, which means $\|\mathbf{w}_t\|_0 \leq B \cdot d_t$. [1] Since $B \cdot d_t$ is usually not an integer, we use $floor(B \cdot d_t)$[2] instead of $B \cdot d_t$.

Algorithm 1 The proposed algorithm FLLS and its two variants FLLS-I and FLLS-II

Input: smoothing parameter $\rho \geq 1$, penalty parameter $C > 0$, regularization parameter $\lambda > 0$, ratio of selected features $B \in (0, 1]$;
Initialization: $\mathbf{w}_1 = (0, \ldots, 0) \in \mathbb{R}^{d_1}$;
for $t = 1, 2, \ldots, T$ **do**
  receive $\mathbf{x}_t \in \mathbb{R}^{d_t}$;
  set $q_t = \mathbf{w}_t \cdot \Pi_{\mathbf{w}_t} \mathbf{x}_t$, predict $\hat{y}_t = sign(q_t)$;
  draw a Bernoulli random variable $\delta_t \in \{0, 1\}$ of probability $\rho/(\rho + |q_t|)$;
  if $\delta_t = 1$ **then**
    query the label of $\mathbf{x}_t$: $y_t \in \{-1, +1\}$;
    suffer loss $\ell_t(\mathbf{w}_t) = \max\{0, 1 - y_t(\mathbf{w}_t \cdot \Pi_{\mathbf{w}_t} \mathbf{x}_t)\}$;
    model update
      compute τ_t according to Eq. (10);
      update model: $\bar{\mathbf{w}}_{t+1} = [\mathbf{w}_t + \tau_t y_t \mathbf{x}_t^s, \tau_t y_t \mathbf{x}_t^a]$;
    model sparse
      project $\bar{\mathbf{w}}_{t+1}$ to a L_1 ball according to Eq. (11);
      truncate $\hat{\mathbf{w}}_{t+1}$ and obtain $\mathbf{w}_{t+1}$:
      $\mathbf{w}_{t+1} = truncate(\hat{\mathbf{w}}_{t+1}, B)$ (See Algorithm 2);
  else
    $\mathbf{w}_{t+1} = \mathbf{w}_t$;
  end if
end for

If we directly select the smallest weights from the classifier $\bar{\mathbf{w}}_t$ and set them to zero, it may perform poorly because the result of the dot product is suddenly changed, besides, we can not be sure that the numerical values of the ignored features are small enough. Therefore, we need a projection step before truncation. Usually, we project the classifier to an L_1 ball, through this way, most numerical values of classifier $\bar{\mathbf{w}}_t$ are concentrated to the largest elements, so setting the smallest weights to zero will not result in a sudden change. Here is the projection,

$$\hat{\mathbf{w}}_{t+1} = \min\left\{1, \frac{\lambda}{\|\bar{\mathbf{w}}_{t+1}\|_1}\right\} \bar{\mathbf{w}}_{t+1}, \tag{11}$$

[1] $\|\mathbf{a}\|_0$ is equal to the number of non-zero elements in the vector $\mathbf{a}$.
[2] $floor(a)$ is equal to the largest integer smaller than a.

Algorithm 2 $\mathbf{w} = truncate(\hat{\mathbf{w}}, B)$

1: **Input**: $\hat{\mathbf{w}} \in \mathbb{R}^{d_{\hat{\mathbf{w}}}}$, $B \in (0, 1]$.
2: **if** $\|\hat{\mathbf{w}}\|_0 \geq B \cdot d_{\hat{\mathbf{w}}}$ **then**
3: retain $\max\{1, floor(B \cdot d_{\hat{\mathbf{w}}})\}$ largest elements in $\hat{\mathbf{w}}$; set the rest elements in $\hat{\mathbf{w}}$ to zero. We denote by $\hat{\mathbf{w}}^B$ the corresponding vector. Then $\mathbf{w} = \hat{\mathbf{w}}^B$
4: **else**
5: $\mathbf{w} = \hat{\mathbf{w}}$
6: **end if**

where λ is a positive regularization parameter. With a ratio of B, we truncate the smallest elements from the classifier $\hat{\mathbf{w}}_{t+1}$ and obtain the final classifier $\mathbf{w}_{t+1}$. This strategy helps to truncate the redundant features. The overall procedure of our methods is given in Algorithm 1 and Algorithm 2.

4 THEORETICAL ANALYSIS

We analyze the error bounds of FLLS and its two variants theoretically. In the following analysis, four theorems discuss the upper error bounds of the proposed algorithms in the linear separable and non-separable cases, and the lemma is the crux to prove the above four theorems.

For clarity, we use the notations: $\mathcal{F} = \{t | t \in [T], \hat{y}_t \neq y_t\}$, and $\mathcal{G} = \{t | t \in [T], \hat{y}_t = y_t, \ell_t(\mathbf{w}_t; (\mathbf{x}_t, y_t)) > 0\}$. where $[T]$ denotes $\{1, 2, \ldots, T\}$.

Lemma 1. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of obtained instances, where $\mathbf{x}_t \in \mathbb{R}^{d_t}$, $d_{t-1} \leq d_t$, $y_t \in \{-1, +1\}$, $t = 1, \ldots, T$. Let τ_t be the step size for FLLS, FLLS-I and FLLS-II as given in Eq. (10). The following bound holds for any $\mathbf{u} \in \mathbb{R}^{d_T}$*

$$\sum_{t=1}^{T} 2\delta_t \tau_t \left[G_t\left(\theta - |q_t|\right) + F_t\left(\theta + |q_t|\right)\right] \leq \theta^2 \|\mathbf{u}\|^2 + \sum_{t=1}^{T} \tau_t^2 \|\mathbf{x}_t\|^2 + \sum_{t=1}^{T} 2\theta\tau_t \ell_t^*(\mathbf{u}). \tag{12}$$

where $F_t = \mathbb{I}_{(t \in \mathcal{F})}$, $G_t = \mathbb{I}_{(t \in \mathcal{G})}$, $\mathbb{I}$ is an indicator function, and $\theta > 0$, $\ell_t^*(\mathbf{u}) = \max\left(0, 1 - y_t\left(\Pi_{\mathbf{x}_t}\mathbf{u} \cdot \mathbf{x}_t\right)\right)$.

Proof. Define $\Delta_t = \left\|\mathbf{w}_t - \theta\Pi_{\mathbf{w}_t}\mathbf{u}\right\|^2 - \left\|\mathbf{w}_{t+1} - \theta\Pi_{\mathbf{w}_{t+1}}\mathbf{u}\right\|^2$. Thus $\sum_t \Delta_t$ collapses to

$$\begin{aligned}\sum_{t=1}^{T} \Delta_t &= \sum_{t=1}^{T} \left(\left\|\mathbf{w}_t - \theta\Pi_{\mathbf{w}_t}\mathbf{u}\right\|^2 - \left\|\mathbf{w}_{t+1} - \theta\Pi_{\mathbf{w}_{t+1}}\mathbf{u}\right\|^2\right) \\ &= \left\|\mathbf{w}_1 - \theta\Pi_{\mathbf{w}_1}\mathbf{u}\right\|^2 - \left\|\mathbf{w}_{T+1} - \theta\Pi_{\mathbf{w}_{T+1}}\mathbf{u}\right\|^2,\end{aligned} \tag{13}$$

where $\left\|\mathbf{w}_{T+1} - \theta\Pi_{\mathbf{w}_{T+1}}\mathbf{u}\right\|^2 \geq 0$ always holds and $\mathbf{w}_1 = (0, \ldots, 0) \in \mathbb{R}^{d_1}$. Thus, we have

$$\sum_{t=1}^{T} \Delta_t \leq \theta^2 \left\|\Pi_{\mathbf{w}_1}\mathbf{u}\right\|^2. \tag{14}$$

Then we prove the following inequality always holds

$$\left(2G_t\delta_t\tau_t\left(\theta - |q_t|\right) + 2F_t\delta_t\tau_t\left(\theta + |q_t|\right)\right) \leq \Delta_t + \tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u}) + \theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2. \tag{15}$$

To prove the above inequality, we enumerate all the possible cases of δ_t, F_t, G_t and have the following discussions:

Case 1: if $\delta_t = 0$, the inequality (15) holds since $\mathbf{w}_t = \mathbf{w}_{t+1}$ and $\tau_t = 0$.

Case 2: if $\delta_t = 1$ and $F_t = 0$, the label of $\mathbf{x}_t$ is queried and the predicted label $\hat{y}_t = y_t$. Here are two sub-cases:

Sub-case 2.1: if $G_t = 0$, we have $\ell_t(\mathbf{w}_t) = 0$, thus $\tau_t = 0$, $\mathbf{w}_t = \mathbf{w}_{t+1}$. According to Case 1, inequality (15) holds.

Sub-case 2.2: if $G_t = 1$, we only need to pay attention to rounds which have $\ell_t(\mathbf{w}_t) > 0$. Note that $\bar{\mathbf{w}}_{t+1} = [\mathbf{w}_t + \tau_t y_t \mathbf{x}_t^s, \tau_t y_t \mathbf{x}_t^a]$, and the fact that $\hat{\mathbf{w}}_{t+1} \leq \bar{\mathbf{w}}_{t+1}$ and $\mathbf{w}_{t+1} \leq \hat{\mathbf{w}}_{t+1}$, we have $\mathbf{w}_{t+1} \leq \bar{\mathbf{w}}_{t+1}$ and $\|\mathbf{w}_{t+1}\|^2 \leq \|\bar{\mathbf{w}}_{t+1}\|^2$. Also, $\left\|\mathbf{w}_{t+1} - \theta\Pi_{\mathbf{w}_{t+1}}\mathbf{u}\right\|^2 - \left\|\bar{\mathbf{w}}_{t+1} - \theta\Pi_{\bar{\mathbf{w}}_{t+1}}\mathbf{u}\right\|^2 = \theta^2\|\mathbf{w}_{t+1}\|^2 - \theta^2\|\bar{\mathbf{w}}_{t+1}\|^2 - 2\theta\left\|\Pi_{\mathbf{w}_{t+1}}\mathbf{u}\right\|(\|\bar{\mathbf{w}}_{t+1}\| - \|\mathbf{w}_{t+1}\|) \leq 0$, we have $\left\|\mathbf{w}_{t+1} - \theta\Pi_{\mathbf{w}_{t+1}}\mathbf{u}\right\|^2 \leq \left\|\bar{\mathbf{w}}_{t+1} - \theta\Pi_{\bar{\mathbf{w}}_{t+1}}\mathbf{u}\right\|^2$. Then

