Journal of Universal Computer Science, vol. 31, no. 11 (2025), 1248-1271
submitted: 25/6/2024, accepted: 21/1/2025, appeared: 28/9/2025 CC BY 4.0

A Feature Evolution Aware Classification Framework for Streaming Data using Dynamic Autoencoder and Ensembled Learning

Jafseer KT
(Department of Computer Applications, Cochin University of Science and Technology, Cochin, India,
https://orcid.org/0000-0002-2127-6271, jafsi@cusat.ac.in)

Shailesh S
(Department of Computer Science, Cochin University of Science and Technology, Cochin, India,
https://orcid.org/0000-0002-5107-8997, shaileshsivan@cusat.ac.in)

Sreekumar A
(Department of Computer Applications, Cochin University of Science and Technology, Cochin, India,
askcusat@gmail.com)

Abstract: Recent advancements in data mining and knowledge discovery have created numerous research opportunities in streaming data analysis. One critical challenge is developing machine learning models that can efficiently handle changes in features and dynamic concepts, including concept drift, feature drift, and feature evolution. State-of-the-art techniques proposed to address these anomalies in data streams often assume that a constant set of features is available for processing. However, in real-time scenarios, the situation is quite different, as the set of features in a stream may vary over time due to factors such as the disappearance of existing features or the emergence of new ones. The proposed work focuses on handling dynamically evolving features by introducing a novel solution that leverages a Dynamic Autoencoder DAE and ensemble learning. Additionally, adaptive windowing and concept-preserving mechanisms improve the proposed architecture by retaining the concept information from previous data windows. The ensemble technique used in the proposed classification framework demonstrates promising performance in diverse datasets, achieving accuracies of 86%, 94%, and 95% in the Weather, Electricity and Forest Cover Type datasets, respectively. This innovative integration of deep learning and traditional methods effectively addresses various challenges in streaming data analysis.

Keywords: Streaming Data, Features Evolution, Data Stream Classification, Dynamic Autoencoder
Categories: H.3.1, H.3.2, H.3.3, H.3.7, H.5.1
DOI: 10.3897/jucs.130450

1 Introduction

Streaming data analysis is a transformative approach that revolutionizes data processing for businesses and organizations. Instead of waiting to collect the entire dataset, streaming data analysis allows real-time analysis as data is received. This enables quick decision-making based on the most up-to-date information available.

Continuous data processing forms the foundation of streaming data analysis. It ensures uninterrupted data analysis, facilitating the identification of patterns and swift decision-making. With streaming data analysis, businesses and organizations can detect data trends as they occur, empowering them to make informed decisions promptly.

To effectively utilize streaming data analysis, businesses and organizations should understand the types of data they receive. Structured data is more manageable and easier to analyze compared to unstructured data. Additionally, it is crucial to have a robust system capable of handling the influx of streaming data. Furthermore, implementing tools like data mining, machine learning, and predictive analytics aids in analyzing the received data. As soon as patterns are identified, businesses and organizations can take immediate action.

The streaming data analysis differs from a batch process in four aspects: processing samples one at a time, inspecting them just once, working with limited storage capacity, working within a time constraint, and being ready to predict at any time[Bifet et al., 2010]. Since data streams are continuously generated with high velocity and volume, a vast amount of data, or the 'infinite length' of the data stream, must be handled in real-time. Furthermore, the ability to effectively adapt to change is required. According to C.Fahy et.al[Fahy et al., 2022], one of the most difficult challenges in streaming data analysis is dealing with changes in a stream. Changes can be sudden or gradual, persistent, recurring, or transient. A stream can change on two levels: feature and concept. Feature changes involve variations in the values of attributes within the stream, while concept changes reflect changes in the underlying data distribution. To deal with these changes, streaming data analysis systems must be able to detect, adapt and respond quickly to these changes.

Streaming data analysis is a powerful tool for dealing with massive amounts of data in real-time. However, one of the challenges of this approach is dealing with concept drift and concept evolution. Concept drift occurs when the statistical properties of data change over time. This could be due to the source of data changing, or even due to changes in the environment. For example, if there is a change in the type of customers buying a product, the data stream will change as well. This can make it difficult to accurately predict future outcomes, as the data is no longer representative of the true underlying concept.

Concept evolution differs from concept drift in that it occurs over a longer period of time. The phenomenon of concept evolution occurs when some existing classes disappear or new classes appear over time in streaming data. For example, if the type of customer buying a product changes over a year, the data stream will gradually drift away from the original concept. This can lead to inaccurate predictions over time, as the data is no longer representative of the true underlying concept.

In order to combat concept drift and concept evolution, streaming data analysis must be able to detect and adapt to changes in the data stream. This can be done using techniques such as anomaly detection and machine learning, which can be used to detect changes in the data stream and adapt to them. By doing this, streaming data analysis can remain accurate and reliable, even in the face of changing concepts.

Feature drift and feature evolution are two other important concepts in streaming data analysis. Feature drift is a phenomenon where the distribution of data points in a feature space changes over time. This can occur when the source of the data changes, or as the data changes due to external factors. Streaming scenarios also involve the appearance and disappearance of features with time, which is a process called feature evolution. This could be due to new data sources being added to the stream, or changes in the values of existing features. A new feature could be included in the learning process in the future if found to be relevant. In the same way, if a feature is unavailable, the learning

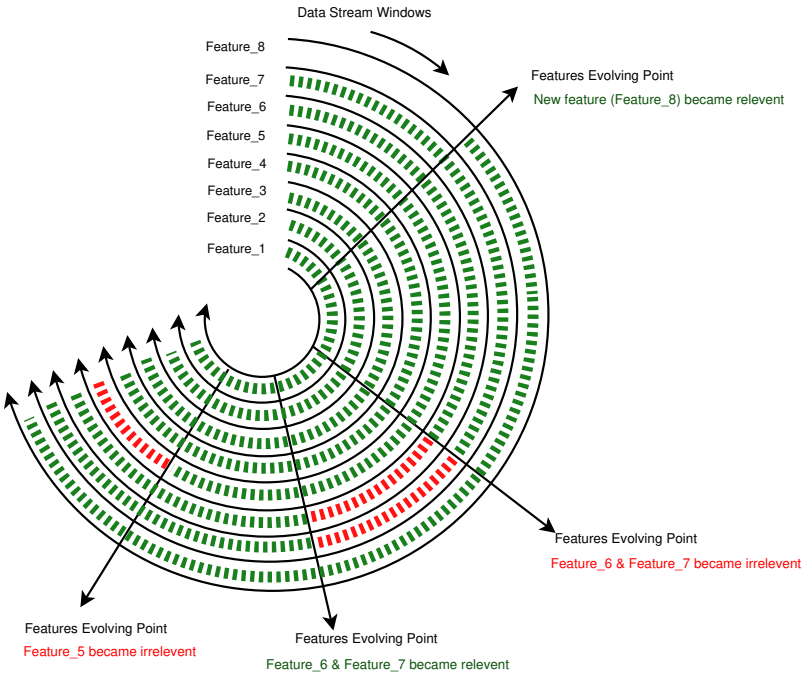

Figure 1: Evolving Features

model should ignore its existence since all its values would be considered missing. Let $DS = [(X_i, y_i)^t]_{t=0}^{\infty}$ represent a data stream in which X_i is a D dimensional vector that describes the class y_i at the time t. The manner in which features change can be divided into two categories: drifting and evolving. Suppose an arriving instance X_i^t in dimensions D at time t, $X_i^t = \{F_1^t, ..., F_D^t\}$. In a stream, feature drift occurs when a feature F_i loses its relevance, importance, or distinctive power. When there is feature drift, the significance of a feature F_i evolves over time. ie, $significance(F_i^t) \neq significance(F_i^{t+\delta})$after the time δ. When new features appear in the stream, the dimensionality of X_i changes, resulting in feature evolution. If there is feature evolution, $D^t \neq D^{t+\delta}$[Fahy et al., 2022]. Consider the Weather data set[Elwell and Polikar, 2011], which contains 18000 instances with 8 features. Data instances were divided into five data sets with varying feature sets, as shown in Figure-1, to create a feature-evolving data stream. In the first set, there are seven features (features 1 to 7), whereas, in the second set, there is one new feature (feature 8). In the third data set, however, features 6 and 7 disappear, whereas, in set 4, they reappear. However, feature 5 disappears in set 5. Most existing models do not accommodate changes in streaming data input vectors. A highly crucial and difficult task is to develop a dynamic input vector representation for streaming data.

In order to address feature drift and feature evolution, it is important for data scientists to monitor the data stream for changes. By doing so, they can adjust the model accordingly in order to ensure that the predictions remain accurate. Additionally, it is important to use data points from different sources, as this will help to ensure that the distribution of data points remains relatively consistent. Finally, data scientists should use techniques such as data augmentation to ensure that the model is able to handle changes in feature values. In

addition to being time-consuming, it requires a lot of effort. We address these challenges of feature evolution in streaming data analysis by using a Dynamic Auto-encoder to convert dynamic feature vectors into a fixed input vector representation.

