Semi-Supervised Streaming Learning with Emerging New Labels

Yong-Nan Zhu and **Yu-Feng Li***
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{zhuyn, liyf}@lamda.nju.edu.cn

Abstract

In many real-world applications, the modeling environment is usually dynamic and evolutionary, especially in a data stream where emerging new class often happens. Great efforts have been devoted to learning with novel concepts recently, which are typically in a supervised setting with completely supervised initialization. However, the data collected in the stream are often in a semi-supervised manner actually, which means only a few of them are labeled while the great majority miss ground-truth labels. Besides, new classes hidden in unlabeled instances bring more challenges for the learning task. In this paper, we tackle these issues by a new approach called SEEN which consists of three major components: an effective novel class detector based on clustering random trees, a robust classifier for predictions on the known classes, and an efficient updating process that ensures the whole framework adapts to the changing environment automatically. The classifier produces known labels via label propagation that utilizes all labeled and part unlabeled data in the past which naturally describe the entire stream seen so far. Empirical studies on several datasets validate that the algorithm can accurately classify points on a dynamic stream with a small number of labeled examples and emerging new classes.

Introduction

In traditional machine learning, many advanced approaches have been proposed based on the assumption that the learning environment is stationary. However, the learning environment is often dynamic in practical applications (Zhou 2016), especially when learning with a data stream. In particular, during the data stream, *novel classes* may often emerge. For example, while mining news webpages, a new hot topic often arises with time. Such new setting attracts much attention in recent years (Masud et al. 2011; Haque, Khan, and Baron 2016; Mu et al. 2017).

Previous efforts on novel class typically work on a supervised setting which assumes that all the training data are completely labeled except for the novel class. However, in reality, the data collected in the stream are often in a semi-supervised manner (Chapelle, Schölkopf, and Zien 2006; Zhu, Goldberg, and Khot 2009), since it is often unavailable to access the true labels for all instances in a data stream due to realistic constraints such as the time, the labeling cost, etc.

To cope with this new yet realistic setting, in this paper we propose SEEN (**SE**mi-supervised streaming learning with **E**merging **N**ew labels) to address the dynamic semi-supervised learning problems. The proposed method contains three main components: (1) a specific designed detector based on the features of instances to determine whether an unlabeled instance belongs to the new class or known classes; (2) a robust graph-based semi-supervised classifier that takes advantage of both labeled and unlabeled data to make predictions on a new sample if it is likely to be part of known classes; (3) a novel updating process which utilizes the detected new class data to remodel both the detector and the classifier. The three parts collaborate to adapt the learned model to the changing environment, and achieve satisfactory outcomes.

Our main contributions are concluded as follows:

- The presented semi-supervised learning framework is capable of handling emerging new class in a dynamic data stream where a few labeled instances are collected together with a large number of unlabeled instances.
- The model is learned without an initial large labeled training set and works naturally and reliably in streaming settings, even if there are only a few labeled data.
- Comprehensive experiments conducted on a number of benchmark datasets validate the effectiveness and efficiency of the proposed method.

The rest of this paper starts with an introduction to related work. Then our proposal is presented, which is followed by extensive empirical justification. The paper ends with the section of the conclusion.

Related Work

Incremental learning requires the flexible and efficient adaptation of models to an open and dynamic environment, and class-incremental learning (C-IL) (Zhou and Chen 2002; Fink et al. 2006; Scheirer et al. 2012; Kuzborskij, Orabona,

*This research was supported by the National Key R&D Program of China (2018YFB1004300) and the NSFC (61772262). Yu-Feng Li is the corresponding author of this work.

and Caputo 2013; Sun et al. 2016) is a particularly significant branch that focuses on the emerging classes. Classification under data stream with emerging new classes is a streaming C-IL problem and some efforts have been put in this field in recent years. These existing methods include clustering-based methods (Masud et al. 2011; Haque, Khan, and Baron 2016), tree-based methods (Mu, Ting, and Zhou 2017; Zhu, Ting, and Zhou 2018), matrix sketch (Mu et al. 2017).

The ECSMiner (Masud et al. 2011) firstly studies three major problems of streaming classification: infinite length, concept drift, and concept evolution. It also maintains an ensemble of classification models to make delayed prediction under a maximum allowable wait time. But the basic assumption of ECSMiner is that true labels of samples can be obtained after some time delay, which may be impractical in many cases. SAND (Haque, Khan, and Baron 2016) estimates classifier confidence in predicting instances from evolving data stream and dynamically determines chunk boundaries. Instead of requiring true labels of all instances, SAND intelligently selects a few instances using the estimated classifier confidence scores. This kind of active selection (Settles 2009) is not naturally in general semi-supervised settings. SENCForest (Mu, Ting, and Zhou 2017) uses the isolation-based anomaly detection method (Liu, Ting, and Zhou 2008) to construct the classifier and detector. It builds completely-random trees for anomaly detection, and an effective model retiring and growing mechanism is proposed to meet limited memory constraints.

Besides, in (Mu et al. 2017), a matrix sketch method based on the technique Frequent-Directions (Liberty 2013) is used to approximate original information. The global sketching is produced on the whole dataset for new class detection, whereas the local sketching is built on each class as local information for classification. These approaches have achieved good performance as most of them rely on a large labeled initial training set. In addition, they don't make the greatest use of unlabeled data.