$$\begin{aligned}
\Delta_t &= \left\|\mathbf{w}_t - \theta\Pi_{\mathbf{w}_t}\mathbf{u}\right\|^2 - \left\|\mathbf{w}_{t+1} - \theta\Pi_{\mathbf{w}_{t+1}}\mathbf{u}\right\|^2 \\
&\geq \left\|\mathbf{w}_t - \theta\Pi_{\mathbf{w}_t}\mathbf{u}\right\|^2 - \left\|\bar{\mathbf{w}}_{t+1} - \theta\Pi_{\bar{\mathbf{w}}_{t+1}}\mathbf{u}\right\|^2 \\
&= \left\|\mathbf{w}_t - \theta\Pi_{\mathbf{w}_t}\mathbf{u}\right\|^2 - \left\|\mathbf{w}_t + \tau_t y_t \Pi_{\mathbf{w}_t}\mathbf{x}_t - \theta\Pi_{\mathbf{w}_t}\mathbf{u}\right\|^2 - \left\|\tau_t y_t \Pi_{\neg\mathbf{w}_t}\mathbf{x}_t - \theta\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \\
&= 2\tau_t y_t \left(\theta\Pi_{\mathbf{w}_t}\mathbf{u} - \mathbf{w}_t\right) \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t - \tau_t^2\left\|\Pi_{\mathbf{w}_t}\mathbf{x}_t\right\|^2 - \left\|\tau_t y_t \Pi_{\neg\mathbf{w}_t}\mathbf{x}_t - \theta\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \\
&= 2\tau_t\left(\theta y_t \Pi_{\mathbf{w}_t}\mathbf{u} \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t + \theta y_t \Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u} \cdot \Pi_{\neg\mathbf{w}_t}\mathbf{x}_t\right) - 2\tau_t y_t \mathbf{w}_t \Pi_{\mathbf{w}_t}\mathbf{x}_t - \tau_t^2\|\mathbf{x}_t\|^2 - \theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2
\end{aligned} \tag{16}$$

According to definition, we have $\Pi_{\neg\mathbf{w}_t}\mathbf{x}_t = \Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{x}_t$, $\left\|\Pi_{\mathbf{w}_t}\mathbf{x}_t\right\|^2 + \left\|\Pi_{\neg\mathbf{w}_t}\mathbf{x}_t\right\|^2 = \|\mathbf{x}_t\|^2$. Since $\ell_t^*(\mathbf{u}) \geq 1 - y_t\left(\Pi_{\mathbf{x}_t}\mathbf{u} \cdot \mathbf{x}_t\right)$, we have $y_t\left(\Pi_{\mathbf{w}_t}\mathbf{u} \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t\right) + y_t\left(\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u} \cdot \Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{x}_t\right) \geq 1 - \ell_t^*(\mathbf{u})$. Then we have

$$\Delta_t + \tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u}) + \theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \geq 2\tau_t\left(\theta - y_t\mathbf{w}_t\Pi_{\mathbf{w}_t}\mathbf{x}_t\right) \tag{17}$$

Since $F_t = 0$ and $G_t = 1$, it implies that $\ell_t(\mathbf{w}_t) > 0$ and $0 < y_t\mathbf{w}_t \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t < 1$. Therefore, inequality (15) holds:

$$\Delta_t + \tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u}) + \theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \geq 2\tau_t\left(\theta - |q_t|\right) \tag{18}$$

Case 3: if $\delta_t = 1$ and $F_t = 1$, the label of $\mathbf{x}_t$ is queried but the predicted label $\hat{y}_t \neq y_t$, thus $G_t = 0$. Similarly, we have

$$\Delta_t + \tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u}) + \theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \geq 2\tau_t\left(\theta - y_t\mathbf{w}_t\Pi_{\mathbf{w}_t}\mathbf{x}_t\right) \tag{19}$$

Since $F_t = 1$, it implies that $y_t\mathbf{w}_t \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t \leq 0$ and $-y_t\mathbf{w}_t \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t = |q_t|$. Therefore, inequality (15) holds:

$$\Delta_t + \tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u}) + \theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \geq 2\tau_t\left(\theta + |q_t|\right). \tag{20}$$

Since inequality (15) always holds, we sum both sides of inequality Eq. (15) for $t = 1, \ldots, T$ and have

$$\begin{aligned}
\sum_{t=1}^{T}\left(2G_t\delta_t\tau_t\left(\theta - |q_t|\right) + 2F_t\delta_t\tau_t\left(\theta + |q_t|\right)\right) &\leq \sum_{t=1}^{T}\Delta_t + \sum_{t=1}^{T}\left(\tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u})\right) + \sum_{t=1}^{T}\theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 \\
&\leq \theta^2\left\|\Pi_{\mathbf{w}_1}\mathbf{u}\right\|^2 + \sum_{t=1}^{T}\theta^2\left\|\Pi_{\mathbf{w}_{t+1}/\mathbf{w}_t}\mathbf{u}\right\|^2 + \sum_{t=1}^{T}\left(\tau_t^2\|\mathbf{x}_t\|^2 + 2\theta\tau_t\ell_t^*(\mathbf{u})\right) \\
&= \theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T}\tau_t^2\|\mathbf{x}_t\|^2 + \sum_{t=1}^{T}2\theta\tau_t\ell_t^*(\mathbf{u})
\end{aligned} \tag{21}$$

The lemma is proved. □

With the help of Lemma 1, we can continue to prove the following four theorems, which discuss the upper error bounds of the proposed algorithms in the linear separable and non-separable cases. First we derive the expected error bound for FLLS in the separable case, i.e., for any $t \in [T]$, we assume that there exists vector $\mathbf{u} \in \mathbb{R}^{d_T}$ such that $y_t(\Pi_{\mathbf{x}_t}\mathbf{u} \cdot \mathbf{x}_t) \geq 1$.

Theorem 1. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of obtained instances, where $\mathbf{x}_t \in \mathbb{R}^{d_t}$, $d_{t-1} \leq d_t$, $y_t \in \{-1, +1\}$ and $\|\mathbf{x}_t\| \leq R$ for all t. Assume that there exists a vector $\mathbf{u} \in \mathbb{R}^{d_T}$ such that $\ell_t^*(\mathbf{u}) = 0$ for all t. Then the expected number of errors made by FLLS is bounded by*

$$\mathbb{E}[\sum_{t=1}^{T} F_t] \leq \mathbb{E}[\sum_{t=1}^{T} F_t \ell_t(\mathbf{w}_t)] \leq \frac{R^2}{4}(\rho + \frac{1}{\rho} + 2)\|\mathbf{u}\|^2. \tag{22}$$

Setting $\rho = 1$, we can achieve the following upper bound:

$$\mathbb{E}[\sum_{t=1}^{T} F_t] \leq \mathbb{E}[\sum_{t=1}^{T} F_t \ell_t(\mathbf{w}_t)] \leq R^2\|\mathbf{u}\|^2. \tag{23}$$

Proof. Combining $\ell_t^*(\mathbf{u}) = 0$ for all t and the result of Lemma 1, we can get

$$\sum_{t=1}^{T} 2\delta_t \tau_t \left[G_t\left(\theta - |q_t|\right) + F_t\left(\theta + |q_t|\right)\right] \leq \theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} \tau_t^2 \|\mathbf{x}_t\|^2. \tag{24}$$

Inequality (24) can be further reformulated as:

$$\begin{aligned}
\theta^2\|\mathbf{u}\|^2 &\geq \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t|) + F_t(\theta + |q_t|)] - \sum_{t=1}^{T} \tau_t^2\|\mathbf{x}_t\|^2 \\
&= \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t| - \frac{\tau_t}{2}\|\mathbf{x}_t\|^2) + F_t(\theta + |q_t| - \frac{\tau_t}{2}\|\mathbf{x}_t\|^2)] \\
&= \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t| - \frac{\ell_t(\mathbf{w}_t)}{2}) + F_t(\theta + |q_t| - \frac{\ell_t(\mathbf{w}_t)}{2})] \\
&= \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t| - \frac{1 - y_t q_t}{2}) + F_t(\theta + |q_t| - \frac{1 - y_t q_t}{2})] \\
&= \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t| - \frac{1 - |q_t|}{2}) + F_t(\theta + |q_t| - \frac{1 + |q_t|}{2})] \\
&= \sum_{t=1}^{T} 2G_t\delta_t\tau_t(\theta - \frac{1 + |q_t|}{2}) + \sum_{t=1}^{T} 2F_t\delta_t 2\tau_t(\theta - \frac{1 - |q_t|}{2}).
\end{aligned} \tag{25}$$

If we set $\theta = (\rho + 1)/2, \rho \geq 1$ and plug it into inequality (25), we have

$$(\frac{1 + \rho}{2})^2\|\mathbf{u}\|^2 \geq \sum_{t=1}^{T} F_t\delta_t\tau_t(\rho + |p_t|), \tag{26}$$

Note that when $G_t = 1, |q_t| \in [0, 1), [\theta - (1 + |q_t|)/2] = (\rho - |q_t|)/2 > 0$, and $[\theta - (1 - |q_t|)/2] = (\rho + |q_t|)/2$.

Plugging $\ell_t(\mathbf{w}_t)/\|\mathbf{x}_t\|^2 \geq \ell_t(\mathbf{w}_t)/R^2$ into inequality (26) we have:

$$(\frac{1 + \rho}{2})^2\|\mathbf{u}\|^2 \geq \frac{1}{R^2}\sum_{t=1}^{T} F_t\delta_t\ell_t(\mathbf{w}_t)(\rho + |q_t|). \tag{27}$$

Theorem 1 can be proved by taking expectation with inequality (27)

$$\begin{aligned}\mathbb{E}[\frac{1}{R^2}\sum_{t=1}^{T}F_t\delta_t\ell_t(\mathbf{w}_t)(\rho+|q_t|)] &= \mathbb{E}[\frac{1}{R^2}\sum_{t=1}^{T}F_t\ell_t(\mathbf{w}_t)(\rho+|q_t|)\mathbb{E}(\delta_t)] \\ &= \frac{1}{R^2}\mathbb{E}[\rho\sum_{t=1}^{T}F_t\ell_t(\mathbf{w}_t)] \leq (\frac{\rho+1}{2})^2\|\mathbf{u}\|^2.\end{aligned} \tag{28}$$

Theorem 1 is proved. □

As can be seen from Theorem 1, the upper error bound is proportional to R and inversely proportional to $1/\|\mathbf{u}\|^2$, revealing the consistency with existing research[32]. The deficiency of Theorem 1 is the assumption that there exists a vector $\mathbf{u} \in \mathbb{R}^{d_T}$ such that $\ell_t^*(\mathbf{u}) = 0$ for all t, which means the classifier $\mathbf{u}$ can perfectly separate the data streams. In Theorem 2, we further extend the above case and assume that $\ell_t^*(\mathbf{u}) = 0$ may not always hold for all t. Besides, we assume that $\|\mathbf{x}_t\|^2$ is normalized, i.e., $\|\mathbf{x}_t\|^2 = 1$. Then we have the following expected error bounds for FLLS algorithm:

