An Adaptive Stacking Method for Multiple Data Streams Learning under Concept Drift

Kun Wang, Jie Lu*, Anjin Liu, Guangquan Zhang

Australia Artificial Intelligence Institute (AAII), University of Technology Sydney, Sydney, NSW 2007, Australia
**E-mail: jie.lu@uts.edu.au*

Multiple data streams learning has attracted high attention recently. However, different feature spaces and uncertain concept drift situations among each stream may lead to the learning decay of machine learning models. To address this issue, this paper proposes an Adaptive Stacking method for multiple data stream learning. Firstly, a stacking-based learning framework is built to handle multiple data streams with different feature spaces. Secondly, a selective retraining scheme is developed for concept drift adaptation. Finally, by testing the proposed method on five data scenarios and comparing it with three benchmarks, the experiment results show the efficiency.

Keywords: Concept drift; Data stream; Ensemble learning; Machine learning.

1. Introduction

Data stream learning has been highly focused recently, especially dealing with concept drift under uncertain environments [1–3]. Streaming data widely exists in real-world life, and many of them are shown as multiple data streams, like daily weather records of multiple regions, and interest changes among different users. However, most previous studies focus on handling a single data stream without considering multiple data streams under concept drift, which is more complicated. Therefore, to further maintain the learning performance in such a complex scenario, it is needed to consider handling multiple streams under concept drift.

In addition, previous studies about multiple data stream learning include regression [4,5] and classification tasks [6]. For the regression task, the relationship among streams has been considered and extracted to support model learning. For the classification task, methods like transfer learning [7] and ensemble learning [8] have been conducted for learning adaptation. However, few of them consider the scenario of multiple data streams with different feature spaces and drift situations, as shown in Fig. 1. Motivated by this, this work aims to focus on dealing with multiple data streams with different feature spaces, and help adapt to different concept drift situations.

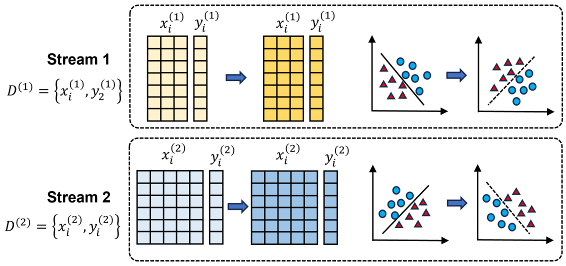


Fig. 1. An illustration of multiple data streams with concept drift.

The main contributions of this paper are listed below:

- A stacking-based learning framework is given to deal with multiple data streams with different feature spaces.
- A selective retrain scheme has been developed to obtain the learning performance.
- Experiments on five data scenarios show the efficiency of the proposed method. The source code of the proposed method is available at `https://github.com/kunkun111/Adaptive-Stacking`

The following contents of this paper are organized as follows: Section 2 summarizes the recent studies. Our proposed method is introduced in Section 3, followed by Section 4, which outlines detailed experiment setting and discussion. Finally, Section 5 gives the conclusion and future study.

2. Related Work

2.1. *Data Stream Learning under Concept Drift*

Concept drift is known as the data distribution changes timely that may lead to a learning decay[1,2], denoted as $P_t(y|x) \neq P_{t+1}(y|x)$, where t and $t+1$ are two consecutive time steps, and $P(y|x)$ expresses the data distribution with data attributes x and data label y.

To handle real-time data stream under concept drift, sufficient methods have been developed. On the one hand, some previous studies aim to identify concept drift firstly to further support model learning. For example, competence models-based concept drift detection methods[9] and Detection

Delay Index (DD Index)[10] are recently proposed concept drift detection methods. On the other hand, sufficient drift adaptation methods have been proposed to obtain performance. As ensemble learning methods perform excellently, they have been chosen and redesigned for data stream learning under concept drift[11]. Methods like Adaptive Random Forests (ARF)[12], Streaming Random Patches (SRP)[13,14], pENsemble[15], and Graph ensemble boosting[16] are widely used ensemble learning methods.

2.2. *Mutiple Data Stream Learning Approaches*

To handle more complex learning scenarios, multiple data stream learning under concept drift attracts attention[17,18]. The supervised tasks for handling multiple data streams mainly include regression and classification. For multiple data stream regression, research[4] gives a fuzzy drift correlation matrix. In addition, a graph neural network-based method has been proposed to handle multiple data stream regression under concept drift[5].

For unsupervised tasks, research[19] obtains ideas of transfer learning for handling multiple data streams. In addition, focusing on the problem of unlabelled drifting streams, a framework called Learn-to-Adapt (L2A) has been developed to handle concept drift in multiple data streams[6]. Also, clustering methods have been used for multiple streams learning[20].

