# Online automated machine learning for class imbalanced data streams

Wang, Zhaoyang; Wang, Shuo

*DOI:*
10.1109/IJCNN54540.2023.10191926

# Online automated machine learning for class imbalanced data streams

1st Zhaoyang Wang
*School of Computer Science*
*University of Birmingham*
Birmingham, UK
zxw180@student.bham.ac.uk

2nd Shuo Wang*
*School of Computer Science*
*University of Birmingham*
Birmingham, UK
s.wang.2@bham.ac.uk

***Abstract*—Automated machine learning (AutoML) has achieved great success in offline class imbalance learning where data are static. However, many real world applications data nowadays tend to evolve over time in the form of data streams and involve class imbalance distributions, e.g., intrusion detection, fault diagnosis systems, and fraud detection. These learning tasks require AutoML processing the instances instantly and adapting to the dynamic data changes. Nevertheless, existing AutoML research either only focuses on class imbalance in static data sets, or discusses data streams with concept drift. No existing work studied the joint learning challenges of class imbalance and online data stream learning in AutoML. To close the gap, this paper focuses on learning dynamic data streams with a skewed class distribution in AutoML. In this paper, we propose two new AutoML approaches, UEvoAutoML and OEvoAutoML, which integrate adaptive resampling techniques into an existing online AutoML framework. Their performance is investigated through a set of synthetic imbalanced data streams under various stationary and non-stationary scenarios and 5 real-world data streams. As the pioneering work of exploring how class imbalance techniques benefit online AutoML, this paper demonstrated that the effectiveness of adaptive resampling in AutoML frameworks.**

***Index Terms*—Automated machine learning, class imbalance, data stream, evolutionary learning, ensemble learning,**

## I. Introduction

AutoML is designed to automatically build machine learning systems given independent and identically distributed (i.i.d) data that are accessible at once. It has achieved great success and has been applied in various fields, such as computer vision, data mining and natural language processing.

However, the well-studied offline setting does not apply to data collected over time as a stream [1] [2]. Firstly, the data are usually infinite. The unbounded data stream should be fed into the learning process constantly whenever new data become available. Secondly, many high-speed data streams require timely processing in a "one-by-one" fashion without storing and reprocessing learned instances (i.e. online learning), and provide real-time predictions. Thirdly, a data stream tends to be dynamic, which leads to concept drift where the underlying distribution of data changes over time. Unfortunately, very little AutoML research has studied these challenges in data streams [3].

The issue becomes more difficult when class imbalance exists in data streams. This problem is called online class imbalance in the literature [2]. Class imbalance refers to a data distribution where some classes are significantly under-represented in comparison to other classes. It causes poor generalization capability and learning bias on majority classes in learners [4]. When class imbalanced data arrive sequentially, new challenges arise in the following three aspects. Firstly, the imbalance degree can not be measured directly because of incomplete data information at certain time. It is vital to reflect the recent imbalance status correctly for using the correct class imbalance techniques. Secondly, the imbalance condition may dynamically alter over time. Additionally, the imbalanced data streams may also involve the real concept drift [5]. A real concept is referred to as a change in posterior probabilities $p(y|x)$, which leads to an out-of-date learner. Because class imbalance and real concept drift degrade a learner differently, they requires different treatments [6]. An AutoML framework should be adaptive to the changing imbalance status as well as new concept distributions.

Although there are several papers studying AutoML for offline class imbalance and online data stream learning, respectively, the joint issue has not been studied yet. Solving this joint problem can benefit many real world applications, such as social media mining [7], fault diagnosis monitoring systems [8], and risk assessment in bank [9]. The most recent online AutoML framework is called EvoAutoML(evolution-based online automated machine learning) [10]. It processes instances strictly online (one-by-one learning) and handles real concept drift, but it cannot deal with class imbalance and any class status changes.

To fill in these research gaps, we propose two online AutoML frameworks based on EvoAutoML by integrating with resampling techniques [2], called Undersampling-based EvoAutoML (UEvoAutoML) and Oversampling-based EvoAutoML(OEvoAutoML). They can deal with the dynamic class imbalance issue and passively adapt to real concept drift through the AutoML process. The two approaches are evaluated by focusing on the following research questions in comparison with EvoAutoML: (1) How do they perform on data streams with a static class imbalance ratio? (2) How do they perform when the data stream has a dynamically changing

* the corresponding author.

imbalance ratio? (3) How do they perform when both real concept drift and class imbalance exist? (4) How do they perform on real-world imbalanced data streams?

Our experiments show that our approaches can effectively alleviate the class imbalance issue compared with EvoAutoML. Firstly, UEvoAutoML performs better when the data streams has a static class imbalance ratio and a dynamically changing imbalance ratio. Secondly, for the real concept drift and class imbalance, UEvoAutoML has good performance on mild changes and OEvoAutoML performs better on severe changes. Thirdly, UEvoAutoML achieves the best performance in real-world data streams.These findings confirm that adaptive resampling is necessary and effective for dealing with the class imbalance in data streams in terms of online AutoML.

## II. PROBLEM STATEMENT

### A. Online Automated machine learning

Online AutoML is to address the online Combine Algorithm Selection and Hyperparameter(CASH) optimization problems. The online CASH follows the definition in [10]. A machine learning(ML) pipeline structure $g \in G$ can be modelled as an arbitrary directed acyclic graph (DAG). We define a possible infinite sequence of instances $S = \{e_0, e_1, ..., e_t, ...\}$ $e_i = (x_i, y_i)$ denotes each learning instance that $x_i$ is a $d-$dimensional input vector belonging to input space $X$ and $y^t$ is the corresponding class label belonging to clas space $Y$) and a set of online algorithms $\mathcal{A} = \{A^1, ..., A^R\}$ (each algorithm $A^j$ having the hyperparameters space $\{\Lambda^1, ..., \Lambda^j\}$). The set of past observed instances is represented as $S^-$. Let $\mathcal{L}(\mathcal{P}_{g,A,\lambda}(S^T), S^V)$ indicates the loss that algorithm combination $P^j$ obtains on a subset of validation instances $S^V \subset S^-$ when trained on $S^T$, and $S^T \cap S^V = \emptyset$