In our proposed model, an adaptive landmark windowing is used to take the most recent data points. Due to its feature-evolving property, each window may have a different number of features. The various features are converted into a fixed number of features using Dynamic Auto-encoder. The classification model uses this feature set. We preserve the properties of previous data points by adding representatives of those data points to the current data points without storing previous data or a summary. The classification model is then used to make predictions. The predictions are tested and evaluated. Which gives good results.

Main contributions

1. The Particular problem itself is novel when it is considered in the broader domain.
2. Novel approach of Dynamic Auto-encoder(DAE) to handle the dynamics of evolving features.
3. Concept preservation mechanism used for handling streaming data.
4. Applications of ensemble classification technique for evolving stream classifier.

This paper is structured as follows: Section-2 provides a review of previous research, focusing primarily on evolving streaming data. Section-3 details our proposed algorithm and methodology. Section-4 covers the evaluation of the model and the analysis of results. Finally, Section-5 discusses the conclusions, open issues, and future work.

2 Related Works

There are several tasks involved in streaming data analysis, including classification, clustering, anomaly detection, frequency counting, regression, and time series analysis. In our work, we focus on the classification of features-evolving data streams.

2.1 Streaming Data Analysis and Feature Drift

[Aggarwal et al., 2006] address the challenges of handling streaming data with infinite length and concept drift. They propose techniques for continuous data influx and adaptability to changing data distributions over time. The paper lays the groundwork for understanding the fundamental issues in streaming data analysis and provides insights into strategies for real-time data processing and classification. [Chen et al., 2008] contribute to the field of streaming data analysis by exploring methods to handle concept drift in dynamic data environments. Their research focuses on developing adaptive classification models that can effectively detect and respond to concept drift, ensuring the stability and accuracy of classification models over time. The paper offers valuable insights into the challenges of concept drift and the importance of adaptive learning algorithms in streaming data analysis. [Hulten et al., 2001] introduce the concept of incremental learning and propose the VFDT (Very Fast Decision Tree) algorithm for processing streaming data. Their work addresses the need for efficient and scalable algorithms capable of handling continuous data influx and evolving feature patterns. The VFDT algorithm offers a practical solution for real-time decision-making in dynamic

data environments, laying the foundation for further research in streaming data analysis. [Kolter and Maloof, 2005] present methods for anomaly detection in streaming data, focusing on identifying unusual patterns or events that deviate from normal behavior. Their research contributes to the field of anomaly detection by proposing techniques to detect and classify anomalies in real-time data streams. The paper highlights the importance of early detection and mitigation of anomalies in dynamic data environments, ensuring the reliability and robustness of streaming data analysis systems. [Masud et al., 2008] investigate techniques for handling imbalanced data in streaming data analysis. Their research focuses on addressing the challenges posed by skewed data distributions and class imbalance in dynamic data environments. The paper proposes methods to improve the classification performance of imbalanced datasets, offering practical solutions for real-world streaming data analysis applications. [Scholz and Klinkenberg, 2005] explore ensemble learning techniques for streaming data analysis, focusing on combining multiple classifiers to improve classification accuracy and robustness. Their research demonstrates the effectiveness of ensemble methods in handling concept drift and evolving feature patterns in dynamic data environments. The paper provides valuable insights into the benefits of ensemble learning for real-time data processing and classification tasks. [Wang et al., 2003] propose techniques for feature selection in streaming data analysis, aiming to identify relevant features and reduce dimensionality in dynamic data environments. Their research addresses the challenges of feature drift and evolution by selecting informative features that capture the underlying structure of streaming data. The paper offers practical methods for improving the efficiency and effectiveness of classification models in dynamic data environments. [Yang et al., 2005] investigate methods for handling incremental updates in streaming data analysis, focusing on efficient algorithms for processing continuous data influx. Their research contributes to the field of incremental learning by proposing techniques to update classification models in real-time and adapt to changing data distributions over time. The paper offers insights into the challenges of processing streaming data and provides practical solutions for scalable and efficient data processing systems.

[Gomes et al., 2019b] contribute to streaming data analysis by addressing the issue of feature drift and evolution. They propose a feature evolution-aware classification framework to adaptively select relevant features and mitigate over-fitting in dynamic data environments. The paper emphasizes the importance of detecting and adapting to feature changes to maintain classification accuracy over time. MOA [Bifet et al., 2010] is an open-source software framework for mining data streams in real-time. Developed by Bifet et al., MOA provides a comprehensive set of algorithms for classification, clustering, and regression on streaming data. The framework supports incremental learning and concept drift detection, making it suitable for dynamic data analysis tasks. SAMOA [Morales and Bifet, 2015] is a distributed streaming machine learning framework built on top of Apache Storm. Morales and Bifet introduce SAMOA as a scalable and fault-tolerant platform for performing large-scale machine learning tasks on streaming data. The framework supports various learning algorithms and provides real-time processing capabilities for handling continuous data influx. Scikit-multiflow [Montiel et al., 2018] is an extension of the popular Scikit-learn library, designed for incremental learning on streaming data. Montiel et al. introduce Scikit-multiflow as a versatile tool for building and evaluating streaming data analysis pipelines. The library offers a wide range of algorithms and evaluation metrics for monitoring model performance in dynamic data environments. Vowpal Wabbit [Agarwal et al., 2014] is a fast and scalable online learning system developed by Agarwal et al. It is designed for large-scale machine learning tasks, particularly on streaming data. VW supports a variety of learning algorithms, including linear models,

tree ensembles, and neural networks, making it suitable for a wide range of streaming data analysis applications. StreamDM [Zaharia et al., 2013] is a distributed machine learning library for mining big data streams. Zaharia et al. introduce StreamDM as an extension of the Apache Spark platform, providing efficient algorithms for processing streaming data in parallel. The library offers support for classification, clustering, and regression tasks, with a focus on scalability and real-time performance. Ray RLlib [Liang et al., 2018] is a reinforcement learning library built on top of the Ray distributed computing framework. Liang et al. present Ray RLlib as a scalable and flexible platform for training reinforcement learning models on streaming data. The library supports various algorithms and environments for tackling dynamic decision-making problems in real time. The literature review delves into pivotal research and frameworks in streaming data analysis, tackling issues like infinite data influx, concept drift, and feature evolution. Researchers such as [Aggarwal et al., 2006], [Chen et al., 2008], and [Hulten et al., 2001]. propose strategies for adaptability, incremental learning, and real-time decision-making to address these challenges. On the other hand, [Kolter and Maloof, 2005] ,[Masud et al., 2008], and [Scholz and Klinkenberg, 2005] focus on anomaly detection, imbalanced data, and ensemble learning techniques. Additionally, [Wang et al., 2003] and [Yang et al., 2005] explore feature selection and incremental updates, offering practical solutions for accurate classification in dynamic data environments. Furthermore, [Gomes et al., 2019b] introduce a feature evolution-aware classification framework to tackle feature drift, while MOA [Bifet et al., 2010] and SAMOA [Morales and Bifet, 2015] provide scalable platforms for real-time mining of data streams. Scikit-multiflow [Montiel et al., 2018] extends Scikit-learn to facilitate incremental learning, while Vowpal Wabbit [Agarwal et al., 2014] offers a fast and scalable online learning system. Lastly, StreamDM [Zaharia et al., 2013] and Ray RLlib [Liang et al., 2018] contribute distributed machine learning libraries, further advancing the field by enabling efficient and adaptive processing of dynamic data environments.

Recent advancements in handling dynamic feature spaces and feature drift in data streams have introduced innovative methods to improve model adaptability and efficiency. [Sajedi and Razzazi, 2024] propose a feature mapping mechanism that aligns evolving feature spaces with a stable representation, enhancing classification accuracy in high-dimensional data streams. [Leonardi et al., 2024] explore contextual feature drift in large language models (LLMs), identifying adaptive retention mechanisms that mitigate performance degradation across sequential inputs, crucial for robust real-time applications.[Magistri et al., 2024] introduce Elastic Feature Consolidation (EFC), a framework for exemplar-free incremental learning that adjusts feature importance dynamically, achieving high accuracy without requiring historical data, ideal for memory-constrained scenarios.[Wu et al., 2023] proposed an efficient feature selection in data streams by dynamically identifying and updating the minimal subset of relevant features, reducing computational overhead while maintaining predictive performance. Together, these studies emphasize dynamic feature alignment, adaptive retention, and efficient feature selection as pivotal strategies for addressing challenges in evolving data stream