Theorem 2. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of obtained instances, where $\mathbf{x}_t \in \mathbb{R}^{d_t}$, $d_{t-1} \leq d_t$, $y_t \in \{-1, +1\}$ and $\|\mathbf{x}_t\|^2 = 1$ for all t. For any vector $\mathbf{u} \in \mathbb{R}^{d_T}$, the expected number of errors made by FLLS is then bounded by*

$$\mathbb{E}[\sum_{t=1}^{T}F_t] \leq \frac{1}{4}(\rho+\frac{1}{\rho}+2)\|\mathbf{u}\|^2 + (\frac{1}{\rho}+1)\sum_{t=1}^{T}\ell_t^*(\mathbf{u}), \tag{29}$$

Setting $\rho = 1$, we can achieve the following upper bound:

$$\mathbb{E}[\sum_{t=1}^{T}F_t] \leq \|\mathbf{u}\|^2 + 2\sum_{t=1}^{T}\ell_t^*(\mathbf{u}). \tag{30}$$

Proof. According to the result of Lemma 1, we have

$$\begin{aligned}\theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T}2\theta\tau_t\ell_t^*(\mathbf{u}) &\geq \sum_{t=1}^{T}2\delta_t\tau_t[G_t(\theta-|q_t|)+F_t(\theta+|q_t|)] - \sum_{t=1}^{T}\tau_t^2\|\mathbf{x}_t\|^2 \\ &= \sum_{t=1}^{T}2\delta_t\tau_t[G_t(\theta-|q_t|-\frac{\tau_t}{2}\|\mathbf{x}_t\|^2)+F_t(\theta+|q_t|-\frac{\tau_t}{2}\|\mathbf{x}_t\|^2] \\ &= \sum_{t=1}^{T}2\delta_t\tau_t[G_t(\theta-|q_t|-\frac{\ell_t(\mathbf{w}_t)}{2})+F_t(\theta+|q_t|-\frac{\ell_t(\mathbf{w}_t)}{2})] \\ &= \sum_{t=1}^{T}2G_t\delta_t\tau_t(\theta-\frac{1+|q_t|}{2}) + \sum_{t=1}^{T}2F_t\delta_t\tau_t(\theta-\frac{1-|q_t|}{2}).\end{aligned} \tag{31}$$

Similarly, plugging $\theta = (1+\rho)/2, \rho \geq 1$ into (31), we have

$$(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T}(\rho+1)\tau_t\ell_t^*(\mathbf{u}) \geq \sum_{t=1}^{T}F_t\delta_t\tau_t(\rho+|q_t|). \tag{32}$$

Since $\|\mathbf{x}_t\|^2 = 1$ and $\tau_t = \ell_t/\|x_t\|^2$, we have $\tau_t = \ell_t$. Thus we can rewrite it as

$$(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T}(\rho+1)\ell_t\ell_t^*(\mathbf{u}) \geq \sum_{t=1}^{T}F_t\delta_t\ell_t(\rho+|p_t|). \tag{33}$$

Note that when $F_t = 1$, it means FLLS made a wrong prediction, then we have $y_t\left(\mathbf{w}_t \cdot \Pi_{\mathbf{w}_t}\mathbf{x}_t\right) \leq 0$ and $\ell_t\left(\mathbf{w}_t\right) \geq 1$. Divide both sides of the inequality by ℓ_t implies:

$$\begin{aligned}\sum_{t=1}^{T} F_t\delta_t(\rho + |q_t|) &\leq (\frac{\rho+1}{2})^2\frac{\|\mathbf{u}\|^2}{\ell_t} + (\rho+1)\sum_{t=1}^{T}\ell_t^*\left(\mathbf{u}\right) \\ &\leq (\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + (\rho+1)\sum_{t=1}^{T}\ell_t^*\left(\mathbf{u}\right)\end{aligned} \tag{34}$$

Theorem 2 can be proved by taking expectation with the above inequality. □

The deficiency of Theorem 1 and 2 is the assumption that the training data is linearly separable, which is inconsistent with most situations in reality. FLLS-I and FLLS-II are more suitable in practice by introducing ξ, the difference between FLLS-I and FLLS-II is their model update strategies, the former scales linearly with ξ and the latter scales quadratically with ξ. Therefore, the expected error bounds for FLLS-I and FLLS-II algorithms are further derived to adapt to the linearly non-separable situations.

Theorem 3. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of obtained instances, where $\mathbf{x}_t \in \mathbb{R}^{d_t}$, $d_{t-1} \leq d_t$, $y_t \in \{-1, +1\}$ and $\|\mathbf{x}_t\| \leq R$ for all t. For any vector $\mathbf{u} \in \mathbb{R}^{d_T}$, the expected number of errors made by FLLS-I is then bounded by*

$$\mathbb{E}[\sum_{t=1}^{T} F_t] \leq \beta[(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + (\rho+1)C\sum_{t=1}^{T}\ell_t^*(\mathbf{u})], \tag{35}$$

where $\beta = \frac{1}{\rho}\max\{\frac{1}{C}, R^2\}$. Setting $\rho = 1$, we can achieve the following upper bound:

$$\mathbb{E}[\sum_{t=1}^{T} F_t] \leq \beta[\|\mathbf{u}\|^2 + 2C\sum_{t=1}^{T}\ell_t^*(\mathbf{u})]. \tag{36}$$

Proof. According to the definition of τ_t in FLLS-I algorithm, we have $\tau_t \leq \ell_t(\mathbf{w}_t)/\|\mathbf{x}_t\|^2$, so $\tau_t\|\mathbf{x}_t\|^2 \leq \ell_t(\mathbf{w}_t)$. Besides, according to the result of Lemma 1, we have

$$\begin{aligned}\theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} 2\theta\tau_t\ell_t^*\left(\mathbf{u}\right) &\geq \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t|) + F_t(\theta + |q_t|)] - \sum_{t=1}^{T}\tau_t^2\|\mathbf{x}_t\|^2 \\ &= \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t| - \frac{\tau_t}{2}\|\mathbf{x}_t\|^2) + F_t(\theta + |q_t| - \frac{\tau_t}{2}\|\mathbf{x}_t\|^2] \\ &\geq \sum_{t=1}^{T} 2\delta_t\tau_t[G_t(\theta - |q_t| - \frac{\ell_t\left(\mathbf{w}_t\right)}{2}) + F_t(\theta + |q_t| - \frac{\ell_t\left(\mathbf{w}_t\right)}{2})] \\ &= \sum_{t=1}^{T} 2G_t\delta_t\tau_t(\theta - \frac{1+|q_t|}{2}) + \sum_{t=1}^{T} 2F_t\delta_t\tau_t(\theta - \frac{1-|q_t|}{2}).\end{aligned} \tag{37}$$

Similarly, plugging $\theta = (1+\rho)/2, \rho \geq 1$ into (37), we have

$$(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T}(\rho+1)\,\tau_t\ell_t^*\left(\mathbf{u}\right) \geq \sum_{t=1}^{T} F_t\delta_t\tau_t(\rho + |q_t|). \tag{38}$$

Note that when $F_t = 1$, it means FLLS-I made a wrong prediction, then we have $y_t \left(w_t \cdot \Pi_{\mathbf{w}_t} \mathbf{x}_t\right) \leq 0$ and $\ell_t \left(\mathbf{w}_t\right) \geq 1$. So $\tau_t \geq \min\{C, \frac{1}{R^2}\}$ when $F_t = 1$, and we have:

$$(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} (\rho+1)\,\tau_t \ell_t^*\left(\mathbf{u}\right) \geq \min\{C, \frac{1}{R^2}\} \sum_{t=1}^{T} F_t \delta_t (\rho + |q_t|). \tag{39}$$

Theorem 3 can be proved by taking expectation with the above inequality. □

Theorem 4. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of obtained instances, where $\mathbf{x}_t \in \mathbb{R}^{d_t}$, $d_{t-1} \leq d_t$, $y_t \in \{-1, +1\}$ and $\|\mathbf{x}_t\| \leq R$ for all t. For any vector $\mathbf{u} \in \mathbb{R}^{d_T}$, the expected number of errors made by FLLS-II is then bounded by*

$$\mathbb{E}[\sum_{t=1}^{T} F_t] \leq \gamma \frac{1}{\rho}[(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + 2C(\frac{\rho+1}{2})^2 \sum_{t=1}^{T} \ell_t^*(\mathbf{u})^2], \tag{40}$$

where $\gamma = R^2 + \frac{1}{2C}$. Setting $\rho = 1$, we can achieve the following upper bound:

$$\mathbb{E}[\sum_{t=1}^{T} F_t] \leq (R^2 + \frac{1}{2C})[\|\mathbf{u}\|^2 + 2C \sum_{t=1}^{T} \ell_t^*(\mathbf{u})^2]. \tag{41}$$

Proof. First we define $\mathcal{O} = \theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} \tau_t^2 \|\mathbf{x}_t\|^2 + \sum_{t=1}^{T} 2\theta\tau_t \ell_t^*\left(\mathbf{u}\right)$, $\mathcal{P} = \sum_{t=1}^{T} \theta(\frac{\tau_t}{\sqrt{2C\theta}} - \sqrt{2C\theta}\ell_t^*(\mathbf{u}))^2$ and $\mathcal{Q} = \theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} \tau_t^2(\|\mathbf{x}_t\|^2 + \frac{1}{2C}) + \sum_{t=1}^{T} 2C\theta^2 \ell_t^*(\mathbf{u})^2$, it can be easily verified that $\mathcal{O} \leq \mathcal{O} + \mathcal{P} = \mathcal{Q}$. Then according to the result of Lemma 1, we have

$$\sum_{t=1}^{T} 2\delta_t \tau_t \left[G_t\left(\theta - |q_t|\right) + F_t\left(\theta + |q_t|\right)\right] \leq \mathcal{Q}. \tag{42}$$

We reformulated the above formulation as:

$$\begin{aligned} \theta^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} 2C\theta^2 \ell_t^*(\mathbf{u})^2 &\geq \sum_{t=1}^{T} 2\delta_t \tau_t [G_t(\theta - |q_t|) + F_t(\theta + |q_t|)] - \tau_t^2(\|\mathbf{x}_t\|^2 + \frac{1}{2C}) \\ &= \sum_{t=1}^{T} 2G_t \delta_t \tau_t (\theta - \frac{1+|q_t|}{2}) + \sum_{t=1}^{T} 2F_t \delta_t \tau_t (\theta - \frac{1-|q_t|}{2}) \end{aligned} \tag{43}$$

Similarly, plugging $\theta = (1+\rho)/2, \rho \geq 1$ into (43), we have

$$(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} 2C(\frac{\rho+1}{2})^2 \ell_t^*(\mathbf{u})^2 \geq \sum_{t=1}^{T} F_t \delta_t \tau_t (\rho + |q_t|). \tag{44}$$

Note that when $F_t = 1$, it means FLLS-II made a wrong prediction, then we have $y_t \left(\mathbf{w}_t \cdot \Pi_{\mathbf{w}_t} \mathbf{x}_t\right) \leq 0$ and $\ell_t \left(\mathbf{w}_t\right) \geq 1$. So $\tau_t \geq R^2 + \frac{1}{2C}$ when $F_t = 1$, and we have

$$(\frac{\rho+1}{2})^2\|\mathbf{u}\|^2 + \sum_{t=1}^{T} 2C(\frac{\rho+1}{2})^2 \ell_t^*(\mathbf{u})^2 \geq (\frac{1}{R^2 + 1/(2C)}) \sum_{t=1}^{T} F_t \delta_t (\rho + |q_t|). \tag{45}$$

Taking expectation with the above inequality will conclude theorem 4. □

5 EXPERIMENTS

First, we give an introduction to the datasets and compared methods used in this paper along with the general settings. Then we present the experimental results on synthetic data sets and real-world applications.

