National College of Ireland

Improving Dynamic Cloud Workload Prediction and Resource Management with AdaptiveCloudEnsemble(ACE): A Concept Drift-aware Approach

MSc Research Project
Cloud Computing

Shreya Dhumal
Student ID: 21195773

School of Computing
National College of Ireland

Supervisor: Shreyas Setlur Arun

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Shreya Dhumal
Student ID:	21195773
Programme:	Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Shreyas Setlur Arun
Submission Due Date:	14/12/2023
Project Title:	Improving Dynamic Cloud Workload Prediction and Resource Management with AdaptiveCloudEnsemble(ACE): A Concept Drift-aware Approach
Word Count:	6533
Page Count:	19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Shreya Dhumal
Date:	11th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Improving Dynamic Cloud Workload Prediction and Resource Management with AdaptiveCloudEnsemble(ACE): A Concept Drift-aware Approach

Shreya Dhumal
21195773

Abstract

In the ever-changing landscape of cloud computing, dynamic resource allocation is a crucial challenge, particularly in handling fluctuating workloads and navigating the intricacies of concept drift. Traditional machine learning-based resource allocation methods, while promising, frequently fail when confronted with the dynamic and unexpected nature of real-world cloud workloads. Although ensemble learning has emerged as a potential answer, many existing models are still struggling with real-time adaptation and effective drift management. This research offers the AdaptiveCloudEnsemble (ACE) technique, a new solution for stream management and development of models done using AWS Cloud resources. Within a unified ensemble architecture, the ACE approach integrates many predictive algorithms and advanced drift detection systems. By integrating AWS services for data streaming and analysis, the ACE model not only improves adaptability and accuracy in resource management but also establishes a new benchmark in leveraging cloud infrastructures for solving complex machine learning problems.

[1]

1 Introduction

The introduction of cloud computing has transformed the way data is handled and maintained; yet, the dynamic nature of cloud workloads poses a substantial issue. This research project tackles conventional models' inadequacies in cloud workload prediction and resource management, particularly in dynamic circumstances typified by concept drift, where the statistical features of target variables change with time. It aims to enhance the domain by combining cutting-edge machine learning models with sophisticated concept drift detection approaches to address the unpredictability of workload patterns in cloud environments. It extends the novel approaches to drift detection such as those developed by Barros et al. (2017) and Jin et al. (2023) in evolving data streams and expands on the fundamental work of Hameed et al. (2016) in resource allocation efficiency. The development of the AdaptiveCloudEnsemble (ACE), an ensemble model aimed to improve

[1] Code Artefact: `https://github.com/Shre02/FinalProject`

the adaptability and accuracy of cloud resource management in the presence of concept drift, is central to this research.

The research question posed intends to investigate the effectiveness of ACE in enhancing cloud workload prediction.

Research Question: How can different forecasting ensemble models be incorporated into the AdaptiveCloudEnsemble (ACE) technique to improve predictive resource management in dynamic cloud workloads, handle concept drift, and improve the existing models?

The study is divided into four essential steps to investigate the research question: Stream Processing, Data Preprocessing, Model Development, and Model Testing and Evaluation. In the Stream Processing step, AWS Kinesis Datastreams and Firehose are used in conjunction with an AWS Managed Apache Flink Zeppelin notebook to manage real-time data streaming. AWS Cloud 9 is crucial in developing a Python producer to feed data streams into Kinesis streams, demonstrating a viable use of streaming technologies in cloud environments. The SageMaker Studio Notebook is widely used for data preprocessing and model development.

For these critical stages, the research employs an extensive set of algorithms. The ACE model contains the Adaptive Random Forest Classifier (ARF), which has variants such as ARF-ADWIN and ARF-DDM for adaptive drift detection Gomes et al. (2017), as well as the Streaming Random Patches (SRP) Classifier, which has SRP-ADWIN and SRP-DDM variants Montiel et al. (2020). Furthermore, for its efficiency in handling huge datasets, the ensemble includes the well-known XGBClassifier from the XGBoost framework Chen and Guestrin (2016). The ensemble model incorporates a drift detection technique based on the Drift Detection Method (DDM) **?**, which improves its capacity to adjust to concept drifts in data streams.

Ensemble approaches improve forecast accuracy and robustness by integrating the characteristics of various algorithms, which is especially useful for dynamic and complicated data streams as suggested in a survey by Saxena et al. (2023). By using the distinct characteristics of each method, this technique maximizes overall model performance in cloud workload prediction and management. The ACE model is intended to attain more accuracy and enhanced drift detection skills in similar settings to the method proposed by Wu et al. (2023). The final stage of the project involves examining the proposed model's accuracy, flexibility, and efficiency to traditional models.