2.2 Ensemble Learning Techniques in Streaming Data Analysis

[Oza and Russell, 2001] introduced Online Bagging and Boosting as ensemble learning techniques suitable for streaming data analysis. Online Bagging combines multiple classifiers trained on bootstrap samples of the data stream, while Online Boosting adapts the weights of classifiers based on their performance over time. These techniques are effective for handling concept drift and achieving high classification accuracy in dynamic

data environments. [Wang et al., 2003] proposed Weighted Ensemble Classifiers as a method to improve classification accuracy by assigning weights to individual classifiers in the ensemble. This approach allows for adaptive weighting of classifiers based on their performance on different subsets of data, leading to more robust and accurate predictions in streaming data scenarios. [Hashemi et al., 2008] introduced Adapted One-vs-All Decision Trees as a technique for multi-class classification in streaming data. This approach adapts decision trees to handle concept drift by dynamically updating tree structures and decision boundaries based on incoming data. OVA decision trees are effective for handling evolving class distributions and maintaining classification accuracy over time. SEA [Street and Kim, 2001] is a streaming ensemble algorithm that continuously updates its base classifiers using new data instances. It adapts to concept drift by dynamically adjusting the weights of individual classifiers based on their performance on recent data. SEA is suitable for handling evolving data distributions and maintaining classification accuracy in real-time streaming environments. [Kotler and Maloof, 2003] introduced Dynamic Weighted Ensemble as an ensemble learning technique for streaming data analysis. DWM adapts to changing data distributions by dynamically adjusting the weights of base classifiers based on their performance over time. This approach improves classification accuracy by giving more weight to classifiers that perform well on recent data instances.[Bifet and Gavalda, 2007] proposed ADWIN as a change detection algorithm for data streams. ADWIN monitors incoming data and dynamically adjusts its window size to detect changes in data distribution. This algorithm is essential for identifying concept drift and triggering updates to ensemble classifiers in streaming data analysis tasks. [Gomes et al., 2017] introduced Adaptive Random Forest as an ensemble learning technique tailored for streaming data analysis. ARF dynamically grows and prunes decision trees based on data stream characteristics, allowing for efficient adaptation to concept drift. This approach improves classification accuracy and scalability in dynamic data environments.[Gomes et al., 2019a] proposed Streaming Random Subspaces as a technique for ensemble learning in streaming data analysis. SRP constructs multiple random subspaces from the input features and trains base classifiers on each subspace. This approach diversifies the ensemble and improves robustness to concept drift and feature evolution in streaming data scenarios. These ensemble learning techniques offer effective solutions for handling concept drift, evolving data distributions, and feature evolution in streaming data analysis. They enable accurate and adaptive classification in dynamic data environments, contributing to advancements in real-time decision-making and data-driven applications.

2.3 Online Learning

[Cesa-Bianchi and Lugosi, 2006] introduced the theoretical foundations of online learning algorithms. Their work focused on understanding the theoretical properties of online learning algorithms, such as regret minimization and convergence rates. This paper laid the groundwork for the study of online learning in sequential decision-making tasks. [Hazan et al., 2007] proposed online convex optimization algorithms for sequential decision-making problems. Their work addressed the challenge of optimizing convex functions in an online setting, where decisions are made sequentially without knowledge of future observations. This paper provided practical algorithms for solving a wide range of online optimization problems.[Shalev-Shwartz et al., 2012] introduced the online mirror descent algorithm for online convex optimization. Their work extended classical mirror descent methods to the online setting, offering efficient algorithms for solving convex optimization problems in an online fashion. This paper contributed to

the development of robust and scalable online learning algorithms. [Li et al., 2014] proposed online learning algorithms for structured prediction tasks. Their work focused on learning predictive models for structured outputs, such as sequences or trees, in an online setting. This paper provided algorithms for solving structured prediction problems efficiently and effectively, enabling applications in natural language processing and computer vision.[Muslea et al., 2002] introduced online learning algorithms for text classification tasks. Their work addressed the challenge of classifying text documents in a streaming fashion, where data arrives sequentially and must be classified in real-time. This paper contributed to the development of online learning techniques for text mining and information retrieval applications. [Xu et al., 2013] proposed online learning algorithms for collaborative filtering and recommendation systems. Their work focused on learning personalized recommendation models from user feedback in an online fashion. This paper provided algorithms for adapting recommendation models to changing user preferences and evolving item popularity trends. [Pan and Yang, 2009] introduced online feature selection algorithms for high-dimensional data streams. Their work addressed the challenge of selecting informative features from streaming data in real-time. This paper provided algorithms for adapting feature selection to changing data distributions and evolving feature patterns.[Raina et al., 2007] proposed online learning algorithms for training large-scale linear models. Their work focused on efficiently updating model parameters in an online fashion, enabling scalable training of linear models on massive datasets. This paper contributed to the development of online learning techniques for big data analytics and machine learning applications. [Guan and Li, 2001] introduced online learning algorithms for training support vector machines (SVMs) in a sequential fashion. Their work addressed the challenge of adapting SVM models to changing data distributions and evolving class boundaries. This paper provided algorithms for updating SVM models incrementally, enabling real-time adaptation to streaming data.

The review discusses key papers in online learning and sequential decision-making, covering theoretical foundations, algorithm development, and applications across diverse domains. [Cesa-Bianchi and Lugosi, 2006] lay theoretical groundwork for online learning, while [Hazan et al., 2007] propose practical online convex optimization algorithms. [Shalev-Shwartz et al., 2012] extend mirror descent methods for online convex optimization. [Li et al., 2014] focus on structured prediction tasks, while [Muslea et al., 2002] address text classification in streaming data. [Xu et al., 2013] introduce online learning for collaborative filtering, and [Pan and Yang, 2009] develop online feature selection algorithms. [Raina et al., 2007] propose online learning for large-scale linear models, and [Guan and Li, 2001] tackle online training of support vector machines. These papers collectively advance online learning theory and provide practical solutions for various real-world applications.

2.4 Concept evolution

[Hashemi et al., 2008] proposed Adapted One-vs-All Decision Trees as a method for multi-class classification in streaming data. This approach dynamically updates decision trees to handle concept drift, ensuring the adaptability of the classification model to changing data distributions over time. The paper contributes to addressing the challenges of evolving class distributions in streaming data analysis. [Ditzler et al., 2010] introduced Adaptive Windowing for Streaming Data Analysis, a technique for adaptively adjusting the window size used for processing data streams. By dynamically resizing the window based on data characteristics, this approach enables efficient and accurate analysis of streaming data while accommodating variations in data arrival rates and concept

drift. [Parker et al., 2012] proposed a method for Incremental Learning in Streaming Environments, focusing on updating classification models incrementally as new data arrives. This approach enables real-time adaptation to concept drift and evolving data distributions, ensuring the accuracy and effectiveness of classification models in dynamic data environments. [Parker and Khan, 2015] introduced a framework for Dynamic Ensemble Selection in Streaming Data Analysis. This framework dynamically selects the most relevant ensemble members based on their performance on recent data instances, improving the robustness and adaptability of ensemble classifiers to concept drift and changing data distributions. [Haque et al., 2016] proposed an Efficient Clustering Algorithm for Streaming Data Analysis. This algorithm efficiently clusters streaming data into meaningful groups, facilitating exploratory analysis and pattern discovery in dynamic data environments. The algorithm's scalability and effectiveness make it suitable for real-time data processing and analysis tasks. [ZareMoodi et al., 2019] investigated Concept Drift Detection and Adaptation in Streaming Data Analysis. Their research focuses on developing techniques to detect and adapt to concept drift in streaming data, ensuring the accuracy and reliability of classification models over time. The paper contributes to addressing one of the key challenges in streaming data analysis. [Chambers et al., 2020] proposed DeepStreamCE, a Deep Learning Framework for Streaming Data Analysis. This framework leverages deep learning techniques to analyze streaming data in real-time, enabling the extraction of complex patterns and relationships from dynamic data streams. DeepStreamCE offers advanced capabilities for handling large volumes of streaming data efficiently. [Zheng et al., 2021] introduced Semi-Supervised Learning Approaches for Streaming Data Analysis. Their research focuses on leveraging semi-supervised learning techniques to improve the performance of classification models in streaming data environments with limited labeled data. The paper offers insights into the potential of semi-supervised learning for enhancing the accuracy and robustness of streaming data analysis systems.

The reviewed papers cover various aspects of streaming data analysis, offering solutions to key challenges like concept drift, concept evolution, adaptability, and real-time processing. [Hashemi et al., 2008] proposed adaptive decision trees for multi-class classification, while [Ditzler et al., 2010] introduced adaptive windowing techniques. [Parker et al., 2012] focused on incremental learning, and [Parker and Khan, 2015] devised dynamic ensemble selection frameworks. [Haque et al., 2016] developed efficient clustering algorithms, and [ZareMoodi et al., 2019] explored concept drift detection. [Chambers et al., 2020] introduced a deep learning framework, and [Zheng et al., 2021] delved into semi-supervised learning approaches. These papers collectively contribute to enhancing the effectiveness and efficiency of streaming data analysis in dynamic environments.

