International Journal of Innovative Computer Science and IT Research

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

Data Stream Mining for Real-Time Knowledge Discovery

Gillian Dobbie

Professor, School of Computer Science, University of Auckland, New Zealand

Abstract

The rapid expansion of connected devices, online platforms, enterprise applications, sensor infrastructures, and digital services has resulted in continuous generation of high-volume and rapidly changing data. Unlike traditional static datasets, data streams require analytical approaches capable of processing information incrementally while maintaining low latency and adapting to changes in underlying patterns. Data stream mining provides a computational framework for discovering useful knowledge from continuously arriving data without requiring the entire dataset to be stored before analysis. This paper examines data stream mining as an approach for real-time knowledge discovery, focusing on stream representation, incremental learning, concept drift, online classification, clustering, anomaly detection, and adaptive decision-making.

The study develops a conceptual analytical framework consisting of stream acquisition, preprocessing, feature extraction, online model learning, knowledge evaluation, and continuous adaptation. Particular attention is given to sliding-window techniques, landmark windows, fading-factor approaches, incremental classifiers, stream clustering, and drift-aware learning. The analysis indicates that real-time stream mining can support timely discovery of emerging patterns in cybersecurity, Internet of Things environments, financial transactions, healthcare monitoring, transportation systems, industrial automation, and online services.

However, continuous data processing introduces challenges related to velocity, volume, memory constraints, concept drift, noisy observations, class imbalance, model stability, and computational latency. The study further examines the importance of adaptive algorithms that can update models continuously while preserving previously learned knowledge. Two analytical tables compare major stream-mining approaches and their application areas. The findings suggest that combining incremental learning, adaptive windows, drift detection, distributed processing, and automated model evaluation can improve the effectiveness of real-time knowledge discovery. The paper concludes that data stream mining represents an important foundation for intelligent information systems that require continuous learning and rapid decision support.

Keywords: Data Stream Mining, Real-Time Analytics, Knowledge Discovery, Incremental Learning, Concept Drift, Stream Processing, Online Learning, Anomaly Detection, Data Analytics, Intelligent Systems.

1. Introduction

The increasing digitization of social, commercial, industrial, and technological environments has created unprecedented quantities of continuously generated data. Sensors, mobile devices, financial platforms, enterprise applications, social networks, cloud services, and Internet of Things systems produce information in the form of uninterrupted streams. Such data are characterized by high velocity, potentially large volume, changing distributions, and varying levels of noise. Conventional data-mining approaches generally assume that datasets are available as relatively stable collections that can be stored and processed repeatedly. This assumption becomes unsuitable when analytical decisions must be generated while data are continuously arriving.

Data stream mining addresses this limitation by providing techniques for analyzing observations incrementally rather than waiting for a complete dataset. In a streaming environment, each observation may need to be processed only once or a limited number of times because retaining every historical observation can be computationally expensive. Stream-mining algorithms therefore emphasize incremental computation, memory efficiency, processing speed, and adaptation. The objective is not simply to analyze large datasets but to identify useful information while the underlying stream is still active and relevant to decision-making.

Real-time knowledge discovery requires more than rapid data processing. The statistical properties of a stream may change over time, resulting in concept drift. For example, patterns of network attacks can change as attackers modify their strategies, customer preferences can evolve, and sensor behavior can shift because of environmental conditions. A model trained on historical observations may consequently become less accurate when the current data distribution differs from the original training distribution. Adaptive stream-mining methods attempt to recognize such changes and update the analytical model accordingly.

Different techniques have been developed to manage continuous data, including sliding windows, landmark windows, fading factors, online classifiers, incremental clustering, and drift-detection mechanisms. Ensemble approaches can also combine multiple models to improve predictive stability in changing environments. Online learning methods are particularly valuable because they can update model parameters as new observations arrive without repeatedly rebuilding the entire model. These characteristics make stream mining relevant to modern information systems where continuous monitoring and immediate response are essential.