Our work also relates to novel class detection (Abdallah et al. 2016; Spinosa and de Leon Ferreira 2004; Ma and Perkins 2003) and anomaly detection (Liu, Ting, and Zhou 2008; Breunig et al. 2000; Na, Kim, and Yu 2018) which focus on the identification of data which have not been seen during the training process. However, they only study subproblems of our setting, ignoring the problem of classification and model update, thus their approaches fail in the streaming context.

Semi-Supervised Learning (SSL) (Chapelle, Schölkopf, and Zien 2006; Zhou and Li 2010; Wei et al. 2018; Li, Guo, and Zhou 2019) aims to make use of unlabeled data for training - typically a small set of labeled data together with a large collection of unlabeled data. Graph-based SSL algorithms (Zhu, Ghahramani, and Lafferty 2003; Zhou et al. 2003; Li, Wang, and Zhou 2016) have a long history of work and propagate limited label information to unlabeled examples following clustering or manifold assumptions. Online graph-based SSL is a relative new research field that has generated considerable interest (Zhu, Goldberg, and Khot 2009; Valko et al. 2010; Ravi and Diao 2016; Wagner et al. 2018). Even though they are applied to points arriving on a stream, they assume unlabeled data have no previously unseen labels and are limited in the stationary environments. Therefore, these solutions cannot handle novel classes in evolving streams.

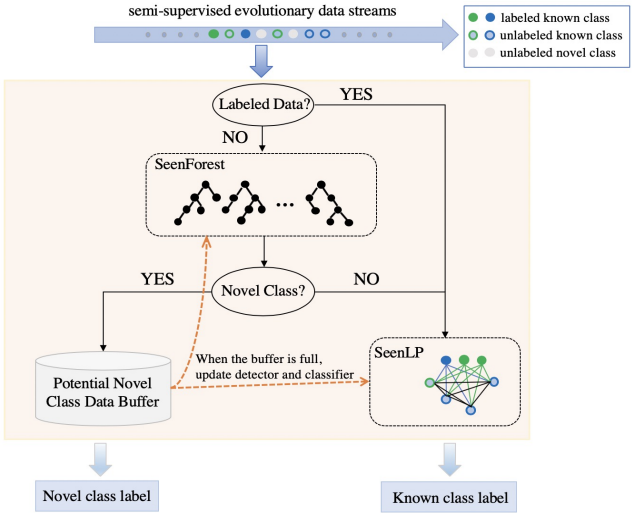

Figure 1: Illustration for our proposed SEEN framework.

The SEEN Method

In this section, we propose an efficient algorithm called SEEN to deal with the dynamic semi-supervised evolutionary data streams. SEEN consists of three main parts: an effective novel class detector, a robust classifier for prediction and an efficient updating process. We first present the problem formulation and then provide an overview of the overall training procedure. The concrete details in the procedure are provided in the following contents.

Problem Formulation

In open dynamic semi-supervised learning problems, the instances are collected successively from a data stream. Due to practical issues, such as time and resource constraints, only a little data are labeled while a large amount of them miss label information. At the beginning time, all samples belong to known classes. Let S denote streaming data and define $S = \{(\boldsymbol{x}_t, y_t)\}_{t=1}^{T_0}$ as the data in the first T_0 time period. Let $Y = \{1, 2, \cdots, K\}$ be the known labels set. The arriving data stream has a instance $\boldsymbol{x}_t$ observed at time t and $y_t \in Y' = \{-1, 1, 2, \cdots, K\}$. If $y_t = -1$, $\boldsymbol{x}_t$ is an unlabeled instance, but the true label of $\boldsymbol{x}_t$ is in set Y. By the way, these relatively pure data can be used to initialize the classifier and detector.

As time goes by, evolution happens in the following data stream $S' = \{(\boldsymbol{x}_t, y_t)\}_{t=T_0+1}^{\infty}$ which contains novel class instances. Specifically, there exists unlabeled instance $(\boldsymbol{x}_{t'}, y_{t'}) \in S'$ where $y_{t'} = -1$, but the true label of $\boldsymbol{x}_{t'}$ is **not** in set Y. Note that if $y_{t'} \neq -1$, $y_{t'} \in Y$ holds forever.

The above problem can have many different variations. For example, after the first period, there follow several pe-

riods each of which includes different novel classes. Further to say, after receiving unlabeled novel class instances in previous periods, novel class labels are available along with subsequent streaming data. As a result, the known labels set Y may be enlarged along with the evolutionary data stream. In empirical studies, the presented method performs well in those variations.

SEEN: An Overview

A schematic description of the overall training procedure of our method is given in Figure 1. As shown in Figure 1, if a newly arrived instance $\boldsymbol{x}_t$ is labeled in the data stream, it is then used to update classifier C directly. Otherwise, it is identified by an effective detector D built with known class data. The detector outputs 1 if $\boldsymbol{x}_t$ is likely to hold a new label, and $\boldsymbol{x}_t$ is then added into a temporal potential novel class data buffer B whose maximum buffer size is s. On the contrary, the detector outputs -1 and $\boldsymbol{x}_t$ is then classified by classifier C. The classifier is also updated by $\boldsymbol{x}_t$ simultaneously while making predictions. For novel class instances, the update process of the classifier and detector begins when the number of examples in buffer B reaches the preset maximum buffer size. We build an initial classifier and detector by the data in the beginning period of the stream. Algorithm 1 summarizes the approach.