The online CASH problem is defined as finding the joint algorithm and hyperparameter setting that minimizes the loss function in (1).

$$g^*, A^*, \lambda^* \in \underset{P^{(j)} \in \mathcal{P}, A^{(j)} \in \mathcal{A}, \lambda \in \Lambda^{(j)}, g \in G}{\arg\min} \mathcal{L}(\mathcal{P}_{g,A,\lambda}(S^T), S^V) \tag{1}$$

## III. RELATED WORK

### A. AutoML for data streams

AutoML techniques in learning data streams have drawn a growing attention in ML communities in recent years. Madrid et al. [11] first attempted to modify the current Autosklearn framework in order to adapt to the dynamic data streams context. Retraining strategies are exploited when the concept drift occurs, which makes the Autosklearn adaptive with the distribution of recent data. Similarly, three AutoML frameworks (GAMA, Autosklearn, and H2O) are integrated with different adaptation strategies [12]. Nevertheless, the computational cost of these frameworks is high. They still use the offline AutoML frameworks that include offline learners in pipelines, which are inappropriate for data streams.

Two frameworks were then proposed to overcome these limitations. EvoAutoML is the first framework for dealing with online data streams one-by-one [10]. Its search space involves online classifiers which are optimised after every certain time steps. Automated Machine Learning (OAML) [13] processes data stream in both batch and one-by-one modes with adaptation capabilities. When the concept drift is not triggered, OAML learns the instance one-by-one. Once a real concept drift is detected, the asynchronous genetic programming in OAML is exploited to search the optimal model and hyperparameter through the recent batch data.

In addition to adaptive AutoML frameworks, some studies focus on continuous hyperparameter optimization for data streams without optimizing the entire pipeline. ChaCha(Champion-Challengers) automatically selects promising hyperparameters in online settings [15]. Veloso et al. [14] extended the Self Parameter Tuning (SPT). It restarts the optimization process to obtain a new optimal hyperparameter configuration using the Nelder-Mead algorithm when a drift is identified. A classifier with a small amount of hyperparameter settings can benefit from this method. All of the aforementioned techniques ignore the class imbalance issue in data streams. It can potentially cause the optimization process having a learning bias toward the majority class and poor generalization in the minority class. The learning difficulty is further exacerbated when the imbalance status changes over time.

### B. AutoML for class imbalanced data

Some AutoML studies have discussed static imbalanced datasets, where a whole dataset is available before training starts. Truong et al. [17] investigated how the current AutoML frameworks are affected by class imbalance. The results have showed that the higher degree of imbalance, the greater negative impact on the AutoML frameworks, including Autokeras, Autosklearn, H2O, and TPOT. Wang et al. [18] proposed an AutoML framework to handle the imbalance issue. In this framework, the Bayesian optimization is employed to choose the optimal pipeline in the search space that consists of 12 resampling techniques(undersampling, oversampling and hybrid sampling), along with two cost-sensitive loss functions. Similarly, the recent AutoBalance framework alleviated class imbalance by tweaking the GAMA framework [16]. Resampling approaches were embedded into the mAML framework that improved the disease prediction [19]. They show the necessity of handling class imbalance, but are not applicable to data streams.

## IV. METHODOLOGY

In this section, we propose OEvoAutoML and UEvoAutoML approaches that integrate adaptive resampling into EvoAutoML. We focus on binary classification.

### A. Adaptive imbalance status in online data streams

In binary classification, the imbalance ratio (IR) of a data stream is defined as the prior probability of the minority class in recent time steps. To reflect a real-time class imbalance status in data streams, the time decay class probability is calculated to determine which class is minority/majority and measure the current class size ratio between classes [2].

At each time $t$, the size of each class is incrementally updated by (2).

$$s_k^t = \theta s_k^{t-1} + I_{y^t=c_k}(1-\theta) \tag{2}$$

where $s_k^t$ represents the decay size of class $c_k$ at time step t, and $I_{y^t=c_k} = 1$ if the true class label of $x_t$ is $c_k$, otherwise, 0. $\theta(0 < \theta < 1)$ is the predetermined time decay factor that gives more emphasis on the current class status of data stream and reduce the impact of older instances. In this work, the label of $c_0$ and $c_1$ is set to ”0” and ”1”, respectively. The time decay class size $c_0$ and $c_1$ denote the $s_0^t$, $s_1^t$, respectively.

At any given time $t$, the time decay class size is able to identify which class is smaller or bigger and help calculate the size ratio between classes. This information is used to adaptively decide the resampling rate in our approaches.

### *B. UEvoAutoML and OEvoAutoML*

UEvoAutoML and OEvoAutoML are proposed to learn the online class imbalanced data streams in an AutoML framework. They are explained in Algorithm 1.

The original training of EvoAutoML ensembles a fixed number of $P$ pipelines in a dynamic manner. Mutation of the best pipeline $\mathcal{P}^{best}$ and deletion of the worst pipeline $\mathcal{P}^{weak}$ are performed at a fixed interval according to the accuracy scores. The interval is controlled by the sampling rate $f$. After each evolution step, each classifier in pipeline is updated $K$ times using the current instance $e_t = (x_t, y_t)$, where $K$ follows the $Poisson(\lambda = 6)$ distribution inspired by [21]. Although the larger $\lambda$ can increase the diversity of weights, it also increases the updating times of each pipeline for each instance, leading to higher computation cost. In the prediction stage, a hard majority voting methods of the heterogeneous algorithm configurations in $p$ is employed to decide the label $\hat{y_t} = model(\mathcal{P}.predict(e_t) \in p)$ in ensemble fashion.

