Transfer Learning for Semi-supervised Classification of Non-stationary Data Streams

Yimin Wen[1](✉), Qi Zhou[1], Yun Xue[2], and Chao Feng[1]

[1] Guangxi Key Laboratory of Image and Graphic Intelligent Processing, Guilin University of Electronic Technology, Guilin, China
ymwen2004@aliyun.com, zqguidian@163.com, henryfung01@126.com

[2] School of Municipal and Surveying Engineering, Hunan City University, Yiyang, China
yunxue1209@163.com

Abstract. In the scenario of data stream classification, the occurrence of recurring concept drift and the scarcity of labeled data are very common, which make the semi-supervised classification of data streams quite challenging. To deal with these issues, a new classification algorithm for partially labeled streaming data with recurring concept drift is proposed. CAPLRD maintains a pool of concept-specific classifiers and utilizes historical classifiers to label unlabeled data, in which the unlabeled data are labeled by a weighted-majority vote strategy, and concept drifts are detected by automatically monitoring the threshold of classification accuracy on different data chunks. The experimental results illustrate that the transfer learning from historical concept-specific classifiers can improve labeling accuracy significantly, the detection of concept drifts and classification accuracy effectively.

Keywords: Concept drift · Semi-supervised learning · Transfer learning

1 Introduction

In the era of big data, data streams are very common, examples include financial transaction of credit cards, network intrusion information, and so on. Concept drift [1] often occurs in these data streams, in which the distribution of data is not stationary and makes the traditional classification methods not applicable to data stream [2,3]. Recurring concept drift is a special type of concept drift, referring to the phenomenon that concepts appeared in the past may reoccur in the future [4]. However, some algorithms like [5–7] don't consider the occurrence of recurring concept. Furthermore, in many real applications, the labeled

The original version of this chapter was revised: The typesetting errors in Tables 2–5 were corrected. The correction to this chapter is available at https://doi.org/10.1007/978-3-030-63823-8_95 instances are always scarce, and the cost of labeling unlabeled instances is high. Therefore, using transfer learning and semi-supervised methods to solve these problems are promising.

In this paper, a new Classification Algorithm for Partially Labeled data stream with Recurring concept Drift (CAPLRD)[1] is proposed. CAPLRD can be employed as a framework for semi-supervised classification of data stream with recurring concept drifts. The main contributions of CAPLRD are as follows: (1) a new algorithm of Semi-Supervised Learning with Node Ensemble (SSLNE) is proposed to label unlabeled instances, which employs labeled instances to locate the similar local areas among historical classifiers, and then employs these local areas to assist labeling unlabeled instances. (2) A new simple method of Recurring Concept Drift Detection (RCDD) is proposed. RCDD mainly finds the classifier that has the best classification accuracy on the current data chunk in the ensemble classifiers. If the highest accuracy exceeds the preset threshold, the current data chunk corresponds to a recurring concept, otherwise corresponds to a new concept. It is very interesting that the threshold can be automatically adjusted according to the number of labeled instances.

2 Related Work

This paper is related to semi-supervised classification of data stream and transfer learning, therefore, we briefly discuss them.

The approaches for semi-supervised classification of data stream can be broadly divided into recurring-concept-based and non-recurring-concept-based. ReaSC [8] and SUN [9] are non-recurring-concept-based method. ReaSC utilizes both labeled and unlabeled instances to train and update the classification model and maintains a pool of classifiers. Each classifier is built over a data chunk as a collection of micro-clusterings which are generated through semi-supervised clustering, and an ensemble of these cluster-based classifiers is used to classify instances. SUN employs a clustering algorithm to produce concept clusters at leaves in an incremental decision tree. If a concept drift is detected, the trained decision tree is pruned to adapt to the new concept.

SPASC [10] and REDLLA [11] are recurring-concept-based approaches. SPASC maintains a pool of historical classifiers and detects the recurring concept drifts by the similarity between the current data chunk with the best classifier. REDLLA adopts decision tree as its classification model, and it detects concept drift by deviations between history and new concept clusters.

