RECORD: Resource Constrained Semi-Supervised Learning under Distribution Shift

Lan-Zhe Guo and Zhi Zhou and Yu-Feng Li
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210023, China
{guolz,zhouz,liyf}@lamda.nju.edu.cn

ABSTRACT

Semi-supervised learning (SSL) tries to improve performance with the use of massive unlabeled data, which typically works in an offline manner with two assumptions. i) Data distribution is static; ii) Data storage overhead is unlimited. In many online tasks, however, none of the above assumptions is valid. For example, in online image classification, a large amount of unlabeled images increases sharply, which makes it difficult to store them in full; meanwhile, the content of unlabeled images changes constantly, and it is no longer suitable to assume a fixed distribution. We call such a novel setting *Resource Constrained SSL under Distribution Shift* (or *Record* for short) and to our best knowledge, it has not been thoroughly studied yet. This paper presents a systemic solution RECORD consisting of three sub-steps, that is, *distribution tracking*, *sample selection* and *model updating*. Specifically, we propose an effective method to track the distribution changes and locate distribution shifted samples. A novel influence-based approach is used to select the most influential samples for the distribution change based on resource constraints. Finally, we free up memory to put the latest unlabeled data with its pseudo-label for the next distribution tracking. Extensive empirical results confirm the effectiveness of our scheme. In the case of diverse and unknown distribution shifts, our solution is consistently and clearly better than many baseline and SOTA methods along with the memory budget and in some cases it can even approximate the performance of oracle.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Learning under covariate shift**; **Semi-supervised learning settings**.

KEYWORDS

semi-supervised learning; resource constraint; distribution shift

ACM Reference Format:
Lan-Zhe Guo and Zhi Zhou and Yu-Feng Li. 2020. RECORD: Resource Constrained Semi-Supervised Learning under Distribution Shift. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403214

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403214

1 INTRODUCTION

Semi-supervised learning (SSL) is well-known in machine learning field, which aims to enhance performance with the exploitation of unlabeled data since in many situations the acquisition of a large number of labeled data is infeasible. Tremendous efforts on SSL have been devoted over the past decades, typically on the enhancement of learning performance by introducing new feature representation [25], model regularization [26], adversarial training [23] and other interesting techniques [1]. These methods have been applied successfully to various applications from medical diagnosis to information retrieval [9, 10, 21, 22, 24, 31].

Previous SSL studies typically operate in an offline manner, i.e., the whole labeled and unlabeled data are given before training, under two assumptions−−i) data distribution is static, i.e., the data are drawn from a fixed distribution; ii) data storage overhead is unlimited, i.e., the entire data set can be stored fully in memory. In many scenarios, however, these assumptions do not hold. For example, in autonomous car [29], some labeled road conditions images are given at the initialization stage. Then, the car will receive massive unlabeled images that are difficult to be stored fully and it is evident that the content of unlabeled images would change constantly under different environments; in Twitter sentiment analysis [2], labeled data is given at the beginning while massive unlabeled data emerge every day with shifted distribution and it is not feasible to store all the data into memory; in Didi comment system[13], the machine learning model is initialized with a small amount of labeled ride-sharing comment data and a huge amount of unlabeled data arrives every day with varied distribution over time, region, etc. Similar cases can be found in other online applications such as spam detection and recommendation [5, 18, 29, 30].

To summarize, we have the data situation as following: i) Unlabeled data is continuously streaming with a shifted distribution over time; ii) Only a small number of manually labeled examples are provided at the beginning of the stream. Moreover, it is demanded that resources are often limited and constrained, e.g., memory budget, since the stream is generally too large to fully store in memory. As can be seen, such a situation is obviously different from previous SSL studies. We call such a novel setting as *Resource Constrained SSL under Distribution Shift* (or *Record* for short). To our best knowledge, this setting has not been thoroughly studied yet. Figure 1 illustrates the *Record* Setting and its formal definition is stated as follows.

***Record* Setting** Given a labeled data set $\mathcal{L} = \{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$ at the beginning stage $t = 0$ and unlabeled data arrives in a stream: $\mathcal{U}^1, \cdots, \mathcal{U}^t, \cdots$, where $\mathcal{U}^t = \{\mathbf{x}_1, \cdots, \mathbf{x}_{m^{(t)}}\}$. The underlying distribution $p(\mathbf{x})$ changes over time gradually. Let B be the memory budget, which means only B examples can be stored. The goal is to train a model in the semi-supervised fashion at every

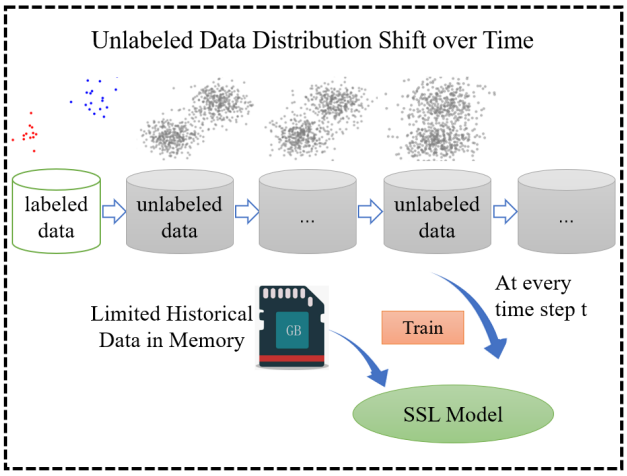

Figure 1: Illustration of the *Record* setting

time step using the stored B examples and the unlabeled data that arrives at the current time step.

It is evident that previous SSL studies could not well tackle the setting concerned in this paper, because they operate in a static environment which is unable to well address streaming data with a distribution shift. There are some SSL attempts designed for data streams [8], however, they either assume a fixed data distribution or neglect the resource constraint. Budget SSL [33] considers the resource constraint, however, they work on static scenarios and could not tackle the distribution shift.

In this paper, we present a systemic solution RECORD. It consists of three sub-steps: *distribution tacking*, *sample selection* and *model updating*. Specifically, we track the distribution changes and locate the distribution shifted samples. Then the most influential sample for the distribution change is selected based on a novel influence-based approach. Finally, the latest selected unlabeled data with its pseudo-label is stored into memory for the next distribution tracking. All the above operations are readily suitable for arbitrary SSL algorithms. Extensive empirical results clearly demonstrate the effectiveness of our scheme in the case of distribution shifts and memory resource limitations.

In the following, we first review several related works in section 2, then present the technical details of the proposed RECORD scheme in section 3. Next, we report the empirical results in section 4. Finally, we conclude this paper.

2 RELATED WORK

SSL is a well-established field and comprehensive reviews are available in [3, 31]. Most SSL studies were designed for the offline setting, and there are some attempts designed for data stream [4, 5, 7, 8, 16, 29, 34]. [7] proposed an online manifold regularization algorithm that can learn with unlabeled data based on convex programming in kernel space. [8] proposed an online active semi-supervised method by integrating a semi-supervised likelihood function and a sequential Monte Carlo scheme. [16] proposed a graph-based online SSL method via local consistency propagation. However, these approaches do not take the distribution shift problem into account. [4, 20] considered to learn from streaming data with a distribution shift. However, they assume that the labeled data is available at every time step which is difficult to satisfy in real-world applications since the label is hard to collect. [14, 34] assumed that there are unseen classes in the arrived data streaming and their goal is to identify instances with new classes.