Algorithm 1 SEEN

Input: detector D, classifier C, buffer B, collector O, maximum buffer size s
Output: y - class label for each unlabeled $\boldsymbol{x}$ in a data stream

```
1: while not end of data stream do
2:   for each (x, y) do
3:     O ← O ∪ {x}
4:     if y ≠ −1 then
5:       Update classifier C by x
6:     else
7:       Identify x by detector D
8:       if x is likely to have novel label then
9:         Output novel label for x
10:        B ← B ∪ {x}
11:        if |B| ≥ s then
12:          Update classifier C by buffer B
13:          Update detector D by collector O
14:          B ← empty set
15:          O ← select a subset of O randomly
16:        end if
17:      else
18:        Output the prediction on x by classifier C
19:      end if
20:    end if
21:  end for
22: end while
```

As shown in lines 3 and 13 of Algorithm 1, to build and update detector D, a data collector O stores a subset of previously observed data. To meet the demand for storage and computational complexity, only a subset of all historical data is randomly sampled. When $|B| = s$, the detected new class data are sufficient to train and update a good performing classifier. In this situation, the instances in D are all labeled with the new label. When the unlabeled data of known classes are classified as presented in line 18 of Algorithm 1, it then follows the workflow in Algorithm 3 (detailed in the following parts). That is to say, the classifier learns from all data with different levels of label information.

In the following sections, we detail the construction of the detector, the classifier, and their updates.

New Class Detection: SEENForest

Inspired by the fact that the appearances of a new class may be attributed to previously unseen set of feature values, we take feature space into account and build SEENForest which is similar to the early work iForest (Liu, Ting, and Zhou 2008) for unsupervised anomaly detection. SEENForest consists of many SEENTrees, and each SEENTree is built using a random subset of input training set O of size ϕ. Algorithm 2 summarizes the construction of SEENTree.

Algorithm 2 SEENTree

Input: input dataset S, current tree height h, maximum tree height h_m, number of randomly selected attributes k
Output: SEENTree

```
1: Construct a ball whose r = max_{x∈S} ‖x − c‖ where
   c = mean_{x∈S}(x)
2: if |S| = 1 or h ≥ h_m then
3:   return LeafNode{Center ← c, Radius ← r }
4: else
5:   Select k attributes q from the input feature set in S
     randomly
6:   Get two cluster centers {c_1, c_2} based on the selected
     k attributes of S
7:   S_l = {x ∈ S | ‖x^q − c_1‖ ≤ ‖x^q − c_2‖}
8:   S_r = {x ∈ S | ‖x^q − c_1‖ > ‖x^q − c_2‖}
9:   return InNode{Center ← c, Radius ← r,
                   SelectedAtt ← q,
                   SplitCenters ← {c_1, c_2},
                   Left ← SEENTree(S_l, h + 1, h_m, k),
                   Right ← SEENTree(S_r, h + 1, h_m, k) }
10: end if
```

While building trees for detection, compared with iForest which randomly selects an attribute and its cutpoint between the minimum and maximum values, SEENForest selects an attribute set with fixed size and then the split is an outcome of a clustering process based on the selected attributes. Projected on a set of randomly selected attributes, each internal node in SEENTree is split based on a cluster center on either branch. This strategy ensures instances within the same leaf node must be similar in some attributes of features.

Another main difference shows in the evaluation procedure. iForest employs the average path length, that the test instance traverses over all trees, as the anomaly score. Shorter path length indicates that the instance is more likely to be an anomaly. To detect emerging new classes, we consider more about the specific character of each SEENTree. Specifically, we first calculate the average path length be-

tween each node and the root which characterizes structural information of the tree. Let $L_{i,t}$ be the i-th node's distance from root node in t-th SEENTree which has m_t nodes in total, including leaf nodes. Then the threshold for identifying a new class is defined as:

$$\tau_t = \frac{1}{m_t} \sum_{i=1}^{m_t} L_{i,t} \tag{1}$$

Recall that a ball is constructed in each node based on all training instances which fall into the node as shown in the first line of Algorithm 2. A testing instance is likely to be a novel class instance if it falls outside the ball; otherwise, it has a known class label. The radius of the ball is defined as:

$$r = \max_{\boldsymbol{x} \in S} \|\boldsymbol{x} - \boldsymbol{c}\| \tag{2}$$

where S is the set of all training instances falling into the node, and $\boldsymbol{c} = \text{mean}_{\boldsymbol{x} \in S}(\boldsymbol{x})$ is the center of the ball. Here the ball-shaped constraint is for local regions, rather than global data distribution. It is very common to adopt ball-shaped constraint for a local area. For the local distribution of non-ball-shaped class, it is often a good approximation to adopt ball-shaped constraint under the case of smoothness.

We then obtain the path length l_t when a testing instance firstly falls outside one node in the t-th SEENTree. If $l_t < \tau_t$, it is classified as novel class instance; otherwise, it owns known label. The final output of SEENForest is decided via majority voting. Note that the travel of testing instances is based on the distance to the centers of nodes and it selects the closest node of two son nodes to travel.

Known Classes Classification: SEENLP

In order to make accurate predictions on a large amount of newly arrived unlabeled points based on a few labeled instances, we introduce SEENLP to solve this issue. SEENLP is an online variant of label propagation algorithm (Zhou et al. 2003; Ravi and Diao 2016; Wagner et al. 2018) and capable of handling with novel class instances in the semi-supervised dynamic data stream.