Transfer learning is an important learning method and employed to address data stream classification in recent years. Condor [12] reuses historical models to build new model and update model pool, by making use of the biased regularization technique for multiple model reuse learning. SCBELS [13] utilizes the local structure mapping strategy [14] to compute local similarity around each sample and combines with semi-supervised Bayesian method to perform concept detection which borrows idea from transfer learning.

[1] Source code: https://gitee.com/ymw12345/caplrdsrc.git.

3 The Proposed Algorithm

3.1 The Framework of CAPLRD

Without loss of generality, this paper assumes that a data stream is processed batch by batch in which some of them are randomly selected to be labeled by a supervisor. For convenience, $B^t = (x_1^t, x_2^t, ..., x_m^t)$ is used to denote a batch of instances collected in the time t. $B_L^t = (x_1^t, x_2^t, ..., x_n^t)$ and $Y_L^t = (y_1^t, y_2^t, ..., y_n^t)$ denote the labeled samples in B^t and their labels, respectively, whereas $B_U^t = (x_{n+1}^t, x_{n+2}^t, ..., x_m^t)$ denotes the remaining unlabeled instances.

CAPLRD is described in the Algorithm 1. CAPLRD employs VFDT (very fast decision tree) [15] as the base model, and many concept-specific classifiers are maintained in a pool. For each coming data chunk, the pool selects a classifier as an "active classifier" to classify the current data chunk. And then, SSLNE is employed to label the unlabeled instances. Next, RCDD is employed to detect concept drifts. If a new concept is detected, a new model is trained and added into the pool, otherwise, a historical model is updated.

```
Algorithm 1: CAPLRD
   Input: a streaming data in the form of batches: $B^t$, $B_L^t$, $Y_L^t$, $B_U^t$; the
          parameter: $\theta$
   Output: the predicted labels of all the instances in $B^t$
 1 Initialization: $activeClf$=NULL; $E$=NULL; $B^t$=NULL;
 2 $activeClf$=createTree($B_L^1$);
 3 $E = E \cup activeClf$; $r$=0; $act$=$r$; $t$=2;
 4 while a new data batch is available do
 5     $B^t$=read_next_batch(); $activeClf$.classify($B^t$);
 6     $(B', Y_L')$=SSLNE($B_L^t$, $Y_L^t$, $B_U^t$, $E$, $\theta$);
 7     $index$=RCDD($B'$, $Y_L'$, $E$, $act$);
 8     if $index == -1$ then
 9         $curClf$=createTree($B'$, $Y_L'$);
10         $curClf.highestAcc$=0; $E = E \cup curClf$; $r$++;
11         $activeClf$=$curClf$; $act$=$r$;
12     else
13         $acc$=$E_{index}$.classify($B'$);
14         $E_{index}.highestAcc$=max($acc$, $E_{index}.highestAcc$);
15         $E_{index}$.update($B'$, $Y_L'$); $activeClf$=$E_{index}$; $act$=$index$;
16     $B'$=NULL; $Y_L'$=NULL; $B^t$=NULL; $t$++;
```

3.2 Employing Historical Classifiers for Transfer Learning

SSLNE are described in the Algorithm 2. SSLNE is proposed to expand the labeled instances in the current data batch and hence alleviate the scarcity of labeled data. There are many semi-supervised learning methods like the well-known self-training [16] and tri-training [17] methods. However, in these methods, if the wrongly classified samples are added to the original training set, the errors will be accumulated in the subsequent training process. SSLNE is based on the facts that even two data batches corresponding to different concepts, their distributions may be similar in some subregions. A trained decision tree can divide an instance space into many subregions, and hence we can use the similar subregions among the historical trees to label the unlabeled instances.