There are very limited studies on resource constrained SSL. [33] is one early study on this topic and proposed an approximation method to adapt graph-based SSL methods into the given resource constraint. [6] adopted a density-based measure to select unlabeled data points and led to a compact graph with reduced computational costs. [19] proposed a resource constraint semi-supervised SVM that leverages the adjacent and distributive information carried in a spectral graph for efficient memory usage. However, all the aforementioned efforts carried on the stationary environment.

There are two related studies [5, 29], where the labeled data is provided at the beginning stage and the unlabeled data arrives in a stream. Specifically, [5] proposed an SSL approach based on a geometry algorithm. However, they do not consider the memory resource limitation, and the proposed geometry method is only useful for low-dimensional data. [29] proposed a graph-based online SSL method via temporal label propagation with the memory constraint. However, the method does not consider the distribution shift and can only be applied to graph-based SSL.

3 PROPOSED RECORD SCHEME

In this section, we propose a systemic scheme RECORD to deal with the *Record* problem. Two major challenges of the *Record* setting are i) There is a distribution shift in the data streams. How to make the algorithm adapt to the distribution shift and avoid performance degradation? ii) The volume of unlabeled data is generally too large to fully be stored into memory. How to leverage unlabeled examples to help improve the performance gain compared with the simple supervised learning model under memory resource constraint? RECORD provides a systemic solution consisting three main sub-steps, *distribution tracking*, *sample selection* and *model updating*. We first present a brief introduction to the basic framework of RECORD, including notations and setting. Then we describe the detail of the main techniques, and next give an analysis of the complexity.

3.1 Overall Framework

Consider a prediction problem from some input space $\mathcal{X} \in \mathbb{R}^d$ to an output space $\mathcal{Y} \in \mathbb{R}^C$ where d is the feature dimension and C is the number of classes. At the beginning stage (time $t = 0$), a small amount of labeled data set $\mathcal{L} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ is given where $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y}$. At all other time t, we receive only unlabeled data $\mathcal{U}^t = \{\mathbf{x}_i\}_{i=1}^{m^t}$ where m^t could be different at different time step. We aim to learn a function $f : \mathcal{X} \to \mathcal{Y}$ that performs well on the underlying distribution $p^t(\mathbf{x}, \mathbf{y})$ at every time step t. Table 1 summarizes the notations used in this paper.

In our paper, we consider the underlying distribution $p^t(\mathbf{x})$ at time t could be shifted from $p^{t-1}(\mathbf{x})$ at time $t-1$ and assume the distribution shift is gradual, i.e., distribution $p^t(\mathbf{x})$ in time t must have a considerable overlap with the distribution $p^{t-1}(\mathbf{x})$ [5]. The assumption is reasonable in practice and a completely random

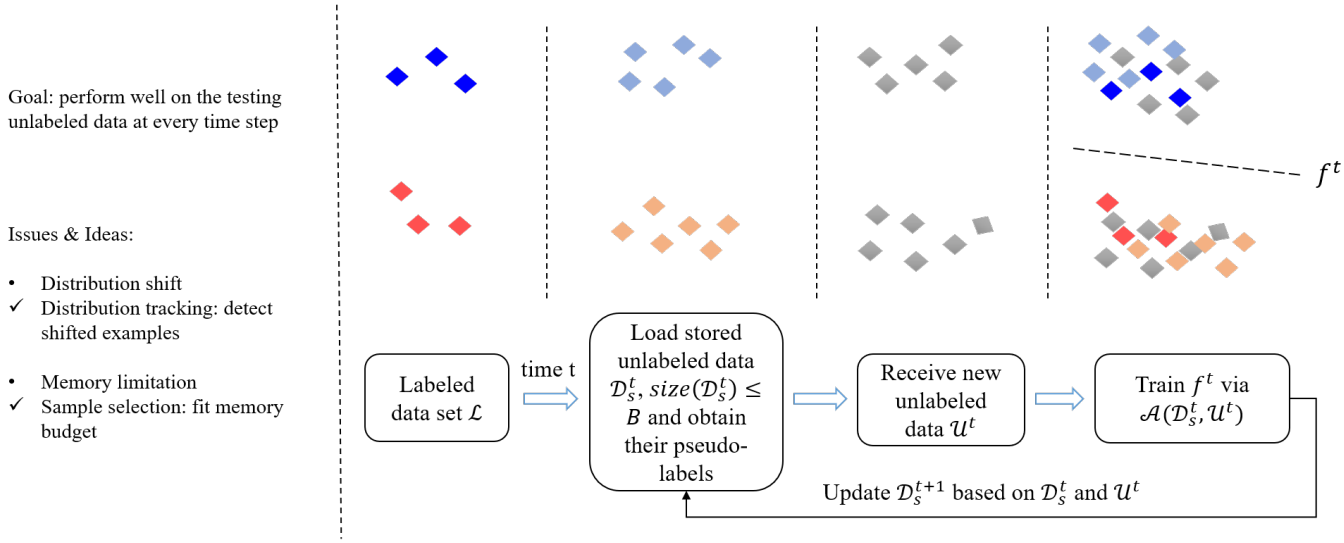

Figure 2: The proposed framework. At time $t = 0$, a small labeled data set is given, where the circle and rectangle indicate two classes. At all subsequent time step steps, unlabeled data $\mathcal{U}^t$ is received and the distribution could shift gradually over time.

Table 1: Summary of Notations.

Notation	Meaning
n	Number of labeled data
m_t	Number of unlabeled data at time t
B	Memory resource budget
C	Number of classes
$\mathbf{x} \in \mathcal{X}$	Feature vector of examples
$\mathrm{y} \in \mathcal{Y} = \{1, \cdots, C\}$	True label of labeled examples
$\hat{\mathrm{y}} \in \mathcal{Y} = \{1, \cdots, C\}$	Pseudo-label of unlabeled examples
$f : \mathcal{X} \rightarrow \mathcal{Y}$	Learned function
$\mathcal{U}^t$	Unlabeled data arrives at time t
$\mathcal{A}$	A semi-supervised algorithm
$\mathcal{D}_s^t$	Stored data at time t with size $\leq B$

fluctuation is intractable for learning. Figure 3 illustrates an example of the distribution shift.

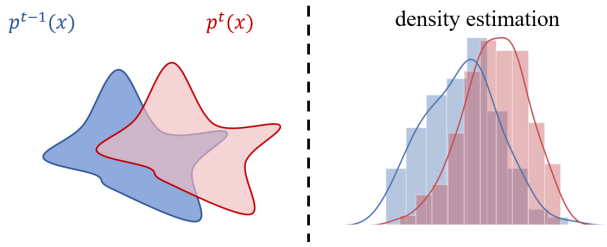

Figure 3: Distribution shift between $p^{t-1}(\mathbf{x})$ and $p^t(\mathbf{x})$.

To make the framework satisfy memory resource constraint, we propose to select unlabeled examples to store at every time step, i.e., maintain a data set $\mathcal{D}_s$ that contains at most B examples and can be continuously updated. $\mathcal{D}_s$ is initialized as $\mathcal{L}$. At time t, we train an SSL model with the stored data $\mathcal{D}_s^t$ and unlabeled data $\mathcal{U}^t$. Therefore, the key problem in our setting is how to select examples and update $\mathcal{D}_s$ continuously to make it adapt to the distribution shift in the subsequent time steps and helpful for model building.