To better learn dynamic imbalanced data streams, $\lambda$ in UEvoAutoML and OEvoAutoML is automatically adjusted in accordance with the current class size ratio at any given time $t$ in the data streams for adaptive resampling. To achieve this, the time decayed class size in Equation2 is integrated. In lines 12-15, the time decay class size of each class ($s_0^t$ and $s_1^t$) is calculated at each time step $t$, and its ratio is used to set $\lambda$. For instance, if the time decay size of $c_1$ is smaller than $c_0(s_0^t > s_1^t)$ at the current time step $t$, the $\lambda = s_0^t/s_1^t$ will be assigned to $c_1$ in OEvoAutoML(i.e. increasing the chance of training with minority class instances) ; the $\lambda = s_1^t/s_0^t$ will be assigned to $c_0$ in UEvoAutoML(i.e. decreasing the chance of training with majority class instances). When time decay class size is equal, it implies a balanced status, and the $\lambda$ is set to 1, following the Online Bagging [24]. This core resampling is inspired by [6]. The prediction stage of UEvoAutoML and OEvoAutoML is the same as the EvoAutoML.

In addition, the evolution process for EvoAutoML uses the accuracy scores, which leads to the evolution's pipelines having biases on majority class in class imbalance learning(see Section V-C). Thus, UEvoAutoML and OEvoAutoML will exploit the G-mean as the evolution metric scores. For fair comparison, the EvoAutoML will also set the G-mean as evolution metric, instead of accuracy metric.

**Algorithm 1** UEvoAutoML and OEvoAutoML Training

```
Input: Data stream $S$, population size $P$, sampling rate $f$,
    loss function $\mathcal{L}$, search space $\mathcal{A}, \Lambda, G$, time decay factor
    $\theta$.
Output: Set of suited algorithm configurations: $p^* =
    \{\mathcal{P}^{(1)}, .., \mathcal{P}^{(P)}\}$
 1: Initialize $p$: $\mathcal{P} \leftarrow$ Random($\mathcal{A}, \Lambda, G$)
 2: if $e_t = (x_t, y_t)$ then
 3:     if $t$ mod $f$==0 then
 4:         $\mathcal{P}^{best} \leftarrow \min_{\mathcal{P} \in p} \mathcal{L}(\mathcal{P}(S^T), S^V)$
 5:         $\mathcal{P}^{weak} \leftarrow \max_{\mathcal{P} \in p} \mathcal{L}(\mathcal{P}(S^T), S^V)$
 6:         $\mathcal{P}^{mut} \leftarrow$ Mutate($\mathcal{P}^{best}$)
 7:         $p \leftarrow p \cup \mathcal{P}^{best}$
 8:         $p \leftarrow p \setminus \mathcal{P}^{best}$
 9:     end if
10:     $s_0^t = \theta s_0^{t-1} + I_{y^t=c_0}(1-\theta)$
11:     $s_1^t = \theta s_1^{t-1} + I_{y^t=c_1}(1-\theta)$
12:     if $y_t = 1$ and $\begin{cases} s_0^t < s_1^t & \text{for UEvoAutoML} \\ s_0^t > s_1^t & \text{for OEvoAutoML} \end{cases}$ then
13:         set $\lambda = s_0^t/s_1^t$
14:     else if $y_t = 0$ and $\begin{cases} s_0^t > s_1^t & \text{for UEvoAutoML} \\ s_0^t < s_1^t & \text{for OEvoAutoML} \end{cases}$ then
15:         set $\lambda = s_1^t/s_0^t$
16:     else
17:         set $\lambda = 1$
18:     end if
19:     $K \sim Poisson(\lambda)$
20:     for $\mathcal{P} \in p$ do
21:         update $K$ times: $\mathcal{P}$.fit($e_t$)
22:     end for
23:     $t = t + 1$
24: end if
```

There are three main advantages of UEvoAutoML and OEvoAutoML. First, any online classifiers are applicable in the search space of online AutoML. This is because the oversampling and undersampling are data level approaches to handle class imbalance, without relying on the algorithms. Second, they are able to dynamically adjust the resamlping rate and adapt to the changing imbalance data streams. Third, to a certain degree, they allow to handle real concept drift with class imbalance due to the search process of passive evolution.

## V. EXPERIMENTAL SETUP

### *A. Data sets*

*1) Synthetic Data sets:* In our experiments, we adopt two commonly used data stream generators, i.e., SEA [22] and SINE [23]. They allow us to control the key factors, e.g, the class imbalance degree, when and what types of drifts being considered. Various class imbalanced scenarios without and with real concept drifts are designed to verify the performance of proposed methods. The detailed settings are given in Section VI.

*2) Real-world Data sets:* Five public real-world data sets commonly used in class imbalance data stream learning are chosen in our experiments. They cover the topics of finance banking, weather, environment, etc.

(1) The Credit Card Frauds(Frauds) [25] data set consists of transactions made by credit cards in September 2013 by European cardholders. It includes 492 frauds out of 284807 transactions (IR 0.172%).

(2) The task of Forest cover type(Covtype) [26] data set is to predict forest cover type(7 classes) given $30 \times 30$ meter cells from the Roosevelt National Forest in Colorado. It includes cartographic information that was determined from US Forest Service (USFS). This multi-class data streams are converted to binary streams using the same methodology as in [5], which chooses one category as the majority and another as the minority. The cover type ”4” is selected as the minority class with 2747 instances and the cover type ”3” is chosen as the majority class with 35754 instances. Thus, IR is approximately 7%.

(3) The Weather [27] data set includes the weather information with eight features from the Nebraska by the National Oceanic and Atmospheric Administration (NOAA).The task is to predict whether it will rain or not. There are total 18159 instances, and the rain instances(i.e. minority class) are 5698. Therefore, IR is approximately 30%.

(4) Given Me Some Credit(GMSC) [28] is a credit scoring data set. The task is to decide whether a loan should be granted. Incorrect loans result in the risk of default and would add extra costs on future lawsuits. This data set consists of 120269 instances after instances of missing values are deleted. The minority class account accounts for around 7% of all borrowers.

(5) The Poker [29] data set has ten classes and ten predictive attributions. Each record of Poker is an example of a hand containing five playing cards drawn from a standard deck. We convert it into a binary dataset by selecting class 3 as the majority with 17541 instances and class 7 as the minority with 195 instances. IR is approximately 1%.