Algorithm 2: SSLNE

```
   Input: the pool of historical classifiers: $E$, the current data batch: $B_L$,
          $Y_L$, $B_U$ and the threshold $\theta$
   Output: the expanded labeled data chunk: $B'$
1  $B'$=NULL; $B' = B' \cup B_L$; $Y'_L$=NULL; $Y'_L = Y'_L \cup Y_L$;
2  for each $E_i$ in $E$ do
3      $E_i.lnode.N$=0; $E_i.lnode.CN$=0;
4      for each $x_i^l$ in $B_L$ do
5          $E_i.lnode$=$E_i$.sort_into_leaf($x_i^l$); $E_i.lnode.N$++;
6          if $x_i^l$ is classified correct then
7              $E_i.lnode.CN$++;
8  for each $x_i^u$ in $B_U$ do
9      for $1 \leq j \leq labelNum$ do
10         each class $WS_j$=0;
11     for each $E_i$ in $E$ do
12         $E_i.lnode$=$E_i$.sort_into_leaf($x_i^u$);
13         if $E_i.lnode.N$==0 then
14             go to step 11;
15         else
16             label $x_i^u$ with $E_i.lnode$ and obtain the predicted label $c$;
17             $acc = E_i.lnode.CN/E_i.lnode.N$; $T = acc - 1.0/labelNum$;
18             if $T > \theta/labelNum$ then
19                 $WS_c = WS_c + T$;
20     $WS_k$=max{$WS_1$, $WS_2$, ..., $WS_{labelnum}$};
21     if $WS_k > \theta/labelNum$ then
22         $x_i^u$ is labeled with the label $k$; $B' = B' \cup x_i^u$; $Y'_L = Y'_L \cup k$;
23 return ($B'$, $Y'_L$);
```

More specifically, for each historical classifier, all the labeled instances in the current data batch are sorted into its leaves. In the process of traversing the decision tree, *lnode.N* is saved for counting the number of instances that are sorted into its corresponding leaf, while *lnode.CN* is saved for counting the number of correct classified instances among them. Then, for each historical classifier, each unlabeled instance in the current data batch is sorted into a leaf of it. The value of $lnode.CN/lnode.N$ to this leaf node can be used to present the classification confidence of the historical classifier for this unlabeled instance. The larger the value of $lnode.CN/lnode.N$, the higher the local similarity is between the historical classifier and the current data batch. At last, the historical trees with the value of $lnode.CN/lnode.N$ are ensembled to give the classification result for the unlabeled instances, then the unlabeled instance with its predicted class label is added into the set of labeled instances in the current data batch.

3.3 Recurring Concept Drift Detection

RCDD is described in the Algorithm 3. In RCDD, if the classification accuracy of a historical classifier E_i on the current data batch is higher than the threshold β ($\beta = E_i.highestAcc - \delta$, $\delta = \sqrt{\frac{2.0}{w}}$, w means the number of labeled instances in B'), then the current data batch is considered include the same concpet with E_i. The larger w means there are more labeled instances in the current data chunk, and a classifier will be evaluated more accurately with more labeled instances. And hence, we set the smaller the w and the larger the δ, even the δ is empirical.

Algorithm 3: RCDD

```
Input: the expanded labeled data batch: $B'$, the label: $Y$, the pool of
       classifiers: $E$, the index of active classifier in $E$: $r$;
Output: the index of classifier that corresponds to the same concept as
        $B'$ in $E$ or -1 which means $B'$ corresponds to a new concept
1 Compute the classification accuracy $acc$ of $E_r$ in $B'$;
2 if $acc > (E_r.highestAcc - \delta)$ then
3     return $r$;
4 $maxAcc$=0; $index$=-1;
5 for each $E_i \in E$ except $E_r$ do
6     Compute the classification accuracy $acc$ of $E_i$ in $B'$;
7     if $acc > (E_i.highestAcc - \delta)$ and $acc > maxAcc$ then
8         $maxAcc = acc$; $index = i$;
9 return $index$;
```

4 Experiments

To evaluate the performance of SSLNE, the first group is conducted to compare SSLNE with the self-training and tri-training algorithms under the framework of CAPLRD. That is, in CAPLRD, SSLNE is replaced by the self-training and tri-training algorithm in turn while the other codes remain the same.

To evaluate the performance of RCDD, the second group is conducted to compare RCDD with CCPRD which is the recurring concept drift detection method of CCP [4] under the framework of CAPLRD. That is, in CAPLRD, RCDD is replaced by CCPRD while the other codes remain the same.

To evaluate the performance of CAPLRD, the third group is conducted to compare CAPLRD with REDLLA and SPASC.

To evaluate the sensitiveness of θ to CAPLRD, in the fourth group, the values of θ is set as 0, 0.2, 0.4, 0.5, 0.6, 0.8, and 0.9, respectively.