Conceptually, RECORD solves the following problem at each time step t:

$$\begin{aligned} \max_{\mathcal{D}_s^t} \quad & performance(f) \\ \text{s.t.} \quad & f = \mathcal{A}(\mathcal{D}_s^t, \mathcal{U}^t) \\ & \mathcal{D}_s^t \subseteq \mathcal{D}_s^{t-1} \cup \mathcal{U}^{t-1} \\ & size(\mathcal{D}_s^t) \leq B \end{aligned} \tag{1}$$

A schematic description of the overall framework of RECORD is shown in Figure 2.

3.2 Shifted Example Detection

A straightforward example selection method is to randomly select B examples to store at each time step. It is evident that this method could not work well since it does not take the structure of unlabeled data into account. Further, an intuitive method is to select examples with high softmax probability since the predicted probability reveals the confidence of the example to the learning task [12]. However, as the underlying probability distribution $p^t(\mathbf{x})$ of $\mathcal{U}^t$ could change over time, we need not only examples with high confidence in current $p^t(\mathbf{x})$ but also examples that reveal the data distribution shift trend. To do so, we first need to identify examples shifted from the previous distribution, i.e., data in the non-overlapped region between $p^{t-1}(\mathbf{x})$ and $p^t(\mathbf{x})$.

Specifically, we adopt probabilities from softmax distribution as a criterion since examples in the previous distribution tend to have high softmax probabilities than examples in the non-overlapped region [15]. The softmax probabilities can be written as $f(\mathbf{x}) \in$

Algorithm 1 RECORD Approach

Input: Any SSL algorithm $\mathcal{A}$, labeled data $\mathcal{L} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$, stored data D_s^0 that initialized as $\mathcal{L}$.

for $t = 0, 1, \cdots$ **do**
  Receive unlabeled data $\mathcal{U}^t = \{\mathbf{x}_i\}_{i=1}^{m^t}$.
  Run SSL algorithm $\mathcal{A}$ with $\mathcal{D}_s^t$ and $\mathcal{U}^t$, Obtain $f^t : \mathcal{X} \to \mathcal{Y}$.
  Obtain the predicted probability $f^t(\mathbf{x}), \forall \mathbf{x} \in \mathcal{U}^t$.
  Assign pseudo-label $\hat{\mathbf{y}}$ to $\mathbf{x}, \forall \mathbf{x} \in \mathcal{U}^t$.
  Let $\mathcal{R} = \emptyset$
  for each class $c = 1, \cdots, C$ **do**
    $\mathcal{U}_c = \{\mathbf{x}_i \in \mathcal{U}^t$ where $\hat{\mathbf{y}}_i = c\}$.
    Split $\mathcal{U}_c$ into $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$.
    Randomly select $|\mathcal{U}_c^{in}|$ examples from other classes, combined with $\mathcal{U}_c^{in}$, named $\mathcal{U}_c^{train}$.
    Randomly select $|\mathcal{U}_c^{out}|$ examples from other classes, combined with $\mathcal{U}_c^{out}$, named $\mathcal{U}_c^{test}$.
    Run Algorithm 2 with $\mathcal{U}_c^{train}, \mathcal{U}_c^{test}, \mathcal{U}_c^{in}$, obtain $\mathcal{R}_c$.
    $\mathcal{R} = \mathcal{R} \cup \mathcal{R}_c$.
  end for
  Combine $\mathcal{R}$ with examples in $\mathcal{D}_s^t$ via Eq.(7) as $\mathcal{D}_s^{t+1}$.
end for

$\mathbb{R}^{1\times C}$ where $f(\mathbf{x})_c$ indicates the posterior probability $p(c|\mathbf{x}, f)$. We can then assign a pseudo-label $\hat{\mathbf{y}}$ to each example based on the probabilities:

$$\hat{\mathbf{y}} = \underset{c\in\{1,\cdots,C\}}{\operatorname{argmax}} \; f(\mathbf{x})_c \tag{2}$$

Let $\mathcal{U}_c$ be the set of all examples in $\mathcal{U}^t$ with pseudo-label c. We split $\mathcal{U}_c$ into two subsets $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$ with equal size based on the predicted probability. Specifically, $\mathcal{U}_c^{in}$ contains examples with the largest probabilities, which are likely to fall into the region of previous data distribution while $\mathcal{U}_c^{out}$ contains the rest examples which consequently lie in the shifted distribution (i.e., examples in the non-overlapped region). Figure 4 illustrates the splitting process and the examples in $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$.

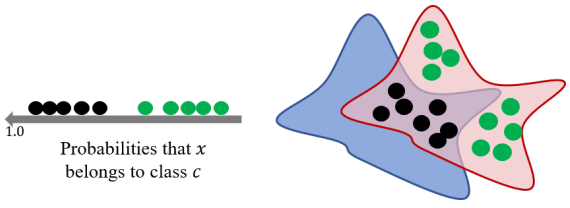

Figure 4: Examples in $\mathcal{U}_c^{in}$ (the black circle) and $\mathcal{U}_c^{out}$ (the green circle). The examples in $\mathcal{U}_c^{in}$ being larger probabilities are in the overlapped region while the examples in $\mathcal{U}_c^{out}$ being lower probability are in the non-overlapped region.

3.3 Influential Example Selection

After obtaining $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$, we come to the problem of how to select examples most helpful for the subsequent time steps. Our basic idea is to select examples from $\mathcal{U}_c^{in}$ based on its influence on $\mathcal{U}_c^{out}$ since examples in $\mathcal{U}_c^{in}$ being a high softmax probability indicate that they are helpful for the learning task meanwhile as they are influential on $\mathcal{U}_c^{out}$, they are to some extent able to reveal the distribution shift trend.

Algorithm 2 Influence Function Computation.

Input: $\mathcal{U}_c^{train}, \mathcal{U}_c^{test}, \mathcal{U}_c^{in}$.

Train a logistic regression model with $\mathcal{U}_c^{train}$.
for example $\mathbf{x}$ in $\mathcal{U}_c^{in}$ **do**
  $\text{IF}(\mathbf{x}) = 0$
  for example $\mathbf{z}_{test}$ in $\mathcal{U}_c^{test}$ **do**
    Calculate $\mathcal{I}(\mathbf{x}, \mathbf{z}_{test})$ via Eq.(5).
    $\text{IF}(\mathbf{x}) = \text{IF}(\mathbf{x}) + \mathcal{I}(\mathbf{x}, \mathbf{z}_{test})$.
  end for
end for
Sort examples in $\mathcal{U}_c^{in}$ according to $\text{IF}(\mathbf{x})$.
return Examples in $\mathcal{U}_c^{in}$ with $\text{IF}(\mathbf{x}) > 0$.

Specifically, we propose to adopt influence function [17] to evaluate the influence of one example in $\mathcal{U}_c^{in}$ on $\mathcal{U}_c^{out}$ since the influence function is a very useful tool to help find the training examples that are most influential for an underlying distribution.

