A Concept Drift Adaption Framework for Online Game Chargeback Detection

Yu-Chih Wei[1], Ching-Huang Lin[1], Yan-Ling Ou[1], Wei-Chen Wu[2]*

[1] Department of Information and Finance Management, National Taipei University of Technology, Taiwan
[2] Department of Finance, National Taipei University of Business, Taiwan
vickrey@mail.ntut.edu.tw, gbox@ntut.edu.tw, t109ab8016@ntut.org.tw, weichen@ntub.edu.tw

Abstract

The burgeoning online gaming market, driven by the rapid advancement of internet infrastructure and hardware performance, has increasingly become a focal point for criminal activity. Service providers in this industry are particularly vulnerable, suffering significant financial losses due to malicious chargebacks. Current countermeasures remain largely reactive, as providers typically block compromised game accounts only after an attack has occurred. Although prior studies have attempted to leverage machine learning techniques to detect fraudulent chargebacks, perpetrators often employ evasion strategies to bypass detection, thereby exacerbating the problem of concept drift in game records. To address this issue, we propose enhancing the detection of malicious behavior through behavioral analysis in online gaming. This study applies machine learning models to detect malicious chargebacks in online games, focusing not only on improving detection capabilities but also on introducing mechanisms for identifying and mitigating concept drift. We propose an adaptive learning model specifically designed to handle concept drift in the context of top-up fraud in online gaming, aiming to proactively prevent malicious chargebacks before financial losses occur. The final experimental results demonstrate that, by employing incremental learning methods after detecting concept drift, the LSTM-Seq2Seq model with an attention mechanism achieved the best MCC performance. Through the proposed incremental learning model, online gaming service providers can effectively enhance their detection capabilities and further reduce operational losses.

Keywords: Concept drift, Machine learning, Online game

1 Introduction

In its 2024 Global Games Market Report, Newzoo forecasted the global game market would expand from $179 billion in 2020 to $213 billion by 2027, reflecting a compound annual growth rate of 3.1%. Additionally, the number of global paying gamers is expected to reach 1.5 billion in 2024 and will increase to 1.67 billion by 2027. The gaming industry can be broadly categorized into developers, publishers (or operators), and distributors, each occupying distinct roles within the value chain. Developers focus on the creation and production of games, while operators manage the acquisition of publishing rights, marketing, daily operations, and after-sales services, generating revenue from distribution fees and player transactions. Distributors, such as the Apple Store and Google Play, serve as platforms that sell these products directly to consumers.

In Taiwan, the predominant business model within the gaming industry emphasizes operational activities, with in-house game development playing a supplementary role. To comply with regulatory standards and protect consumer rights, game companies have implemented chargeback mechanisms that allow players to rectify erroneous purchases. However, these systems have increasingly become targets of abuse by individuals seeking financial gain through fraudulent means. Several Korean game companies have reported instances of players exploiting these refund mechanisms, prompting the need for operators to scrutinise payment data and game logs to identify and block users who engage in recurrent malicious refund activities.

Global chargeback value forecasted to rise from $33.79 billion in 2025 to $41.69 billion by 2028 [1]. In China, there are professional chargeback firms that specialize in arranging chargebacks for consumers. They charge 55% of chargeback amounts as their fees [2]. The Korea Times [3] reports that many Korean gamers profit from the loopholes in the chargeback mechanisms. A pertinent example is the digital game platform STEAM, which allows game developers to distribute their games to users for purchase and in-game item transactions. According to STEAM's refund policy, players are entitled to a refund if they have played for less than two hours within 14 days of purchase, provided that the game has not been consumed, modified, or transferred. However, this system has been subject to misuse, with malicious chargeback tutorials circulating online that instruct users on how to circumvent system checks and play online games without paying. This analysis illustrates that while developers are responsible for game production, the operational aspects—including the distribution of in-game content and the processing of player transactions—fall under the remit of game operators. As a result, in cases of fraudulent chargeback activities, operators bear the financial burden of refunding

*Corresponding Author: Ching-Huang Lin; Email: gbox@ntut.edu.tw
DOI: https://doi.org/10.70003/160792642026072704012

payments and reclaiming distributed items.

Malicious chargebacks are a frequent issue on platforms such as Google Play and the App Store, often arising from circumstances such as the purchase of incorrect products, erroneous top-ups of gaming credits, or accidental transactions made by children. Beyond these common occurrences, unauthorized chargebacks are particularly prevalent on Google Play, where users exploit the chargeback system to counteract payment reversals initiated by family members or friends, either mistakenly or deliberately. Unfortunately, online gaming service providers are largely powerless in influencing the chargeback process, rendering them unable to prevent ongoing financial losses. Although the chargeback system was originally designed to be a consumer-friendly safeguard, it has become a target for fraudulent exploitation. Criminals now offer charge tutorials that guide gamers through the process of making a chargeback, taking commissions of 20-50% of the chargeback amount. This emergent grey market poses a serious threat to gaming service providers, as the systematic abuse of chargebacks leads to significant financial harm. The aforementioned user behaviors have affected the online gaming ecosystem and caused significant impacts on the business operations of online gaming companies. To combat fraudsters, online gaming service providers establish risk management systems in accordance with guidelines established by experts. Unfortunately, these rules cannot detect new fraudulent activities in real time, which leads to many loopholes in the risk management systems [4]. For now, the majority of gaming companies can only passively react to malicious chargeback behaviors, i.e., blocking the users' accounts after malicious chargebacks have been made. Malicious chargebacks continue to cause huge losses to online gaming service providers.

To address this issue, this study will employ machine learning techniques to detect fraudulent chargeback activities in online games, utilising real game logs provided by the gaming provider for research purposes. However, this data presents two primary challenges: data imbalance, as normal transactions significantly outnumber abnormal ones in real-world scenarios, and concept drift, which occurs as user behavior, including deposits and spending patterns, evolves over time, resulting in shifts in the underlying data distribution. In light of these challenges, the research will focus on developing sampling techniques to manage imbalanced datasets and constructing incremental learning models to accommodate concept drift. The aim is to mitigate the degradation of model performance caused by dynamic transaction data. The study will train models using two distinct types of game data, with the objective of enhancing the general detection mechanism for evaluating malicious refund behaviors. Ultimately, the goal is to improve the identification of players engaging in fraudulent chargeback activities and minimise the financial losses of game developers.