### *B. AutoML setting and configurations*

To verify whether the adaptive resampling is effective, the search spaces of UEvoAutoML and OEvoAutOML are identical to the EvoAutoML [10]. Similarly, the population size is 10 and sampling rate is 1000. The search space includes three preprocessing algorithms (i.e. a missing value cleaner, min-max scaler and a standard scaler) and four online classifiers with their hyperparameters(i.e. Gaussian Naive Bayes, HoeffdingTreeClassifier, k-Nearest Neighbors, and Logistic Regression classifiers). The detailed algorithm settings of searching space can be found in [10].

The time decay factor $\theta$ in Equation 2 is set to 0.9 for UEvoAutoML and OEvoAutoML. This value can best balance their reacting speed and prediction variance based on preliminary experiments.

### *C. Performance Metrics and Evaluation*

The overall accuracy metric biases toward the majority class when the dataset is class imbalanced. Thus, the geometric mean(G-mean) that is not sensitive to class imbalance is commonly used for performance evaluation [6]. In binary classification, it is defined as the G-mean = $\sqrt{(Recall\_0) \times (Recall\_1)}$. Recall represents the classification accuracy on a single class. Specifically, recall on class 0 denotes the “Recall_0” and recall on class 1 refers to “Recall_1”. G-mean measures the overall performance and a high G-mean implies a classifier that has high accuracy on both classes.

The prequential evaluation (aka. test-then-train) is used to evaluate and compare the three online AutoML frameworks. Data collected at each time step are first used to test the AutoML pipeline before being used for updating it. The performance is updated incrementally. and is commonly employed in the online learning literature.

Additionally, in all experiments, Wilcoxon Sign Rank [6] is used to test the statistical significance and determines which of the approach is significantly different, based on 30 runs of the AutoML training. The significance level is set to 0.05. For space reason, the p-value of this test will be presented in close performance between AutoML approaches in the next experiments.

## VI. Experimental Results

In this section, we discuss our experimental results on the synthetic and real-world data streams.

### *A. Stationary Data Streams*

The aim of this experiment is designed to answer Research Question 1, i.e., how do UEvoAutoML and OEvoAutoML perform on data streams with a static class imbalance ratio? We focus on analysing that to what extent do the UEvoAutoML and OEvoAutoML help cope with the stationary class imbalance without concept drift. To assess this, the SINE and SEA are leveraged to generate four data streams with a different IR, i.e. 0.1%, 0.5%, 1%, and 10%, respectively. Each data stream has 100000 instances, and the class 1 is fixed as the minority class in this experiment.

The three AutoML algorithms(i.e. UEvoAutoML, OEvoAutoML, and EvoAutoML) are compared through the prequential G-mean and Recall of class 1 (minority) at the final step averaged over 30 repetitions. Their means and standard deviations are shown in Table I and Table II.

In Table I and Table II, we can see that the UEvoAutoML has the best performance in all of the cases in terms of G-mean and Recall_1. The OEvoAutoML is the second best and usually outperforms the EvoAutoML in most situations. Particularly, UEvoAutoML is robust to highly skewed class distribution(i.e. 0.1%). The performance of other AutoML declines in this case and the minority class is difficulty identified.

TABLE I
THE FINAL STEP G-MEAN ON FOUR DATA STREAMS WITH VARYING IR FOR SINE AND SEA

| Data sets | IR | UEvoAutoML | OEvoAutoML | EvoAutoML |
|---|---|---|---|---|
| SINE | 0.1% | 0.8280±0.0275 | 0.3160±0.0117 | 0.5457±0.0438 |
| | 0.5% | 0.9434±0.0192 | 0.6700±0.0042 | 0.3785±0.0832 |
| | 1% | 0.9616±0.0030 | 0.7864±0.0026 | 0.5052±0.0019 |
| | 10% | 0.9713±0.0007 | 0.9381±0.0007 | 0.8540±0.0003 |
| SEA | 0.1% | 0.9024±0.0214 | 0.6524±0.0363 | 0.6210±0.0258 |
| | 0.5% | 0.9458±0.0067 | 0.6588±0.0382 | 0.7701±0.0006 |
| | 1% | 0.9597±0.0027 | 0.7522±0.0038 | 0.4611±0.0021 |
| | 10% | 0.9662±0.0007 | 0.9400±0.0009 | 0.8527±0.0004 |

TABLE II
THE FINAL STEP RECALL_1 ON FOUR DATA STREAMS WITH VARYING IR FOR SINE AND SEA

| Data sets | IR | UEvoAutoML | OEvoAutoML | EvoAutoML |
|---|---|---|---|---|
| SINE | 0.1% | 0.7907±0.0587 | 0.1000±0.0074 | 0.2997±0.0473 |
| | 0.5% | 0.9355±0.0247 | 0.4492±0.0057 | 0.1500±0.0982 |
| | 1% | 0.9559±0.0058 | 0.6193±0.0042 | 0.2554±0.0019 |
| | 10% | 0.9690±0.0020 | 0.8914±0.0013 | 0.7350±0.0005 |
| SEA | 0.1% | 0.8990±0.0437 | 0.4270±0.0491 | 0.3863±0.0322 |
| | 0.5% | 0.9683±0.0148 | 0.4359±0.0575 | 0.5971±0.0009 |
| | 1% | 0.9757±0.0066 | 0.5678±0.0057 | 0.2130±0.0020 |
| | 10% | 0.9842±0.0014 | 0.9067±0.0016 | 0.7370±0.0006 |

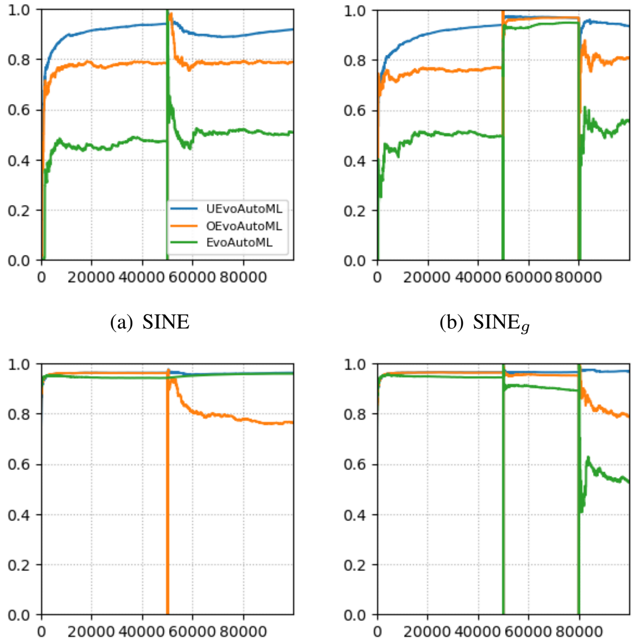
The reason for UEvoAutoML's advantage of identifying the minority class is that with higher IR, UEvoAutoML reduces the number of majority class by adaptive undersampling. As a result, it gives more emphasis on the minority class after the entire data streams have been learned. In contrast, a smaller number of minority class under higher IR can not be sufficiently learned for the OEvoAutoMl and EvoAutoML, which may even cause overfitting.