To compute the influence function, we first need to construct a training and testing data set and build a logistic regression model [17]. To do so, we randomly sample $|\mathcal{U}_c^{in}|$ and $|\mathcal{U}_c^{out}|$ ($|\mathcal{U}_c^{in}|$ and $|\mathcal{U}_c^{out}|$ indicate the numbers of examples in $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$) examples from $\mathcal{U}_{c'}^{in}$ and $\mathcal{U}_{c'}^{out}$ (where $c' \neq c$) as negative examples (i.e., $\tilde{y} = -1$) and combine these examples with $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$ as positive class (i.e., $\tilde{y} = 1$) to construct a binary classification training set $\mathcal{U}_c^{train}$ and testing set $\mathcal{U}_c^{test}$,

$$\mathcal{U}_c^{train} = \mathcal{U}_c^{in} \cup \text{Random}_{|\mathcal{U}_c^{in}|}(\mathcal{U}_{c'}^{in}) \tag{3}$$

$$\mathcal{U}_c^{test} = \mathcal{U}_c^{out} \cup \text{Random}_{|\mathcal{U}_c^{out}|}(\mathcal{U}_{c'}^{out}) \tag{4}$$

where $\text{Random}_n(\mathcal{D})$ refers to n examples randomly selected from the data set $\mathcal{D}$.

Then, we can train a logistic regression model on $\mathcal{U}_c^{train}$ and calculate the influence $\mathcal{I}(\mathbf{x}, \mathbf{z}_{test})$ for example $\mathbf{x}$ in $\mathcal{U}_c^{in}$ and test data point $\mathbf{z}_{test} = (\mathbf{x}_{test}, \tilde{y})$ in $\mathcal{U}_c^{test}$.

Specifically, let θ be the parameter of the logistic regression model and $\sigma(t) = \frac{1}{1+\exp(-t)}$. The logistic loss function for the example $\mathbf{x} \in \mathcal{U}_c^{in}$ refers to $L(\mathbf{x}, \theta) = \log(1 + \exp(-\theta^\top \mathbf{x}))$, and based on [17] the $\mathcal{I}(\mathbf{x}, \mathbf{z}_{test})$ can be written as:

$$\tilde{y}\sigma(-\tilde{y}\theta^\top \mathbf{x}_{test}) \cdot \sigma(-\theta^\top \mathbf{x})\mathbf{x}_{test}^\top \mathbf{H}_\theta^{-1}\mathbf{x} \tag{5}$$

where $\mathbf{H}_\theta$ is the Hessian of the logistic loss function.

The influence of example $\mathbf{x}$ can be written as,

$$\text{IF}(\mathbf{x}) = \frac{1}{|\mathcal{U}_c^{test}|} \sum_{\mathbf{z}_{test} \in \mathcal{U}_c^{test}} \mathcal{I}(\mathbf{x}, \mathbf{z}_{test}) \tag{6}$$

Finally, for each class c, let $\mathcal{R}_c$ be the set containing examples in $\mathcal{U}_c^{in}$ with positive influence, i.e., $\text{IF}(\mathbf{x}) \geq 0$. If the number of these examples $|\mathcal{R}_c|$ is greater than the memory resource constrained number $\frac{B}{C}$, we retain $\frac{B}{C}$ most influential examples, otherwise we replace the oldest examples in $\mathcal{D}_s^t$ to feed these examples,

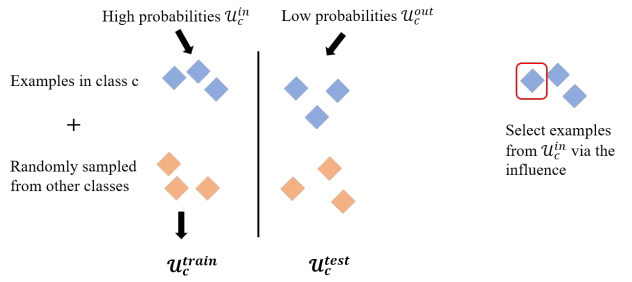

Figure 5: Illustration of data selection in class c.

$$\mathcal{D}_s^{t+1} = \bigcup_{c=1}^{C} \left\{ \begin{array}{ll} \text{Top-IF}_{\frac{B}{C}}(\mathcal{R}_c) & |\mathcal{R}_c| \geq \frac{B}{C} \\ \text{Latest}_{\frac{B}{C}-|\mathcal{R}_c|}(\mathcal{D}_s^t) \cup \mathcal{R}_c & |\mathcal{R}_c| < \frac{B}{C} \end{array} \right\} \quad (7)$$

The retained examples along with their pseudo-labels can be used in the subsequent steps. Figure 5 illustrates an example of data selection for one class c. The detailed procedure of RECORD is summarized in Algorithm 1.

3.4 Complexity Analysis

At every time step t, RECORD needs to split examples in each class c into $\mathcal{U}_c^{in}$ and $\mathcal{U}_c^{out}$, and the time complexity is $O(m^t \log(m^t))$ as the examples need to be sorted according the the predicted probability. As for the calculation of the influence for each example in $\mathcal{U}_c^{out}$, according to [17], the computation time complexity to calculate influence function is $O\left(\frac{(m^t)^2}{4}d\right)$ where d is the feature dimension. Overall, the whole computation time complexity for RECORD at each time step is $O\left(\frac{(m^t)^2}{4}d + m^t \log(m^t)\right)$. And obviously, the space complexity of RECORD is $O(B + m^t)$.

4 EXPERIMENTS

In this section, we conduct extensive experiments on various data sets and competitive methods to evaluate the effectiveness of the RECORD approach.

4.1 Setup

Experiments are conducted on various data sets, including four commonly used classification data sets (*Optdigits*, *Satimage*, *Twonorm*, *Spam*) and four benchmark data sets (*1CHT*, *2CDT*, *UG_2C_2D*, *UG_2C_5D*) [27] which is designed for evaluating learning algorithm on streaming data. The data sizes vary from 5,620 to 200,000 and feature dimensions vary from 2 to 500. The statistics of these data sets is summarized in Table 2.

To validate the generality of RECORD that can be incorporated with various SSL algorithms, three diverse kinds of SSL algorithms are considered.

- Mean Teacher [28]: Mean Teacher is one representative deep SSL method. It adopts consistency regularization as the unsupervised loss, and averages model parameters to form a high quality generated targets for unlabeled examples. The consistency regularization based deep SSL methods have achieved SOTA results on various SSL tasks [24].

Table 2: Statistics of data sets

Dataset	# instance	# feature	# class	# labeled examples (per class)
Optdigits	5,620	64	10	10
Satimage	6,435	36	7	10
Twonorm	7,400	20	2	10
Spam	9,324	500	2	50
1CHT	16,000	2	2	1
2CDT	16,000	2	2	5
UG_2C_2D	100,000	2	2	1
UG_2C_5D	200,000	5	2	5

- Label Propagation: Label propagation is a graph-based SSL method that transfers class information from labeled vertices to neighboring examples. The underlying assumption of label propagation is the smooth assumption, i.e., similar examples should have similar labels.
- S^3VM: Semi-Supervised Support Vector Machine (S^3VM) is a classical SSL method based on the low-density assumption that aims to construct a decision boundary that passes the low-density region.