We suppose that there are l labeled points $\{(\boldsymbol{x}_i, y_i)_{i=1}^{l}\}$, and u unlabeled points $\{(\boldsymbol{x}_i)_{i=l+1}^{l+u}\}$; typically $l \ll u$. Let $n = l + u$ be the total number of data points. Consider a connected graph $\mathcal{G} = (V, E)$ with nodes V corresponding to the data points, with nodes $L = \{1, \cdots, l\}$ corresponding to the labeled points with labels, and nodes $U = \{l+1, \cdots, l+u\}$ corresponding to the unlabeled points.

For vanilla label propagation, the task is to learn a real-valued function $f : V \to \mathcal{R}$ on $\mathcal{G}$ to assign labels for unlabeled points. Then the energy function (Zhu, Ghahramani, and Lafferty 2003) is defined as:

$$E(f) = \frac{1}{2} \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j} W_{\boldsymbol{x}_i, \boldsymbol{x}_j} (f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j))^2 \tag{3}$$

where W is an $n \times n$ symmetric weight matrix.

By the harmonic solution, we get the closed-form formula of the above optimization problem:

$$\boldsymbol{f}_u = -G_{uu}^{-1} G_{ul} \boldsymbol{f}_l \tag{4}$$

Algorithm 3 SEENLP

Input: label set $Y = \{1, 2, \cdots, K\}$, super nodes set $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_K\}$, labeled nodes set $V_l = \{L_1, L_2, \cdots, L_K\}$, unlabeled nodes set V_u, weight matrix W, τ
Output: y - class label for each unlabeled $\boldsymbol{x}$ in a data stream

initialize V, V_l, V_u as empty sets and W as zero matrix
while not end of data stream **do**
  for each $(\boldsymbol{x}, y)$ **do**
    if $y \neq -1$ **then**
      if y not in Y **then**
        $Y \leftarrow Y \cup \{y\}$; // add new class
        $V \leftarrow V \cup \{\boldsymbol{v}_y\}$; $V_l \leftarrow V_l \cup \{L_y\}$
      end if
      $L_y \leftarrow L_y \cup \{\boldsymbol{x}\}$ // add labeled node
      for each $\boldsymbol{x_u}$ in V_u **do**
        $W_{\boldsymbol{x}_u, \boldsymbol{v}_y} \leftarrow W_{\boldsymbol{x}_u, \boldsymbol{v}_y} + \text{dist}(\boldsymbol{x}_u, \boldsymbol{x})$
      end for
    else
      for each $\boldsymbol{v}_i$ in V_l **do**
        $W_{\boldsymbol{x}, \boldsymbol{v}_i} \leftarrow \sum_{\boldsymbol{x}' \in L_i} \text{dist}(\boldsymbol{x}, \boldsymbol{x}')$
      end for
      for each $\boldsymbol{x}'$ in V_u **do**
        $W_{\boldsymbol{x}, \boldsymbol{x}'} \leftarrow \text{dist}(\boldsymbol{x}, \boldsymbol{x}')$
      end for
      $V_u \leftarrow V_u \cup \{\boldsymbol{x}\}$ // add unlabeled node
      if $|V_u| > \tau$ **then**
        $\boldsymbol{x}_o \leftarrow$ oldest point in V_u
        Remove $\boldsymbol{x}_o$ from V_u
        for each pair $\boldsymbol{p}, \boldsymbol{q} \in V_u$ **do**
          $W_{\boldsymbol{pq}} \leftarrow W_{\boldsymbol{pq}} + W_{\boldsymbol{p x}_o} W_{\boldsymbol{q x}_o} / \sum_{\boldsymbol{x}'} W_{\boldsymbol{x}' \boldsymbol{x}_o}$
        end for
      end if
      Output the harmonic solution of $\boldsymbol{x}$ based on W
    end if
  end for
end while

where $\boldsymbol{f}_l = (f(\boldsymbol{x}_1); \cdots; f(\boldsymbol{x}_l)) = (y_1; \cdots; y_l)$, $\boldsymbol{f}_u = (f(\boldsymbol{x}_{l+1}); \cdots; f(\boldsymbol{x}_{l+u}))$ are the predictions on labeled and unlabeled data, respectively. $G \in \mathbb{R}^{n \times n}$ is the Laplacian matrix of the graph $\mathcal{G}$.

Despite the simpleness and effectivity of label propagation, computing the inverse matrix takes $O(n^3)$ time and $O(n^2)$ memory complexity in general, which makes it infeasible on large-scale datasets (Liang and Li 2018), such as data streams.

Inspired by recent advances in electric networks (Dörfler and Bullo 2013) and online graph-based algorithms (Ravi and Diao 2016; Wagner et al. 2018), SEENLP is presented in Algorithm 3. We introduce *star-mesh* transform on a node $\boldsymbol{v}$ in a graph $\mathcal{G} = (V, E)$. The *star* operation means removing node $\boldsymbol{v}$ from $\mathcal{G}$ with its incident edges while the *mesh* operation indicates updating weight matrix W.

Specifically, if $\boldsymbol{v}$ is removed from the graph, for each pair $\boldsymbol{p}, \boldsymbol{q} \in V$ such that $(\boldsymbol{p}, \boldsymbol{v}) \in E$ and $(\boldsymbol{q}, \boldsymbol{v}) \in E$, we add $W_{\boldsymbol{pq}}$ by $W_{\boldsymbol{pv}} W_{\boldsymbol{qv}} / \sum_{\boldsymbol{v}'} W_{\boldsymbol{v}' \boldsymbol{v}}$.