2 Related Work

2.1 Concept Drift in Online Games

Concept drift refers to the phenomenon where the statistical properties of the target variable, which a model seeks to predict, shift over time in unpredictable ways [5]. In the context of online games, large volumes of in-game logs are generated based on gamer actions and purchases. In fraud detection, concept drift refers to the phenomenon where, as the system continuously imports data and conducts detection over an extended period, the behaviors of both legitimate users and fraudsters evolve over time [6]. For instance, in the case of credit card transactions, concept drift in fraud detection occurs as a cardholder's transaction behavior changes due to external factors. Changes in the cardholder's consumption patterns, income levels, and lifestyle can lead to adjustments in the transaction amount and frequency.

As illustrated, when concept drift occurs, relying on older data to retrain the detection model can be problematic. With the continuous influx of new data, the model may be misled by outdated information, resulting in erroneous predictions [7]. Concept drift is also referred to by alternative terms such as concept shift [5, 8-9] or dataset shift [10]. Concept drift can be classified into four categories: sudden drift, gradual drift, incremental drift, and recurring concepts, as depicted in Figure 1. Sudden drift occurs when a new concept emerges unexpectedly. Gradual drift occurs when a new concept replaces the old one progressively over time. Incremental drift occurs when an old concept gradually transitions to a new concept over a period of time. Recurring concepts refer to the reappearance of an old concept after some time.

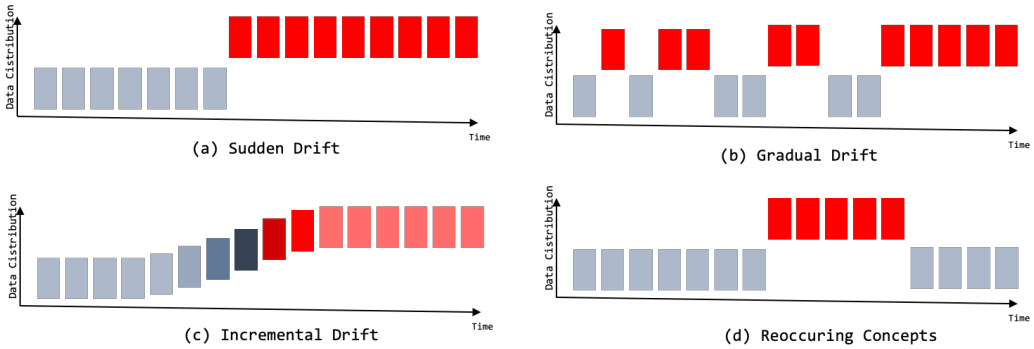

Figure 1. The types of concept drift can be classified as sudden drift, gradual drift, incremental drift, and recurring concepts

2.2 Concept Drift Detection

In [11], the author provides a comprehensive framework for studying concept drift, categorizing prior research into three main areas: concept drift detection, understanding, and adaptation. Concept drift detection can be achieved through various methods, including error rate-based detection, data distribution-based detection, and multiple hypothesis validation. Error rate-based detection triggers drift detection when a significant increase or decrease in the classification error rate is observed. Data distribution-based detection involves measuring differences in data distributions to detect drift, which can account for both temporal drift and conceptual differences within a dataset. Typically, two sliding windows are employed to track these changes over time. Multiple hypothesis validation detection methods are further divided into parallel and hierarchical approaches. However, this study focuses solely on error rate-based detection methods.

The most well-known error rate-based detection method is the Drift Detection Method (DDM) [12], which is based on the probably approximately correct learning model. DDM assumes that the classifier's error rate should either decrease or remain stable as the number of instances increases; any significant deviation from this trend indicates drift. Early Drift Detection Method (EDDM) [13], is particularly effective for detecting both gradual and abrupt changes. EDDM tracks the distance between classification errors rather than just the number of errors. It assumes that drift may occur when this distance starts to shrink, signaling potential changes in the underlying data. Hoeffding Drift Detection Method (HDDM) [14], which utilises Hoeffding's inequality, is divided into two variants of tests. One of the methods for comparing moving averages to detect abrupt drifts is another that employs the Exponentially Weighted Moving Average (EWMA) forgetting mechanism, which gives more weight to recent data, thereby enhancing its ability to detect drift in gradually changing environments. Hoeffding's inequality is applied to set an upper bound on the difference between the means, thus providing a threshold for drift detection.

In addition to these methods, sub-window and main-window approaches are also used for detecting changes in error rates. ADaptive WINdowing (ADWIN) [15] operates by using a window to adjust its size dynamically based on changes in the data distribution, which is divided into two sub-windows, and statistical comparisons between them are used to detect drift. When changes are identified, the main window shrinks; when no drift is detected, it expands. Furthermore, ADWIN uses Hoeffding Bound to manage the detection of these changes. Kolmogorov-Smirnov WINdowing (KSWIN) [16], similar to ADWIN, uses the Kolmogorov-Smirnov (KS) test to detect data distribution changes without assuming any specific underlying distribution. KSWIN maintains a fixed window size and compares the cumulative distributions of the current and previous windows. If the KS test rejects the null hypothesis that both windows have the same distribution, it signals the presence of drift.