Data set preparation. For the four commonly used classification data sets, we simulate the distribution shift manually by regrouping the instances. Specifically, for *Optdigits*, *Twonorm* and *Satimage*, 200 instances arrive every time step in which 160 as unlabeled data, 40 as test data; for *Spam*, 400 instances arrive every time step and 280 as unlabeled data, 120 as test data. For the four benchmark data sets, the distribution shift is the same as the default setup described in [27]. Specifically, at every time step, 400 instances arrive for *1CHT* and *2CDT*, 1,000 instances arrive for *UG_2C_2D* and 2,000 instances arrive for *UG_2C_5D*, 30% as test data. For S^3VM, we construct a binary classification task (1 vs 7) for *Optdigits* and *Satimage* data sets since it is inefficient to handle multi-class classification task. The number of labeled examples given at the beginning stage for each data set are listed in Table 2.

Parameter detail. For S^3VM and Label Propagation, we adopt the implementation in sklearn[1]. The RBF kernel is adopted for S^3VM and 9NN is adopted for Label Propagation on *Spam* data set. All other parameters are set as default. For Mean Teacher, we adopt the official implementation [2] and run 1,000 training iterations. For our framework RECORD the memory resource budget B is set as 100, i.e., only 100 unlabeled examples can be stored into memory.

4.2 Compared with Baseline Methods

As there is no literature working on the same setting concerned in the paper, we first study how effective RECORD is at improving classification performance gain in comparison with a simple supervised learning method,

- Supervised: Simply train a supervised model using only the labeled examples while ignoring all coming unlabeled examples. The Supervised method can be seen as the lower bound of the performance.

[1] https://scikit-learn.org/
[2] https://github.com/CuriousAI/mean-teacher

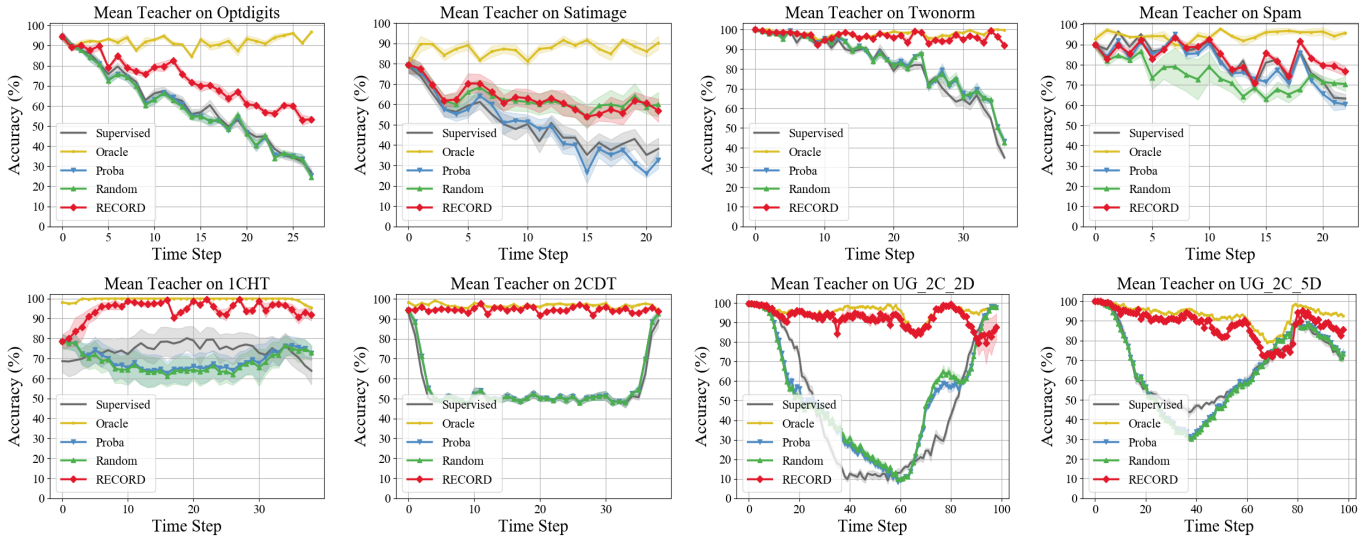

Figure 6: Accuracy of compared methods on 8 data sets with *Mean Teacher* as the basic SSL model. Shaded regions indicate standard deviation.

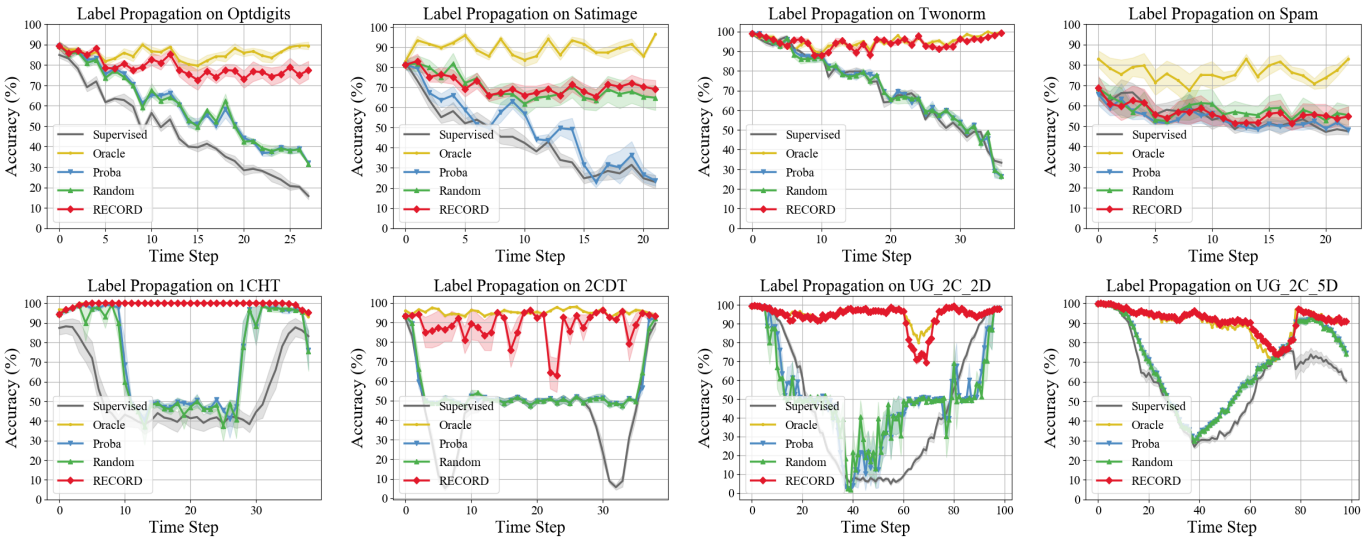

Figure 7: Accuracy of compared methods on 8 data sets with *Label Propagation* as the basic SSL model.

and two baseline example selection methods,

- Proba: Select examples according to the prediction probability, i.e., examples with a higher softmax probability are selected. This method has been proved effective in various tasks, e.g., out of distribution examples detection [15].
- Random: Randomly select the unlabeled examples to store at every time step. This is the simplest method to satisfy the memory resource constraint.

For the other configurations, Proba and Random follow the same setups as those in RECORD.

We also compare with an Oracle method,

- Oracle: Assume that the ground-truth labels are available and a supervised model is trained at every time step. Oracle is impossible in real practice and can be seen as the upper bound of the performance to the learning task.