This report's structure indicates a methodical approach to researching this topic. Following this introduction, a review of the literature provides a broader background, following the development of cloud workload prediction as well as the role of ensemble learning and concept drift detection. The following parts will go into the methodology, implementation, and rigorous evaluation of the ACE framework, concluding with a discussion of the results and their implications in the broader field of cloud computing.

2 Related Work

The Internet of Things (IoT) and cloud computing landscape is inherently dynamic, characterized by the ever-changing nature of data streams. This environment is particularly challenged by the phenomena of concept drift, where the statistical features of data change with time, having a substantial influence on predictive modeling and workload

management in cloud systems. Concept drift in IoT and cloud computing is more than simply a data characteristic; it is a critical aspect in evaluating the success of predictive analytics. A complete evaluation of learning models under the influence of concept drift was discovered by critically assessing the research of Lu et al. (2019). Their findings highlight the importance of adaptive learning models in IoT environments, where data features can change unexpectedly. Similarly, Ullah and Mahmoud (2020) underlined the dynamic and real-time nature of IoT data, emphasizing the necessity for models that can react to these changes quickly and accurately.

While these studies set the framework for understanding concept drift, they also highlight limitations in present approaches, particularly in dealing with high-volume, high-velocity data streams prevalent in cloud computing. This gap highlights the promising potential of ensemble learning models, which aggregate predictions from numerous models to improve overall performance Lu et al. (2019). With their inherent flexibility and adaptability, ensemble models are well-suited to addressing the issues provided by concept drift in cloud computing systems.

This section lays the groundwork for investigating ensemble learning as a solution to the issues of concept drift, recognizing the need for novel methods that capitalize on the strengths of multiple predictive and drift detection models. It prompts consideration of how the integration of several algorithms within an ensemble framework can improve forecast accuracy and adaptability in the setting of dynamic and complex IoT data streams.

2.1 Concept Drift Detection Techniques

The detection of concept drift in data stream mining, particularly in IoT and cloud computing, is critical for sustaining predictive model accuracy. This field has witnessed a development of solutions, each tackling a different component of drift while simultaneously presenting new obstacles. The survey by Lu et al. (2019) categorizes drift detection approaches into error rate-based, data distribution-based, and ensemble-based strategies. The "Drift Detection Method (DDM)" uses a landmark time frame to detect large changes in the online error rate of base classifiers under error rate-based drift detection Gama et al. (2004). The "PCA-based drift detection" method signals drift in data distribution-based drift detection when the p-value of the generalized Wilcoxon test statistic is sufficiently big Shao et al. (2014). The power of numerous models is harnessed by ensemble-based drift detection methods such as "Accuracy Weighted Ensemble (AWE)" and "Adaptive Random Forest (ARF)" to identify drifts more effectively Gomes et al. (2017). However, real-world datasets pose difficulties: precise times of drifts are frequently unknown, and records may contain a mix of drift types. This uncertainty makes evaluating drift understanding approaches more difficult and may induce biases when comparing machine learning models. Furthermore, defining an accurate and robust dissimilarity measurement for drift detection is a challenge that has yet to be solved.

The study by Suárez-Cetrulo et al. (2023) investigates numerous approaches for detecting drift in real-world datasets. The uncertainty of drift timings, as well as the presence of mixed drift types in datasets, might make evaluating drift detection algorithms more difficult. The Restricted Boltzmann Machine (RBM-IM), introduced by Korycki and Krawczyk (2021), adopts an imbalance-aware loss function to anticipate changes in multi-label online learning environments. Furthermore, Barros et al. (2017)'s Reactive drift detection approach (RDDM) continuously recomputes statistics notifying warnings

and drifts. However, providing an efficient and accurate dissimilarity measurement for drift detection remains a continuous difficulty across various methodologies.

These investigations serve as the foundation for the proposed ACE model. The ACE model attempts to combine advanced drift detection methodologies with ensemble modeling, hence improving adaptability and accuracy in cloud workload forecasts. Wu et al. (2023)'s reference work backs up this method by proving the effectiveness of mixing different models, such as ARF-ADWIN and ARF-EDDM, in an ensemble to address both rapid and gradual drifts. The ACE model demonstrates an advancement in concept drift handling by merging prediction and drift detection methods in an ensemble model, to fill gaps found in previous studies.