Consider a data stream $\{(\boldsymbol{x}_t, y_t)\}_{t=1}^{\infty}$ in which some

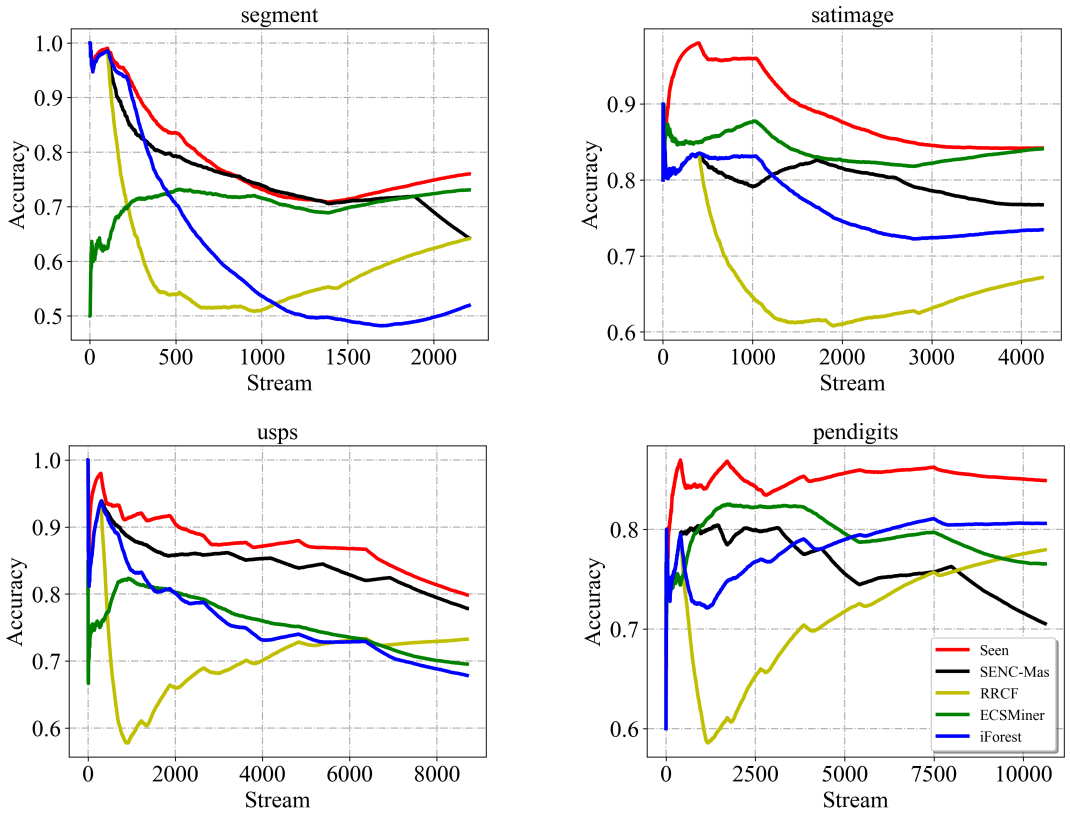

Figure 2: Accuracy results on the data streams of four datasets.

points are labeled and most points are unlabeled, and $y_t \in \{-1, 1, 2, \cdots, K\}$. SeenLP maintains a graph $\mathcal{H}$ that contains the most recent τ unlabeled points and K *super* nodes $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_K\}$ that each *super* node v_i represents one kind labeled points L_i. We categorize the update and classification of SeenLP into two parts according to the label information of the input data. Note that since the points sent for classification have been identified by SeenForest, they can be regarded as known class data.

When a labeled sample arrives, we update the weight matrix of the corresponding *super* node directly without any update with other nodes. When an unlabeled point arrives, we add it to $\mathcal{H}$ firstly and then remove the oldest unlabeled point by a *star-mesh* transform if there already exists τ unlabeled samples. In a nutshell, for each new unlabeled point, there are at most $\tau + K$ nodes for computing the harmonic solution on the graph $\mathcal{H}$.

Even though the *star* operation removes the unlabeled data and their edges, the *mesh* operation reserves the structural information of these points which helps for label propagation as if they still stay in the original graph. Through *star-mesh* transform, the time and space consumption for each newly arrived unlabeled point in the data stream are independent of the length of the data stream which indicates the efficiency of SeenLP. Considering the complexity in solving the harmonic solution, the time and space cost are $O((\tau + K)^3)$ and $O(\tau^2)$, respectively.

Model Update

As shown in Figure 1 and Algorithm 1, when the number of samples in novel class data buffer B reaches maximum buffer size, the model update procedure starts. To update the classifier with these potential novel class examples, the instances in B are marked as the same new label. To make a more robust classification, we select the data which are closest to the center of these candidates for update. Furthermore, a new *super* node and new class node set will be added in V and V_l as shown in lines 5 to 8 in Algorithm 3. The new class data can then be viewed as labeled data with known label.

To update the detector, we sample instances randomly in data collector O which is a subset of previous data to rebuild SeenTrees. This kind of update considers all detected novel class instances as known class instances and is beneficial to detect prospective new classes.

Experiments

In this section, we conduct experiments on four common-used datasets to evaluate the effectiveness of our proposed algorithm.