In this paper, The size of the pool is unlimited. The unlabeled ratio is set as 0.9, which means 90% of labels are not available. These experiments are repeated 50 times for the synthetic datasets and 20 times for the real datasets. The parameters of CCPRD are set as follows: $\theta = 0.5$, $Cmax = 3$. The paired t-test at 95% confidence level has been conducted to detect whether the differences between our approach and other compared methods are statistically significant.

4.1 Datasets

In Table 1, the synthetic datasets of sea and cir are generated by MOA [18], the definition of sine is if $a * sin(b * x_1 + \theta) + c > x_2$, the label is 0, otherwise is 1. While the electricity dataset is collected from the Australia New South Wales electricity market, the weather dataset comes from an air force base with a 50-year time span and diverse weather patterns and the spam dataset is collected from various kinds of email about advertisements for products/web sites etc.

Table 1. Datasets with concept drifts.

Datasets	Drift value	Instances	Dim	Class	Chunk size
Cir	A:a = 0 b = 0 R = 2, B:a = 0 b = 0 R = 3 C:a = 2 b = 2 R = 2, D:a = 2 b = 2 R = 3 A-B-C-D-A-B-C-D	40,000	2	2	200
Sea	θ = 5-8-12-15-5-8-12-15	40,000	3	2	200
Sine	b = 1-π-3π-1-π-3π, $\theta = 0$, $a = 1$, $c = 0$	30,000	3	2	200
Electricity	Unknown	45,312	14	2	200
Spam	Unknown	4,601	57	2	100
Weather	Unknown	18,159	8	2	200

4.2 Experimental Results

From Table 2, it can be observed that the accumulative accuracy of CAPLRD based on SSLNE is significantly better than it based on self-training or tri-training in all synthetic and real datasets. These results illustrate that the transfer learning from historical classifiers can bring effective improvement for semi-supervised learning.

From Table 3, it can be observed that the accumulative accuracy of CAPLRD based on RCDD is better than it based on CCPRD on all the datasets, except slightly worse than on the weather dataset. The reason why RCDD is better than CCPRD may be that CCPRD uses the distance between the concept vector and the concept cluster to judge whether is it a recurring concept, which has a certain

Table 2. The accumulative accuracy (%) of CAPLRD on each dataset when its module for semi-supervised learning is replaced with SSLNE, self-training, and tri-training, respectively. •/◦ indicates CAPLRD based on SSLNE is significantly better/worse than the compared methods.

Datasets	Cir	Sea	Sine	Electricity	Spam	Weather
Self-training	77.03±0.10•	78.39±0.08•	66.16±0.15•	65.66±0.19•	60.22±0.32•	60.49±0.23•
Tri-training	80.66±0.09•	81.94±0.11•	69.18±0.15•	68.49±0.19•	63.40±0.43•	66.52±0.20•
SSLNE	**82.47±0.22**	**87.76±0.12**	**81.46±0.18**	**73.70±0.21**	**71.93±0.87**	**68.45±0.12**

Table 3. The accumulative accuracy (%) of CAPLRD on each dataset when its module for concept drift detection is set as RCDD and CCPRD, respectively. •/◦ indicates CAPLRD based on RCDD is significantly better/worse than it based on CCPRD.

Datasets	Cir	Sea	Sine	Electricity	Spam	Weather
CCPRD	81.45±0.16•	80.02±0.23•	77.76±0.26•	72.70±0.29•	70.65±1.03•	**68.76±0.30**◦
RCDD	**82.47±0.22**	**87.76±0.12**	**81.46±0.18**	**73.70±0.21**	**71.93±0.87**	68.45±0.12

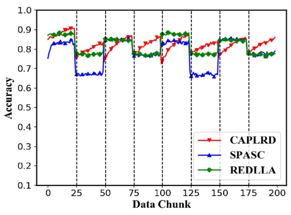
ambiguity. For CCPRD, the effect of the classifier corresponding to this concept is that the classification of current data chunk cannot be optimal. RCDD is to find a corresponding classifier with the highest classification accuracy in the ensemble model, so RCDD is more accurate than CCPRD detection.