2.5 Feature Evolution

Hierarchical Stream Miner (HSMiner)[Parker et al., 2012] decomposes the classification problem into tiers, prunes irrelevant features, and uses weighted voting and boosting to counter individual errors. [Hou and Zhou, 2017] proposed a one-pass learning approach that attempts to compress the information of vanished features into functions of survived features, and then expand that information to include augmented features. In a subsequent work, [Hou et al., 2017] further explored Dynamic Ensemble Selection in Streaming Environments. They introduced refinements and extensions to existing techniques, improving the efficiency and effectiveness of ensemble learning in streaming data analysis tasks. To improve performance,their method attempt to recover vanished

features and exploit them with current features. [Savitha et al., 2020] proposed the O-RMB framework to handle feature-evolving data streams. Before performing feature selection, LOSSA[Wu et al., 2021] uses latent factor analysis to pre-estimate missing data in sparse streaming features.[Wambura et al., 2020] [Wambura et al., 2022] use the One sketch Fits All algorithm (OFA) to handle feature-evolving data streams. [Zhang et al., 2020] proposed Online Learning Algorithms for Streaming Data Analysis. They developed algorithms for continuously updating classification models and adapting to changing data distributions in real-time streaming data environments. [Hou et al., 2021] introduced a method for Dynamic Feature Selection in Streaming Data Analysis. Their approach focuses on selecting relevant features dynamically based on their importance and relevance to the classification task, enabling efficient and accurate analysis of streaming data with evolving feature spaces. [Jafseer et al., 2023] proposes a technique for detecting feature drift in streaming data using an overlapping window approach and the Mann-Whitney U test. By continuously monitoring feature distributions over time, the method identifies significant changes that indicate feature drift, enabling timely adaptation of classification models to evolving data environments.

While most streaming data algorithms assume consistent features across all samples, only a few address feature evolution.To tackle feature-evolving data streams, we propose a novel deep learning approach distinct from existing methods. By employing Dynamic Auto-encoder during preprocessing, we transform evolving features into a fixed-dimensional representation. This facilitates subsequent analysis steps, culminating in the construction of an ensemble model for classification. Leveraging the transformed features, our approach enhances adaptability to evolving data, promising more accurate and robust analysis in dynamic streaming environments.

3 Methodology

In this section, we present a detailed overview of the proposed architecture and algorithms designed to handle evolving data streams in a machine-learning pipeline. The dynamic nature of data streams necessitates a robust and adaptive approach to ensure real-time processing with better model accuracy.

3.1 Overview of Proposed Architecture

The proposed model for classifying feature-evolving data streams is illustrated in Figure-2. The most recent data should be analyzed in streaming data analyses rather than the entire dataset. In this case, the proposed model takes into account the most recent data points by using the adaptive landmark window model[Calinski, 1974]. The landmark window model includes data between two landmarks. Elapsed time or instance count determines the window length. This model uses successive windows that do not overlap, so new windows start where the old windows end. Each window may have a different number of features due to the feature-evolving property. In other words, some features may disappear, or even some new features may appear, or some features may reappear as shown in the figure-1. So, windows may contain data points with different features. We employ the Dynamic Auto-encoder to extract features in our model for each data window, ensuring that any evolution in features does not affect the performance of the model. Our next step is to classify data streams using an ensemble model. Due to the fully online nature of the model, no summary of data is stored. We preserve the properties of previous windows by adding representatives to the current window while storing no previous data or a synopsis.

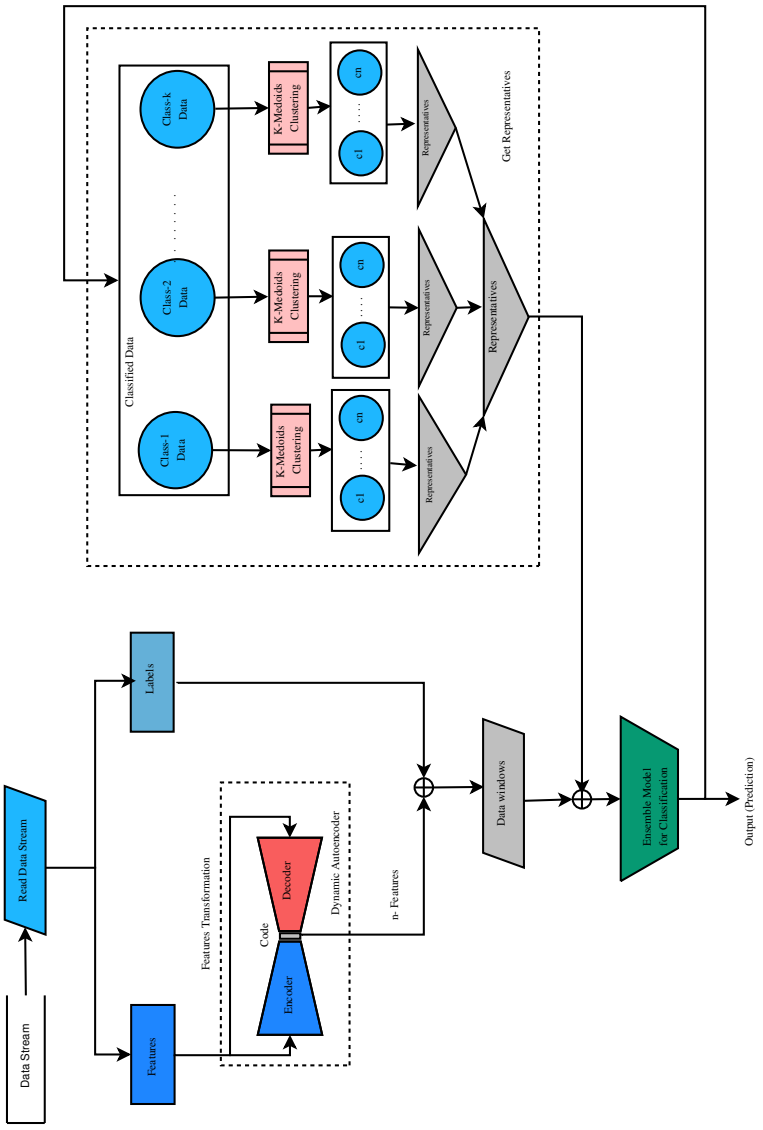

Figure 2: Architecture of proposed Model

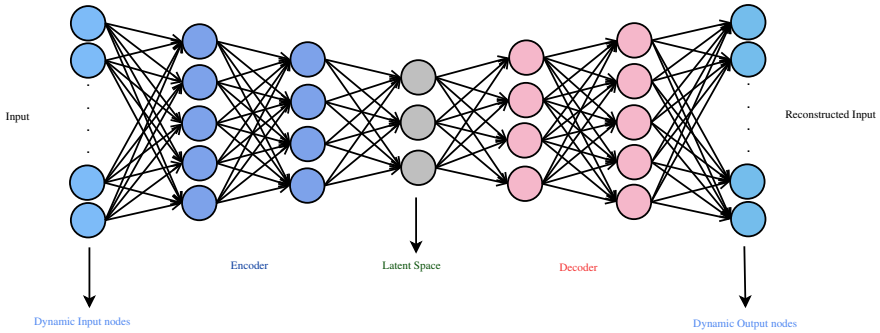

Figure 3: Dynamic Autoencoder

3.2 Handling Evolving Features using DAE

To handle the dynamics of evolving features we utilize Dynamic Auto-encoder (DAE) as shown in figure-3 to maintain a fixed number of features, ensuring consistency despite variations over time. This approach effectively addresses the challenge of evolving feature sets. By incorporating new latent values and their corresponding labels, the classification process remains robust, maintaining performance levels even as the dataset evolves. In our proposed model, the number of input and output nodes varies based on the features in the data stream, while the hidden layers and bottleneck layer maintain a fixed number of nodes. Our experimental findings demonstrate superior performance when employing four hidden layers in the encoding and decoding parts, with node configurations of 100, 50, 20, and 10, respectively, for encoding, and 10, 20, 50, and 100, respectively, for decoding. To prevent over-fitting, we incorporated a dropout mechanism, removing 20 percent of learned nodes from each layer. After training and fitting the model, we utilize the generated code in the latent space as features for classification. Further investigation revealed that varying the number of nodes in the bottleneck layer (ranging from 2 to 6) did not significantly impact the classification process. An illustration of these results is provided in the figure.

3.3 Classification

For streaming data classification, we developed a voting ensemble model. This includes four machine learning techniques: Support Vector Machines, Logistic Regression, Decision Tree, and K-Nearest Neighbor(KNN). We use the code generated by Dynamic Auto-encoder as features for training and testing the model. Subsequently, encoded features and original labels are combined to train the model. In addition, the prediction of the model is done using unknown data with coded features. With these coded features, the model performs reasonably. We chose some representatives to represent the various

Algorithm 1 FeaturesEvolvingDataClassification (W_C)

Require: W_C- Current window o Data stream and e-elitism rate
Ensure: Classes
1: $R_P = \phi$ ▷ Representatives of previous window initialized as Null
2: **while** $W_C \neq \phi$ **do**
3: $W_C.Features = E(W_C.Features)$ ▷ Encoded Features
4: **if** $R_P \neq \phi$ **then**
5: $W_C = W_C \cup R_P$
6: **end if**
7: $Result = classifier(W_C.Features, W_C.Labels)$ ▷ Classifications
8: $R_P = getRepresentatives(W_C, e)$
9: **end while**

Algorithm 2 getRepresentatives(D_p, e)

Require: D_p-Data points of the Class C and e-elitism rate
Ensure: representatives of each Class $reps$ for appending with next data window W
while $D_p \neq \phi$ **do**
$k = MAX(1, e \times |D_p|)$
$km = KMedoids(C, k)$
$samples = km.Centers$
$reps = reps \cup samples$
end while
$return\ reps$

classes in the current window. These representatives will be added to the data points in the upcoming windows to preserving the properties of the current data points.