Experimental Setup

Datasets. To evaluate the predictive performance of the proposed Seen approach, we use four multi-class benchmark datasets ("segment", "satimage", "usps", "pendigits", de-

Table 1: Accuracy and F1 (mean ± std) for SEEN and the compared methods. The best results are in bold.

Dataset	Metric	iForest	RRCF	ECSMiner	SENC-Mas	SEEN
segment	Accuracy	0.520 ± 0.028	0.642 ± 0.025	0.731 ± 0.019	0.641 ± 0.035	$\mathbf{0.760 \pm 0.019}$
	F1	0.567 ± 0.020	0.690 ± 0.020	0.661 ± 0.013	0.695 ± 0.028	$\mathbf{0.790 \pm 0.013}$
satimage	Accuracy	0.735 ± 0.007	0.672 ± 0.036	$\mathbf{0.841 \pm 0.021}$	0.767 ± 0.024	$\mathbf{0.842 \pm 0.008}$
	F1	0.698 ± 0.006	0.685 ± 0.031	0.705 ± 0.017	0.761 ± 0.028	$\mathbf{0.806 \pm 0.008}$
usps	Accuracy	0.678 ± 0.007	0.733 ± 0.021	0.695 ± 0.023	0.778 ± 0.006	$\mathbf{0.798 \pm 0.008}$
	F1	0.656 ± 0.011	0.743 ± 0.024	0.628 ± 0.021	$\mathbf{0.785 \pm 0.006}$	$\mathbf{0.784 \pm 0.010}$
pendigits	Accuracy	0.806 ± 0.020	0.779 ± 0.021	0.765 ± 0.014	0.705 ± 0.008	$\mathbf{0.849 \pm 0.014}$
	F1	0.819 ± 0.018	0.792 ± 0.019	0.704 ± 0.013	0.744 ± 0.006	$\mathbf{0.856 \pm 0.013}$

tailed information is shown in Table 2)[1]. The instances in segment dataset are drawn randomly from a database of 7 outdoor images. The satimage database consists of the multi-spectral values of pixels in 3×3 neighborhoods in a satellite image. The aim is to predict the classification associated with the central pixel in each neighborhood. The usps and pendigit datasets are both digit samples. The usps dataset refers to numeric data obtained from the scanning of handwritten digits from envelopes by the U.S. Postal Service. The images here have been normalized, resulting in 16×16 grayscale images.

Table 2: A summary of datasets used in the experiments.

Dataset	# classes	# attributes	# instances
segment	7	19	2310
satimage	6	32	4435
usps	10	256	9298
pendigits	10	16	10992

Data streams. For a given dataset, the data stream is simulated as follows. In the first period, the semi-supervised data stream only contains known class instances, such as the ground-truth label set of unlabeled points is the same as that of labeled points. In the following each period, a new class appears and the labeled data have labels that have shown in previous periods. The new class labels in different periods except the first period are different. Let C_i^l and C_i^u be the label sets of the labeled and unlabeled data in the i-th period, respectively. For example, we have $C_1^l = C_1^u = C_2^l = \{1, 2\}$, $C_2^u = C_3^l = \{1, 2, 3\}$, $C_3^u = \{1, 2, 3, 4\}$, and so on.

The experiments on each dataset are repeated 10 times with different simulated data streams and both the mean and standard variance of the performance are reported.

Competing algorithms. We compared with: **iForest** (Liu, Ting, and Zhou 2008): it is an unsupervised anomaly detector which can be treated as a new class detector. **RRCF** (Guha et al. 2016): this method investigates a random cut data structure that can be used as a sketch of the input dynamic stream. **ECSMiner** (Masud et al. 2011): it maintains an ensemble framework for classifying data streams and addresses both of concept drift and evolution problems. **SNEC-Mas** (Mu et al. 2017): it uses two low-dimensional matrix sketches for detecting new class and classifying known classes. Since iForest and RRCF do not make classifications, we combine them with SVM as a classifier.

Algorithm settings. Number of trees in iForest is set to 50 and $\psi = 200$. For RRCF, the shingle size is 4. The SVM classifiers in iForest and RRCF are both set to RBF kernel. ECSMiner employs K-means and K is set to 5. In SENC-Mas, $L = N \times 0.8, l_i = n_i \times 0.8$, as suggested in the paper. For all the above methods, the initialization is realized by the data in the first period. For SEENForest, $h_m = 7, s = 50, \phi = 128$, and k is half the number of features. For SEENLP, τ = 50 and we use the standard RBF similarity to inialize the weight matrix, $W_{\boldsymbol{x}_i, \boldsymbol{x}_j} = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\sigma^2)$, where σ can be obtained by cross-validation from the first period. The number of labeled instances in each class in the four datasets as ordered in Table 2 is 20, 50, 80, and 50.

Performance metrics. To evaluate the performance of these approaches, two measures are used in this paper. One is *Accuracy*, $Accuracy = \frac{N_{new}+N_{known}}{N}$, where N is the total number of examples, N_{new} and N_{known} are the number of emerging and known class instances identified correctly. Another measure is macro-averaged F1 performances which produces a combined effect of precision (P) and recall (R) of the detection performance in each class, $F1 = \frac{2 \times P \times R}{P+R}$.

Results

Simulated Streams. We have run 10 independent runs with different simulated streams on the four datasets and both the mean and the standard variance of the performance are reported in Table 1. As can be seen, SEEN outperforms all the other compared approaches and maintains good performance in the whole data streams with different emerging new classes. Besides, compared with the best performance of other methods, our method improves accuracy and F1 by an average of 0.023 and 0.044 respectively.