5.1 Data Sets and General Settings

We conduct our experiments on twelve data sets from UCI Repository[3] and LIBSVM Library[4], and two real-world data sets which are rcv1[35] and RFID[3]. The details of the 14 data sets used in our experiments are listed in TABLE 2.

Table 2. The details of experimental data sets.

Dataset	# Inst.	# Feat.	Dataset	# Inst.	# Feat.
air	210	64	splice	3175	60
vote	435	16	kr-V-kp1	3196	36
wdbc	569	29	spambase	4601	57
dna	949	180	phishing	11055	68
svmguide3	1243	22	a9a	32561	123
PCMAC	1943	3289	rcv1	20242	47236
basehock	1993	4862	RFID	940	150

Our proposed algorithms have the advantage of simultaneously processing label scarcity and feature incremental learning problems. To verify the above conclusion, we compare our proposed algorithms with PEA, RPEA, RFESL, and RFLLS algorithms. PEA is only designed for the label scarcity problem, it uses the Perceptron update strategy for learning, i.e., $\mathbf{w}_{t+1} = [\mathbf{w}_t + y_t\mathbf{x}_t^s, y_t\mathbf{x}_t^q]$, and updates the current model only when the model makes a wrong prediction. RFESL is only designed for the feature incremental learning problem, It is based on the FESL algorithm, the features of data streams would vanish or occur in the basic setting of FESL. To adapt FESL to our new setting, we query labels uniformly and randomly for FESL to handle label scarcity and complete the vanished features with true values. By adding these assumptions, FESL can be applied in our incremental feature spaces learning with label scarcity. The adjusted FESL is named as RFESL. RPEA is the random version of PEA, which is not designed for any of the above two problems. RFLLS are the random version of the proposed algorithms, including RFLLS, RFLLS-I, and RFLLS-II. All the above nine methods are as follows:

- "RFLLS": the Random FLLS algorithms, including RFLLS, RFLLS-I, RFLLS-II, which will uniformly and randomly query labels;
- "PEA"[24]: the Perceptron-based Active learning algorithm;
- "RPEA"[36]: the Random PEA algorithm, which uniformly and randomly queries labels;
- "RFESL": the Random FESL algorithm[3], which uniformly and randomly queries labels;
- "FLLS": the proposed algorithm, including FLLS, FLLS-I and FLLS-II;

All the nine methods learn a linear binary classifier. Each dataset is randomly divided into two parts with 80% as the training data and 20% as the testing data, which is non-overlapping. We use the training data to learn a classification model and the testing data to evaluate the learning performance. We simulate trapezoidal streams as follows, the training data is split into 10 chunks, the first chunk carries the first 10 percent instances and features. The second chunk carries the second 10 percent instances with another 10 percent features (20 percent features in total). The testing data carries all features.

[3] http://archive.ics.uci.edu/ml
[4] http://www.csie.ntu.edu.tw/~cjlin/libsvm

We set $\lambda = 30$ and $B = 0.5$ unless otherwise specified, i.e., 50 percent features are used for learning at each round t. Cross-validation is used for all the methods and datasets to determine parameter C, which is searched from $2^{[-5:5]}$. Parameter ρ, which determines the query ratio, is set as $2^{[-20:20]}$ to examine varied query ratios. After T rounds, we measure the performance of the learned classifier on test data. Both ACC and AUC are used for evaluating metrics. For each data set, we run the experiment 20 times, each with a random partition. The results are reported by an average performance.

5.2 Comparisons with Benchmarks

In this subsection, we present the experimental results of nine methods on 12 benchmark data sets. Fig. 3 and Fig. 4 show the average ACC results of these nine methods with varied ratios of queried instances. Table 3 and Table 4 show the average AUC values and running time of these nine methods with the query ratio near 10% and 20%, the best AUC result and its comparable results are highlighted in boldface based on the paired t-tests at 95% significance level.

From Fig. 3 and Fig. 4, Table 3 and Table 4, we can have several conclusions. First, we can observe that all these three proposed algorithms are better than their corresponding random versions on ACC and AUC results, validating the effectiveness of the label querying strategy. Second, among FLLS, FLLS-I, and FLLS-II, FLLS-I performs the best on dna, svmguide3, splice, spambase, phishing, and a9a. FLLS-II performs well on high-dimensional data sets PCMAC and basehock. FLLS performs the worst on most data sets. There are two reasons, one is the noise contained in these data sets, the other one is the sensitivity of FLLS to noise since it conducts a more aggressive update. In contrast, FLLS-I and FLLS-II avoid overfitting to noise by using a "soft" update strategy. Third, these two soft algorithms FLLS-I and FLLS-II also achieve significantly higher ACC and AUC values than PEA and RPEA algorithms, which indicates that the proposed model update strategy can effectively exploit the information of labeled data. Forth, on six datasets, i.e., air, vote, PCMAC, basehock, spambase, a9a, PEA performs better than RPEA, on the rest, their performance is equal. Since PEA and RPEA only use a simple update strategy to deal with the feature incremental learning problem, it results in a small performance difference between PEA and RPEA on some datasets. Fifth, RFESL performs better than RPEA and RFLLS on most datasets, which again validates the importance of using an effective model update strategy to exploit the information of labeled data with different feature spaces. Sixth, in Fig. 3 and Fig. 4, an abnormal phenomenon is that when the query ratio exceeds some thresholds, some curves could decrease as the query ratio increases. We think this is mainly caused by over-fitting on the noisy training data because FLLS-I and FLLS-II have fewer such phenomena than FLLS. Finally, we find that most curves of the proposed methods tend to be stable quickly after the query ratio exceeds some thresholds, which indicates the effectiveness of the proposed algorithms in alleviating the problem of label scarcity.

We can also observe from Table 3 and Table 4 that the running time of the proposed algorithms is consistently higher than the versions with random queries and PEA algorithms, the reasons are as follows. First, compared with the versions with random queries, the proposed algorithms decide whether to query the label of the current instance by a Bernoulli random variable $\delta_t \in \{0, 1\}$ with probability $\rho/(\rho + |q_t|)$, $\rho \geq 1$, which guarantees the superior to the versions with random queries. Thus, the proposed algorithms would cost more time. Second, compared with the PEA algorithm, the model update strategy of the proposed algorithms is more complicated. PEA updates the current model only when the model makes a wrong prediction, however, the proposed algorithms aggressively update the current model whenever the loss is nonzero (even if the prediction is correct) to use more instances for model updating and getting better results than the PEA algorithm. Besides, as can be seen from the numerical results in Table 3 and Table 4, we think that the additional computational cost is not so large and it is affordable for real applications.

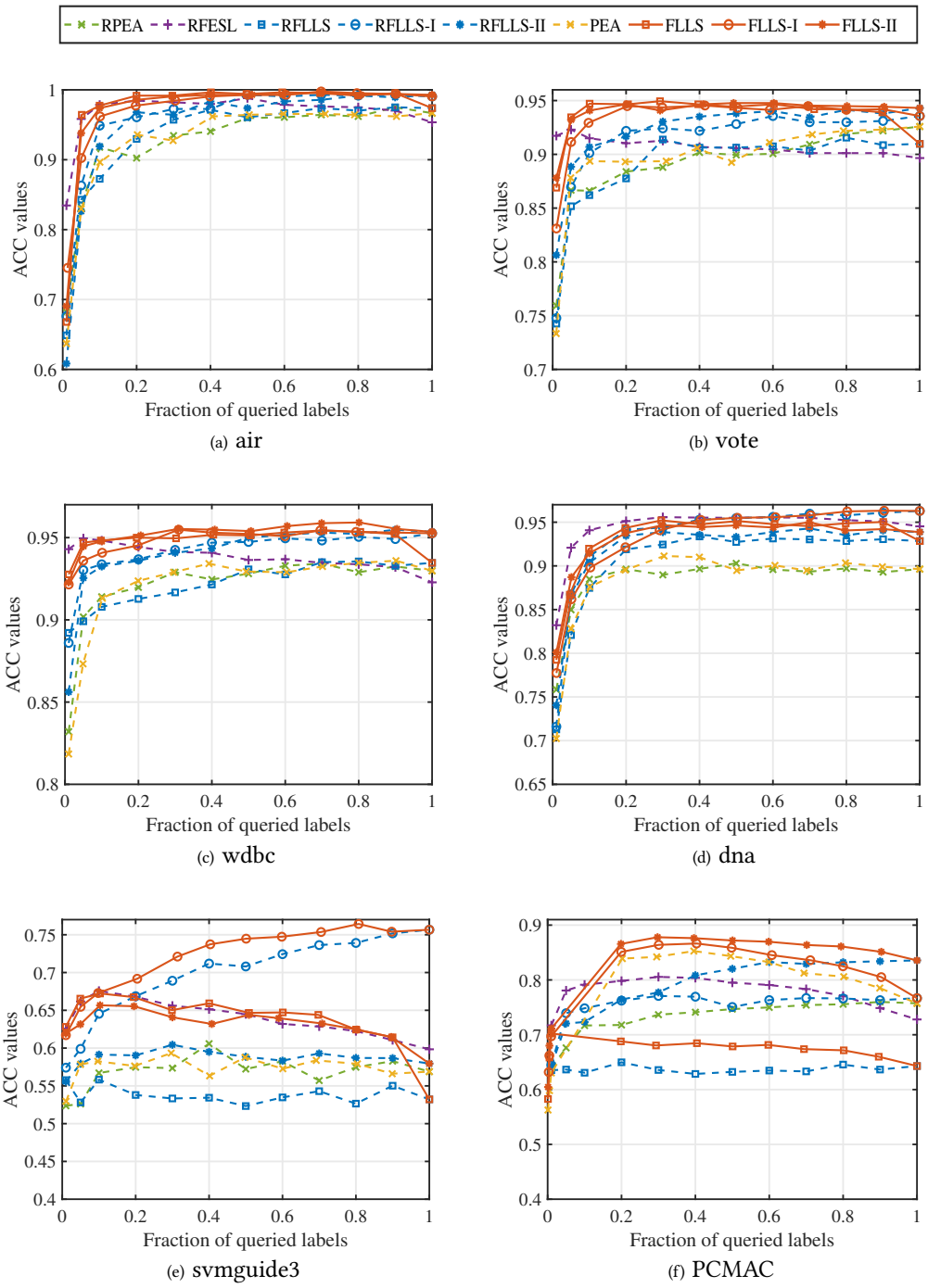

Fig. 3. The ACC results for the 6 data sets: air, vote, wdbc, dna, svmguide3, and PCMAC. The curve shows the average learning accuracy over the fraction of queried labels.