3.4 Electing Representatives

We preserve the properties of previous windows rather than storing previous data. Using the K-Medoid clustering technique, each class's data points are clustered and the cluster centers are taken as stratified samples for representing the class. For each class, the number of clusters 'k' is estimated to be 10 percent of the data size. All properties of data points in the previous data window can be updated by merging all representatives and adding them to the next data window. Thus, the model retains the characteristics of its predecessors without storing any data or synopsis about those predecessors.

4 Result Analysis

The results of the experiment are discussed in this section. There are two hyperparameters for the proposed model: window size (amount of data per chunk) and the number of class members from the previous window. It is most effective to use a window size of 500 and 10 percent of each class size as a representative,e for weather datasets (Fig:4) and

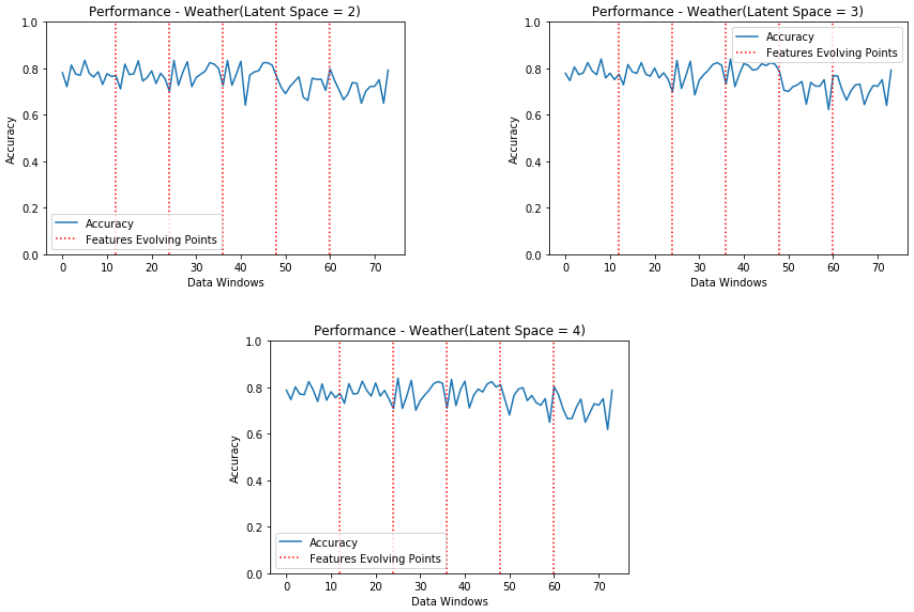

Figure 4: Accuracy of the model for each window with the different Latent Space - Weather Data

Measure	Latent Space-2	Latent Space-3	Latent Space-4
Sensitivity/Recall	0.8269	0.8385	0.8423
Specificity	0.8750	0.8833	0.8917
Precision	0.8776	0.8862	0.8939
Negative Predictive Value	0.8235	0.8346	0.8392
False Positive Rate	0.1250	0.1167	0.1083
False Discovery Rate	0.1224	0.1138	0.1061
False Negative Rate	0.1731	0.1615	0.1577
Accuracy	0.8500	0.8600	0.8660
F1 Score	0.8515	0.8617	0.8673

Table 1: Weather Data Classification Report

electricity datasets (Fig:8). Forest Cover types datasets do better with window size 2000 (Fig:6).

Using auto-encoder, we convert the features of data windows into a fixed number of features. By using Dynamic Auto-encoder, features are converted into different latent spaces, such as 2,3, and 4, and the values from these latent spaces are then used for classification. The evolving features of data streams handle in this way.

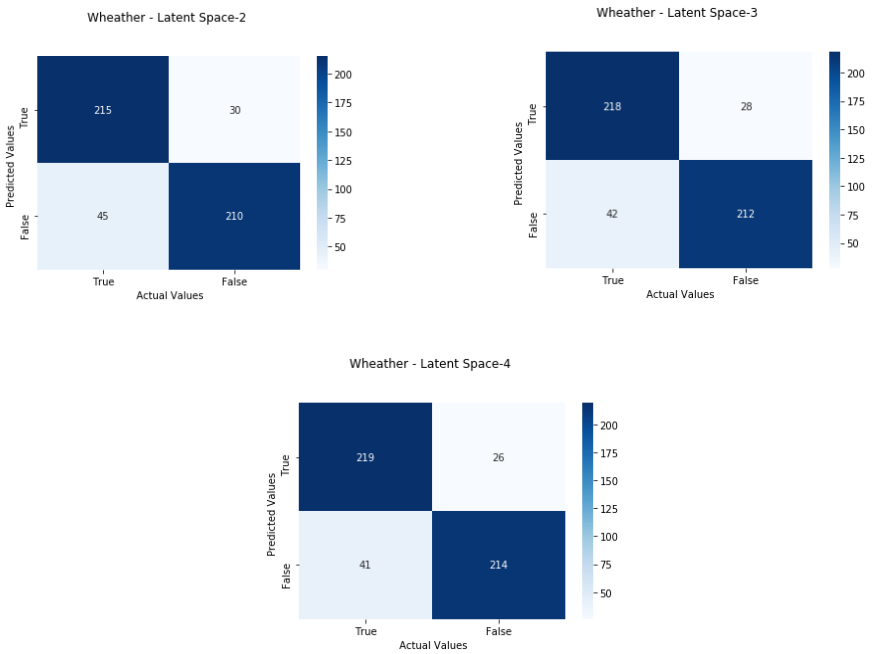

Figure 5: Confusion Matrices for Weather data

Classifier	Recall	Precision	Accuracy	F1 Score
HSMiner [Parker et al., 2012]	0.824	0.874	0.853	0.850
O-RMB [Savitha et al., 2020]	0.815	0.857	0.844	0.845
LOSSA [Wu et al., 2021]	0.815	0.867	0.846	0.854
OFA [Wambura et al., 2022]	0.827	0.864	0.844	0.850
Proposed Model	0.843	0.894	0.867	0.871

Table 2: Weather Data Classification Report Comparison

With each data stream, we predicted the model. In the testing data window, there are 500 instances of weather data (Fig:5, Tab:1), 500 instances of electricity data (Fig:9, Tab:5), and 1000 instances of forest cover types data (Fig:7, Tab:3).

The model was compared to four existing machine-learning methods: Support Vector Machines, Logistic Regression, Decision Tree, and K-Nearest Neighbor (KNN) using

Measure	Latent Space-2			Latent Space-3			Latent Space-4		
	Class-1	Class-2	Class-5	Class-1	Class-2	Class-5	Class-1	Class-2	Class-5
Sensitivity/Recall	0.9164	0.9374	0.9299	0.9200	0.9413	0.9252	0.9164	0.9335	0.9346
Specificity	0.9683	0.9632	0.9631	0.9683	0.9652	0.9644	0.9683	0.9632	0.9618
Precision	0.9164	0.9638	0.8728	0.9167	0.9659	0.8761	0.9164	0.9636	0.8696
Negative Predictive Value	0.9683	0.9364	0.9806	0.9696	0.9402	0.9793	0.9683	0.9327	0.9818
False Positive Rate	0.0317	0.0368	0.0369	0.0317	0.0348	0.0356	0.0317	0.0368	0.0382
False Discovery Rate	0.0836	0.0362	0.1272	0.0833	0.0341	0.1239	0.0836	0.0364	0.1304
False Negative Rate	0.0836	0.0626	0.0701	0.0800	0.0587	0.0748	0.0836	0.0665	0.0654
Accuracy	0.9540	0.9500	0.9560	0.9550	0.9530	0.9560	0.9540	0.9480	0.9560
F1 Score	0.9164	0.9504	0.9005	0.9183	0.9534	0.9000	0.9164	0.9483	0.9009