3. Methodology

3.1. *Stacking-Based Multiple Data Stream Learning*

Given two data streams $S_1 = \{x_i^{S_1}, y_i^{S_1}\}_{i=1}^n$, $S_2 = \{x_i^{S_2}, y_i^{S_2}\}_{i=1}^n$, both of them consists of data chunks D with the same size at each time point. The data chunks of two streams are expressed as $\{D_t^{S_1}\}_{t=1}^T$ and $\{D_t^{S_2}\}_{t=1}^T$, where T is the total time steps. First, we initially train two base models f, g on chunks $D_t^{S_1}$ and $D_t^{S_2}$ sepatately, and then test them on $D_{t+1}^{S_1}$, $D_{t+1}^{S_2}$. Thus, we can get the initial prediction results of two chunks of data

$$\hat{y}_{t+1}^{S_1} = \{\hat{y}_{t+1}^{f,S_1}, \hat{y}_{t+1}^{g,S_1}\}, \quad \hat{y}_{t+1}^{S_2} = \{\hat{y}_{t+1}^{f,S_2}, \hat{y}_{t+1}^{g,S_2}\}. \tag{1}$$

Second, we aim to reconstruct a meta train set to combine the knowledge of two data streams together. So, we give a label α for each data stream, and reconstruct the meta train set as

$$D_{t+1}^{meta} = \{\alpha_{t+1}, \hat{y}_{t+1}, y\}, \tag{2}$$

where $\alpha = \{\alpha_{t+1}^{S_1} \cup \alpha_{t+1}^{S_2}\}$, $\hat{y} = \{\hat{y}_{t+1}^{S_1} \cup \hat{y}_{t+1}^{S_2}\}$. $y = \{y_{t+1}^{S_1} \cup y_{t+1}^{S_2}\}$ is the true label of two data chunks. Then, we train a new learning model on this meta training set, denoted as

$$\hat{y} = F(\alpha_{t+1}, \hat{y}_{t+1}) + \epsilon, \tag{3}$$

where ϵ is the learning error. This new learning model can be an ensemble model with several base learners.

3.2. *Selective Retraining Scheme for Learning Adaption*

To maintain learning performance under concept drift, a selective retraining scheme has been utilized. First, after testing the model at time $t+1$, the accuracy of model on each data stream at time t and $t+1$ can be obtained, denoted as $\{\varepsilon_t^{S_1}, \varepsilon_{t+1}^{S_1}\}, \{\varepsilon_t^{S_2}, \varepsilon_{t+1}^{S_2}\}$. Then, we retrain the base model of each stream once the accuracy decreases, denoted as

$$\begin{cases} \text{If } \varepsilon_t^{S_1} > \varepsilon_{t+1}^{S_1} \textbf{ or } \varepsilon_t^{S_2} > \varepsilon_{t+1}^{S_2}, \ \text{retrain } f^{S_1}, g^{S_1} \textbf{ or } f^{S_2}, g^{S_2} \\ \qquad\qquad\qquad\qquad\qquad\quad \text{Else, } \ \text{test } f^{S_1}, g^{S_1} \textbf{ and } f^{S_2}, g^{S_2} \end{cases}. \tag{4}$$

Then, to further reduce the impact of concept drift, we also consider updating the meta train set D_{t+1}^{meta} adaptively to help retrain the meta model. The meta train set updation depends on whether the accuracy is decreased, expressed as

$$\begin{cases} \text{If } \varepsilon_t^{S_1} > \varepsilon_{t+1}^{S_1}, \varepsilon_t^{S_2} \leq \varepsilon_{t+1}^{S_2}, \ D_{t+1}^{meta} = D_{t+1}^{meta,S_1} \cup D_{t+1}^{meta,S_2} \cup D_t^{meta,S_2} \\ \text{If } \varepsilon_t^{S_1} \leq \varepsilon_{t+1}^{S_1}, \varepsilon_t^{S_2} > \varepsilon_{t+1}^{S_2}, \ D_{t+1}^{meta} = D_{t+1}^{meta,S_1} \cup D_t^{meta,S_1} \cup D_{t+1}^{meta,S_2} \\ \text{If } \varepsilon_t^{S_1} \leq \varepsilon_{t+1}^{S_1}, \varepsilon_t^{S_2} \leq \varepsilon_{t+1}^{S_2}, \ D_{t+1}^{meta} = D_{t+1}^{meta} \cup D_t^{meta} \\ \text{If } \varepsilon_t^{S_1} > \varepsilon_{t+1}^{S_1}, \varepsilon_t^{S_2} > \varepsilon_{t+1}^{S_2}, \ D_{t+1}^{meta} = D_{t+1}^{meta} \end{cases}. \tag{5}$$

Finally, retrain the meta model on the updated meta train set, thus helping the learning model handle different types of concept drift.