Mean ± Std accuracy of 5 runs with 3 different SSL implementations on 8 data sets are shown in Figure 6-8. From the results, we can observe that RECORD achieve the best performance among the compared methods regardless of the SSL methods and data sets whereas the simple Proba and Random methods all suffer performance degradation with the distribution shift and can not achieve performance gain compared with the Supervised method. Moreover, the RECORD very closely follows the performance of the Oracle

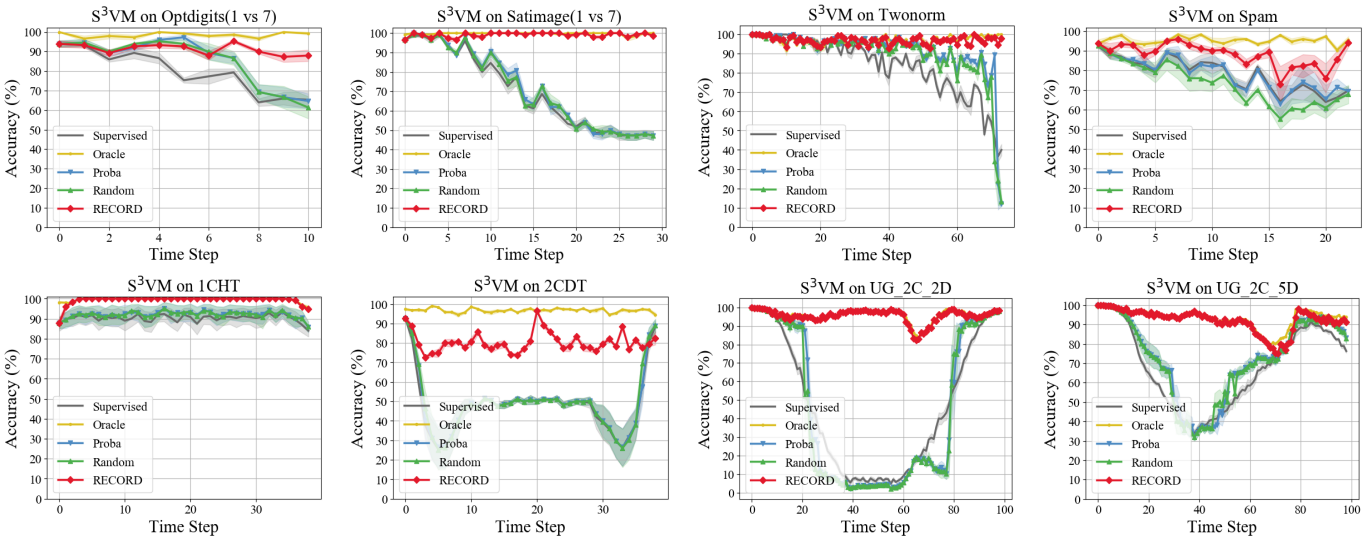

Figure 8: Accuracy of compared methods on 8 data sets with S^3VM as the basic SSL model.

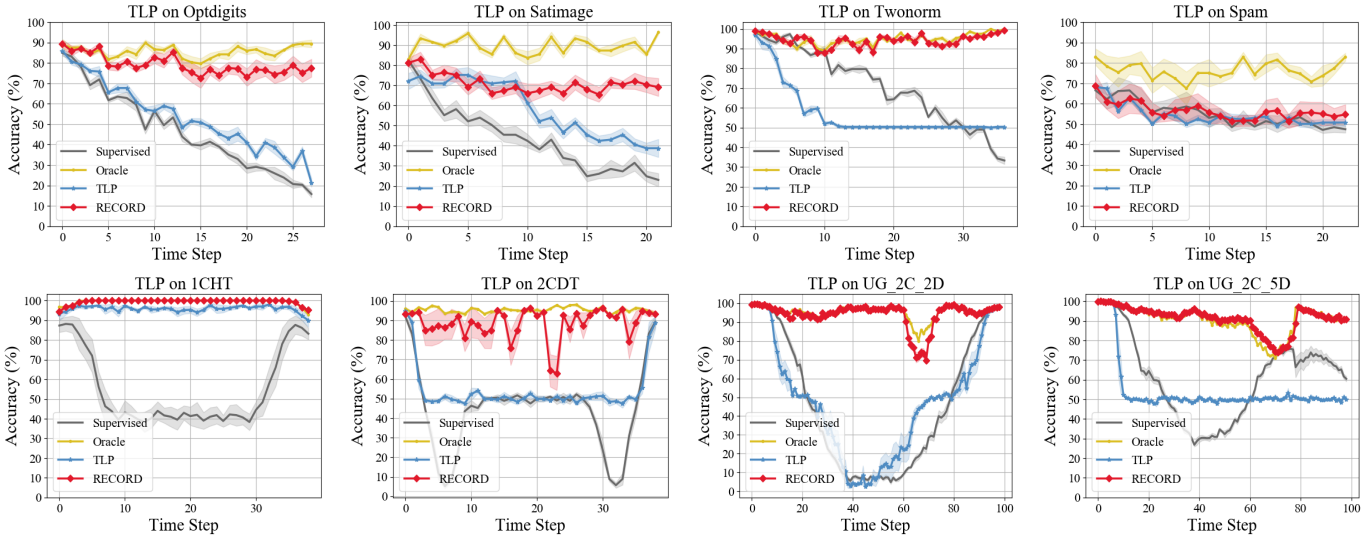

Figure 9: Comparison results with TLP on 8 data sets

method and even better than the Oracle in some cases. The main reason is RECORD can access to the stored examples from previous distributions. This phenomenon validates RECORD can retain helpful examples for subsequent time steps. These demonstrate the effectiveness and generality of the proposed RECORD scheme.

4.3 Compared with State-of-the-art Methods

We also compare RECORD to two works that are most related to our work: COMPOSE [5] and TLP [29]. TLP is specifically designed for graph-based SSL methods. TLP maintains a small synopsis of the data stream that can be quickly updated as new examples arrive. The effectiveness of TLP has been demonstrated in various real-world tasks such as ECG analysis, autonomous car [29].

The comparison results of 5 runs on 8 data sets are shown in Figure 9. We can observe that the performance of TLP method degrades dramatically with the distribution shift in many cases while our RECORD performs more stable with the distribution varies and achieves a significant performance gain compared with TLP. We also report the average performance on the whole data streams in Table 3. We can see that, TLP method performs even worse than the baseline Supervised method on *Twonorm*, *Spam* and *UG_2C_5D* data sets while our RECORD works much better, and even approximates the performance of oracle in some cases.

COMPOSE adopts a compacted polytope sample extraction algorithm and generates pseudo-labels for previously unlabeled examples and combine these examples with incoming unlabeled data.