The three AutoML approaches differ significantly in terms of G-mean and Recall_1 according to Wilcoxon Sign Rank test. For example, in SEA data streams with IR 0.1% for G-mean, OEvoAutoML significantly outperforms the EvoAutoML with p-value 0.0018, and the significant superiority of UEvoAutoML against EvoAutoML with p-value 1.7344e-06. The similar statistic test results confirm that adaptive undersampling in UEvoAutoML is more effective and significantly improve the prediction performance with regard to balancing accuracy between the two classes.

The findings from the stationary data streams tell us, adaptive resampling techniques are necessary to alleviate the class imbalance problem in online AutoML. UEvoAutoML is more resistant to extremely skewed class distributions and can significantly improve the performance of EvoAutoML in stationary imbalanced data streams.

### B. Nonstationary data streams

*1) Data with a changing IR and fixed concepts:* The goal of the experiment is to answer the research question 2, i.e., how do they perform when the data stream has a dynamically changing imbalance ratio? The effectiveness of UEvoAutoML and OEvoAutoML are studied in data streams with a dynamic IR and fixed posterior probabilities.



(a) SINE

(b) SINE$_g$

(c) SEA

(d) SEA$_g$

Fig. 1. Prequential G-mean of the three AutoML approaches on data streams with a dynamic IR.

Each data stream is fixed to have 100000 instances like the previous section. We vary the IR at the time step 50000. From time step 50001, we change the IR at a different speed (an abrupt or gradual change) and severity (a severe or mild change). An abrupt change means that the new distribution entirely replace the old one from time step 500001, and a gradual change indicates that the change lasts for 30000 time steps. The SINE data stream involves a severe IR change, and the SEA data stream involves a mild IR change, as shown below:

For SINE:

$$\begin{aligned} P(y=1) &= 0.01 \rightarrow 0.99 \\ P(y=0) &= 0.99 \rightarrow 0.01 \end{aligned} \quad (3)$$

For SEA:

$$\begin{aligned} P(y=1) &= 0.50 \rightarrow 0.01 \\ P(y=0) &= 0.50 \rightarrow 0.99 \end{aligned} \quad (4)$$

Thus, four data streams are generated in total, i.e. SINE–severe abrupt change, SINE$_g$–severe gradual change, SEA–mild abrupt change, and SEA$_g$–mild gradual change. These settings allow us to analyse if the proposed methods can deal with dynamic IR changing and quickly adjust the adaptive resampling when they occur. To understand the dynamic change, the learning curve of prequential G-mean is produced and compared in Fig. 1. Each point on the curve represents the mean value across 30 runs. G-mean in the curve is reset to 0 when the change begins and ends in order to observe the impact of the change.

To quantitatively analyze how these changes affect the overall performance of three frameworks, the means and standard

TABLE III
MEANS AND STANDARD DEVIATIONS OF G-MEAN AND RECALL OF THE THREE AUTOML APPROACHES AFTER THE CHANGE ON DATA STREAMS WITH A DYNAMIC IR.

| Data sets | Frameworks | G-mean | Recall_1 | Recall_0 |
|---|---|---|---|---|
| SINE | UEvoAutoML | 0.9194±0.0301 | 0.8832±0.0501 | 0.9576±0.0099 |
| | OEvoAutoML | 0.7898±0.0025 | 0.9975±0.0001 | 0.6253±0.0040 |
| | EvoAutoML | 0.5089±0.0013 | 0.9990±0.0000 | 0.2592±0.0013 |
| $\text{SINE}_g$ | UEvoAutoML | 0.9356±0.0076 | 0.9102±0.0092 | 0.9617±0.0104 |
| | OEvoAutoML | 0.8079±0.0056 | 0.9986±0.0002 | 0.6537±0.0090 |
| | EvoAutoML | 0.5531±0.0019 | 0.9995±0.0000 | 0.3062±0.0022 |
| SEA | UEvoAutoML | 0.9632±0.0040 | 0.9891±0.0065 | 0.9381±0.0065 |
| | OEvoAutoML | 0.7642±0.0048 | 0.5864±0.0074 | 0.9959±0.0002 |
| | EvoAutoML | 0.9598±0.0001 | 0.9460±0.0002 | 0.9739±0.0000 |
| $\text{SEA}_g$ | UEvoAutoML | 0.9681±0.0043 | 0.9850±0.0072 | 0.9515±0.0048 |
| | OEvoAutoML | 0.7873±0.0073 | 0.6222±0.0116 | 0.9963±0.0000 |
| | EvoAutoML | 0.5276±0.0027 | 0.2788±0.0028 | 0.9983±0.0000 |

deviations of metrics on the new class status in data streams are further presented over all the time steps after the change completely ends, i.e., time step 50000-100000 for an abrupt change and time step 80000-100000 for a gradual change. The detail G-mean and Recall of the new status are shown in Table III.