The purpose of this paper is to examine the role of data stream mining in real-time knowledge discovery. The study focuses on the methodological foundations of streaming analytics, incremental learning mechanisms, concept-drift management, analytical applications, and implementation challenges. A conceptual framework is developed to demonstrate how continuously arriving data can be transformed into actionable knowledge through adaptive processing and continuous model evaluation.

2. Objectives

2.1 Primary Objective

The primary objective of this study is to examine how data stream mining techniques can support real-time knowledge discovery from continuously generated data. The research emphasizes incremental processing, adaptive learning, pattern detection, concept-drift management, and timely decision-making in information systems where data arrive continuously and analytical requirements change over time.

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

2.2 Evaluation of Stream-Mining Mechanisms

The first objective is to analyze techniques used for managing continuous data streams, including window-based processing, incremental learning, online classification, stream clustering, and anomaly detection. These approaches are designed to operate under memory and latency constraints while maintaining useful representations of recent and historically relevant observations.

A further objective is to determine how streaming algorithms balance computational efficiency with analytical quality. Since complete historical datasets may not be available during processing, stream-mining systems must determine which information should be retained, discarded, weighted, or updated. Effective knowledge discovery therefore depends on appropriate strategies for maintaining relevant information.

2.3 Examination of Adaptive Learning

The second objective is to investigate how stream-mining models respond to changes in data distributions. Concept drift can cause previously accurate models to become ineffective when relationships among variables change. Adaptive algorithms must therefore detect changes and modify their representations or decision boundaries.

The study also considers the importance of continuous model evaluation. Accuracy alone may not be sufficient for streaming environments because performance can vary over time. Monitoring prediction quality, latency, memory consumption, drift frequency, and model stability can provide a more comprehensive assessment of real-time analytical performance.

2.4 Identification of Practical Challenges

The final objective is to identify technical and operational challenges associated with real-time stream mining. These include high data velocity, limited memory, noisy data, class imbalance, evolving distributions, delayed labels, computational overhead, and scalability. Understanding these limitations can assist researchers and system developers in designing robust and adaptive stream-mining architectures.

3. Review of Literature

3.1 Foundations of Data Stream Mining

Data stream mining emerged as a response to the increasing need to analyze continuously generated information. Early research emphasized the differences between traditional database processing and streaming environments, particularly the requirement for incremental computation and limited memory usage. Stream-processing systems cannot generally assume that all observations are permanently available for repeated analysis.

Aggarwal et al. introduced a framework for data-stream clustering that emphasized compact representations and evolving data distributions. Their work demonstrated the importance of summarizing incoming information rather than retaining every individual observation.

3.2 Incremental Learning and Concept Drift

Incremental learning allows analytical models to update their knowledge as new observations become available. Unlike batch learning, where a model is retrained using a complete dataset, incremental

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

learning modifies an existing model using newly arriving information. This reduces repeated computational requirements and enables continuous operation.

Concept drift represents one of the most significant challenges in streaming analytics. Gama et al. examined concept-drift adaptation and highlighted the importance of distinguishing between stable and changing data distributions. Drift may be sudden, gradual, recurring, or incremental. Effective systems therefore require mechanisms capable of detecting different types of changes and adapting without unnecessarily discarding useful historical knowledge.

3.3 Online Classification and Ensemble Learning

Online classification has become an important component of real-time knowledge discovery. Algorithms such as Hoeffding Trees are designed to learn decision structures incrementally using statistical properties of streaming observations. The Hoeffding bound provides a mechanism for deciding when sufficient observations have been collected to select an attribute with high confidence.

Ensemble approaches provide another strategy for handling evolving data. Adaptive ensembles can maintain multiple models and adjust their contribution according to recent performance. Adaptive Random Forests, for example, combine online tree learners with mechanisms for dealing with evolving streams and concept drift. Such methods demonstrate how model diversity and continuous adaptation can improve streaming prediction.