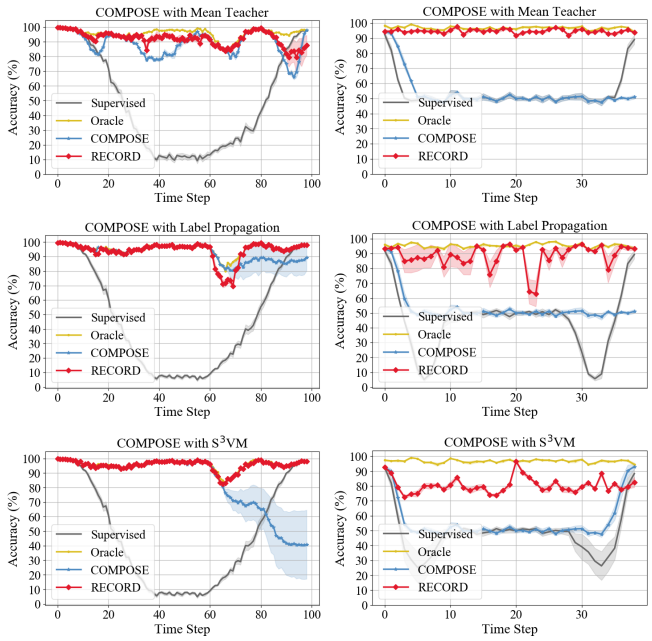

Figure 10: Compared with COMPOSE on UG_2C_2D (left) and 2CDT (right) data sets with 3 basic SSL models.

Table 3: Mean ± Std accuracy on the whole data streams with Label Propagation as the base SSL method. The best two methods (including Oracle method) are emphasized in bold. Moreover, the underline indicates that the method is even worse than the supervised method.

Data set	Supervised	TLP	RECORD	Oracle
Optdigits	46.64 ± 1.04	53.56 ± 1.42	**79.20 ± 2.01**	**85.61 ± 0.25**
Twonorm	70.84 ± 0.36	57.24 ± 0.86	**94.31 ± 0.38**	**94.99 ± 0.13**
Satimage	43.47 ± 2.61	58.43 ± 2.54	**71.02 ± 2.93**	**89.59 ± 0.70**
Spam	55.12 ± 3.48	54.18 ± 1.39	**56.55 ± 4.72**	**76.35 ± 2.41**
1CHT	54.79 ± 4.93	95.78 ± 1.27	**99.44 ± 0.05**	**99.44 ± 0.01**
2CDT	44.47 ± 0.46	54.28 ± 0.48	**88.68 ± 1.57**	**95.36 ± 0.13**
UG_2C_2D	46.40 ± 0.23	47.56 ± 1.12	**94.11 ± 0.12**	**95.35 ± 0.04**
UG_2C_5D	61.00 ± 1.01	54.40 ± 0.17	**91.39 ± 0.08**	**90.41 ± 0.05**

However, the geometry-based method can only be applied to low-dimensional data set, so we only run COMPOSE on *1CHT* and *2CDT* data sets based on three base SSL methods. For COMPOSE, the two important parameters α and cp are set to 0.4 and 0.7 separately.

The experimental results on two data sets are summarized in Figure 10. From the results, we can see that, on *UG_2C_2D* data set, both RECORD and COMPOSE perform well at first 60 time steps, however, after 60 steps, COMPOSE suffers a severe performance degradation problem while RECORD still maintain the good performance that close to the Oracle method. And on *2CDT* data set, the performance of COMPOSE degrades even in the first 5 time steps whereas RECORD achieves a significant performance gain compared with COMPOSE. The average performance on the whole data streams is reported in Table 4, we can observe the similar results

Table 4: Mean ± Std accuracy with the 3 SSL implementations on 2CDT and UG_2C_2D data sets.

SSL methods	Supervised	COMPOSE	RECORD	Oracle
		2CDT		
Mean Teacher	54.40 ± 0.36	54.09 ± 0.24	**94.60 ± 0.13**	**96.72 ± 0.18**
Label Propagation	44.47 ± 0.46	53.19± 0.20	**88.70 ± 1.57**	**95.36 ± 0.13**
S^3VM	48.40 ± 2.29	56.05 ± 0.70	**80.41 ± 0.19**	**96.71 ± 0.16**
		UG_2C_2D		
Mean Teacher	47.82 ± 0.65	88.76 ± 0.35	**92.34 ± 0.66**	**95.94 ± 0.02**
Label Propagation	46.40 ± 0.23	92.63 ± 2.89	**94.11 ± 0.12**	**95.35 ± 0.04**
S^3VM	46.49 ± 0.19	83.05 ± 5.09	**95.55 ± 0.05**	**96.02 ± 0.04**

that our RECORD always better than COMPOSE and very close to the Oracle method while COMPOSE performs even worse than Supervised method in some case. All results demonstrate that our proposal achieves state-of-the-art performance in this setting.

4.4 Impact of Memory Budget

It is meaningful to study the impact of the memory budget. We investigate how the performance varies with the memory budget decreased from 80 to 10 on four benchmark data sets. S^3VM is adopted as the base SSL method. The results are plotted in Figure 11. We can see that RECORD does not suffer severe performance degradation as long as we can store more than 20 examples.

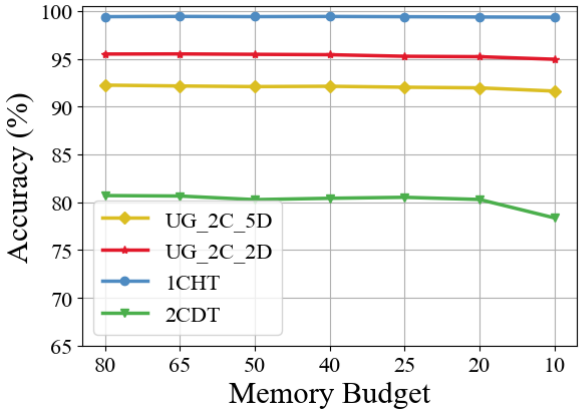

Figure 11: Performance of RECORD with varied memory budgets on 4 benchmark data sets.

4.5 Visualization of Retained Examples

To further show the validity of examples retained by RECORD, we visualize the distribution shift and retained examples in Figure 12. The experiment is conducted on *UG_2C_2D* data set. The blue and red circles indicate retained examples while shaded circles indicate all unlabeled examples arrived at the current time step. We can see that RECORD can retain examples in the overlapped region of $p^t(\mathbf{x})$ and $p^{t+1}(\mathbf{x})$ which helpful for current distribution and track the distribution shift trend.

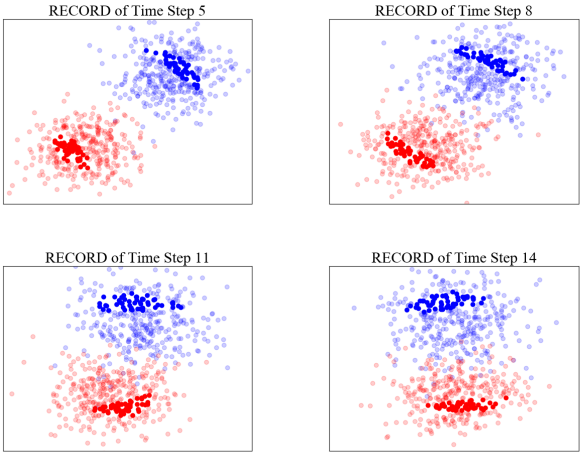

Figure 12: Visualization of distribution shift and retained examples on *UG_2C_2D* data set, from time step 5 to 14.

5 CONCLUSION

In this paper, we tackle the novel and realistic *Record* setting that has rarely been studied before and have proposed a systemic solution RECORD. RECORD tracks the distribution shift trends based on a novel influence-based approach. We select the most influential samples for the distribution change based on memory constraint. The learned model is constantly updated via the newly stored data for the next distribution tracking. Extensive empirical results on various data sets demonstrate that RECORD perform clearly better than many competitive baseline and SOTA methods. Moreover, RECORD is readily suitable for arbitrary SSL implementations.