In Fig. 1(a), UEvoAutoML has achieved the best performance for balancing the accuracy on the two classes before the change happens. After the abrupt change occurs(i.e. the class 1 from the minority class becoming the majority class), the G-mean rises in all of the AutoML frameworks due to the appearance of class 1 more frequent in the data stream. The G-mean then decreases over time steps as class 0 becomes the minority class, the UEvoAutoML slightly declines as a result of its ability to quickly adjust the adaptive undersampling method on the right class(i.e. current majority class 1) and precisely identify the current minority class 0. For space limitations, Recall learning curve is not included in here. In Fig. 1(b), due to gradual change, the growth trend is obvious in the change process and in the new class status. The UEvoAutoML and OEvoAutoML are more rapid since the current class imbalance status is immediately evaluated via calculating the time decay class size ratio. They are able to faster tune the resampling's attention for the right class after the change. Besides, UEvoAutoML is more effective for coping with dynamic IR.

In Fig. 1(c), after the balance stream becomes imbalanced (the class 1 is minority class in the new status), the G-mean of UEvoAutoML still performs the best after this type of change due to its superior performance on identifying the minority class. However, OEvoAutoML declines more than EvoAutoML. This is because the pipelines in EvoAutoML are updated more times(see Section IV-B). It results in retaining more knowledge about class 1. Even if the class 1 becomes the minority class after this change, it still performs better than OEvoAutoML at the expense of sacrificing computation time. This worse performance of OEvoAutoML is alleviated for identifying the current minority class in the new status in the scenario with a mild gradual change in Fig. 1(d).

After the changes, the class 0 becomes the minority class in SINE and $\text{SINE}_g$ , and the class 1 becomes the minority class in SEA and $\text{SEA}_g$. In Table III, for G-mean and the minority class recall, the UEvoAutoML has achieved the best performance in all of cases since it quickly adapts to the changes and improves the performance on the minority class by using undersampling for the right class based on the accurate real-time class size ratio estimate. In the close performance in mild abrupt change in SEA, the UEvoAutoML outperforms the EvoAutoML significantly with p-value 0.0003 according to Wilcoxon Sign Rank test. The adaptive Undersamlping in UEvoAutoML is more effective than Oversampling in OEvoAutoML in terms of changing IR.

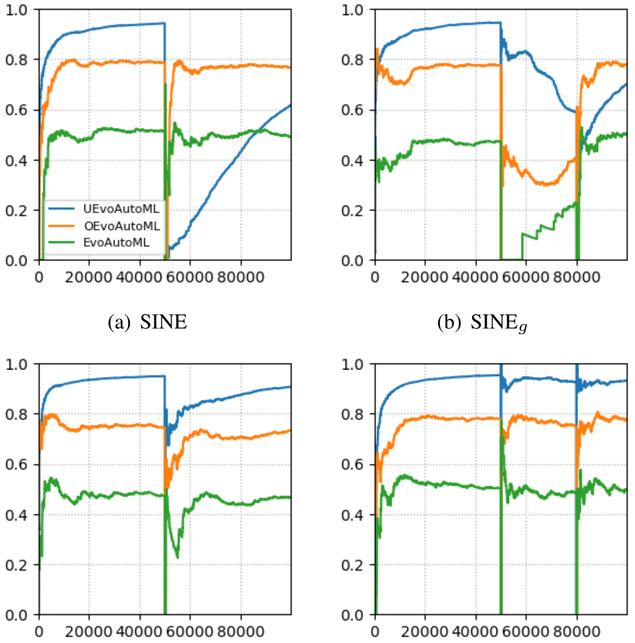
*2) Data with real concept drift and a fixed IR:* The aim of the experiment is designed to answer research question 3, i.e., how do they perform when both real concept drift and class imbalance occur simultaneously? We study the scenarios where both a real drift and class imbalance exist in data streams. Even though the UEvoAutoML and OEvoAutoML do not explicitly cope with real concept drift, the dynamic searching process(i.e, mutation of the best pipeline and removal of the worst pipeline) can passively respond to it through periodic pipeline optimisation.

The change follows the settings in experiment VI-B1. The class imbalance is fixed to 1% and the class 1 is the minority class. For a severe change, the SINE synthetic data are involved with concept swap, i.e. the concept that is reversed. A gradual change in $\text{SINE}_g$ occurs through probability choosing of examples from the old and the new concept. For a mild change, the threshold of old concept in SEA is set to $\theta = 7$, and the threshold of new concept status is $\theta = 9.5$. A gradual change in $\text{SEA}_g$ data distribution is generated by means of linearly moving the threshold. The SEA data stream has a smaller change severity degree than SINE since examples of the new concept in the SEA data set still include some previous concept examples after the threshold changes. Fig. 2 depicts the prequential G-mean throughout the time steps to compare the three AutoML approaches in these four scenarios.

In Fig. 2(a), before the change, UEvoAutoML and OEvoAutoML outperform EvoAutoML in terms of G-mean. After the concept abrupt drifts, all of the online AutoML suffer from the performance reduction since this severe drift is involved with the concept swap, which causes the rules learnt from old concept being not suitable to the current new concept. All of the online AutoML are able to recover thanks to the passive adaptation to the concept change. OEvoAutoML quickly adapts to this severe change and has the best performance in terms of trade-off between the two classes in the new data concept. The similar results can be seen in Fig. 2(b).

In Fig. 2(c) and Fig. 2(d), a similar decreasing trend can be observed. At the new concept status, we can see that UEvoAutoML has the best performance on G-mean. This is attributed to the fact that adaptive undersampling is more effective when the UEvoAutoML retains some knowledge about the minority class in the new concept in terms of the process of mild real concept change in the SEA and $\text{SEA}_g$.

Table IV shows the performance of three online AutoML approaches on new data concept with different types of real



(a) SINE

(b) SINE$_g$

(c) SEA

(d) SEA$_g$

Fig. 2. Prequential G-mean of the three AutoML approaches on data streams with real concept drift and a fixed IR.