2.2 Forecasting Cloud Resources Using Predictor Models

Predictor models are critical in cloud computing for effectively predicting resource utilization in the presence of different service types and dynamic workloads. The development of machine learning-based forecasting models, as investigated by Saxena et al. (2023) in a survey, emphasizes their capacity to harness computing strength, delivering precise predictions critical for optimum resource management in cloud environments. Based on their theoretical concepts and mathematical functioning, the author divided these models into five categories: Evolutionary Neural Network-based prediction models, Deep Learning-based prediction models, Hybrid Learning-based prediction models, Ensemble Learning-based prediction models, and Quantum Learning-based prediction models.

Evolutionary learning, which is inspired by natural evolution, iteratively refines solutions. Kumar and Singh (2018) addressed typical parameter selection issues in evolutionary approaches by combining a neural network with a self-adaptive differential evolution algorithm. Their technique learns parameter values and mutation procedures, increasing the robustness of the solution. It may, however, be sensitive to parameter changes, demanding iterative modification for optimal outcomes.

Deep learning, which employs multi-layered neural networks, has been applied to cloud computing to address straggler workloads, which cause response times to be prolonged. Tuli et al. (2023) developed the Straggler Prediction and Mitigation Technique (START), which uses an Encoder LSTM network to forecast future stragglers, improving SLA adherence and decreasing execution time. While START exceeds many existing approaches, it may neglect task execution time details and may fail to fully incorporate host capabilities, thus leading to errors in results.

Hybrid learning-based prediction models combine various strategies to leverage their strengths while mitigating their drawbacks. To address resource scaling issues in cloud systems, Kardani-Moghaddam et al. (2021) developed the ADRL approach, which combines anomaly detection with deep reinforcement learning. While ADRL improves service quality and stability by managing huge state spaces and assuring quick scaling decisions, combining approaches can bring complications. Furthermore, hybrid models may have higher processing costs, thereby delaying real-time decisions.

Kumar et al. (2020) propose an ensemble learning-based workload forecasting system that employs extreme learning machines and weights their forecasts with a voting engine. This system, which was optimized using a meta-heuristic technique inspired by blackhole theory, significantly reduced resource waste and energy usage, attaining up to a 99.20% improvement in root mean squared error. However, because the model is heuristic, network parameters must be manually selected, indicating a need for automation

improvements. While the technique is advanced, it still faces adaption and automation challenges, highlighting possibilities for additional improvement to increase its practical applicability and efficiency in real-world cloud computing systems.

Quantum Learning models leverage quantum physics to improve computation. Singh et al. (2021) presented an Evolutionary Quantum Neural Network (EQNN) for accurate cloud workload forecasts, which outperformed conventional approaches by up to 91.6%. However, while quantum approaches offer exciting results, they also have their own set of challenges, such as the need for specialist quantum equipment and the possibility of quantum decoherence.

Taking these achievements and limits into account, the proposed ACE model in this study employs ensemble learning because of its capacity to combine different models, resulting in higher accuracy and resilience against overfitting. This technique is consistent with Wu et al. (2023)'s findings, which show the usefulness of ensemble models such as ARF-ADWIN and ARF-EDDM in IoT data stream analytics. The ACE model aims to enhance these techniques to provide a more balanced, accurate, and efficient solution to cloud computing resource forecasts.

2.3 Ensemble Learning Approaches for Cloud Workload Prediction

Ensemble learning has emerged as a major approach for improving workload forecast accuracy in the evolving cloud computing ecosystem. Researchers have delved into the complexities of cloud computing resource prediction, resulting in the development of several ensemble models. One such study by Wang et al. (2021). uses the capability of ensemble learning to improve computing resource estimates in cloud environments. They combine models such as support vector machine, decision tree, and k-nearest neighbor, and they improve the ensemble by utilizing an Accuracy and Relative Error-based Pruning approach. To overcome feature redundancy, a forward search Feature Selection method is added, with a focus on impacting features. However, challenges with managing feature redundancy and ensuring consistent performance across varied cloud applications persist.