In RECORD, we assume the class space of all unlabeled examples is fixed. In some online applications, unseen classes may emerge [11]. We will consider extending this work to scenarios with unseen classes and prior knowledge [32] in the future.

The code and data for the experiments could be freely downloaded at https://www.lamda.nju.edu.cn/code_RECORD.ashx or https://github.com/WNJXYK/RECORD.

6 ACKNOWLEDGMENTS

This research was supported by the National Key R&D Program of China (2017YFB1001903), the NSFC (61772262) and the Fundamental Research Funds for the Central Universities (14380061). We would like to thank Jieping Ye and Wei-Wei Tu who provided valuable insight to this work. Dr. Yu-Feng Li is the corresponding author.

REFERENCES

[1] David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. 2019. MixMatch: A Holistic Approach to Semi-Supervised Learning. In *NeurIPS*. 5050–5060.
[2] Albert Bifet, Geoffrey Holmes, Bernhard Pfahringer, and Ricard Gavaldà. 2011. Detecting Sentiment Change in Twitter Streaming Data. In *WAPA*. 5–11.
[3] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien (Eds.). 2006. *Semi-Supervised Learning*. The MIT Press.
[4] Gregory Ditzler and Robi Polikar. 2011. Semi-Supervised Learning in Nonstationary Environments. In *IJCNN*. 2741–2748.
[5] Karl B Dyer, Robert Capo, and Robi Polikar. 2013. Compose: A Semi-Supervised Learning Framework for Initially Labeled Nonstationary Streaming Data. *IEEE Transactions on Neural Networks and Learning Systems* 25, 1 (2013), 12–26.
[6] Sandra Ebert, Mario Fritz, and Bernt Schiele. 2012. Semi-Supervised Learning on a Budget: Scaling Up to Large Datasets. In *ACCV*. 232–245.
[7] Andrew B Goldberg, Ming Li, and Xiaojin Zhu. 2008. Online Manifold Regularization: A New Learning Setting and Empirical Study. In *ECML/PKDD*. 393–407.
[8] Andrew B Goldberg, Xiaojin Zhu, Alex Furger, and Jun-Ming Xu. 2011. Oasis: Online Active Semi-Supervised Learning. In *AAAI*. 362–367.
[9] C Gong, D Tao, SJ Maybank, W Liu, G Kang, and J Yang. 2016. Multi-modal curriculum learning for semi-supervised image classification. *IEEE Transactions on Image Processing* 25, 7 (2016), 3249–3260.
[10] Lan-Zhe Guo and Yu-Feng Li. 2018. A General Formulation for Safely Exploiting Weakly Supervised Data. In *AAAI*. 3126–3133.
[11] Lan-Zhe Guo, Zhen-Yu Zhang, Yuan Jiang, Yu-Feng Li, and Zhi-Hua Zhou. 2020. Safe Deep Semi-Supervised Learning for Unseen-Class Unlabeled Data. In *ICML*.
[12] Lan-Zhe Guo, Tao Han, and Yu-Feng Li. 2019. Robust Semi-supervised Representation Learning for Graph-Structured Data. In *PAKDD*. 131–143.
[13] Lan-Zhe Guo, Feng Kuang, Zhang-Xun Liu, Yu-Feng Li, Nan Ma, and Xiao-Hu Qie. 2020. Weakly Supervised Learning Meets Ride-Sharing User Experience Enhancement. In *AAAI*.
[14] Ahsanul Haque, Latifur Khan, and Michael Baron. 2016. Sand: Semi-Supervised Adaptive Novel Class Detection and Classification over Data Stream. In *AAAI*. 1652–1658.
[15] Dan Hendrycks and Kevin Gimpel. 2017. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In *ICLR*.
[16] Lei Huang, Xianglong Liu, Binqiang Ma, and Bo Lang. 2015. Online Semi-Supervised Annotation via Proxy-Based Local Consistency Propagation. *Neurocomputing* 149 (2015), 1573–1586.
[17] Pang Wei Koh and Percy Liang. 2017. Understanding Black-Box Predictions via Influence Functions. In *ICML*. 1885–1894.
[18] Georg Krempl, Indre Zliobaite, Dariusz Brzeziński, Eyke Hüllermeier, Mark Last, Vincent Lemaire, Tino Noack, Ammar Shaker, Sonja Sievi, Myra Spiliopoulou, et al. 2014. Open Challenges for Data Stream Mining Research. *SIGKDD Explorations* 16, 1 (2014), 1–10.
[19] Trung Le, Phuong Duong, Mi Dinh, Tu Dinh Nguyen, Vu Nguyen, and Dinh Q Phung. 2016. Budgeted Semi-supervised Support Vector Machine.. In *UAI*. 377–386.
[20] Pei-Pei Li, Xindong Wu, and Xuegang Hu. 2010. Mining Recurring Concept Drifts with Limited Labeled Streaming Data. In *ACML*. 241–252.
[21] Yu-Feng Li, Lan-Zhe Guo, and Zhi-Hua Zhou. 2019. Towards Safe Weakly Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).
[22] Yu-Feng Li and De-Ming Liang. 2019. Safe Semi-Supervised Learning: A Brief Introduction. *Frontiers of Computer Science* 13, 4 (2019), 669–676.
[23] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 8 (2018), 1979–1993.
[24] Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. 2018. Realistic Evaluation of Deep Semi-Supervised Learning Algorithms. In *NeurIPS*. 3235–3246.
[25] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. 2015. Semi-Supervised Learning with Ladder Networks. In *NeurIPS*. 3546–3554.
[26] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. 2016. Regularization with Stochastic Transformations and Perturbations for Deep Semi-Supervised Learning. In *NeurIPS*. 1163–1171.
[27] V. M. A. Souza, D. F. Silva, J. Gama, and G. E. A. P. A. Batista. 2015. Data Stream Classification Guided by Clustering on Nonstationary Environments and Extreme Verification Latency. In *ICDM*. 873–881.
[28] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*. 1195–1204.
[29] Tal Wagner, Sudipto Guha, Shiva Kasiviswanathan, and Nina Mishra. 2018. Semi-Supervised Learning on Data Streams via Temporal Label Propagation. In *ICML*. 5095–5104.
[30] Peng Zhao, Xinqiang Wang, Siyu Xie, Lei Guo, and Zhi-Hua Zhou. 2019. Distribution-free one-pass learning. *IEEE Transactions on Knowledge and Data Engineering* (2019).
[31] Zhi-Hua Zhou. 2017. A Brief Introduction to Weakly Supervised Learning. *National Science Review* 5, 1 (2017), 44–53.
[32] Zhi-Hua Zhou. 2019. Abductive learning: Towards bridging machine learning and logical reasoning. *Science China Information Sciences* 62, 7 (2019), 76101:1–76101:3.
[33] Zhi-Hua Zhou, Michael Ng, Qiao-Qiao She, and Yuan Jiang. 2009. Budget Semi-Supervised Learning. In *PAKDD*. 588–595.
[34] Yong-Nan Zhu and Yu-Feng Li. 2020. Semi-Supervised Streaming Learning with Emerging New Labels. In *AAAI*.