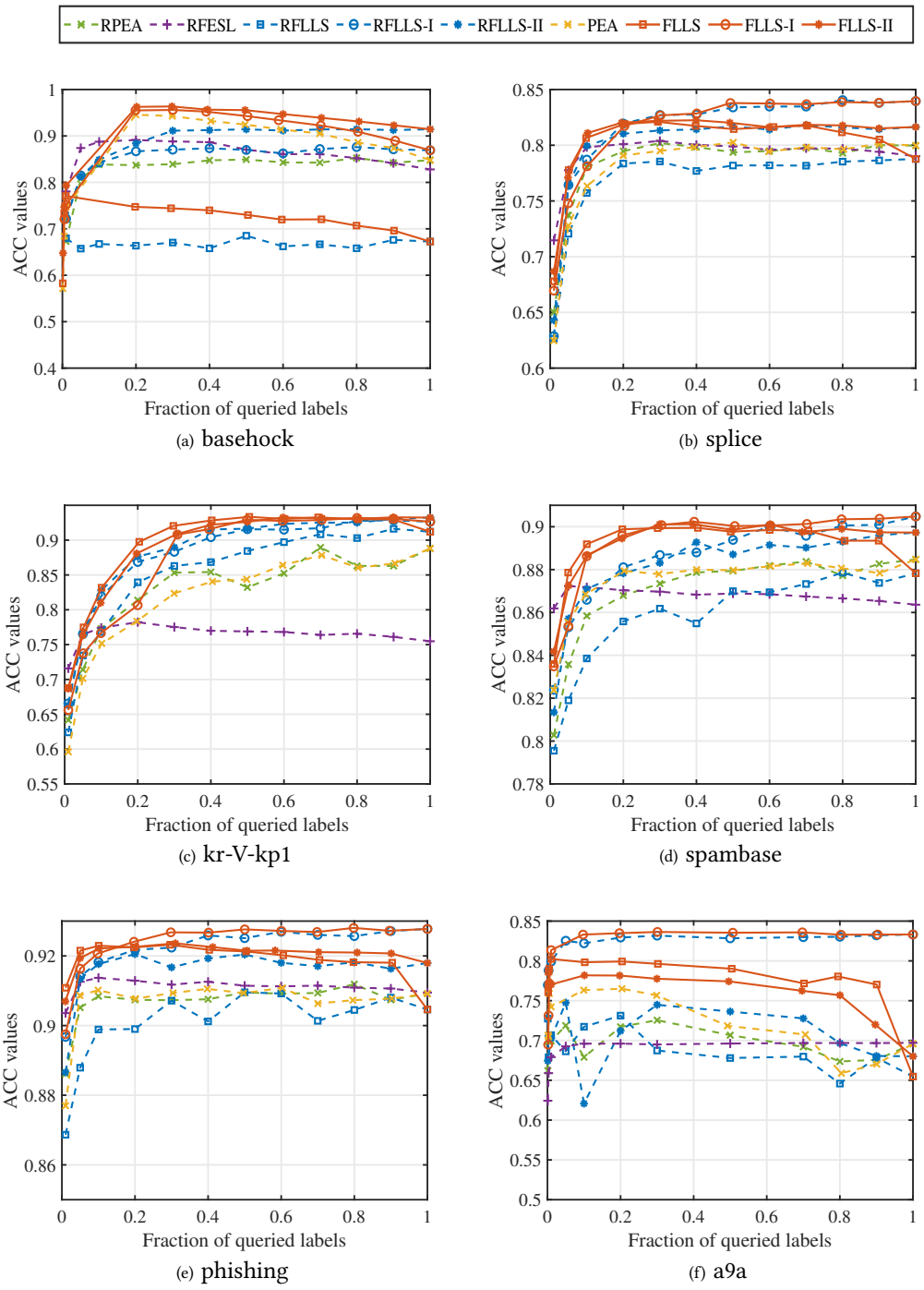

Fig. 4. The ACC results for the 6 data sets: basehock, splice, kr-V-kp1, spambase, phishing, and a9a. The curve shows the average learning accuracy over the fraction of queried labels.

Table 3. AUC results of algorithms with the query ratio fixed to about 10%.

Algorithm	air			vote			wdbc		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.917(.041)	.0014	10.00(.00)	.877(.041)	.0015	10.00(.00)	.913(.032)	.0017	10.00(.00)
RFESL	**.979(.018)**	.0030	10.00(.00)	.930(.021)	.0042	10.00(.00)	.948(.019)	.0053	10.00(.00)
RFLLS	.875(.077)	.0014	10.00(.00)	.876(.070)	.0016	10.00(.00)	.913(.043)	.0017	10.00(.00)
RFLLS-I	**.959(.029)**	.0015	10.00(.00)	.911(.036)	.0018	10.00(.00)	.931(.022)	.0018	10.00(.00)
RFLLS-II	.918(.051)	.0017	10.00(.00)	.917(.036)	.0018	10.00(.00)	.931(.024)	.0018	10.00(.00)
PEA	.898(.075)	.0042	10.06(.47)	.905(.041)	.0056	10.13(.56)	.913(.029)	.0072	10.18(.31)
FLLS	**.978(.015)**	.0044	9.97(.36)	**.956(.017)**	.0066	10.06(.65)	**.950(.023)**	.0073	9.77(.40)
FLLS-I	**.963(.025)**	.0045	10.09(.46)	.940(.026)	.0066	9.77(.77)	.939(.018)	.0074	10.11(.54)
FLLS-II	**.975(.022)**	.0045	9.97(.65)	.950(.021)	.0067	10.10(.51)	.949(.015)	.0078	10.02(.36)

Algorithm	dna			svmguide3			PCMAC		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.884(.030)	.0033	10.00(.00)	.581(.051)	.0026	10.00(.00)	.717(.047)	.0433	10.00(.00)
RFESL	**.941(.015)**	.0113	10.00(.00)	**.689(.036)**	.0129	10.00(.00)	.792(.036)	.0636	10.00(.00)
RFLLS	.874(.024)	.0041	10.00(.00)	.557(.090)	.0029	10.00(.00)	.631(.041)	.0450	10.00(.00)
RFLLS-I	.910(.015)	.0046	10.00(.00)	.639(.043)	.0032	10.00(.00)	.748(.039)	.0460	10.00(.00)
RFLLS-II	.904(.023)	.0050	10.00(.00)	.594(.070)	.0033	10.00(.00)	.723(.050)	.0476	10.00(.00)
PEA	.876(.031)	.0186	10.13(.24)	.576(.065)	.0148	9.69(.42)	.793(.061)	.2995	9.86(.36)
FLLS	.920(.020)	.0191	9.89(.27)	.659(.028)	.0155	10.31(.36)	.692(.028)	.3007	10.10(.15)
FLLS-I	.898(.019)	.0191	10.26(.91)	**.667(.040)**	.0159	9.80(.50)	.820(.020)	.3011	9.74(.23)
FLLS-II	.914(.016)	.0201	9.99(.27)	.662(.024)	.0181	10.31(.39)	**.843(.014)**	.3012	10.02(.70)

Algorithm	basehock			splice			kr-V-kp		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.838(.039)	.0704	10.00(.00)	.781(.020)	.0064	10.00(.00)	.772(.045)	.0064	10.00(.00)
RFESL	.888(.016)	.0954	10.00(.00)	.799(.018)	.0234	10.00(.00)	.773(.026)	.0217	10.00(.00)
RFLLS	.666(.058)	.0724	10.00(.00)	.759(.035)	.0065	10.00(.00)	.768(.052)	.0064	10.00(.00)
RFLLS-I	.842(.040)	.0742	10.00(.00)	.788(.022)	.0069	10.00(.00)	.818(.044)	.0067	10.00(.00)
RFLLS-II	.849(.052)	.0751	10.00(.00)	.801(.024)	.0079	10.00(.00)	.827(.037)	.0072	10.00(.00)
PEA	.925(.021)	.4886	9.97(.24)	.765(.032)	.0437	10.11(.18)	.750(.030)	.0404	10.15(.45)
FLLS	.762(.031)	.4994	10.34(.18)	.808(.014)	.0439	9.81(.27)	**.831(.018)**	.0426	10.10(.36)
FLLS-I	.936(.014)	.5071	9.85(.36)	.782(.026)	.0463	10.09(.57)	.765(.025)	.0426	9.91(.96)
FLLS-II	**.948(.009)**	.5071	10.10(.15)	**.812(.012)**	.0469	10.23(.24)	.809(.016)	.0441	9.86(.45)

Algorithm	spambase			phishing			a9a		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.856(.015)	.0080	10.00(.00)	.907(.007)	.0176	10.00(.00)	.587(.077)	.0611	10.00(.00)
RFESL	.868(.005)	.0315	10.00(.00)	.913(.006)	.1036	10.00(.00)	**.780(.005)**	.3058	10.00(.00)
RFLLS	.835(.027)	.0086	10.00(.00)	.898(.018)	.0186	10.00(.00)	.587(.081)	.0627	10.00(.00)
RFLLS-I	.862(.018)	.0086	10.00(.00)	**.917(.006)**	.0189	10.00(.00)	.732(.041)	.0655	10.00(.00)
RFLLS-II	.870(.018)	.0099	10.00(.00)	**.917(.005)**	.0199	10.00(.00)	.611(.079)	.0656	10.00(.00)
PEA	.866(.019)	.0542	9.68(.22)	.909(.007)	.1234	9.93(.31)	.523(.059)	.3505	9.91(.72)
FLLS	**.889(.014)**	.0589	10.01(.23)	**.922(.005)**	.1371	10.07(.35)	.732(.028)	.3618	10.20(.16)
FLLS-I	**.882(.013)**	.0595	9.98(.33)	**.919(.006)**	.1375	9.78(.60)	**.765(.021)**	.3719	9.75(.37)
FLLS-II	**.884(.011)**	.0601	10.16(.46)	**.921(.005)**	.1418	10.16(.58)	.682(.070)	.3747	10.11(.26)

Table 4. AUC results of algorithms with the query ratio fixed to about 20%.