TABLE IV
MEANS AND STANDARD DEVIATIONS OF G-MEAN AND RECALL OF THE THREE AUTOML APPROACHES AFTER THE CHANGE ON DATA STREAMS WITH REAL CONCEPT DRIFT AND A FIXED IR.

| Data sets | Frameworks | G-mean | Recall_1 | Recall_0 |
|---|---|---|---|---|
| SINE | UEvoAutoML | 0.6187±0.0283 | 0.5683±0.0305 | 0.6741±0.0375 |
| | OEvoAutoML | 0.7677±0.0039 | 0.5913±0.0061 | 0.9968±0.0002 |
| | EvoAutoML | 0.4906±0.0012 | 0.2409±0.0011 | 0.9990±0.0000 |
| SINE$_g$ | UEvoAutoML | 0.7018±0.0192 | 0.6758±0.0278 | 0.7294±0.0304 |
| | OEvoAutoML | 0.7764±0.0068 | 0.6040±0.0106 | 0.9981±0.0002 |
| | EvoAutoML | 0.4975±0.0047 | 0.2477±0.0047 | 0.9993±0.0000 |
| SEA | UEvoAutoML | 0.9077±0.0103 | 0.8502±0.0214 | 0.9693±0.0049 |
| | OEvoAutoML | 0.7325±0.0056 | 0.5378±0.0083 | 0.9979±0.0000 |
| | EvoAutoML | 0.4663±0.0029 | 0.2177±0.0027 | 0.9990±0.0000 |
| SEA$_g$ | UEvoAutoML | 0.9317±0.0071 | 0.8818±0.0151 | 0.9844±0.0031 |
| | OEvoAutoML | 0.7788±0.0044 | 0.6082±0.0069 | 0.9974±0.0002 |
| | EvoAutoML | 0.4948±0.0071 | 0.2452±0.0070 | 0.9988±0.0000 |

concept drifts. In terms of G-mean, the UEvoAutoML has a better performance in mild changes, and the OEvoAutoML has a better performance for dealing with severe changes. Additionally, UEvoAutoML and OEvoAutoML both perform better than EvoAutoML in the new concept. This demonstrates that adaptive resampling can assist EvoAutoML framework in addressing class imbalance issue in real concept drift. Comparing the results in Table III, it can be seen that the real concept drift with class imbalance affects the online AutoML more severely and results in a greater performance degradation.

### C. Real-world data streams

This experiment aims to answer research question 4. i.e., how do they perform on real-world imbalanced data streams? The previous experiments allow us to look into how the UEvoAutoML and OEvoAutoML to handle and adapt to imbalanced data streams in various manipulated scenarios. Real-world data streams can be more complex. Thus, studying how robust and adaptive in real world data streams is essential to verify their effectiveness. The prequential G-mean of real-world data streams is shown in Fig. 3. The means and standard deviations of the final step G-mean averaged over 30 runs are presented in Table V.

TABLE V
MEANS AND STANDARD DEVIATIONS OF G-MEAN OF THE THREE AUTOML APPROACHES ON THE REAL-WORLD DATA.

| Data sets | UEvoAutoML | OEvoAutoML | EvoAutoML |
|---|---|---|---|
| Frauds | 0.8269±0.1313 | 0.7403±0.0234 | 0.5942±0.0200 |
| Covtype34 | 0.8986±0.0019 | 0.8358±0.0028 | 0.6490±0.0017 |
| GMSC | 0.6150±0.0046 | 0.4300±0.0036 | 0.2700±0.0022 |
| Weather | 0.7262±0.0026 | 0.7195±0.0030 | 0.6759±0.0011 |
| Poker27 | 0.9201±0.0092 | 0.6187±0.0094 | 0.1662±0.0247 |

Some similar results are obtained compared with the synthetic data cases. UEvoAutomL performs the best in all of the real-world data streams, which implies the adaptive undersampling having a positive impact on practical applications. It also indicates that UEvoAutoML is more effective and necessary in terms of handling class imbalance data streams in real-world applications.

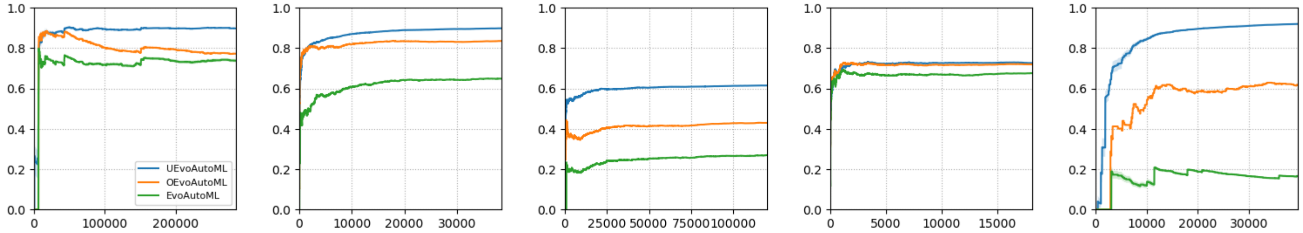
In Fig. 3(a), the Fraud plot of the G-mean performance of UEvoAutoML is relatively stable over time, but OEvoAutoML and EvoAutoML show the fluctuation. The reason is that the UEvoAutoML has the superior performance on the minority class, and the other two suffer from class imbalance issue, resulting in G-mean degradation.The similar trend is also shown in Fig. 3(e). The stable upward trend can be observed in Fig. 3(b)-(d).

In Table V, UEvoAutoML and OEvoAutoML outperform EvoAutoML by more than 20%, and 10%, respectively, on the Frauts data stream. In Covetype34, UEvoAutoML and OEvoAutoML outperform EvoAutoML by more than 18%. In GMSC, UEvoAutoML and OEvoAutoML both achieve a superior performance, outperforming EvoAutoML by more than 30%, and 15%, respectively. The similar advantage is also observed in Poker27. UEvoAutoML and OEvoAutoML slightly outperform EvoAutoML in the Weather dataset.

## VII. CONCLUSIONS

In this paper, we propose two adaptive resampling-based online AutoML frameworks(i.e. UEvoAutoML and OEvoAutoML)to tackle the online class imbalanced data streams. The class size ratio between the time decay size of each class is used to guide resampling methods in UEvoAutoML and OEvoAutoML at any given time step. They are evaluated in comparison with EvoAutoML by answering four research questions as listed in Section I.