From Table 4, it can be observed that CAPLRD achieves higher accumulative accuracy than SPASC and REDLLA on the synthetic datasets, and performs better than SPASC on real datasets. From Fig. 1, it can be observed that CAPLRD can track concept drifts more accurately, and recover to a high classification accuracy quickly on the first four datasets, REDLLA performs better than CAPLRD on the weather and spam datasets. The reason for this phenomenon may be that it is impossible to determine where the concept drift occurs for real data steams, and the artificially dividing the data streams to chunks may cause the current data chunk to be impure, that is it may contain other concepts.

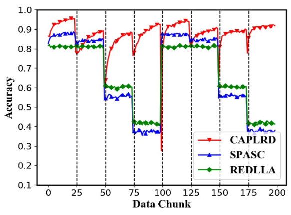
Table 4. The accumulative accuracies (%) of CAPLRD, SPASC, and REDLLA. •/◦ indicates CAPLRD is significantly better/worse than SPASC and REDLLA.

Datasets	Cir	Sea	Sine	Electricity	Spam	Weather
SAPSC	78.00±0.20•	66.13±0.41•	67.63±0.23•	58.68±0.17•	63.22±0.36•	65.31±0.28•
REDLLA	81.74±0.12•	66.05±0.24•	71.67±0.11•	66.50±0.50•	**75.26±0.35**◦	**72.16±0.19**◦
CAPLRD	**82.47±0.22**	**87.76±0.12**	**81.46±0.18**	**73.70±0.21**	71.93±0.87	68.45±0.12

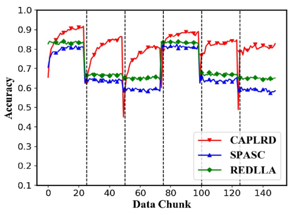
Table 5 is about the influence of θ on the CAPLRD algorithm when the value of θ was set to 0, 0.2, 0.4, 0.5, 0.6, 0.8, and 0.9, respectively. From Table 5, it can be observed that the accumulative accuracy of CAPLRD is not sensitive to θ.



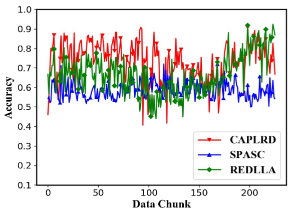
(a) cir.



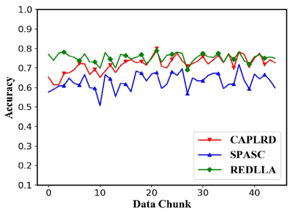
(b) sea.



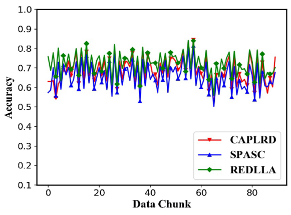
(c) sine.



(d) electricity.



(e) spam.



(f) weather.

Fig. 1. Drift tracking of CAPLRD, SPASC, and REDLLA on each dataset.

Table 5. Accumulative accuracy (%) of CAPLRD on all datasets with different θ.

Datasets	Cir	Sea	Sine	Electricity	Spam	Weather
$\theta = 0$	81.71±0.23	86.78±0.15	79.98±0.18	73.08±0.25	74.69±0.93	68.77±0.15
$\theta = 0.2$	82.16±0.20	87.09±0.15	80.27±0.16	73.28±0.17	73.50±1.25	69.08±0.17
$\theta = 0.4$	82.40±0.19	87.26±0.14	80.90±0.19	73.70±0.21	72.20±1.09	68.78±0.20
$\theta = 0.5$	82.46±0.23	87.76±0.12	81.45±0.17	73.70±0.21	71.93±0.87	68.45±0.12
$\theta = 0.6$	82.30±0.23	87.61±0.15	81.08±0.19	73.11±0.18	73.47±1.13	68.94±0.20
$\theta = 0.8$	83.01±0.23	88.06±0.13	81.35±0.15	73.61±0.16	72.77±1.03	68.38±0.12
$\theta = 0.9$	82.88±0.21	88.17±0.14	81.76±0.17	73.33±0.21	74.89±1.07	68.66±0.15