Algorithm	air			vote			wdbc		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.904(.049)	.0015	20.00(.00)	.894(.046)	.0016	20.00(.00)	.920(.026)	.0019	20.00(.00)
RFESL	.985(.018)	.0047	20.00(.00)	.925(.023)	.0051	20.00(.00)	**.941(.019)**	.0062	20.00(.00)
RFLLS	.931(.035)	.0018	20.00(.00)	.889(.045)	.0019	20.00(.00)	.915(.028)	.0025	20.00(.00)
RFLLS-I	.961(.026)	.0018	20.00(.00)	.931(.031)	.0021	20.00(.00)	.935(.019)	.0025	20.00(.00)
RFLLS-II	.970(.013)	.0019	20.00(.00)	.927(.038)	.0021	20.00(.00)	**.934(.023)**	.0023	20.00(.00)
PEA	.937(.030)	.0052	20.21(.54)	.896(.053)	.0057	19.76(.65)	.923(.026)	.0084	19.93(.61)
FLLS	**.992(.006)**	.0057	19.97(.73)	**.955(.019)**	.0063	20.52(.61)	**.951(.017)**	.0086	19.52(.53)
FLLS-I	.978(.012)	.0058	19.76(.98)	**.954(.022)**	.0067	20.30(.77)	**.946(.015)**	.0089	19.75(.39)
FLLS-II	.986(.010)	.0061	20.03(.68)	**.955(.019)**	.0079	20.06(.94)	**.950(.019)**	.0093	20.62(.71)

Algorithm	dna			svmguide3			PCMAC		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.896(.030)	.0040	20.00(.00)	.584(.075)	.0042	20.00(.00)	.718(.045)	.0693	20.00(.00)
RFESL	**.952(.014)**	.0126	20.00(.00)	**.688(.029)**	.0137	20.00(.00)	.799(.024)	.0820	20.00(.00)
RFLLS	.918(.024)	.0044	20.00(.00)	.529(.073)	.0045	20.00(.00)	.650(.048)	.0700	20.00(.00)
RFLLS-I	**.941(.015)**	.0047	20.00(.00)	.665(.032)	.0046	20.00(.00)	.763(.034)	.0706	20.00(.00)
RFLLS-II	.934(.018)	.0052	20.00(.00)	.598(.071)	.0046	20.00(.00)	.765(.033)	.0754	20.00(.00)
PEA	.895(.024)	.0215	19.76(.43)	.576(.055)	.0172	19.72(.41)	.839(.021)	.3112	20.21(.67)
FLLS	**.943(.009)**	.0236	19.80(.35)	.665(.034)	.0175	19.46(.58)	.688(.023)	.3143	19.82(.23)
FLLS-I	.921(.018)	.0241	19.82(.83)	**.677(.025)**	.0179	20.42(.92)	.851(.016)	.3156	19.93(.49)
FLLS-II	.937(.012)	.0246	19.82(.51)	.653(.034)	.0191	19.55(.19)	**.866(.016)**	.3166	19.84(.13)

Algorithm	basehock			splice			kr-V-kp		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.837(.055)	.1148	20.00(.00)	.796(.019)	.0072	20.00(.00)	.812(.040)	.0070	20.00(.00)
RFESL	.892(.025)	.3481	20.00(.00)	.801(.013)	.0279	20.00(.00)	.781(.011)	.0280	20.00(.00)
RFLLS	.664(.065)	.1161	20.00(.00)	.785(.021)	.0078	20.00(.00)	.839(.040)	.0073	20.00(.00)
RFLLS-I	.867(.033)	.1163	20.00(.00)	**.821(.014)**	.0080	20.00(.00)	.868(.037)	.0074	20.00(.00)
RFLLS-II	.884(.021)	.1187	20.00(.00)	**.812(.018)**	.0088	20.00(.00)	.875(.029)	.0082	20.00(.00)
PEA	.946(.015)	.5014	19.96(.08)	.792(.018)	.0440	19.98(.35)	.782(.028)	.0450	19.77(.42)
FLLS	.747(.046)	.5148	19.79(.37)	**.820(.012)**	.0465	20.34(.07)	**.897(.020)**	.0454	20.41(.24)
FLLS-I	.955(.010)	.5159	19.87(.60)	**.819(.019)**	.0458	20.31(.49)	.805(.027)	.0455	19.98(.35)
FLLS-II	**.963(.009)**	.5164	20.19(.10)	**.822(.012)**	.0487	19.58(.30)	.879(.028)	.0458	19.66(.25)

Algorithm	spambase			phishing			a9a		
	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RPEA	.862(.022)	.0094	20.00(.00)	.907(.013)	.0201	20.00(.00)	.570(.079)	.0848	20.00(.00)
RFESL	.865(.005)	.0410	20.00(.00)	.912(.006)	.1129	20.00(.00)	**.778(.004)**	.3305	20.00(.00)
RFLLS	.853(.033)	.0102	20.00(.00)	.900(.015)	.0217	20.00(.00)	.571(.083)	.0895	20.00(.00)
RFLLS-I	.877(.015)	.0107	20.00(.00)	**.920(.005)**	.0217	20.00(.00)	.749(.054)	.0956	20.00(.00)
RFLLS-II	.875(.019)	.0113	20.00(.00)	**.920(.007)**	.0232	20.00(.00)	.618(.082)	.0960	20.00(.00)
PEA	.875(.017)	.0565	20.23(.27)	.907(.010)	.1250	20.15(.42)	.544(.074)	.3604	20.39(.09)
FLLS	**.896(.013)**	.0617	19.73(.31)	**.921(.005)**	.1394	19.76(.53)	.726(.045)	.3730	20.39(.65)
FLLS-I	**.893(.013)**	.0638	19.72(.33)	**.923(.005)**	.1416	19.71(.53)	**.757(.024)**	.3818	19.69(.65)
FLLS-II	**.892(.011)**	.0638	19.63(.38)	**.922(.005)**	.1437	20.28(.83)	.654(.085)	.3881	19.76(.27)

5.3 Comparisons with Simple and Direct Approaches

To verify the necessity of using all the information of the data streams as much as possible, in this subsection, we take FLLS-I as an example and compare it with its two simplified versions, FLLS -I$_{SN}$ and FLLS -I$_{SI}$ on six data sets. FLLS -I$_{SN}$ simply takes advantage of the newly obtained labeled instance to learn a new model for classification when the feature dimension of instance increases. FLLS -I$_{SI}$ only uses the initial features to update its model, that is, the data streams in FLLS -I$_{SI}$ is $\mathbf{x}_t \in \mathbb{R}^{d_1}$.

Table 5. AUC values of FLLS-I and its two simplified versions on six datasets with the query ratio fixed to about 10%.

Dataset	FLLS -I$_{SN}$		FLLS -I$_{SI}$		FLLS-I	
	AUC	Query(%)	AUC	Query(%)	AUC	Query(%)
air	.712(.141)	10.11(.09)	.760(.054)	10.33(.56)	**.951(.039)**	10.00(.39)
vote	.678(.208)	10.01(.03)	.682(.049)	10.04(.25)	**.937(.028)**	10.01(.76)
basehock	.546(.078)	9.97(.07)	**.929(.018)**	9.97(.23)	**.934(.015)**	9.99(.19)
spambase	.604(.065)	9.94(.07)	.727(.013)	9.95(.52)	**.897(.006)**	10.11(.41)
phishing	.582(.083)	9.98(.08)	.618(.007)	10.08(.71)	**.925(.005)**	10.00(.26)
a9a	.751(.030)	10.07(.09)	.765(.006)	10.50(.17)	**.837(.008)**	9.92(.80)

We set $B = 1$, i.e., all features are used for learning at each round t. We adjust ρ to make the query ratio near 10%. From Table 5 we can observe that FLLS-I achieves significantly better results than FLLS -I$_{SN}$ and FLLS -I$_{SI}$, which indicates that the abandonment of data information will lead to a significant degeneration of model performance. The above results further reveal the necessity of our methods to make full use of data information.

5.4 Comparisons with all Features Accessed FLLS-I

In this subsection, we take FLLS-I as an example and conduct experiments on a special case of FLLS-I, marked as FLLS -I$_f$. Specifically, we assume that FLLS -I$_f$ can access the full features at each round for training, i.e., the data streams in FLLS -I$_f$ is $\mathbf{x}_t \in \mathbb{R}^{d_T}$. We try to explore how close the performance between FLLS-I and FLLS -I$_f$.

We conduct experiments on four data sets, i.e., vote, basehock, phishing, a9a. As can be seen from Fig. 5, the difference in experimental results between FLLS-I and FLLS -I$_f$ is relatively small. FLLS-I is even comparable to FLLS -I$_f$ on some data sets, such as basehock and a9a. Fig. 5 again proves that our proposed methods can effectively extract the information in the data streams, and it can achieve satisfying results even if some features are missing.

5.5 Analysis of Sparsity Strategy

In this subsection, we conduct experiments to test the effectiveness of model sparseness in our algorithms. We replicate different proportions of the original features as the additional features. Thus, the redundancy relative to the features already included is increasing gradually. Then, we test the performance of the proposed algorithms on the original features and redundant features, respectively.

We conduct experiments on four datasets, i.e., vote, svmguide3, PCMAC, a9a. The ACC results with varied query ratios (10%, 50%, 90%) are displayed in Fig. 6. We use the stacked bar to show the results. In each group, from left to right, the algorithms are FLLS, FLLS-I, and FLLS-II. The ACC results on redundant features are plotted by green face. If the results on original features are better than the results on redundant features, we add a yellow bar on the green face. It can be observed that the performances of our algorithms on the original dataset, with and without additional redundant features, are similar, which indicates the effectiveness of the model sparseness part in our proposed algorithms.