4. Methodology

4.1 Research Design and Analytical Framework

This research adopts a conceptual and analytical design to examine the application of data stream mining for real-time knowledge discovery. The framework considers a continuous pipeline beginning with stream acquisition and followed by preprocessing, feature extraction, incremental learning, pattern evaluation, and model adaptation. The approach is designed for environments where data arrive continuously and decisions must be produced without waiting for complete historical datasets.

The conceptual framework represents the stream as a sequence of observations:

S={x1,x2,x3,…,xt,…}S=\{x_1,x_2,x_3,\ldots,x_t,\ldots\}

where xtx_t represents an observation arriving at time tt. Each observation is processed incrementally, and the analytical model is updated according to the current state of the stream. The methodology emphasizes memory efficiency, processing latency, model stability, and adaptation to changing distributions.

4.2 Data Acquisition and Stream Preprocessing

The first methodological stage involves acquiring continuous observations from appropriate digital sources. These may include IoT sensors, network monitoring systems, financial transaction platforms, web applications, enterprise systems, social platforms, or industrial equipment. Incoming observations are assigned timestamps and organized according to their analytical relevance. Data quality checks are performed to identify incomplete, duplicated, inconsistent, or anomalous observations before they enter the learning process.

Preprocessing is performed incrementally to avoid repeatedly scanning historical data. Numerical variables can be normalized using online statistics, while categorical variables can be encoded through appropriate streaming representations. Missing information is handled through lightweight imputation

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

strategies, and noisy observations may be filtered according to predefined rules. The preprocessing layer must operate with low latency because excessive preprocessing delays can undermine the real-time objective of the overall system.

Key methodological activities:

- Continuous stream acquisition
- Timestamp synchronization
- Duplicate detection
- Missing-value handling
- Incremental normalization
- Noise reduction
- Feature extraction
- Stream partitioning
- Real-time quality monitoring

4.3 Incremental Learning and Model Adaptation

Incremental learning represents the central analytical component of the proposed methodology. Instead of rebuilding a model from the entire dataset, the model receives new observations and updates its internal parameters. This reduces computational requirements and allows the system to maintain current knowledge. Suitable approaches include incremental decision trees, online linear models, adaptive ensembles, online neural models, and streaming clustering algorithms.

The learning process can incorporate window-based or weighted strategies to emphasize recent observations. A sliding window retains only a predefined number of recent observations, whereas a fading-factor approach assigns decreasing importance to older observations. These mechanisms help models respond to evolving patterns. Model adaptation can also be triggered by explicit drift detectors or by significant degradation in recent predictive performance.

4.4 Knowledge Evaluation and Real-Time Decision Support

The final methodological stage evaluates whether the continuously updated model is generating useful knowledge. Evaluation can include predictive accuracy, precision, recall, F1-score, detection rate, false-alarm rate, processing latency, memory consumption, and throughput. Unlike static machine-learning evaluation, streaming evaluation must consider temporal performance because model quality can change as the data distribution evolves.

The resulting knowledge can be used for real-time alerts, recommendations, anomaly identification, resource allocation, forecasting, and operational decisions. Continuous evaluation also provides feedback to the learning component. When performance deteriorates, the system can initiate model updating, adjust the window size, increase the influence of recent observations, or activate a drift-management procedure. This creates an adaptive feedback loop between incoming data, learning, evaluation, and decision support.

5. Analysis

5.1 Major Data Stream Mining Approaches

Data stream mining approaches differ according to how they represent historical information and respond to newly arriving observations. Batch-oriented methods may provide strong analytical

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

performance but are generally inefficient when continuous retraining is required. Streaming approaches instead maintain compact representations and update models incrementally.

Window-based methods provide an effective mechanism for controlling memory consumption. Sliding windows emphasize recent observations, while landmark windows maintain information from a defined starting point. Fading-factor methods provide a continuous compromise by gradually reducing the influence of older observations.