Figure 2 further shows the change of average accuracy in the whole data stream of the four datasets. The accuracy rate of each point represents the average performance of 10 runs from the very beginning to the present time point. The main reason that RRCF performs badly at the beginning time is the lack of enough training data for novel class detection, and it works better when it is trained with more data. The comparison between SEEN and iForest validates the effectiveness of SEENForest and the necessity of utilizing unlabeled data. The outcome of SENC-Mas is not very robust. Even though ECSMiner is provided with ground-truth labels of unlabeled data for an update, it still performs worse than

[1]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/ multiclass.html

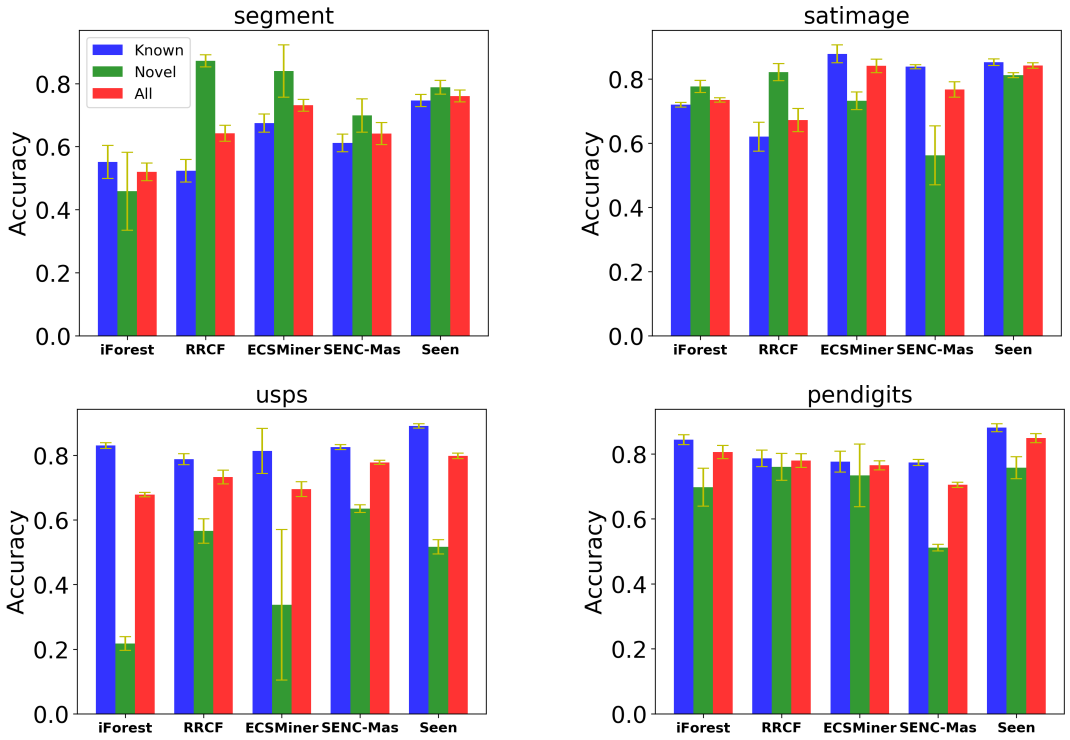

Figure 3: Results of known, novel and all classes, respectively. The yellow segment represents the corresponding variance.

SEEN in most cases.

Figure 3 validates the effectiveness of our systematic solution. The competing approaches either perform badly on one kind of the data or both, or achieve unstable predictions with high variances. On the one hand, for most cases except usps dataset, our framework shows similar performance on known and novel class data, which indicates the adaptivity of SEEN method. On the other hand, the small accuracy rate variance of of our proposal verifies the robustness.

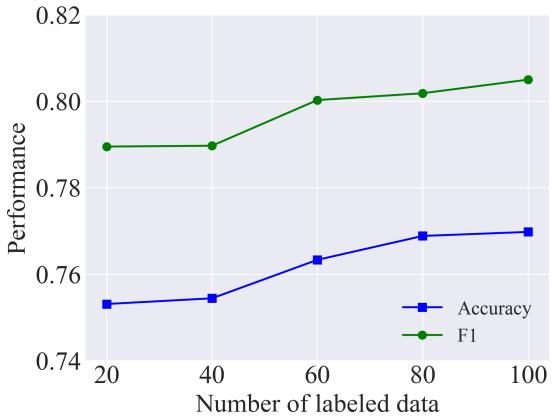

Figure 4: The influence of the number of labeled data in each class on segment dataset.

Parameter analysis. The most significant parameter that impacts the predictions on unlabeled instances is the number of labeled instances. We study its influence on segment dataset and the results are reported in Figure 4. Even with a few labeled examples, SEEN still works well without sacrificing much performance. The performance trends to increase with more labeled instances in general as expectation. Similar results can be obtained from other datasets.

Conclusion

Learning with emerging new classes in a semi-supervised data stream is a very practical yet challenging problem. This is a new kind of learning scenario that to the best of our knowledge, has not been thoroughly studied in literature. In this paper, we propose the SEEN method to tackle the problem. The proposed method consists of a detector for detecting new classes, a classifier based on label propagation that classifies known classes, and their update procedure. Empirical studies on a number of real-world datasets validate the effectiveness of SEEN in handling emerging new classes under semi-supervised streaming data environment. In future, we will consider extending this work to multi-class scenarios with multiple new classes.