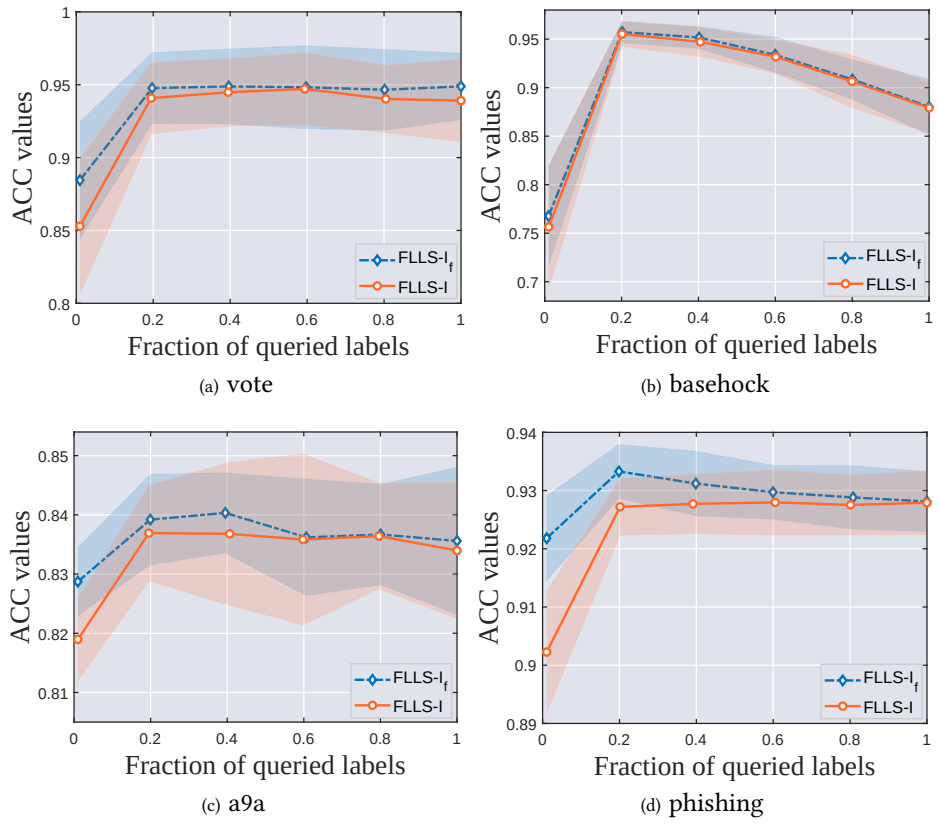

Fig. 5. ACC Performance of algorithms FLLS -I$_f$ and FLLS-I on four different datasets.

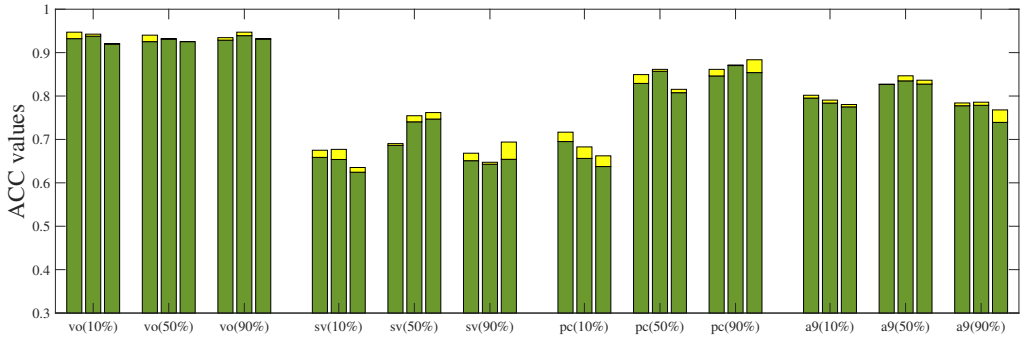

Fig. 6. ACC Performance of the proposed algorithms on original features and redundant features, respectively. Each group corresponds to the results on a data set (vo, sv, pc and a9 represent the abbreviations for data sets vote, svmguide3, PCMAC, and a9a, respectively) with query ratio (10%, 50%, 90%). In each group, the results on redundant features are plotted by green face and the yellow face represents the decrease of the results on original datasets compared to the results on redundant features. In each group, from left to right, the algorithms are FLLS, FLLS-I, and FLLS-II.

5.6 Parameter Analysis

Here we study the parameter sensitivity of our algorithms on two data sets, dna and splice. There are two important parameters in proposed algorithms, i.e., penalty cost parameter C and ratio of selected features B. First we fix $B = 0.5$ and tune C in $2^{[-5:5]}$, then we fix $C = 1$ and tune B in $\{0.04, 0.08, 0.16, 0.32, 0.64\}$. All these experiments are conducted under the query ratio fixed to about 50%.

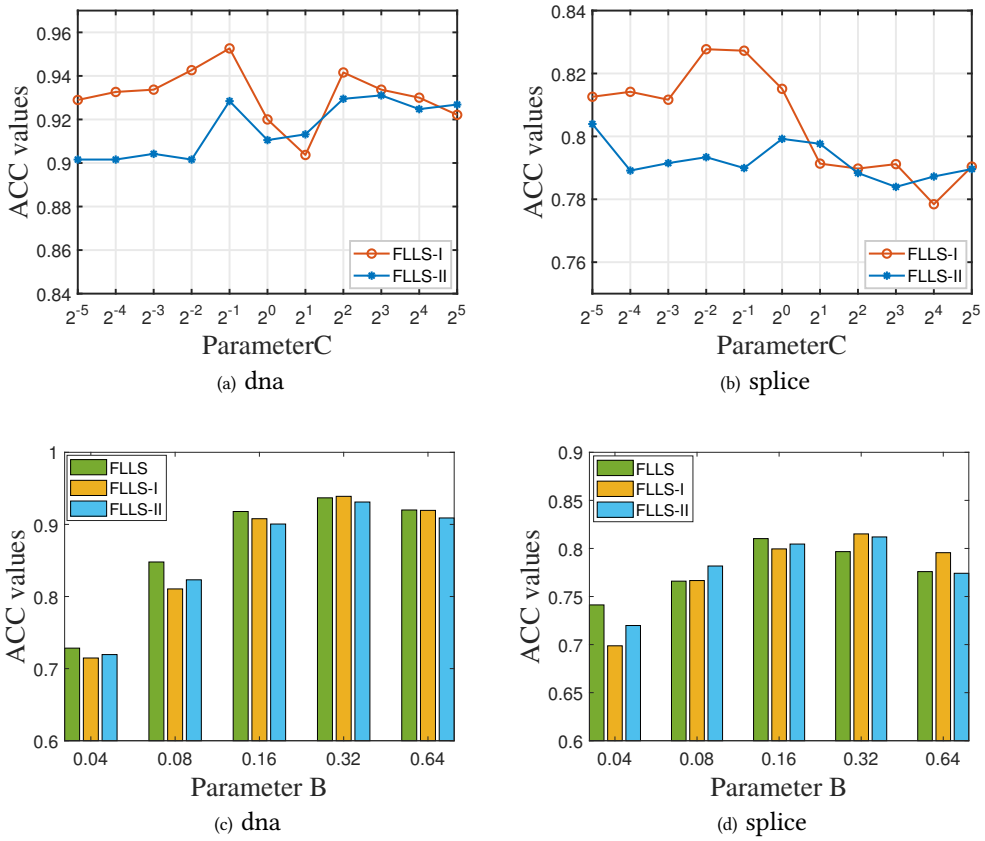

Fig. 7. The ACC results on two data sets with respect to parameter C and B.

As can be seen from Fig. 7, (a) and (b) show the performance of FLLS-I and FLLS-II under different values of C. FLLS-I and FLLS-II are not very sensitive to parameter C in a wide range. Besides, the larger the value of C, the closer the performance of FLLS-I and FLLS-II. This is because the value of τ_t in FLLS-I and FLLS-II is getting closer and closer as C increases, resulting in their models tending to be consistent. When the value of C is large enough, FLLS-I and FLLS-II degenerate to FLLS. (c) and (d) show the performance of the proposed algorithms on different values of B. We can see that a larger value of B may not necessarily lead to better performance, indicating that the sparsity strategy can not only improve the memory usage and running time efficiency, but also ensure that the proposed algorithms have superior performance.

5.7 Real-World Applications

In this subsection, we apply FLLS and its two variants on two real-world datasets, i.e., rcv1 and RFID. rcv1[5] is a text classification dataset, which aims to classify the JMLR articles into different groups. Generally, new articles are published continuously with new research topics, so this setting can be regarded as trapezoidal data streams learning. The "RFID" data stream[6] is collected by Hou et al[3] using the RFID technique. Each RFID aerial keeps receiving the tag signals on each round. To ensure continuous signal reception, new aerials are deployed beside the old ones before the aerials expired. During this overlapping period, we achieve data streams from both historical and augmented feature spaces, which indicates that the volume and dimension of data streams increase over time. Therefore, the RFID data collected by Hou also satisfies our assumptions.

Table 6 shows the average AUC values and running time of these nine methods with the query ratio near 10% and 20%, the best AUC result and its comparable results are highlighted in boldface based on the paired t-tests at 95% significance level. Fig. 8 show the average ACC results of these nine methods with varied ratios of queried instances. We can observe from Table 6 and Fig. 8 that the proposed methods achieve significantly better results than the compared methods, which indicates that our proposed methods can also achieve good performance in real-world applications.

Table 6. AUC Performance of algorithms on two real-world data sets with the query ratio fixed to about 10% and 20%.