In terms of the first research question, a data stream with static class imbalance ratio can be easily handled by UEvoAutoML. UEvoAutoML is more robust to extremely skewed class distributions. For the second research question, the adaptive undersampling technique in UEvoAutoML is



(a) Frauds (b) Covtype34 (c) GMSC (d) Weather (e) Poker27

Fig. 3. Prequential G-mean of the three AutoML approaches on the real-world data streams.

more effective to cope with a dynamically changing imbalance ratio. With respect to the third research question, the prediction performance on the real concept drift with class imbalance is able to be effectively enhanced by the adaptive undersampling in UEvoAutoML and adaptive oversampling in OEvoAutoML. By contrast, EvoAutoML can not perform well in the new concept and exhibits biases towards the majority class. For the final research question, UEvoAutoML achieves the best performance in all of the real-world data streams, and the OEvoAutoML's performance is the second best based on the observation on the G-mean. These indicate that our approaches are effective on practical applications.

Future work includes: (1)adding resampling techniques into the searching space of the AutoML framework, so that the framework can find the best resampling technique for any data stream automatically; (2) considering active detection of real concept drift and continuously evolving process when searching the online algorithms space.

## References

[1] Imbrea, Alexandru-Ionut. ”Automated Machine Learning Techniques for Data Streams.” arXiv:2106.07317, 2021.

[2] Wang, Shuo, Leandro L. Minku, and Xin Yao. ”Resampling-based ensemble methods for online class imbalance learning.” IEEE Transactions on Knowledge and Data Engineering, vol. 27, no.5, pp. 1356-1368, 2014.

[3] Wu, Qingyun, et al. ”ChaCha for Online AutoML.” International Conference on Machine Learning. PMLR, 2021.

[4] He, Haibo, and Edwardo A. Garcia. ”Learning from imbalanced data.” IEEE Transactions on knowledge and data engineering, vol. 21, no.9, pp. 1263-12848, 2009.

[5] Malialis, Kleanthis, Christos G. Panayiotou, and Marios M. Polycarpou. ”Online learning with adaptive rebalancing in nonstationary environments.” IEEE Transactions on Neural Networks and Learning Systems,vol.32, no.10, pp. 4445-4459, 2020.

[6] Wang, Shuo, Leandro L. Minku, and Xin Yao. ”A systematic study of online class imbalance learning with concept drift.” IEEE transactions on neural networks and learning systems,vol.29, no.10, pp.4802-4821, 2018.

[7] Sun, Yu, et al. ”Online ensemble learning of data streams with gradually evolved classes.” IEEE Transactions on Knowledge and Data Engineering, vol.28, no.6, pp.1532-1545, 2016.

[8] Meseguer, Jordi, Vicenç Puig, and Teresa Escobet. ”Fault diagnosis using a timed discrete-event approach based on interval observers: Application to sewer networks.” IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol.40, no.5, pp.900-916, 2010.

[9] Sousa, Maria Rocha, João Gama, and Elísio Brandão. ”A new dynamic modeling framework for credit risk assessment.” Expert Systems with Applications,vol.45, pp.341-351, 2016.

[10] Kulbach, Cedric, et al. ”Evolution-Based Online Automated Machine Learning.” Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, Cham, 2022.

[11] Madrid, Jorge G., et al. ”Towards AutoML in the presence of Drift: first results.” arXiv preprint arXiv:1907.10772, 2019.

[12] Celik, Bilge, and Joaquin Vanschoren. ”Adaptation strategies for automated machine learning on evolving data.” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.43, no.9, pp.3067-3078, 2021.

[13] Celik, Bilge, Prabhant Singh, and Joaquin Vanschoren. ”Online AutoML: An adaptive AutoML framework for online learning.” unpublished.

[14] Veloso, Bruno, et al. ”Hyperparameter self-tuning for data streams.” Information Fusion, vol.76, pp.75-86, 2021.

[15] Wu, Qingyun, et al. ”ChaCha for Online AutoML.” International Conference on Machine Learning. PMLR, 2021.

[16] Singh, Prabhant, and Joaquin Vanschoren. ”Automated Imbalanced Learning.” unpublished.

[17] Truong, Anh, et al. ”Towards automated machine learning: Evaluation and comparison of AutoML approaches and tools.” IEEE 31st international conference on tools with artificial intelligence (ICTAI). IEEE, 2019.

[18] Wang, Ke, Qingwen Xue, and Jian John Lu. ”Risky driver recognition with class imbalance data and automated machine learning framework.” International journal of environmental research and public health, 2021.

[19] Yang, Fenglong, and Quan Zou. ”mAML: an automated machine learning pipeline with a microbiome repository for human disease classification.” , 2020.

[20] Nguyen, Duc Anh, et al. ”Improved automated cash optimization with tree parzen estimators for class imbalance problems.” IEEE 8th international conference on data science and advanced analytics (DSAA), 2021.

[21] Bifet, Albert, Geoff Holmes, and Bernhard Pfahringer. ”Leveraging bagging for evolving data streams.” Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part I 21. Springer Berlin Heidelberg, 2010.

[22] Street, W. Nick, and YongSeog Kim. ”A streaming ensemble algorithm (SEA) for large-scale classification.” Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. 2001.

[23] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, “Learning with drift detection,” in Proc. Brazilian Symp. Artif. Intell. Berlin, Germany: Springer, 2004, pp. 286–295.

[24] Oza, Nikunj C., and Stuart J. Russell. ”Online bagging and boosting.” International Workshop on Artificial Intelligence and Statistics. PMLR, 2001.

[25] Dal Pozzolo, Andrea, et al. ”Calibrating probability with undersampling for unbalanced classification.” IEEE symposium series on computational intelligence, 2015.

[26] Blackard, Jock A., and Denis J. Dean. ”Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables.” Computers and electronics in agriculture, vol.24, no.3, pp.131-151, 1999.

[27] Ditzler, Gregory, and Robi Polikar. ”Incremental learning of concept drift from streaming imbalanced data.” IEEE transactions on knowledge and data engineering, vol.25, no.10, pp.2283-2301, 2012.

[28] Gomes, Heitor M., et al. ”Adaptive random forests for evolving data stream classification.” Machine Learning, vol.106, pp.1469-1495, 2017.

[29] Zhang, Hang, et al. ”Resample-based ensemble framework for drifting imbalanced data streams.” IEEE Access, vol.7, pp.65103-65115, 2019.