Table 3: Forest Cover type Data Classification Report

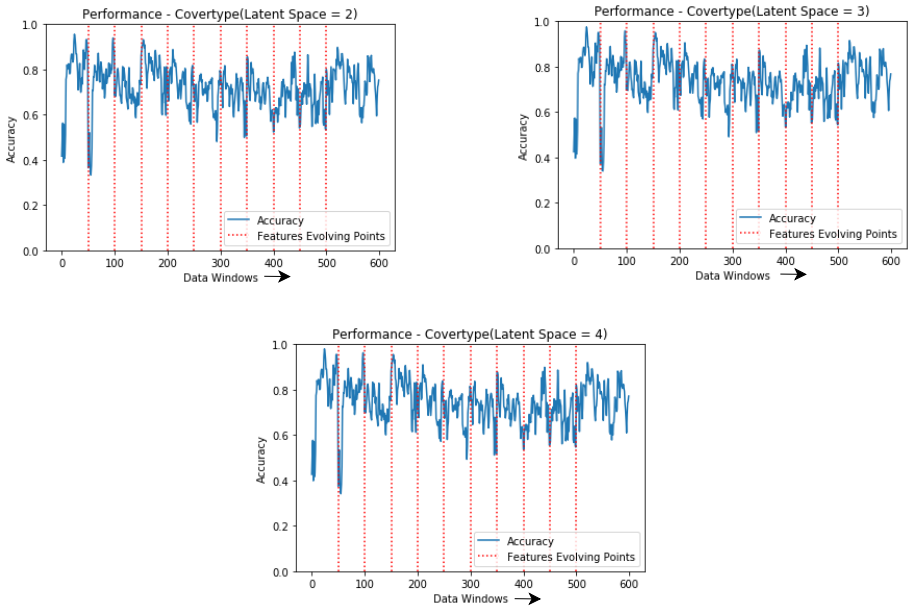

Figure 6: Accuracy of the model for each window with the different Latent Space - Forest Cover type Data

three datasets: Weather data(Tab:2), Electricity data(Tab:6), and Forest Cover data(Tab: 4). The tables show that the proposed model performs better than other models. Furthermore, only a small change in performance is observed as a result of reducing the latent space. Therefore, there is no substantial loss of information as a result of reducing latent space.

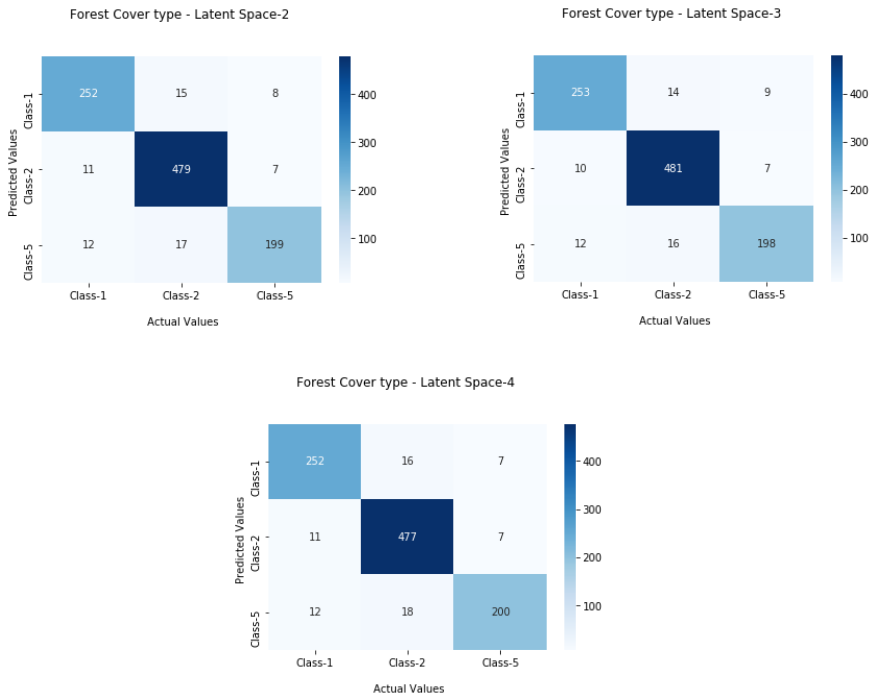

Figure 7: Confusion Matrices for Forest Cover type data

Classifier	Recall	Precision	Accuracy	F1 Score
HSMiner [Parker et al., 2012]	0.894	0.906	0.934	0.905
O-RMB [Savitha et al., 2020]	0.906	0.884	0.914	0.803
LOSSA [Wu et al., 2021]	0.895	0.904	0.934	0.905
OFA [Wambura et al., 2022]	0.887	0.880	0.904	0.881
Proposed Model	0.916	0.921	0.954	0.917

Table 4: Forest Cover type Data Classification Report Comparison

4.1 Statistical Analysis: T-Test Comparison

The comparison of performance metrics using the t-test between the original feature space and Latent Space-4 shows comparable results across all measures, with no signifi-

Measure	Latent Space-2	Latent Space-3	Latent Space-4
Sensitivity/Recall	0.9388	0.9469	0.9510
Specificity	0.9216	0.9294	0.9451
Precision	0.9200	0.9280	0.9433
Negative Predictive Value	0.9400	0.9480	0.9526
False Positive Rate	0.0784	0.0706	0.0549
False Discovery Rate	0.0800	0.0720	0.0567
False Negative Rate	0.0612	0.0531	0.0490
Accuracy	0.9300	0.9380	0.9480
F1 Score	0.9293	0.9374	0.9472

Table 5: Electricity Data Classification Report

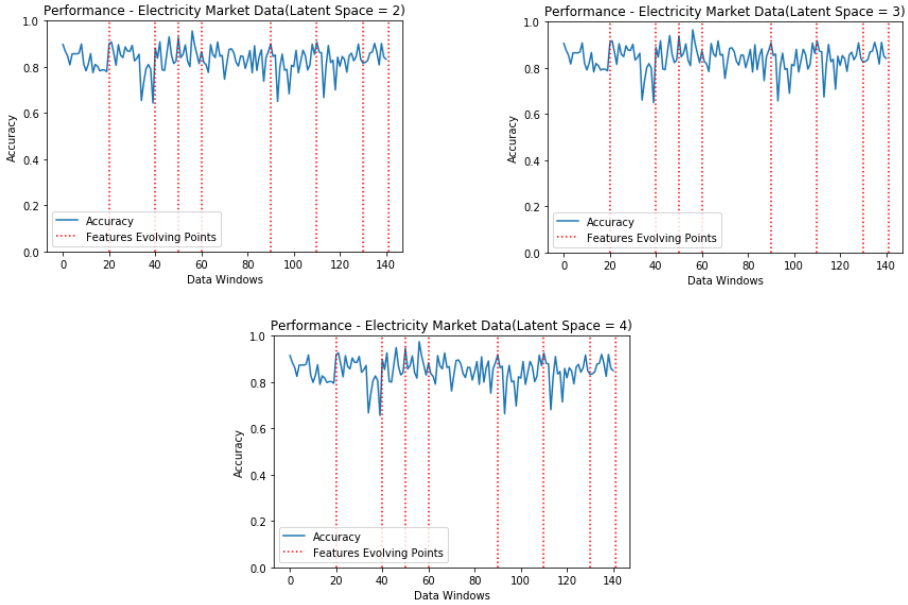

Figure 8: Accuracy of the model for each window with the different Latent Space - Electricity Data

cant differences observed for most metrics, as indicated by p-values greater than 0.05 across all datasets(Tab:7,Tab:8, and Tab:9). The Latent Space-4 representation maintains the model's predictive capabilities while effectively addressing feature evolution and potentially improving performance by reducing computational overhead.

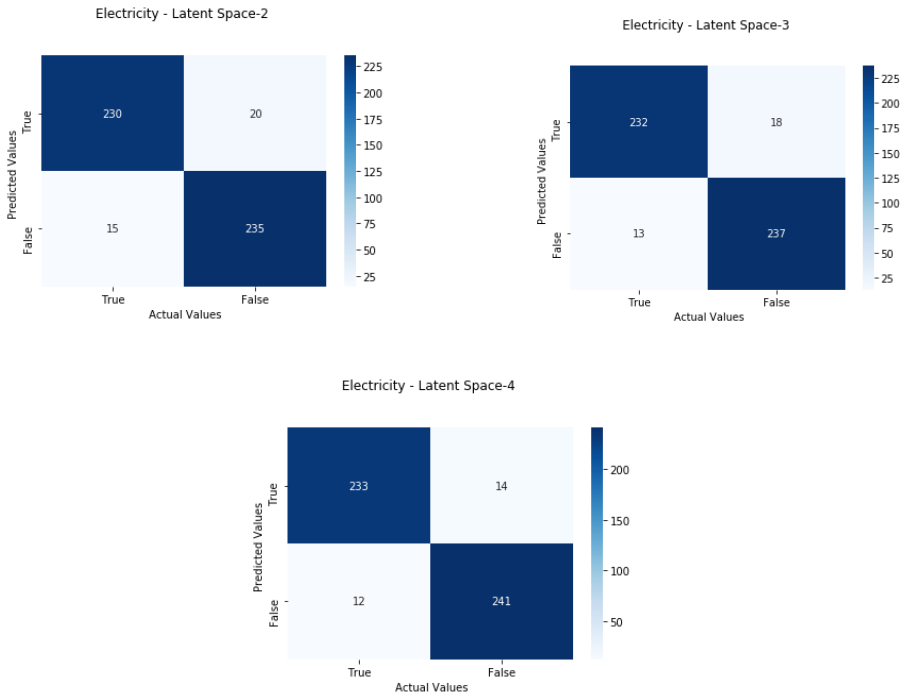

Figure 9: Confusion Matrices for Electricity data

Classifier	Recall	Precision	Accuracy	F1 Score
HSMiner [Parker et al., 2012]	0.921	0.933	0.931	0.929
O-RMB [Savitha et al., 2020]	0.931	0.923	0.938	0.925
LOSSA [Wu et al., 2021]	0.921	0.904	0.920	0.922
OFA [Wambura et al., 2022]	0.928	0.911	0.922	0.900
Proposed Model	0.951	0.943	0.948	0.977