Dataset	Algorithm	Request 10% labels			Request 20% labels		
		AUC	Time(s)	Query(%)	AUC	Time(s)	Query(%)
RFID	RPEA	.721(.060)	0.0031	10.00(.00)	.746(.049)	0.0035	20.00(.00)
	RFESL	.752(.065)	0.0101	10.00(.00)	.778(.038)	0.0118	20.00(.00)
	RFLLS	.729(.037)	0.0039	10.00(.00)	.750(.048)	0.0042	20.00(.00)
	RFLLS-I	.749(.040)	0.0042	10.00(.00)	.775(.029)	0.0045	20.00(.00)
	RFLLS-II	.740(.035)	0.0048	10.00(.00)	.787(.037)	0.0051	20.00(.00)
	PEA	.705(.051)	0.0167	9.93(.10)	.763(.041)	0.0192	20.21(.16)
	FLLS	**.785(.022)**	0.0183	9.96(.15)	**.809(.020)**	0.0203	19.52(.18)
	FLLS-I	.759(.030)	0.0185	10.06(.46)	.777(.021)	0.0213	19.57(.72)
	FLLS-II	**.789(.029)**	0.0190	10.09(.08)	**.806(.016)**	0.0235	20.27(.16)
rcv	RPEA	.835(.044)	0.3495	10.00(.00)	.853(.034)	0.3520	20.00(.00)
	RFESL	.912(.004)	0.5575	10.00(.00)	.912(.048)	0.5844	20.00(.00)
	RFLLS	.797(.034)	0.3496	10.00(.00)	.794(.048)	0.3521	20.00(.00)
	RFLLS-I	.919(.006)	0.3506	10.00(.00)	.920(.006)	0.3533	20.00(.00)
	RFLLS-II	.891(.022)	0.3512	10.00(.00)	.894(.017)	0.3542	20.00(.00)
	PEA	.890(.011)	1.3259	10.06(.03)	.886(.017)	1.3516	19.88(.03)
	FLLS	.848(.014)	1.3407	9.87(.03)	.843(.021)	1.3763	20.40(.03)
	FLLS-I	**.937(.003)**	1.3626	9.82(.23)	**.937(.004)**	1.3778	19.87(.16)
	FLLS-II	.912(.007)	1.3712	9.96(.04)	.910(.005)	1.3799	20.42(.06)

6 CONCLUSION

In this paper, we aim to learn a highly dynamic model from trapezoidal data streams with label scarcity and propose a new algorithm called incremental feature spaces learning with label scarcity (FLLS), together with its two variants

[5] http://www.csie.ntu.edu.tw/~cjlin/libsvm
[6] http://www.lamda.nju.edu.cn/data_RFID.ashx

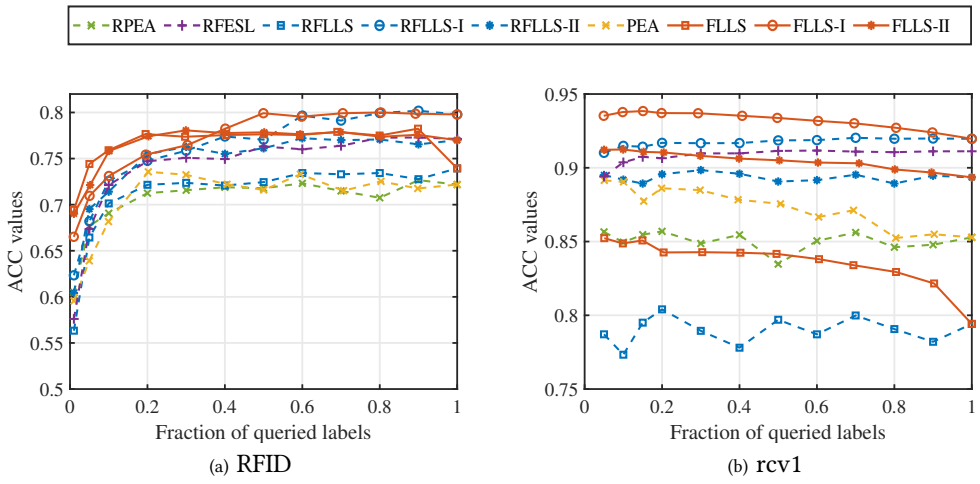

Fig. 8. The ACC results on two real-world data sets.

FLLS-I and FLLS-II. Our approaches are particularly useful when labels are scarce and feature spaces are increasing. We first leverage the margin-based online active learning to annotate the most valuable instances and thus a build superior model with minimal supervision. After receiving the label, we combine the passive-aggressive update rule and margin-maximum principle to jointly update the dynamic classifier in the shared and augmented feature space. Theoretical and empirical studies demonstrate the effectiveness of our proposed algorithms.

Since our proposed algorithms are designed for linear tasks, we think it is an interesting and vital work to extend our proposal into a nonlinear case. How to design a nonlinear classifier in the setting of incremental feature spaces learning is one of our future works. In addition, how to exploit more valuable information within the data streams, such as second-order information, distribution information, and how to conduct clustering with evolving feature space are also interesting works. These works are quite useful in practice and we will do further study on them.

7 ACKNOWLEDGEMENTS

This work was partially supported by the Key NSF of China under Grant No. 62136005, the NSF of China under Grant No. 61922087, Grant No. 61906201, and Grant No. 62006238, the NSF for Distinguished Young Scholars of Hunan Province under Grant No. 2019JJ20020. Chenping Hou and Yuhua Qian are the corresponding authors.

REFERENCES

[1] Zhenyu Zhang, Peng Zhao, Yuan Jiang, and Zhi-Hua Zhou. Learning with feature and distribution evolvable streams. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119, pages 11317–11327. PMLR, 2020.

[2] Qin Zhang, Peng Zhang, Guodong Long, Wei Ding, Chengqi Zhang, and Xindong Wu. Online learning from trapezoidal data streams. *IEEE Transactions on Knowledge and Data Engineering*, 28(10):2709–2723, 2016.

[3] Bo-Jian Hou, Lijun Zhang, and Zhi-Hua Zhou. Learning with feature evolvable streams. In *Advances in Neural Information Processing Systems 30, Long Beach, CA, USA*, pages 1417–1427, 2017.

[4] Chenping Hou and Zhi-Hua Zhou. One-pass learning with incremental and decremental features. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(11):2776–2792, 2018.

[5] Di Wu, Yi He, Xin Luo, Mingsheng Shang, and Xindong Wu. Online feature selection with capricious streaming features: A general framework. In *IEEE International Conference on Big Data, Los Angeles, CA, USA*, pages 683–688. IEEE, 2019.

[6] Steven C. H. Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *CoRR*, abs/1802.02871, 2018.

[7] Dehua Liu, Peng Zhang, and Qinghua Zheng. An efficient online active learning algorithm for binary classification. *Pattern Recognit. Lett.*, 68:22–26, 2015.

[8] Liyao Ma, Sébastien Destercke, and Yong Wang. Online active learning of decision trees with evidential data. *Pattern Recognit.*, 52:33–45, 2016.

[9] Shuji Hao, Jing Lu, Peilin Zhao, Chi Zhang, Steven C. H. Hoi, and Chunyan Miao. Second-order online active learning and its applications. *IEEE Trans. Knowl. Data Eng.*, 30(7):1338–1351, 2018.

[10] Lei Zhu, Shaoning Pang, Abdolhossein Sarrafzadeh, Tao Ban, and Daisuke Inoue. Incremental and decremental max-flow for online semi-supervised learning. *IEEE Trans. Knowl. Data Eng.*, 28(8):2115–2127, 2016.

[11] Peilin Zhao, Dayong Wang, Pengcheng Wu, and Steven C. H. Hoi. A unified framework for sparse online learning. *ACM Trans. Knowl. Discov. Data*, 14(5), August 2020.

[12] Bo-Jian Hou, Lijun Zhang, and Zhi-Hua Zhou. Prediction with unpredictable feature evolution. *CoRR*, abs/1904.12171, 2019.

[13] Bo-Jian Hou, Yu-Hu Yan, Peng Zhao, and Zhi-Hua Zhou. Storage fit learning with feature evolvable streams. *CoRR*, abs/2007.11280, 2020.

[14] Peng Zhou, Peipei Li, Shu Zhao, and Xindong Wu. Feature interaction for streaming feature selection. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[15] Xuegang Hu, Peng Zhou, Pei-Pei Li, Jing Wang, and Xindong Wu. A survey on online feature selection with streaming features. *Frontiers Comput. Sci.*, 12(3):479–493, 2018.

[16] Xindong Wu, Kui Yu, Wei Ding, Hao Wang, and Xingquan Zhu. Online feature selection with streaming features. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(5):1178–1192, 2013.

[17] Ege Beyazit, Jeevithan Alagurajah, and Xindong Wu. Online learning from data streams with varying feature spaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3232–3239. AAAI Press, 2019.

[18] Yi He, Baijun Wu, Di Wu, Ege Beyazit, and Xindong Wu. Toward mining capricious data streams: A generative approach. *IEEE Transactions on Neural Networks and Learning Systems*, PP(99):1–13, 2020.

[19] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434, 2006.

[20] Andrew B. Goldberg, Ming Li, and Xiaojin Zhu. Online manifold regularization: A new learning setting and empirical study. In *Machine Learning and Knowledge Discovery in Databases*, volume 5211, pages 393–407, 2008.

[21] Mehrdad Farajtabar, Amirreza Shaban, Hamid Reza Rabiee, and Mohammad Hossein" Rohban. Manifold coarse graining for online semi-supervised learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 391–406, 2011.

[22] Atsutoshi Kumagai and Tomoharu Iwata. Learning dynamics of decision boundaries without additional labeled data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 1627–1636. ACM, 2018.

[23] Nicolo Cesa-Bianchi, Alex Conconi, and Claudio Gentile. Learning probabilistic linear-threshold classifiers via selective sampling. In *Learning Theory and Kernel Machines*, pages 373–387. Springer, 2003.

[24] Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Worst-case analysis of selective sampling for linear classification. *J. Mach. Learn. Res.*, 7:1205–1230, 2006.

[25] Peilin Zhao and Steven C. H. Hoi. Cost-sensitive online active learning with application to malicious URL detection. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 919–927. ACM, 2013.

[26] Jing Lu, Peilin Zhao, and Steven C. H. Hoi. Online passive aggressive active learning and its applications. In *Proceedings of the Sixth Asian Conference on Machine Learning, ACML ,Nha Trang City, Vietnam*, volume 39. JMLR.org, 2014.

[27] Yoram Baram, Ran El-Yaniv, and Kobi Luz. Online choice of active learning algorithms. *J. Mach. Learn. Res.*, 5:255–291, 2004.

[28] Shuji Hao, Peiying Hu, Peilin Zhao, Steven C. H. Hoi, and Chunyan Miao. Online active learning with expert advice. *ACM Trans. Knowl. Discov. Data*, 12(5):58:1–58:22, 2018.

[29] Shuji Hao, Steven C. H. Hoi, Chunyan Miao, and Peilin Zhao. Active crowdsourcing for annotation. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT, Volume II*, pages 1–8. IEEE Computer Society, 2015.

[30] H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory, COLT*, pages 287–294. ACM, 1992.

[31] Yi He, Xu Yuan, Sheng Chen, and Xindong Wu. Online learning in variable feature spaces under incomplete supervision. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pages 4106–4114. AAAI Press, 2021.

[32] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, 2006.

[33] S. Boyd and L. Vandenberghe. *Convex Optimization*. Convex Optimization, 2004.

[34] Jing Lu, Peilin Zhao, and Steven C. H. Hoi. Online passive-aggressive active learning. *Mach. Learn.*, 103(2):141–183, 2016.

[35] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[36] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.