2.3 Concept Drift Adaption

Gama et al. [17] conceptualised the ability to adapt to concept drift as an extension of incremental learning systems, allowing them to accommodate evolving data over time. The authors suggested that predictive models should detect concept drift swiftly and adjust accordingly, distinguish between drift and random noise, demonstrate adaptability to changing data while maintaining robustness against noise, process incoming data in a timeframe shorter than the interval between new data arrivals, and retain a fixed number of memories. Several strategies for managing concept drift in incremental learning environments have been proposed, including instance selection or window-based methods, weight-based methods, and classifier ensembles. Sliding window methods are commonly employed in online learning scenarios, where they preserve a subset of the most recent data and update the model with this retained data alongside newly arrived instances. Other approaches integrate concept drift detection directly into the learning algorithm. In cases where drift is not detected, the model continues to learn from new data; however, if drift is detected, the current model—whether it consists of a single learner or an ensemble—is discarded, and a new model is trained from scratch without retaining the knowledge from the previous model [18-19]. In [20], a TS-DM has been proposed, which is a concept drift adaptation method designed for data stream learning. It focuses on handling non-stationary environments where the data distribution changes over time. The core idea of TS-DM is to divide incoming data streams into time-based segments.

The comparison of concept drift detection and concept drift adaptation methods is presented in Table 1.

Table 1. The comparison of conception drift detection and adaptation method

Method	Category	Basic Idea	Limitations
DDM [12]	Detection	Error Rate-based	Less effective for gradual drift; sensitive to noise
EDDM [13]	Detection	Error Rate-based	Higher memory use; slower reaction to sudden drift
HDDM [14]	Detection	Error Rate-based	Requires careful parameter tuning; complexity increases
ADWIN [15]	Detection	Window-based, Distribution	Computationally heavier; may over-shrink with noise
KSWIN [16]	Detection	Window-based, Distribution	A fixed window size may miss long-term drift
DTEL [18]	Adaptation	Incremental learning / Transfer learning	Computationally costly
TS-DM [20]	Adaptation	Time-based segments	Segment Size Sensitivity;

In this study, we will first examine the foundational principles of incremental learning and then explore methods that prioritise the preservation of historical models, leveraging them for incremental learning. The subsequent sections will discuss relevant literature addressing these approaches.

3 Research Methodology

3.1 Concept Drift Adaption Framework

In this paper, we propose a concept drift adaptation learning model designed to detect chargeback fraud in online games as shown in Figure 2. The model is structured into three primary stages: pre-processing, basic classifier construction, and the incremental learning stage. The architecture of the proposed model is illustrated in the as follows:

1. **Pre-processing Stage:** In this initial phase, the key tasks include feature construction, feature selection, and the handling of imbalanced datasets. This ensures that the data is prepared for effective model training by addressing issues like irrelevant features and skewed class distributions.
2. **Basic Classifier Construction Stage:** During this stage, machine learning and deep learning models are built. The trained models are evaluated through cross-validation to ensure robustness, and the test set is employed for the final performance verification. This stage focuses on building an initial model that can detect chargeback fraud under normal conditions.
3. **Incremental Learning Stage:** This stage is responsible for detecting and adapting to concept drift. Based on the results of the drift detection process, three possible outcomes are identified: no drift, warning zone, or drift occurred.

The objective of this study is to predict whether a top-up transaction will result in a malicious chargeback by analyzing the top-up behaviors of game users. The framework proposed in this study aims to predict the likelihood of a malicious chargeback at the time of a top-up. This would enable game providers to be alerted early and pay closer attention to the activities and spending patterns of users who top up, thereby providing an opportunity to prevent potential malicious chargeback behavior in advance. However, perpetrators of malicious chargebacks may become aware of the detection mechanisms in place and deliberately disturb or alter their behaviors to evade detection. Therefore, to prevent the detection rate from significantly declining over time, this study proposes a concept drift adaptation framework to address potential behavioral changes.

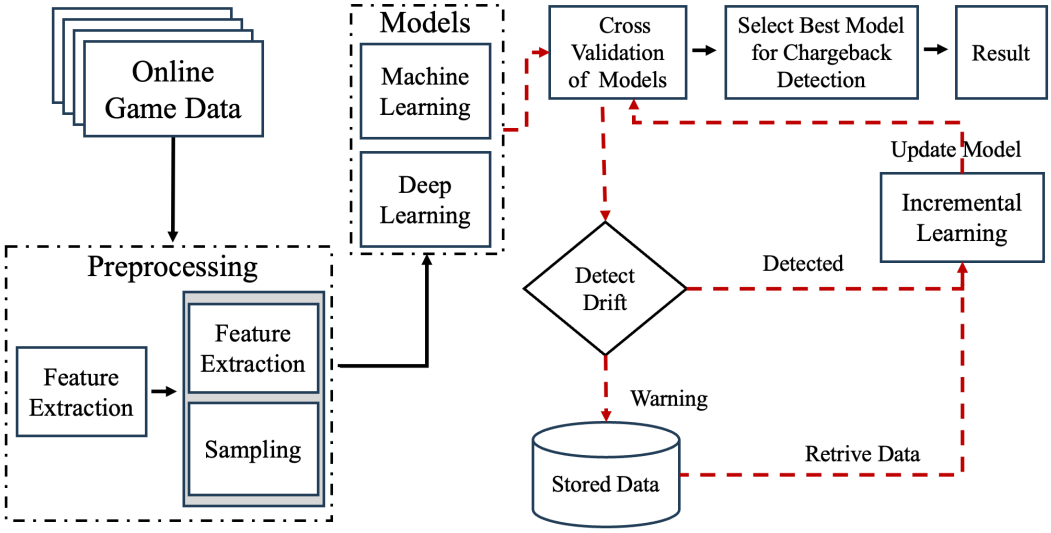

Figure 2. The framework of concept drift adaption learning model

To achieve this, the required feature values are extracted from real-world game log files for training the model. Since past data must be used to predict future outcomes, the time factor must be taken into consideration, with the goal of distinguishing feature data across different time intervals. This study assumes that each top-up transaction record contains n features and m time intervals, forming a two-dimensional matrix of size n x m to serve as the feature set for model training. To predict whether a top-up behavior leads to a malicious chargeback, the analysis requires the user's past behavioral data. Therefore, the EventCommand field in the game logs records various user behaviors during gameplay. The feature values are primarily sampled following the approach used in previous research [2] and can be broadly categorized into four types: transactions, general events, gameplay events, and network-related features.