Table 6: Electricity Data Classification Report Comparison

4.2 Computational Overhead Comparison

Compared to models such as HSMiner, O-RMB, LOSSA, and OFA, the proposed ensemble model incorporates Dynamic Autoencoder, Ensemble Learning, Adaptive

Measure	Latent Space-4	Original	Mean ± SD	95% CI	t-statistic	p-value
Sensitivity/Recall	0.8423	0.8443	0.8325 ± 0.0050	[0.8260, 0.8390]	0.135	0.895
Specificity	0.8917	0.9023	0.8875 ± 0.0105	[0.8800, 0.8950]	-1.000	0.326
Precision	0.8939	0.9001	0.8892 ± 0.0071	[0.8840, 0.8940]	-0.456	0.653
Negative Predictive Value	0.8392	0.8394	0.8342 ± 0.0060	[0.8300, 0.8380]	-0.062	0.951
False Positive Rate	0.1083	0.1071	0.1143 ± 0.0070	[0.1100, 0.1180]	0.602	0.556
False Discovery Rate	0.1061	0.1012	0.1112 ± 0.0080	[0.1070, 0.1150]	1.725	0.099
False Negative Rate	0.1577	0.1522	0.1611 ± 0.0080	[0.1570, 0.1650]	0.978	0.336
Accuracy	0.8660	0.8704	0.8616 ± 0.0082	[0.8560, 0.8660]	-1.032	0.316
F1 Score	0.8673	0.8682	0.8622 ± 0.0070	[0.8570, 0.8670]	-0.486	0.638

Table 7: Performance Metrics and Statistical Comparisons for Weather Data

Measure	Latent Space-4 (Avg)	Original	Mean ± SD	95% CI	t-statistic	p-value
Sensitivity/Recall	0.9282	0.9341	0.9282 ± 0.0076	[0.9130, 0.9434]	-0.528	0.634
Specificity	0.9644	0.9683	0.9644 ± 0.0029	[0.9586, 0.9702]	-1.264	0.372
Precision	0.9165	0.8786	0.9165 ± 0.0330	[0.8507, 0.9823]	2.675	0.070
Negative Predictive Value	0.9609	0.9867	0.9609 ± 0.0069	[0.9473, 0.9745]	-2.080	0.169
False Positive Rate	0.0356	0.0367	0.0356 ± 0.0025	[0.0307, 0.0405]	-0.337	0.759
False Discovery Rate	0.0835	0.1312	0.0835 ± 0.0202	[0.0434, 0.1237]	-3.328	0.045
False Negative Rate	0.0718	0.0680	0.0718 ± 0.0046	[0.0628, 0.0808]	0.730	0.554
Accuracy	0.9527	0.9583	0.9527 ± 0.0036	[0.9458, 0.9596]	-1.402	0.318
F1 Score	0.9225	0.9180	0.9225 ± 0.0200	[0.9026, 0.9424]	0.391	0.725

Table 8: Performance Metrics and Statistical Comparisons for Forest Cover type Data

Windowing, and Concept Preservation. Its complexity leads to longer training times as it adapts to changing features, while existing models like O-RMB and OFA benefit from simpler architectures for faster training. The ensemble structure and dynamic nature of this model increase inference time, whereas HSMiner and O-RMB achieve faster inference through decision tree-based or memory-efficient approaches. Moreover, the proposed model requires substantial memory resources to maintain multiple models, making it less resource efficient than HSMiner and OFA. Despite these demands, its exceptional adaptability allows it to handle feature evolution more effectively than existing models, which generally lack the flexibility to manage dynamic changes, and it can also preserve the previous concept for dealing with recurring concept drift. While the proposed model involves more computational overhead, it is justified by superior accuracy and adaptability for real-time streaming data, ultimately outperforming simpler yet faster models.

4.3 Limitation of the Proposed Model

The proposed method is designed to address feature evolution effectively. However, it does not explicitly deal with concept drift, which occurs when the statistical properties of the data change over time. While the framework can handle recurring concept drift by preserving the characteristics of previously observed concepts, it may struggle with completely new or non-recurring drifts. This indicates the need for further development to include specific strategies for managing different types of concept drift.

Measure	Latent Space-4	Original	Mean ± SD	95% CI	t-statistic	p-value
Sensitivity/Recall	0.9510	0.9541	0.9520 ± 0.02	[0.9300, 0.9740]	-0.45	0.665
Specificity	0.9451	0.9483	0.9460 ± 0.01	[0.9250, 0.9670]	-0.49	0.640
Precision	0.9280	0.9486	0.9300 ± 0.02	[0.9080, 0.9520]	-1.50	0.215
Negative Predictive Value	0.9526	0.9567	0.9530 ± 0.01	[0.9400, 0.9660]	-0.49	0.640
False Positive Rate	0.0549	0.0567	0.0550 ± 0.01	[0.0400, 0.0700]	-0.22	0.830
False Discovery Rate	0.0567	0.0502	0.0550 ± 0.01	[0.0400, 0.0700]	0.56	0.586
False Negative Rate	0.0490	0.0510	0.0500 ± 0.01	[0.0350, 0.0650]	-0.29	0.784
Accuracy	0.9480	0.9513	0.9500 ± 0.01	[0.9400, 0.9600]	-0.34	0.736
F1 Score	0.9472	0.9489	0.9470 ± 0.01	[0.9350, 0.9600]	-0.35	0.730

Table 9: Performance Metrics and Statistical Comparisons for Electricity Data

5 Conclusions and Future Work

The paper proposed a novel classification framework for streaming data with evolving features. The major components of the framework include a Dynamic Auto-encoder (DAE) employed for handling dynamics in feature-evolving data streams, a concept-preserving technique using clustering, and an ensemble classification model. Together these components provide an efficient and robust classification framework for evolving data streams. Such models can be deployed in real-time applications for IoT and social networks. In future, the proposed model could be improved by incorporating concept drift detection through statistical methods or advanced machine learning techniques, allowing for smoother adaptation. Furthermore, integrating distributed learning frameworks would enhance scalability for handling large data streams. Lastly, expanding the model to work with a variety of streaming data sources, like social media, in addition to numeric data, would make it more versatile and applicable across different contexts.

References

[Agarwal et al., 2014] Agarwal, A., Chapelle, O., Dudík, M., and Langford, J. (2014). A reliable effective terascale linear learning system. The Journal of Machine Learning Research, 15(1):1111–1133.

[Aggarwal et al., 2006] Aggarwal, C. C., Han, J., Wang, J., and Yu, P. S. (2006). A framework for on-demand classification of evolving data streams. IEEE Transactions on Knowledge and Data Engineering, 18(5):577–589.

[Bifet and Gavalda, 2007] Bifet, A. and Gavalda, R. (2007). Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM international conference on data mining, pages 443–448. SIAM.

[Bifet et al., 2010] Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., and Seidl, T. (2010). Moa: Massive online analysis, a framework for stream classification and clustering. In Proceedings of the first workshop on applications of pattern analysis, pages 44–50. PMLR.

[Calinski, 1974] Calinski, T. (1974). A dendrite method for cluster analysis. Communication in statistics, 3:1–27.

[Cesa-Bianchi and Lugosi, 2006] Cesa-Bianchi, N. and Lugosi, G. (2006). Prediction, learning, and games. Cambridge university press.

[Chambers et al., 2020] Chambers, L., Gaber, M. M., and Abdallah, Z. S. (2020). Deepstreamce: A streaming approach to concept evolution detection in deep neural networks. arXiv preprint arXiv:2004.04116.

[Chen et al., 2008] Chen, S., Wang, H., Zhou, S., and Philip, S. Y. (2008). Stop chasing trends: Discovering high order models in evolving data. In 2008 IEEE 24th International Conference on Data Engineering, pages 923–932. IEEE.

[Ditzler et al., 2010] Ditzler, G., Muhlbaier, M. D., and Polikar, R. (2010). Incremental learning of new classes in unbalanced datasets: Learn++. udnc. In International Workshop on Multiple Classifier Systems, pages 33–42. Springer.

[Elwell and Polikar, 2011] Elwell, R. and Polikar, R. (2011). Incremental learning of concept drift in nonstationary environments. IEEE Transactions on Neural Networks, 22(10):1517–1531.

[Fahy et al., 2022] Fahy, C., Yang, S., and Gongora, M. (2022). Scarcity of labels in non-stationary data streams: A survey. ACM Computing Surveys (CSUR), 55(2):1–39.