Table 1: Comparative Characteristics of Data Stream Mining Approaches

Approach	Processing Strategy	Main Advantage	Major Limitation	Suitable Environment
Sliding Window	Retains recent observations	Strong adaptation to recent changes	May discard useful historical information	Rapidly changing streams
Landmark Window	Processes data from a defined reference point	Simple historical organization	Less flexible under sudden drift	Periodic analytical systems
Fading Factor	Gradually reduces historical influence	Smooth adaptation	Requires parameter selection	Continuously evolving streams
Incremental Learning	Updates model with new observations	Low retraining cost	Model accumulation may occur	Real-time prediction
Online Clustering	Updates cluster structures continuously	Supports evolving groups	Sensitive to noise	Customer and sensor streams
Adaptive Ensemble	Combines multiple evolving models	Robustness to drift	Higher computational complexity	Complex dynamic environments

Interpretation: The comparison demonstrates that no single stream-mining strategy is optimal for every environment. Sliding windows are effective when recent information is more relevant than older observations, whereas fading-factor approaches provide smoother historical adaptation. Incremental learning is useful when models must be continuously updated, while adaptive ensembles can provide stronger robustness in complex environments. The selection of an approach should therefore consider stream velocity, memory availability, expected drift behavior, computational resources, and the consequences of delayed or incorrect decisions.

5.2 Real-Time Knowledge Discovery Tasks

Real-time knowledge discovery can involve multiple analytical tasks. Classification systems can identify incoming observations according to learned categories, while anomaly detection can identify unusual

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

events that differ from established patterns. Stream clustering can discover evolving groups without requiring predefined class labels.

Financial monitoring, cybersecurity, industrial systems, and IoT environments particularly benefit from real-time discovery because delayed analysis may reduce the value of detected information. For example, identifying a suspicious network behavior several hours after its occurrence may be considerably less useful than generating an alert while the activity is still developing.

Table 2: Applications of Data Stream Mining for Real-Time Knowledge Discovery

Application Domain	Streaming Data	Knowledge Discovery Task	Real-Time Outcome
Cybersecurity	Network and event logs	Anomaly and intrusion detection	Immediate security alerts
Finance	Transaction streams	Fraud and risk detection	Rapid transaction assessment
Internet of Things	Sensor observations	Pattern and fault detection	Automated monitoring
Healthcare	Patient monitoring data	Event and anomaly detection	Timely clinical alerts
Transportation	Traffic and vehicle data	Congestion prediction	Dynamic routing decisions
E-Commerce	Click and transaction streams	Recommendation and behavior analysis	Personalized interaction
Manufacturing	Machine telemetry	Predictive maintenance	Early fault identification

Interpretation: The table illustrates the broad applicability of data stream mining to information systems where continuous observations have operational significance. Cybersecurity and financial applications require particularly rapid responses because anomalies may represent active threats or fraudulent activities. IoT and manufacturing environments similarly benefit from continuous monitoring because early identification of abnormal behavior can support preventive action. The common requirement across these domains is the ability to transform incoming observations into actionable knowledge with minimal processing delay.

5.3 Concept Drift and Adaptive Knowledge Discovery

Concept drift occurs when the relationship between input variables and target outcomes changes over time. It may result from seasonal behavior, changes in user preferences, new attack strategies, market conditions, equipment degradation, or environmental changes. A static model can gradually become outdated when these changes are not incorporated.

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

Adaptive stream-mining systems address this problem through drift detection, window adjustment, model replacement, or ensemble reweighting. The effectiveness of adaptation depends on detecting meaningful changes without responding excessively to temporary noise. Therefore, a practical system should distinguish between natural short-term variation and genuine structural changes in the data-generating process.

6. Discussion

6.1 Importance of Continuous Learning

Continuous learning changes the role of machine learning from periodic model construction to ongoing knowledge adaptation. In real-time systems, information received today may be more relevant than historical observations collected months earlier. Incremental learning allows models to incorporate this information without repeatedly rebuilding the complete analytical pipeline.