When no drift is detected, the system loops back to the second stage, where the previously trained model is used for predictions. If the concept drift status enters the

warning zone, data collected during this period is buffered for incremental learning. Here, small windows and small batches are utilized for incremental learning, forming a backup classifier that is prepared until the drift returns to a stable state. When drift is confirmed as having occurred, larger batches are used for incremental learning, and the classifier constructed in the second stage is updated to reflect the new data distribution and patterns. This adaptive approach ensures that the model remains resilient against changes in player behavior and fraud tactics over time, maintaining high performance in chargeback fraud detection despite the presence of concept drift.

3.2 Basic Machine Learning for Chargeback Detection

The two machine learning models used in the research [21], which include k-nearest neighbor algorithm (k-NN) and random forests (RF), are continued to be used in this research for the subsequent comparison and evaluation of the effectiveness of deep learning.

One such deep learning method, recurrent neural networks (RNNs) [22], is an extension of feedforward neural networks (FNNs) [23-24]. While FNNs compute the output of a layer and pass it to the next layer in a single direction, with no feedback loops, RNNs differ in that they can return values stored in their hidden layers to themselves. This allows them to retain information over time, forming a recurrent structure. Two notable forms of RNNs include the Elman network, where hidden layer outputs are fed back to the hidden layer, and the Jordan network, where overall outputs are returned to the hidden layer. This structure enables RNNs to process input sequences of variable lengths, making them suitable for tasks involving temporal data. However, a limitation of RNNs is their difficulty in learning long-term dependencies due to the vanishing gradient problem. This issue arises because, as time progresses, the impact of earlier information in the sequence diminishes, weakening its influence on future decision-making. To address this problem, Hochreiter and Schmidhuber [25] developed a long short-term memory network (LSTM), which is a specialized type of RNN that effectively captures long-term dependencies by incorporating a memory unit that retains its state over time. Unlike standard RNNs, LSTMs use memory cells in their hidden layers, which are regulated by three gates: the input gate, which controls whether new information enters the memory; the forget gate, which decides whether to erase stored information; and the output gate, which determines whether the stored information is used in the final output [26]. This gate mechanism enables LSTMs to retain and utilize important information from earlier time points, mitigating the vanishing gradient problem and enhancing the network's ability to remember long-term dependencies.

Although LSTMs address the long-term memory issue inherent in RNNs, they come with a trade-off: execution time is relatively long due to their complex structure. To improve computational efficiency, Bahdanau et al. [27] introduced the gated recurrent unit (GRU) in 2014. GRUs accelerate execution time and reduce memory usage while maintaining the ability to manage long-term dependencies. Structurally similar to LSTMs, GRUs differ in that they use an update gate, which combines the functions of the input and forget gates in LSTMs, and integrate the unit state and hidden state into a single vector. With one less gate, GRUs are computationally simpler than LSTMs and offer advantages in terms of speed and resource consumption. Both LSTMs and GRUs address the limitations of standard RNNs; GRUs offer a more computationally efficient solution with fewer parameters, making them an attractive alternative for tasks requiring long-term memory retention with faster execution.

3.3 Incremental Learning

The traditional machine learning process involves first labelling each training dataset and inputting it into the model for initial training, which generates a classification model. However, when new data becomes available, the old training dataset must be merged with the new data, and the model must be retrained from scratch. As a result, when the dataset continues to grow, each retraining can be highly time-consuming. Additionally, when new categories emerge, the model cannot automatically recognise and update itself. Consequently, incremental learning is gaining more attention. Incremental learning allows for the continuous updating of the model with minimal processing time and storage space [28]. The following are the algorithmic definitions of incremental learning [29]:

(1) Acquiring new knowledge from new data.

(2) Training only on new data.

(3) Retaining old knowledge while learning new knowledge.

(4) Recognising and updating categories.

3.3.1 SAM-kNN

Losing et al. [30] utilized kNN combined with SAM, thus constructing two types of memory: STM, which contains current concept data, and LTM, which preserves past knowledge. Conflicting information from previous concepts is removed, and the remaining information is retained through compression methods. When incoming data is stored in STM, the cleaning phase ensures consistency between LTM and STM. Whenever the size of STM changes, knowledge that is no longer needed is transferred to LTM. When LTM runs out of space, it compresses its accumulated knowledge. Ultimately, both models are considered when making predictions.

The adaptation process of SAM-kNN consists of four steps. First, the adaptation of STM is introduced. STM contains a dynamic sliding window of the latest data, which grows as new data is input, ensuring it contains the most current concept data. Whenever a concept is updated, the previous version is deleted, thereby reducing its size. However, if no concept change is detected, the size is adjusted to minimise the interleaved test-train error of the remaining STM. The second step involves cleaning and transferring. LTM contains all data consistent with STM's concepts. Therefore, when STM is reduced, LTM data is not discarded at random, as it may contain important information for future predictions. In the case of recurring drifts, the retained knowledge avoids the need to relearn previous concepts. The third step is LTM compression.

Contrary to STM's first-in, first-out principle, once LTM reaches its threshold, instances do not disappear. Instead, clustering is used to condense the available information into sparse knowledge representations, allowing longer retention periods. This process is repeated each time the threshold is reached, completing the adaptive compression. Finally, model weighting is adapted accordingly.

3.3.2 ARF Model

The incremental learning method for Random Forest in this study employs the ARF model proposed by [31]. ARF includes a resampling method and an adaptive approach, which can handle various types of concept drift without requiring complex processing for different datasets. Random Forest [32] is a learning model widely used in batch classification and regression tasks. It grows multiple trees while decorrelating them through bootstrap aggregation [33] and random feature selection during node splitting, thereby preventing overfitting.