[Gomes et al., 2017] Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., and Abdessalem, T. (2017). Adaptive random forests for evolving data stream classification. Machine Learning, 106(9):1469–1495.

[Gomes et al., 2019a] Gomes, H. M., Read, J., and Bifet, A. (2019a). Streaming random patches for evolving data stream classification. In 2019 IEEE International Conference on Data Mining (ICDM), pages 240–249. IEEE.

[Gomes et al., 2019b] Gomes, H. M., Read, J., Bifet, A., Barddal, J. P., and Gama, J. (2019b). Machine learning for streaming data: state of the art, challenges, and opportunities. ACM SIGKDD Explorations Newsletter, 21(2):6–22.

[Guan and Li, 2001] Guan, S.-U. and Li, S. (2001). Incremental learning with respect to new incoming input attributes. Neural Processing Letters, 14(3):241–260.

[Haque et al., 2016] Haque, A., Khan, L., Baron, M., Thuraisingham, B., and Aggarwal, C. (2016). Efficient handling of concept drift and concept evolution over stream data. In 2016 IEEE 32nd International Conference on Data Engineering (ICDE), pages 481–492. IEEE.

[Hashemi et al., 2008] Hashemi, S., Yang, Y., Mirzamomen, Z., and Kangavari, M. (2008). Adapted one-versus-all decision trees for data stream classification. IEEE Transactions on Knowledge and Data Engineering, 21(5):624–637.

[Hazan et al., 2007] Hazan, E., Agarwal, A., and Kale, S. (2007). Logarithmic regret algorithms for online convex optimization. Machine Learning, 69(2):169–192.

[Hou et al., 2017] Hou, B.-J., Zhang, L., and Zhou, Z.-H. (2017). Learning with feature evolvable streams. Advances in Neural Information Processing Systems, 30.

[Hou et al., 2021] Hou, B.-J., Zhang, L., and Zhou, Z.-H. (2021). Prediction with unpredictable feature evolution. IEEE Transactions on Neural Networks and Learning Systems.

[Hou and Zhou, 2017] Hou, C. and Zhou, Z.-H. (2017). One-pass learning with incremental and decremental features. IEEE transactions on pattern analysis and machine intelligence, 40(11):2776–2792.

[Hulten et al., 2001] Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 97–106.

[Jafseer et al., 2023] Jafseer, K., Shailesh, S., and Sreekumar, A. (2023). Feature drift detection using overlapping window and mann-whitney u test. In 2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT), pages 1–5. IEEE.

[Kolter and Maloof, 2005] Kolter, J. Z. and Maloof, M. A. (2005). Using additive expert ensembles to cope with concept drift. In Proceedings of the 22nd international conference on Machine

learning, pages 449–456.

[Kotler and Maloof, 2003] Kotler, J. and Maloof, M. (2003). Dynamic weighted majority: A new ensemble method for tracking concept drift. In IEEE International Conference on Data Mining, pages 123–130.

[Leonardi et al., 2024] Leonardi, F., Feldman, P., Almeida, M., Moretti, W., and Iverson, C. (2024). Contextual feature drift in large language models: An examination of adaptive retention across sequential inputs.

[Li et al., 2014] Li, S.-Y., Jiang, Y., and Zhou, Z.-H. (2014). Partial multi-view clustering. In Proceedings of the AAAI conference on artificial intelligence, volume 28.

[Liang et al., 2018] Liang, E., Liaw, R., Nishihara, R., Moritz, P., Fox, R., Goldberg, K., Gonzalez, J., Jordan, M., and Stoica, I. (2018). Rllib: Abstractions for distributed reinforcement learning. In International Conference on Machine Learning, pages 3053–3062. PMLR.

[Magistri et al., 2024] Magistri, S., Trinci, T., Soutif-Cormerais, A., van de Weijer, J., and Bagdanov, A. D. (2024). Elastic feature consolidation for cold start exemplar-free incremental learning. arXiv preprint arXiv:2402.03917.

[Masud et al., 2008] Masud, M. M., Gao, J., Khan, L., Han, J., and Thuraisingham, B. (2008). A practical approach to classify evolving data streams: Training with limited amount of labeled data. In 2008 Eighth IEEE International Conference on Data Mining, pages 929–934. IEEE.

[Montiel et al., 2018] Montiel, J., Read, J., Bifet, A., and Abdessalem, T. (2018). Scikit-multiflow: A multi-output streaming framework. The Journal of Machine Learning Research, 19(1):2915–2914.

[Morales and Bifet, 2015] Morales, G. D. F. and Bifet, A. (2015). Samoa: scalable advanced massive online analysis. J. Mach. Learn. Res., 16(1):149–153.

[Muslea et al., 2002] Muslea, I., Minton, S., and Knoblock, C. A. (2002). Active+ semi-supervised learning= robust multi-view learning. In ICML, volume 2, pages 435–442. Citeseer.

[Oza and Russell, 2001] Oza, N. C. and Russell, S. J. (2001). Online bagging and boosting. In International Workshop on Artificial Intelligence and Statistics, pages 229–236. PMLR.

[Pan and Yang, 2009] Pan, S. J. and Yang, Q. (2009). A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10):1345–1359.

[Parker and Khan, 2015] Parker, B. and Khan, L. (2015). Detecting and tracking concept class drift and emergence in non-stationary fast data streams. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 29.

[Parker et al., 2012] Parker, B., Mustafa, A. M., and Khan, L. (2012). Novel class detection and feature via a tiered ensemble approach for stream mining. In 2012 IEEE 24th International Conference on Tools with Artificial Intelligence, volume 1, pages 1171–1178. IEEE.

[Raina et al., 2007] Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In Proceedings of the 24th international conference on Machine learning, pages 759–766.

[Sajedi and Razzazi, 2024] Sajedi, R. and Razzazi, M. (2024). Data stream classification in dynamic feature space using feature mapping. The Journal of Supercomputing, pages 1–19.

[Savitha et al., 2020] Savitha, R., Ambikapathi, A., and Rajaraman, K. (2020). Online rbm: Growing restricted boltzmann machine on the fly for unsupervised representation. Applied Soft Computing, 92:106278.

[Scholz and Klinkenberg, 2005] Scholz, M. and Klinkenberg, R. (2005). An ensemble classifier for drifting concepts. In Proceedings of the Second International Workshop on Knowledge Discovery in Data Streams, volume 6, pages 53–64. Porto, Portugal.

[Shalev-Shwartz et al., 2012] Shalev-Shwartz, S. et al. (2012). Online learning and online convex optimization. Foundations and Trends® in Machine Learning, 4(2):107–194.

[Street and Kim, 2001] Street, W. N. and Kim, Y. (2001). A streaming ensemble algorithm (sea) for large-scale classification. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 377–382.

[Wambura et al., 2020] Wambura, S., Huang, J., and Li, H. (2020). Long-range forecasting in feature-evolving data streams. Knowledge-Based Systems, 206:106405.

[Wambura et al., 2022] Wambura, S., Huang, J., and Li, H. (2022). Robust anomaly detection in feature-evolving time series. The Computer Journal, 65(5):1242–1256.

[Wang et al., 2003] Wang, H., Fan, W., Yu, P. S., and Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 226–235.

[Wu et al., 2021] Wu, D., He, Y., Luo, X., and Zhou, M. (2021). A latent factor analysis-based approach to online sparse streaming feature selection. IEEE Transactions on Systems, Man, and Cybernetics: Systems.

[Wu et al., 2023] Wu, X., Jiang, B., Wang, X., Ban, T., and Chen, H. (2023). Feature selection in the data stream based on incremental markov boundary learning. IEEE Transactions on Neural Networks and Learning Systems, 34(10):6740–6754.

[Xu et al., 2013] Xu, C., Tao, D., and Xu, C. (2013). A survey on multi-view learning. arXiv preprint arXiv:1304.5634.

[Yang et al., 2005] Yang, Y., Wu, X., and Zhu, X. (2005). Combining proactive and reactive predictions for data streams. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 710–715.

[Zaharia et al., 2013] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., and Stoica, I. (2013). Discretized streams: Fault-tolerant streaming computation at scale. In Proceedings of the twenty-fourth ACM symposium on operating systems principles, pages 423–438.

[ZareMoodi et al., 2019] ZareMoodi, P., Kamali Siahroudi, S., and Beigy, H. (2019). Concept-evolution detection in non-stationary data streams: a fuzzy clustering approach. Knowledge and Information Systems, 60:1329–1352.

[Zhang et al., 2020] Zhang, Z.-Y., Zhao, P., Jiang, Y., and Zhou, Z.-H. (2020). Learning with feature and distribution evolvable streams. In International Conference on Machine Learning, pages 11317–11327. PMLR.

[Zheng et al., 2021] Zheng, X., Li, P., Hu, X., and Yu, K. (2021). Semi-supervised classification on data streams with recurring concept drift and concept evolution. Knowledge-Based Systems, 215:106749.