4. Experiment

4.1. *Experiment Setting*

Datasets To simulate the scenarios of multiple data streams with different drift situations, we choose six commonly used synthetic datasets (SEAa (sudden drift), RTG (no drift), RBF (incremental drift), RBFr (incremental drift), AGRa (sudden drift), HYP (incremental drift)) for the experiment. We simply deal with two data streams at the same time, five scenarios with

Table 1. Data scenarios

Scenarios	#Sample	#Feature	#Class	#Drift type
SEAa → RTG	10,000	3	2	No
SEAa → RBF	10,000	3	2	Incremental
SEAa → RBFr	10,000	3	2	Incremental
SEAa → AGRa	10,000	3	2	Sudden
SEAa → HYP	10,000	3	2	Incremental

different drift situations are given, as shown in Table 1. We repeat the experiment on each scenario 15 times and calculate the average results of accuracy and F1 score.

Benchmarks In the experiment, we compare our proposed self-adaptative Stacking method with three benchmark methods, as described below:

- **Baseline:** is the initial stacking-based framework, which is trained on the first chunk and tested on the rest of the chunks.
- **Retrain-B:** retrains base models in the stacking-based framework once the learning accuracy is reduced.
- **Retrain-M:** retrains the meta model in the stacking-based framework on the updated meta train set once the learning accuracy is degraded.
- **Adaptive Stacking:** combines the processes of Retrain-B and Retrain-M in the stacking-based framework.

The experiment is run under Python 3.10, the computing environment is 12th Gen Intel(R) Core(TM) i7-1255U 1.70 GHz with 16.0 GB RAM.

4.2. *Experiment Results Discussion*

We have evaluated the proposed method on five scenarios of multiple data streams, and collect the average accuracy and F1 score, as shown in Tables 2, 3. According to the experiment results, it can be shown that the Baseline method performs not ideal because of the lack of real-time adaptation. Then, by selectively retraining the base model or meta model in methods of Retrain-B and Retrain-M, the average performance has been improved respectively. Finally, the proposed Adaptive Stacking method got a relatively higher performance with a lower standard error. Such performance improvement is ideal in five scenarios, which can deal with multiple data streams with different feature spaces and drift situations.

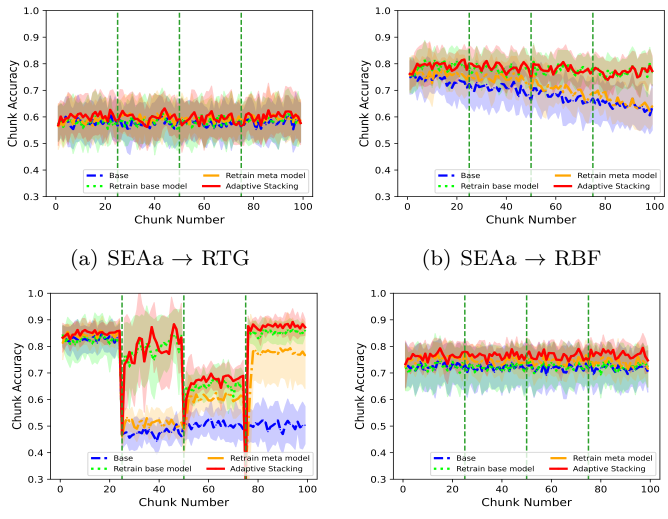
To further observe the experiment results, the average chunk accuracy

Table 2. Average Accuracy of the Adaptive Stacking Method

Scenarios	Baseline	Retrain-B	Retrain-M	Adaptive Stacking
SEAa → RTG	57.86 ± 5.9	58.31 ± 6.4	59.25 ± 5.4	59.79 ± 5.3
SEAa → RBF	69.11 ± 5.2	77.83 ± 3.4	71.65 ± 4.6	78.16 ± 3.0
SEAa → RBFr	68.20 ± 5.0	72.60 ± 6.3	71.42 ± 3.5	75.21 ± 2.5
SEAa → AGRa	57.54 ± 1.3	77.04 ± 2.6	67.23 ± 2.4	78.75 ± 1.2
SEAa → HYP	72.02 ± 6.7	72.28 ± 6.3	74.56 ± 3.2	76.54 ± 0.5

Table 3. Average F1 score of the Adaptive Stacking Method

Scenarios	Baseline	Retrain-B	Retrain-M	Adaptive Stacking
SEAa → RTG	49.56 ± 7.5	50.74 ± 9.3	50.19 ± 6.8	53.19 ± 6.5
SEAa → RBF	66.98 ± 6.3	76.92 ± 3.3	70.27 ± 5.0	77.32 ± 2.9
SEAa → RBFr	65.58 ± 6.3	70.22 ± 8.7	69.84 ± 3.7	74.09 ± 2.8
SEAa → AGRa	57.14 ± 1.4	76.95 ± 2.8	67.22 ± 2.4	78.73 ± 1.2
SEAa → HYP	70.58 ± 10.5	70.89 ± 10.3	74.42 ± 3.3	76.53 ± 0.5



(a) SEAa → RTG

(b) SEAa → RBF

(c) SEAa → AGRa

(d) SEAa → HYP

Fig. 2. A plot of the chunk accuracy of the proposed method with several datasets.

of each benchmark method on different scenarios has been visualized, as shown in Figure 2. For handling sudden drift, the proposed method can help recover the learning performance better when drift occurs. For dealing with incremental drift, the Adaptive Stacking method got a higher performance.