To handle constantly changing data streams, drift detection algorithms are typically combined with ensemble algorithms [34-35]. In ARF, when drift is detected, the trees are not immediately reset; instead, a more relaxed threshold is used to detect warnings and create sub-trees, which are trained along with the ensemble without affecting the predictions. If a tree that issues a warning signal detects drift, it is replaced by the corresponding sub-tree. In ARF, the voting weights are based on the tree's test-then-train accuracy. Suppose the tree has observed n instances since its last reset and has correctly classified k instances, the weight would be k/n. Assuming that drift and warning detection methods are accurate, this weighting reflects the tree's performance on the current concept. The advantage of using this weighting mechanism is that there is no need, as with other data streams, to define a window or a fading factor to estimate accuracy [36-37].

3.3.3 Online RNN-AD

For the incremental learning of LSTM and GRU, we use the Online RNN-AD method proposed by [38], which employs multi-layer RNNs to model time series. This model is capable of incremental training and adapting to changes in data distribution as new data arrives. After making predictions, the prediction error is used to update the model and detect anomalies and change points. Large prediction errors may indicate the occurrence of drift. The Online RNN-AD model uses the most recent prediction errors to calculate anomaly scores and perform incremental model updates. The RNN can be replaced with LSTM or GRU models as needed.

3.4 Evaluation Method

This study evaluates the model through F-Measure and Matthews correlation coefficient (MCC). In this research, false positive (FP) means that legitimate top-ups of game credits are classified as malicious chargebacks. False negative (FN) means that malicious chargebacks are classified as legitimate top-ups of game credits.

$F1$ is the harmonic average of the precision and recall rates. It also includes the precision and recall rates of a classifier model, so this score is used as one of the evaluation indicators. The formula of $F1$ is:

$$F1 = 2 * \frac{precision * recall}{precision + recall} \quad (1)$$

Chicco et al. believed that when evaluating binary models, *MCC* performs better than accuracy and $F1$ due to can produce a better reference and more authentic assessment than the other two [39]. The reasons being that the other two may produce misleading results on imbalanced datasets. *MCC* can only produce high scores when the binary model correctly predicts most of the true positives and most of the true negatives. Therefore, this research uses *MCC* as the main evaluation model indicators. It is noted that the range of MCC scores is [-1, +1], which is negative 1 and positive 1 in cases of complete error and perfect classification, respectively. The formula of MCC is:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}} \quad (2)$$

4 Experiments and Evaluation

4.1 Evaluation Environment and Data Set

As this study aims to evaluate the efficiency of model training, this section provides an introduction to the experimental environment and some of the packages used. Table 2 presents the environment details and versions of some machine learning packages used.

Table 2. The environment details and versions of some machine learning packages

Item	Specification
Workstation	HPE ProLiant ML350 Gen10
CPU	Intel Xeon Silver 4208 CPU 2.10GHz
GPU	GeForce RTX 2080 Ti*2
System OS	SUSE Linux Enterprise Server 15 SP1
Anaconda	4.9.2
Python	3.7.11
Scikit-Learn	0.24.1
Tensorflow	2.2.0

This study utilized data from a real-world game provided by a Taiwanese gaming provider. The data includes game logs, top-up records, refund records, blacklist records, unblacklist records, and game registration records, covering the period from 1 March 2019 to 30 April 2022. As shown in Table 3, during this time, the game recorded over 80 million game logs, with over 157,648 registered users. However, only slightly more than 16,052 users engaged in valid top-up transactions, with more than 165,103 valid top-up records. Thus, the data actually used and labelled for this study includes game

data from 16,052 valid registered users of the game and 171 blacklisted users who were "blocked from logging in" by the game company. This data will be used in subsequent experimental research. Since this study is a continuation of the research by [2], detailed introductions to the game data and the method for marking blacklisted users can be referenced in that paper, and will not be elaborated upon here.

Table 3. The game dataset provided by the game company

User	Top-ups	Chargeback	Blacklist	Players
Total	506,264	1,563	339	157,648
Valid	165,103			16,052
Blocked	3,597	1,111	235	171

In this study, 80% of the dataset was used as the training set, and 20% as the test set. The training data underwent feature selection and sampling methods to train and identify an optimal combination or method. However, each preprocessing technique was subjected to cross-validation, using a 5-fold cross-validation process. Based on the cross-validation results, the best preprocessing method was selected for each model. The chosen preprocessing method was then consistently applied to the 20% test set for evaluation.

4.2 Evaluation Results

This study will utilize SMOTE, Borderline-SMOTE, and a combined sampling method of SMOTE and Tomek link (hereafter referred to as combined sampling) to conduct experiments. These methods will be applied to the originally imbalanced dataset to sample it into a balanced dataset. The sampling results and the selection of methods will be explained in subsequent sections. The training results for all sampling techniques, including cases where no sampling methods were applied. By examining the cross-validation results, we assess the feasibility of the proposed method for detecting malicious refund behavior in online games. The the cross-validation results evaluate by F1 and MCC are shown in Table 4 and Table 5 respectively. From the results shown in Table 5, we can see that the highest overall MCC score was achieved by the models using only Borderline-SMOTE, and the GRU-Seq2Seq model with an attention mechanism that utilized both feature selection and Borderline-SMOTE, with a score of 0.98. However, the LSTM-Seq2Seq models with attention mechanisms also consistently scored above 0.97, achieving their best MCC score of 0.98 when using both Borderline-SMOTE. The result also supported by the evaluation result of Table 4. Based on the above, the best-performing method and model is the LSTM-Seq2Seq model with an attention mechanism, using Borderline-SMOTE sampling method. Thus, this model will be used for subsequent training and testing.