However, continuous updating also creates risks. A model may adapt too strongly to temporary fluctuations and lose previously learned general patterns. Therefore, stream-mining systems must balance plasticity and stability. The appropriate balance depends on the velocity and volatility of the underlying data stream.

6.2 Role of Concept-Drift Management

Concept-drift management is essential for maintaining predictive reliability. A model that remains unchanged despite significant distributional changes may produce increasingly inaccurate results. Conversely, unnecessary model adaptation can reduce stability and increase computational overhead.

Effective drift management should therefore combine statistical monitoring with model-performance evaluation. Recent observations can be compared with historical summaries, while predictive performance can be monitored over time. When meaningful degradation is detected, the system can modify its learning strategy or replace outdated model components.

6.3 Real-Time Decision Support

The ultimate value of stream mining lies in its ability to convert continuous information into timely decisions. A real-time analytical system should not merely process data quickly; it should identify patterns that are sufficiently reliable to support action. This requires coordination between stream ingestion, analytical models, evaluation mechanisms, and decision-support interfaces.

In operational environments, latency can be as important as predictive accuracy. A highly accurate prediction generated too late may have limited practical value. Consequently, system developers should evaluate both analytical quality and response time when designing real-time knowledge-discovery architectures.

7. Challenges

One major challenge is the combination of high data velocity and limited computational resources. Continuous streams can produce observations faster than a system can process them, resulting in backlogs and increased latency. Memory constraints are also significant because retaining all historical observations may be impossible. Stream-mining systems must therefore use efficient summaries, sampling, windows, or compression mechanisms. Distributed processing can improve throughput, but it introduces synchronization and communication overhead that must be carefully managed.

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

Concept drift and noisy observations represent another major challenge. Real-world streams rarely remain statistically stable, and sudden changes may be difficult to distinguish from temporary fluctuations. A drift detector that reacts too aggressively can cause unnecessary model replacement, while a slow detector can allow outdated models to continue producing inaccurate results. Developing robust adaptive mechanisms that balance stability and responsiveness remains an important research problem.

A further challenge involves class imbalance, delayed labels, and limited ground-truth information. In cybersecurity and fraud detection, for example, abnormal events may represent only a very small proportion of all observations. Moreover, the true class of an observation may not be known immediately. These characteristics make conventional supervised evaluation difficult. Real-time systems therefore require evaluation mechanisms capable of operating with incomplete feedback and evolving class distributions.

8. Future Research

8.1 Distributed Stream Mining

Future research should investigate distributed architectures capable of processing extremely high-volume data streams across multiple computing nodes. Stream-processing frameworks can distribute ingestion, preprocessing, model training, and prediction tasks. However, research is needed to minimize communication overhead and maintain consistency among distributed model components. Adaptive workload allocation and resource-aware scheduling could further improve scalability when stream velocity changes dynamically.

8.2 Explainable Real-Time Analytics

Explainability is becoming increasingly important for automated real-time decision systems. Future stream-mining models should not only identify an anomaly or classify an event but also provide understandable reasons for the generated decision. Research could investigate incremental feature attribution, evolving rule extraction, interpretable ensembles, and explanation mechanisms that remain stable as models continuously change.

8.3 Integration with Edge and Intelligent Systems

The combination of stream mining with edge computing represents another promising research direction. Processing data closer to their source can reduce network latency and bandwidth requirements. Edge-based stream mining could support IoT monitoring, industrial automation, intelligent transportation, and distributed cybersecurity. Future research should investigate lightweight adaptive models, resource-aware learning, privacy-preserving stream analytics, and coordinated learning between edge devices and cloud infrastructures.