5 Conclusion

The innovation of the proposed CAPLRD lies in that it includes two components of SSLNE and RCDD. The experimental results demonstrate that the proposed SSLNE can utilizes historical classifiers to label the unlabeled instances effectively, which can expand the set of limited labeled instances and improve the generalization ability. The proposed RCDD is sensitive to the recurring concept drift detection and responds fast due to the threshold can be automatically adjusted according to the number of labeled instances. Besides, CAPLRD performs much better than REDLLA and SPASC. However it has the limitation that the base learner has to be decision tree model. How to extend the semi-supervised classification method so that any type of supervised classification model can be adopted as base learner is still challenging and interesting for future work.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China (61866007, 61662014), Natural Science Foundation of Guangxi District (2018GXNSFDA138006), and Image Intelligent Processing Project of Key Laboratory Fund (GIIP2005).

References

1. Schlimmer, J.C., Granger, R.H.: Incremental learning from noisy data. Mach. Learn. **1**(3), 317–354 (1986)
2. Sun, Y., Tang, K., Zhu, Z.X., et al.: Concept drift adaptation by exploiting historical knowledge. IEEE Trans. Neural Netw. Learn. Syst. **29**(10), 4822–4832 (2017)
3. Brzezinski, D., Stefanowski, J.: Reacting to different types of concept drift: the accuracy updated ensemble algorithm. IEEE Trans. Neural Netw. Learn. Syst. **25**(1), 81–94 (2013)
4. Katakis, I., Tsoumakas, G., Vlahavas, I.: Tracking recurring contexts using ensemble classifiers: an application to email filtering. Knowl. Inf. Syst. **22**(3), 371–391 (2010)
5. Ditzler G, Polikar R.: Semi-supervised learning in nonstationary environment. In: Proceedings of the 2011 International Joint Conference on Neural Networks, Piscataway, NJ, pp. 2741–2748. IEEE (2011)
6. Bertini, J.R., Lopes, A.D., Zhao, L.: Partially labeled data stream classification with the semi-supervised k-associated graph. J. Braz. Comput. Soc. **18**(4), 299–310 (2012)
7. Dyer, K.B., Capo, R., Polikar, R.: Compose: a semi-supervised learning framework for initially labeled nonstationary streaming data. IEEE Trans. Neural Netw. Learn. Syst. **25**(1), 12–26 (2014)
8. Masud, M.M., Woolam, C., Gao, J.: Facing the reality of data stream classification: coping with scarcity of labeled data. Knowl. Inf. Syst. **33**(1), 213–244 (2012)
9. Wu, X.D., Li, P.P., Hu, X.G.: Learning from concept drifting data streams with unlabeled data. Neurocomputing **92**, 145–155 (2012)
10. Hosseini, M.J., Gholipour, A., Beigy, H.: An ensemble of cluster-based classifiers for semi-supervised classification of non-stationary data streams. Knowl. Inf. Syst. **46**(3), 567–597 (2016)
11. Li, P.P., Wu, X.D., Hu, X.G.: Mining recurring concept drifts with limited labeled streaming data. ACM Trans. Intell. Syst. Technol. **3**(2), 1–32 (2012)
12. Zhao, P., Cai, L.W., Zhou, Z.H.: Handling concept drift via model reuse. Mach. Learn. **109**(3), 533–568 (2018)
13. Wen, Y.M., Liu, S.: Semi-supervised classification of data streams by BIRCH ensemble and local structure mapping. J. Comput. Sci. Technol. **35**(2), 295–304 (2020)
14. Gao, J., Fan, W., Jiang, J.: Knowledge transfer via multiple model local structure mapping. In: Proceedings 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, pp. 283–291. ACM (2008)
15. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the 6th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, pp. 71–80. ACM (2000)

16. Zhu, X.J., Goldberg, A.B.: Introduction to semi-supervised learning. Synth. Lect. Artif. Intell. Mach. Learn. **3**(1), 1–130 (2009)
17. Zhou, Z.H., Li, M.: Tri-training: exploiting unlabeled data using three classifier. IEEE Trans. Knowl. Data Eng. **17**(11), 1529–1541 (2005)
18. Bifet, A., Holmes, G., Kirkby, R.: MOA: massive online analysis. J. Mach. Learn. Res. **11**(2), 1601–1604 (2010)