Table 4. F1 Scores for data after different sampling methods

	kNN	RF	LSTM-Seq	LSTM-Seq-att	GRU-Seq	GRU-Seq-att
SMOTE	0.971	0.965	0.972	0.972	0.965	0.974
Borderline-SMOTE	0.971	0.963	0.969	0.975	0.95	0.975
Combined sampling	0.971	0.959	0.964	0.973	0.969	0.97

Table 5. MCC scores for data after different sampling methods

	kNN	RF	LSTM-Seq	LSTM-Seq-att	GRU-Seq	GRU-Seq-att
SMOTE	0.971	0.972	0.978	0.978	0.972	0.979
Borderline-SMOTE	0.971	0.971	0.976	0.98	0.96	0.98
Combined sampling	0.971	0.967	0.972	0.978	0.975	0.976

4.3 Comparison of Basic and Incremental Learners

Subsequently, a performance comparison between the basic models and the incremental learners was conducted. To facilitate a more convenient comparison of the performance of different learners, in addition to the original game (named HS), we introduced another game (named GHT) to compare the differences in learning effectiveness between base learners and incremental learners. Table 6 shows the comparison results of MCC scores. In Figure 3, we can observe that, except for the LSTM-Seq2Seq incremental learner in the GHT game, which scored lower than the baseline model, all other models outperformed their baseline counterparts. However, in the HS game, the scores of the GRU-LSTM baseline model and the incremental learning model were identical, while the RF incremental learning model scored lower than the baseline. Therefore, it can be concluded that when concept drift occurs, incremental learners are better able to handle different concepts. In particular, the kNN incremental learner with the SAM mechanism outperforms the standard kNN in dealing with concept drift, achieving the greatest performance improvement in terms of score. Other baseline models beyond kNN also showed decent adaptability when facing concept drift, but after applying incremental learning, they became more effective at addressing concept drift.

Table 6. MCC Scores of basic models and incremental learners tested using Borderline-SMOTE

Model	HS		GHT	
	Basic	Incremental	Basic	Incremental
kNN	0.929	0.95	0.894	0.961
RF	0.975	0.959	0.979	0.986
LSTM-Seq	0.936	0.96	0.954	0.953
LSTM-Seq-att	0.972	0.975	0.986	0.993
GRU-Seq	0.94	0.94	0.93	0.952
GRU-Seq-att	0.968	0.97	0.964	0.979

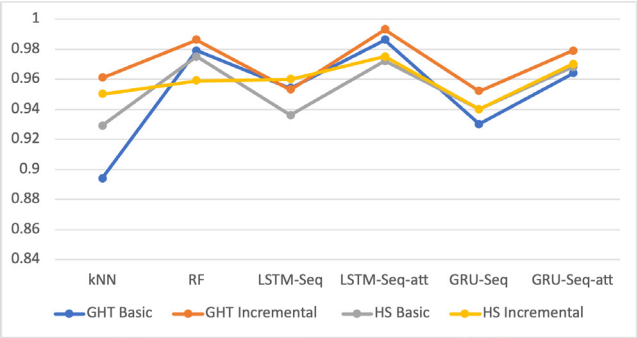

Figure 3. Comparison of basic models and incremental learners tested using Borderline-SMOTE

Furthermore, we provide a complete comparison of the time required from feature selection using the genetic algorithm to model training, testing, and prediction, with all time units measured in seconds. The reported training, testing, and prediction times are based on the use of the optimal preprocessing combination, which includes feature selection and the Borderline-SMOTE sampling method. It can be observed that when performing feature selection, the LSTM-Seq2Seq and GRU-Seq2Seq models with the attention mechanism required relatively long training times for both games. Notably, the LSTM consistently took longer to train than the GRU. In contrast, kNN and RF required considerably less time than the other four models; in particular, kNN required only about two minutes on average to complete feature selection training.

We then discuss the performance time for training, testing, and prediction of the baseline models and incremental learners individually. As shown in Figure 4, incremental learning consistently took longer than the baseline models. Among the models for both games, the LSTM-Seq2Seq with the attention mechanism still required the most time, averaging 57.157 seconds, followed by the GRU-Seq2Seq with the attention mechanism, which averaged 54.899 seconds.

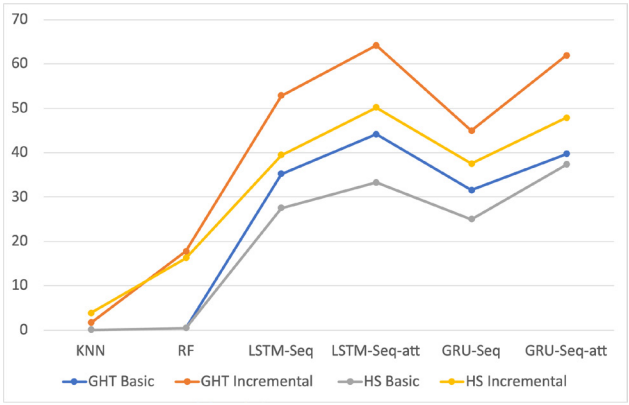

Figure 4. Comparison of execution time on basic models and incremental learners

5 Conclusion and Future Work

With rapid technological advancements and the impact of the pandemic, people's lifestyles have changed, and payment methods have also evolved. This shift has led to a gradual increase in digital fraud incidents across various industries, with the gaming industry being the most affected. When gaming companies detect a large number of fraud cases, they typically seek assistance from digital application distribution platforms as a first step. However, to protect player privacy, these platforms cannot immediately disclose information about which players initiated chargebacks or the reasons behind them. By the time gaming companies obtain the chargeback information, they are often unable to recover the services or items purchased by players. The time gap between platforms and gaming companies allows many malicious players to exploit the chargeback mechanism, resulting in substantial losses for game companies. This has become one of the major challenges currently faced by many in the gaming industry.

This study aims to predict user behavior based on historical data. However, real-world data is not static and may undergo changes in distribution over time or due to shifts in user spending behavior, a phenomenon known as concept drift. Concept drift can lead to a decline in the model's predictive performance, as models trained on outdated data may fail to accurately predict new data, resulting in errors. To detect whether concept drift has occurred in this study's data, we employed two concept drift detectors, ADWIN and KSWIN. The results indicated that both detectors consistently identified ongoing drift in the data. To prevent the potential decline in model performance due to concept drift, this study employed incremental updates to mitigate its impact on the model. When concept drift occurs, incremental learning allows the model to learn new concepts while retaining past knowledge, which can be useful if that knowledge reappears in the future. For incremental learning models, this study utilized SAM-kNN and ARF as incremental learning methods for kNN and RF, respectively, while the parameters of LSTM and GRU incremental learning algorithms were adjusted to enable incremental updates. From the results comparison, it was observed that, aside from RF showing a slightly lower MCC score than its basic learner, all other models outperformed their basic counterparts. The best-performing model was the LSTM-Seq2Seq with an attention mechanism, achieving an average MCC score of 0.982 across the two games, followed by the GRU-Seq2Seq model with an attention mechanism, which scored 0.970.