9. Conclusion

Data stream mining provides an important foundation for real-time knowledge discovery in modern information systems characterized by continuous data generation. Unlike conventional batch-oriented analytics, stream mining processes observations incrementally and seeks to extract meaningful information while the data are still arriving. This capability is particularly valuable in environments where decisions depend on current conditions rather than solely on historical records. Through

incremental classification, stream clustering, anomaly detection, window-based processing, and adaptive learning, stream-mining systems can transform continuously arriving information into actionable knowledge. The conceptual framework presented in this study emphasizes a continuous analytical pipeline consisting of stream acquisition, preprocessing, feature extraction, model learning, evaluation, and adaptation. This approach supports the development of intelligent systems capable of operating under strict latency and memory constraints.

The analysis further demonstrates that effective real-time knowledge discovery requires more than simply increasing processing speed. Streaming environments are inherently dynamic, and the statistical properties of incoming information can change over time. Concept drift therefore represents a fundamental consideration in stream-mining system design. Sliding windows, fading factors, incremental learning, drift detectors, and adaptive ensembles provide different mechanisms for responding to changing patterns. However, these techniques require careful parameter selection because excessive adaptation can make a model unstable, while insufficient adaptation can result in outdated knowledge. In addition, high data velocity, memory limitations, noisy observations, class imbalance, delayed labels, and distributed-processing requirements create substantial implementation challenges. Consequently, evaluation should consider predictive performance alongside latency, throughput, memory consumption, stability, and adaptation effectiveness.

Future development of data stream mining should focus on scalable, adaptive, explainable, and resource-efficient architectures. Distributed processing can support high-volume streams, while edge-based analytics can reduce latency by moving computation closer to data sources. Explainable stream-mining mechanisms can improve trust in automated decisions by identifying the factors responsible for detected patterns or anomalies. Privacy-preserving learning can further support applications involving sensitive information. The integration of these capabilities can enable more responsive intelligent information systems across cybersecurity, finance, healthcare, manufacturing, transportation, e-commerce, and Internet of Things environments.

References

1. Aggarwal, C. C. (2007). Data streams: Models and algorithms. *Springer*.
2. Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). A framework for clustering evolving data streams. *Proceedings of the 29th International Conference on Very Large Data Bases*, 81–92.
3. Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: Massive online analysis. *Journal of Machine Learning Research, 11*, 1601–1604.
4. Bifet, A., & Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. *Proceedings of the 2007 SIAM International Conference on Data Mining*, 443–448.
5. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., & Gavalda, R. (2009). New ensemble methods for evolving data streams. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 139–148.
6. Domingos, P., & Hulten, G. (2000). Mining high-speed data streams. *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 71–80.
7. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys, 46*(4), 1–37.
8. Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. *Advances in Artificial Intelligence—SBIA 2004*, 286–295.

9. Hulten, G., Spencer, L., & Domingos, P. (2001). Mining time-changing data streams. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 97–106.
10. Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association, 58*(301), 13–30.
11. Kifer, D., Ben-David, S., & Gehrke, J. (2004). Detecting change in data streams. *Proceedings of the Thirtieth International Conference on Very Large Data Bases*, 180–191.
12. Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. Wiley.
13. Masud, M. M., Gao, J., Khan, L., Han, J., & Thuraisingham, B. (2011). Classification and novel class detection of data streams in a dynamic environment. *Proceedings of the 2011 ACM Symposium on Applied Computing*, 1535–1540.
14. Oza, N. C., & Russell, S. (2001). Online bagging and boosting. *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*.
15. Read, J., Bifet, A., Holmes, G., & Pfahringer, B. (2012). Scalable and efficient multi-label classification for evolving data streams. *Machine Learning, 88*, 131–154.
16. Street, W. N., & Kim, Y. (2001). A streaming ensemble algorithm for large-scale classification. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 377–382.
17. Wang, H., Fan, W., Yu, P. S., & Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 226–235.
18. Wang, H., Fan, W., Yu, P. S., & Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 226–235.
19. Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., & Petitjean, F. (2016). Characterizing concept drift. *Data Mining and Knowledge Discovery, 30*, 964–994.
20. Widmer, G., & Kubat, M. (1996). Learning flexible concepts from streams of examples: FLORA2. *Proceedings of the European Conference on Machine Learning*, 74–85.