5. Conclusion

This paper develops an Adaptive Stacking method for dealing with multiple data streams under concept drift. A stacking-based learning framework has been built to deal with multiple data streams with different feature spaces and drift situations. Appropriately and selectively retrain the base model and meta model to help maintain the learning performance. The experiment results on five data scenarios show the efficiency of the proposed method. In our future study, we will continue to optimize the current method for better performance and robustness.

Acknowledgments

This work was supported by the Australian Research Council under Grant FL190100149.

References

1. J. Lu, A. Liu, F. Dong, F. Gu, J. Gama and G. Zhang, Learning under concept drift: A review, *IEEE Transactions on Knowledge and Data Engineering* **31**, 2346 (2019).
2. J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy and A. Bouchachia, A survey on concept drift adaptation, *ACM Computing Surveys* **46**, 1 (2014).
3. J. Shao, Y. Tan, L. Gao, Q. Yang, C. Plant and I. Assent, Synchronization-based clustering on evolving data stream, *Information Sciences* **501**, 573 (2019).
4. Y. Song, G. Zhang, H. Lu and J. Lu, A fuzzy drift correlation matrix for multiple data stream regression, in *2020 IEEE International Conference on Fuzzy Systems*, (IEEE, 2020).
5. M. Zhou, J. Lu, Y. Song and G. Zhang, Multi-stream concept drift self-adaptation using graph neural network, *IEEE Transactions on Knowledge and Data Engineering* **35**, 12828 (2023).
6. E. Yu, Y. Song, G. Zhang and J. Lu, Learn-to-adapt: Concept drift adaptation for hybrid multiple streams, *Neurocomputing* **496**, 121 (2022).
7. A. Haque, Z. Wang, S. Chandra, B. Dong, L. Khan and K. W. Hamlen, Fusion: An online method for multistream classification, in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, (ACM, 2017).

8. K. Wang, J. Lu, A. Liu and G. Zhang, An augmented learning approach for multiple data streams under concept drift, in *36th Australasian Joint Conference on Artificial Intelligence, AI 2023*, (Springer, 2023).
9. N. Lu, G. Zhang and J. Lu, Concept drift detection via competence models, *Artificial Intelligence* **209**, 11 (2014).
10. A. Liu, J. Lu, Y. Song, J. Xuan and G. Zhang, Concept drift detection delay index, *IEEE Transactions on Knowledge and Data Engineering* **35**, 4585 (2022).
11. B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski and M. Woźniak, Ensemble learning for data stream analysis: A survey, *Information Fusion* **37**, 132 (2017).
12. H. M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfharinger, G. Holmes and T. Abdessalem, Adaptive random forests for evolving data stream classification, *Machine Learning* **106**, 1469 (2017).
13. H. M. Gomes, J. Read and A. Bifet, Streaming random patches for evolving data stream classification, in *2019 IEEE International Conference on Data Mining*, (IEEE, 2019).
14. H. M. Gomes, J. Read, A. Bifet and R. J. Durrant, Learning from evolving data streams through ensembles of random patches, *Knowledge and Information Systems* **63**, 1597 (2021).
15. M. Pratama, W. Pedrycz and E. Lughofer, Evolving ensemble fuzzy classifier, *IEEE Transactions on Fuzzy Systems* **26**, 2552 (2018).
16. S. Pan, J. Wu, X. Zhu and C. Zhang, Graph ensemble boosting for imbalanced noisy graph stream classification, *IEEE Transactions on Cybernetics* **45**, 954 (2014).
17. J. W. Sangma, V. Pal, N. Kumar, R. Kushwaha *et al.*, Fhc-nds: Fuzzy hierarchical clustering of multiple nominal data streams, *IEEE Transactions on Fuzzy Systems* **31**, 786 (2022).
18. H. Yu, W. Liu, J. Lu, Y. Wen, X. Luo and G. Zhang, Detecting group concept drift from multiple data streams, *Pattern Recognition* **134**, p. 109113 (2023).
19. S. Chandra, A. Haque, L. Khan and C. Aggarwal, An adaptive framework for multistream classification, in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*, (ACM, 2016).
20. A. Otero, P. Félix, D. G. Márquez, C. A. García and G. Caffarena, A fault-tolerant clustering algorithm for processing data from multiple streams, *Information Sciences* **584**, 649 (2022).