Currently, this study is still in the early stages of detecting malicious refunds in the gaming industry, and many areas remain for further research and refinement. One issue is the low utilization of features. Although a large amount of data has been collected, the features do not fully represent user behavior. Therefore, future research and experiments should include features that have not yet been incorporated to build a more comprehensive feature set. This would enable the identification of more malicious refund behaviors and facilitate clearer differentiation between normal user behavior and fraudulent activity. Another area for further study is concept drift. This study only considered changes in data distribution to determine

whether drift had occurred. However, there are different types of concept drift, each requiring specific detection methods. Future research could explore various detection methods to identify different types of drift. Additionally, when different types of drift are detected, model adaptation strategies should be adjusted accordingly to better mitigate the effects of drift.

In this study, incremental learning was employed to adapt to concept drift, while the learning process was still conducted in a batch manner. However, there are more recent approaches that utilize online learning for incremental learning, which could accelerate the learning process and reduce the risk of overfitting. Future research could explore the use of online learning methods for more efficient incremental model training.

Acknowledgement

This research was partially funded by the National Science and Technology Council (MOST 111-2221-E-027-137-, NSTC 114-2637-8-027-001-).

References

[1] Mastercard, *The chargeback window of opportunity: A global view of the 2025 chargeback trends and how to turn them into opportunities*, Mastercard Research, March, 2025.

[2] Y.-C. Wei, Y.-X. Lai, M.-E. Wu, An evaluation of deep learning models for chargeback Fraud detection in online games, *Cluster Computing,* Vol. 26, No. 2, pp. 927-943, April, 2023.
https://doi.org/10.1007/s10586-022-03674-4

[3] S.-W. Yoon, *Apple blamed for loophole in mobile app refund policy*, in The Korea Times, December, 2016.

[4] N. Lee, H. Yoon, D. Choi, Detecting Online Game Chargeback Fraud Based on Transaction Sequence Modeling Using Recurrent Neural Network, *18th International Workshop on Information Security Applications(WISA)*, Jeju Island, Korea, 2017, pp. 297-309.
https://doi.org/10.1007/978-3-319-93563-8_25

[5] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, *Machine Learning*, Vol. 23, No. 1, pp. 69-101, April , 1996.
https://doi.org/10.1007/BF00116900

[6] A. D. Pozzolo, G. Boracchi, O. Caelen, C. Alippi, G. Bontempi, Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 29, No. 8, pp. 3784-3797, August, 2018.
https://doi.org/10.1109/TNNLS.2017.2736643

[7] A. Abdallah, M. A. Maarof, A. Zainal, Fraud detection system: A survey, *Journal of Network and Computer Applications*, Vol. 68, pp. 90-113, June, 2016.
https://doi.org/10.1016/j.jnca.2016.04.007

[8] P. Vorburger, A. Bernstein, Entropy-based Concept Shift Detection, *Sixth International Conference on Data Mining (ICDM'06)*, Hong Kong, China, 2006, pp. 1113-1118.
https://doi.org/10.1109/ICDM.2006.66

[9] J. Coble, D. J. Cook, Real-Time Learning when Concepts Shift, *Thirteenth International Florida Artificial Intelligence Research Society Conference*, Orlando, Florida, USA, 2000, pp. 1-5.
https://cdn.aaai.org/FLAIRS/2000/FLAIRS00-037.pdf

[10] S. Amos, When Training and Test Sets Are Different: Characterizing Learning Transfer, in: J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofe, N. D. Lawrence (Eds.), *Dataset Shift in Machine Learning*, MIT Press, 2009.

[11] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, G. Zhang, Learning under concept drift: A review, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 31, No. 12, pp. 2346-2363, December, 2019.
https://doi.org/10.1109/TKDE.2018.2876857

[12] J. Gama, P. Medas, G. Castillo, P. Rodrigues, Learning with drift detection, *Brazilian symposium on artificial intelligence*, São Luís, Maranhão, Brazil, September, 2004, pp. 286-295.
https://doi.org/10.1007/978-3-540-28645-5_29

[13] M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, R. Morales-Bueno, Early drift detection method, *Fourth international workshop on knowledge discovery from data streams*, Philadelphia, USA, 2006, pp. 77-86.

[14] I. Frías-Blanco, J. del Campo-Ávila, G. Ramos-Jiménez, R. Morales-Bueno, A. Ortiz-Díaz, Y. Caballero-Mota, Online and non-parametric drift detection methods based on Hoeffding's bounds, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 27, No. 3, pp. 810-823, March, 2015.
https://doi.org/10.1109/TKDE.2014.2345382

[15] A. Bifet, R. Gavalda, Learning from time-changing data with adaptive windowing, *2007 SIAM international conference on data mining*, Minnesota, USA, 2007, pp. 443-448.
https://doi.org/10.1137/1.9781611972771.42

[16] C. Raab, M. Heusinger, F.-M. Schleif, Reactive soft prototype computing for concept drift streams, *Neurocomputing*, Vol. 416, pp. 340-351, November, 2020.
https://doi.org/10.1016/j.neucom.2019.11.111

[17] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. J. A. C. S. Bouchachia, A survey on concept drift adaptation, *ACM Computing Surveys (CSUR)*, Vol. 46, No. 4, pp. 1-37, April, 2014.
https://doi.org/10.1145/2523813

[18] Y. Sun, K. Tang, Z. Zhu, X. Yao, Concept drift adaptation by exploiting historical knowledge, *IEEE transactions on neural networks and learning systems*, Vol. 29, No. 10, pp. 4822-4832, October, 2018.
https://doi.org/10.1109/TNNLS.2017.2775225

[19] A. S. Iwashita, J. P. Papa, An overview on concept drift learning, *IEEE Access*, Vol. 7, pp. 1532-1547, December, 2018.
https://doi.org/10.1109/ACCESS.2018.2886026

[20] K. Wang, J. Lu, A. Liu, G. Zhang, TS-DM: A Time Segmentation-Based Data Stream Learning Method for Concept Drift Adaptation, *IEEE Transactions on Cybernetics*, Vol. 54, No. 10, pp. 6000-6011, October, 2024.
https://doi.org/10.1109/TCYB.2024.3429459

[21] Y.-X. Lai, T.-Y. Liao, Y.-S. Wu, Y.-C. Wei, Based on Genetic Algorithm for Feature Selection of Chargeback Fraud Detection in Online Games, 2020 *Taiwan Academic Network (TANET2020)*, Taipei, Taiwan, 2020.