References

Abdallah, Z. S.; Gaber, M. M.; Srinivasan, B.; and Krishnaswamy, S. 2016. Anynovel: detection of novel concepts in evolving data streams. *Evolving Systems* 7(2):73–93.

Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; and Sander, J. 2000. Lof: identifying density-based local outliers. *ACM SIGMOD Record* 29(2):93–104.

Chapelle, O.; Schölkopf, B.; and Zien, A., eds. 2006. *Semi-Supervised Learning*. The MIT Press.

Dörfler, F., and Bullo, F. 2013. Kron reduction of graphs with applications to electrical networks. *IEEE Transactions on Circuits and Systems* 60-I(1):150–163.

Fink, M.; Shalev-Shwartz, S.; Singer, Y.; and Ullman, S. 2006. Online multiclass learning by interclass hypothesis sharing. In *Proceedings of the 23rd International Conference on Machine Learning*, 313–320.

Guha, S.; Mishra, N.; Roy, G.; and Schrijvers, O. 2016. Robust random cut forest based anomaly detection on streams. In *Proceedings of the 33rd International Conference on Machine Learning*, 2712–2721.

Haque, A.; Khan, L.; and Baron, M. 2016. SAND: semi-supervised adaptive novel class detection and classification over data stream. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 1652–1658.

Kuzborskij, I.; Orabona, F.; and Caputo, B. 2013. From N to N+1: Multiclass Transfer Incremental Learning. In *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition*, 3358–3365.

Li, Y.; Guo, L.; and Zhou, Z. 2019. Towards safe weakly supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Li, Y.; Wang, S.; and Zhou, Z. 2016. Graph quality judgement: A large margin expedition. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 1725–1731.

Liang, D., and Li, Y. 2018. Lightweight label propagation for large-scale network data. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 3421–3427.

Liberty, E. 2013. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 581–588.

Liu, F. T.; Ting, K. M.; and Zhou, Z. 2008. Isolation forest. In *Proceedings of the 8th IEEE International Conference on Data Mining*, 413–422.

Ma, J., and Perkins, S. 2003. Time-series novelty detection using one-class support vector machines. In *Proceedings of the International Joint Conference on Neural Networks*, 1741–1745.

Masud, M. M.; Gao, J.; Khan, L.; Han, J.; and Thuraisingham, B. M. 2011. Classification and novel class detection in concept-drifting data streams under time constraints. *IEEE Transactions on Knowledge and Data Engineering* 23(6):859–874.

Mu, X.; Zhu, F.; Du, J.; Lim, E.; and Zhou, Z. 2017. Streaming classification with emerging new class by class matrix sketching. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2373–2379.

Mu, X.; Ting, K. M.; and Zhou, Z. 2017. Classification under streaming emerging new classes: A solution using completely-random trees. *IEEE Transactions on Knowledge and Data Engineering* 29(8):1605–1618.

Na, G. S.; Kim, D. H.; and Yu, H. 2018. DILOF: effective and memory efficient local outlier detection in data streams. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1993–2002.

Ravi, S., and Diao, Q. 2016. Large scale distributed semi-supervised learning using streaming approximation. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, 519–528.

Scheirer, W. J.; de Rezende Rocha, A.; Sapkota, A.; and Boult, T. E. 2012. Toward open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(7):1757–1772.

Settles, B. 2009. Active learning literature survey. Technical Report 1648, University of Wisconsin–Madison.

Spinosa, E. J., and de Leon Ferreira, A. C. P. 2004. Svms for novel class detection in bioinformatics. In *III Brazilian Workshop on Bioinformatics*, 81–88.

Sun, Y.; Tang, K.; Minku, L. L.; Wang, S.; and Yao, X. 2016. Online ensemble learning of data streams with gradually evolved classes. *IEEE Transactions on Knowledge and Data Engineering* 28(6):1532–1545.

Valko, M.; Kveton, B.; Huang, L.; and Ting, D. 2010. Online semi-supervised learning on quantized graphs. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 606–614.

Wagner, T.; Guha, S.; Kasiviswanathan, S.; and Mishra, N. 2018. Semi-supervised learning on data streams via temporal label propagation. In *Proceedings of the 35th International Conference on Machine Learning*, 5095–5104.

Wei, T.; Guo, L.; Li, Y.; and Gao, W. 2018. Learning safe multi-label prediction for weakly labeled data. *Machine Learning* 107(4):703–725.

Zhou, Z., and Chen, Z. 2002. Hybrid decision tree. *Knowledge Based Systems* 15(8):515–528.

Zhou, Z., and Li, M. 2010. Semi-supervised learning by disagreement. *Knowledge and Information Systems* 24(3):415–439.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2003. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16 (NIPS 2003)*, 321–328.

Zhou, Z. 2016. Learnware: on the future of machine learning. *Frontiers of Computer Science* 10(4):589–590.

Zhu, X.; Ghahramani, Z.; and Lafferty, J. D. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference*, 912–919.

Zhu, X.; Goldberg, A. B.; and Khot, T. 2009. Some new directions in graph-based semi-supervised learning. In *Proceedings of the 2009 IEEE International Conference on Multimedia and Expo*, 1504–1507.

Zhu, Y.; Ting, K. M.; and Zhou, Z. 2018. Multi-label learning with emerging new labels. *IEEE Transactions on Knowledge and Data Engineering* 30(10):1901–1914.