[22] J. L. Elman, Finding structure in time, *Cognitive science*, Vol. 14, No. 2, pp. 179-211, April-June, 1990.
https://doi.org/10.1016/0364-0213(90)90002-E

[23] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, *arXiv preprint*, arXiv:1412.3555, December, 2014.
https://arxiv.org/abs/1412.3555

[24] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams, P. Beling, Deep learning detecting fraud in credit card transactions, *2018 Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA, USA, 2018, pp. 129-134.
https://doi.org/10.1109/SIEDS.2018.8374722

[25] S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural computation*, Vol. 9, No. 8, pp. 1735-1780, November, 1997.
https://doi.org/10.1162/neco.1997.9.8.1735

[26] A. Graves, Long short-term memory, *Supervised sequence labelling with recurrent neural networks*, Springer, Berlin, Heidelberg, 2012, pp. 37-45.
https://doi.org/10.1007/978-3-642-24797-2_4

[27] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, *arXiv preprint*, arXiv:1409.0473, September, 2014.
https://arxiv.org/abs/1409.0473

[28] Q. Yang, Y. Gu, and D. Wu, Survey of incremental learning, *2019 Chinese Control And Decision Conference (CCDC)*, Nanchang, China, 2019, pp. 399-404.

[29] R. Polikar, J. Byorick, S. Krause, A. Marino, M. Moreton, Learn++: A classifier independent incremental learning algorithm for supervised neural networks, *2002 International Joint Conference on Neural Network (IJCNN'02)*, Honolulu, Hawaii, USA, 2002, Vol. 2, pp. 1742-1747.
https://doi.org/10.1109/IJCNN.2002.1007781

[30] V. Losing, B. Hammer, H. Wersing, KNN classifier with self adjusting memory for heterogeneous concept drift, *2016 IEEE 16th international conference on data mining (ICDM)*, Barcelona, Spain, 2016, pp. 291-300.
https://doi.org/10.1109/ICDM.2016.0040

[31] Heitor M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfharinger, G. Holmes, T. Abdessalem, Adaptive random forests for evolving data stream classification, *Machine Learning*, Vol. 106, No. 9-10, pp. 1469-1495, October, 2017.
https://doi.org/10.1007/s10994-017-5642-8

[32] L. Breiman, Random forests, *Machine learning*, Vol. 45, No. 1, pp. 5-32, October, 2001.
https://doi.org/10.1023/A:1010933404324

[33] L. Breiman, Bagging predictors, *Machine learning*, Vol. 24, No. 2, pp. 123-140, August, 1996.
https://doi.org/10.1023/A:1018054314350

[34] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, R. Gavalda, New ensemble methods for evolving data streams, *15th ACM SIGKDD international conference on Knowledge discovery and data mining*, Paris, France, 2009, pp. 139-148.
https://doi.org/10.1145/1557019.1557041

[35] A. Bifet, G. Holmes, B. Pfahringer, Leveraging bagging for evolving data streams, *in Joint European conference on machine learning and knowledge discovery in databases*, Barcelona, Spain, 2010, pp. 135-150.
https://doi.org/10.1007/978-3-642-15880-3_15

[36] D. Brzeziński, J. Stefanowski, Accuracy updated ensemble for data streams with concept drift, *International conference on hybrid artificial intelligence systems*, Wroclaw, Poland, 2011, pp. 155-163.
https://doi.org/10.1007/978-3-642-21222-2_19

[37] D. Brzezinski, J. Stefanowski, Combining block-based and online methods in learning ensembles from concept drifting data streams, *Information Sciences*, Vol. 265, pp. 50-67, May, 2014.
https://doi.org/10.1016/j.ins.2013.12.011

[38] S. Saurav, P. Malhotra, V. TV, N. Gugulothu, L. Vig, P. Agarwal, G. Shroff, Online anomaly detection with concept drift adaptation using recurrent neural networks, *Acm india joint international conference on data science and management of data*, Goa, India, 2018, pp. 78-87.
https://doi.org/10.1145/3152494.3152501

[39] D. Chicco, G. Jurman, The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation, *BMC genomics*, Vol. 21, No. 1, pp. 1-13, January, 2020.
https://doi.org/10.1186/s12864-019-6413-7

Biographies

Yu-Chih Wei is an Associate Professor at National Taipei University of Technology, Taiwan. He received his Ph.D. in Information Management from National Central University, Taiwan, in 2013. His research interests include information security, AI system impact assessment, and supervisory technology (SupTech). He currently serves as a board director or supervisor in several professional associations, including CCISA, TAMI, TEHA, ARTA, and ACFD.

Ching-Huang Lin is an assistant professor at National Taipei University of Technology. He received his Ph.D. from the Department of Electrical Engineering at National Cheng Kung University in Taiwan. His research interests include medical information security, network security, and digital forensics.

Yan-Ling Ou graduated from National Taipei University of Technology, Taiwan, and is an engineer at Onward Security, a DEKRA company. Her work focuses on security assessment and verification of IoT devices.

Wei-Chen Wu is Associate Professor in the Department of Finance at the National Taipei University of Business. He received his Ph.D. degree in Information Management from National Central University. His current research interests include blockchain technology, fintech cybersecurity, network security.