Several novel approaches have been proposed in the field of ensemble modeling to solve the issues of accurate prediction and optimization. The "Ensemble Learning-based Approach for Predicting (ELAP)" was introduced by Xiao et al. (2022), with the goal of predicting CPU utilization in cloud systems. Given the huge volatility of cloud data, the strategy mixes numerous regression models to improve prediction accuracy. The ensemble technique successfully mitigates individual model biases and provides a more robust prediction. However, its key weakness is its possible inability to adapt to various workloads beyond the single Cybershake dataset on which it was evaluated. Another study by Rahmanian et al. (2018) describes an ensemble cloud resource utilization prediction system based on the Learning Automata (LA) theory. This method integrates different cutting-edge prediction models, weighting individual constituent models based on their performance. While this method provides a dynamic mechanism to alter weights and increase forecast accuracy, it may face difficulties adjusting to rapidly changing cloud workloads, particularly when the variability is significant. Kaur et al. (2019) delves into the "Intelligent Regressive Ensemble Approach for Prediction (REAP)," which focuses on predicting cloud computing resource utilization. REAP delivers great performance by combining feature selection and resource usage prediction algorithms. A major part of this approach is the employment of a meta-heuristic, the Genetic Algorithm, for feature

selection. However, the method's narrow focus on the Cybershake dataset on a workflow management system may cause difficulties in more general applications.

While ensemble modeling improves prediction accuracy and resilience across all methodologies, the difficulty of assuring flexibility to various and dynamic workloads remains a major concern. Hence, to improve the adaptability and accuracy of prediction models, addressing the phenomena of concept drift, where data distributions vary over time, is critical, necessitating specialized drift detection strategies to preserve model efficacy.

2.4 Ensemble Predictors with Drift Detection Techniques

The combination of drift detection approaches and ensemble classifiers improves the accuracy of cloud computing workload forecasts. Ensemble classifiers, by definition, combine the strengths of numerous basic classifiers, resulting in a more comprehensive approach to prediction. These classifiers, when combined with drift detection, can quickly adjust to changes in data distribution, ensuring constant performance even in dynamic contexts. This collaboration not only improves forecast accuracy but also ensures that the model remains relevant over time, accommodating changing workloads and system dynamics. D and Prem M (2023) provides a unique ensemble approach to concept drift in non-stationary environments that combines several drift detection approaches with ensemble predictors. This method combines the benefits of drift detection and ensemble prediction to achieve greater accuracy and versatility. However, its complications can cause problems in real-time applications. Furthermore,Martínez Pérez et al. (2021)'s study emphasizes the importance of diversity in base learners while dealing with sudden and gradual concept drifts. Their proposed improvements to current ensemble algorithms demonstrated improved performance in various concept drift scenarios by changing voting procedures and adding heterogeneous base learners. However, striking the proper balance between diversity and accuracy remains critical.

Wu et al. (2023)'s reference article effectively demonstrates the utilization of multiple models in an ensemble, such as ARF-ADWIN, ARF-EDDM, OPA, and KNN-ADWIN, to control both sudden and gradual drifts. However, it highlights a significant disadvantage: post-drift detection, the model's accuracy tends to decline due to the fluctuating nature of data and the lack of retraining mechanisms to adjust to these drifts. This restriction highlights the difficulty of maintaining constant accuracy in the dynamic data environment.

In this project, the ACE Model skillfully incorporates complex algorithms such as the Adaptive Random Forest Classifier (ARF), Streaming Random Patches (SRP) Classifier, and XGBClassifier from the XGBoost framework. ARF, including ARF-ADWIN and ARF-DDM versions, and SRP, including SRP-ADWIN and SRP-DDM, provide concept drift flexibility, which is critical in dynamic cloud environments. XGBoost is included for the efficiency in processing massive data sets. The model also contains the Drift Detection Method (DDM) for detecting and adapting to concept drifts in real-time. Notably, the ACE Model is set up to retrain components such as the XGBClassifier after drift detection, ensuring constant adaptation to workload changes. This strategy ensures that the model remains adaptable to changes in workload, retaining overall accuracy and effectiveness in real-time cloud computing settings striving for improved results.

3 Methodology

This research methodology encompasses a disciplined and methodical approach to addressing the difficulties of cloud workload prediction and concept drift. The project's fundamental component is the development and testing of a novel predictive model called the AdaptiveCloudEnsemble (ACE), which employs advanced ensemble learning algorithms and drift detection strategies. Data collection, preprocessing, model creation, and evaluation are all independent but interrelated stages of the methodology implemented using various resources in the Amazon Web Service(AWS) cloud to simulate the real-world cloud environment. This methodical flow not only assures that the project adheres to scientific standards, but it also allows for a complete understanding of the model's efficacy in real-world cloud computing scenarios.

3.1 Data Collection and Preprocessing

The IoT2020 dataset is an important part of our research because it provides a comprehensive and accurate depiction of IoT network behavior, including both neutral and malicious behaviors. This dataset, as described by Ullah and Mahmoud (2020), is a comprehensive collection of data meant to capture the complexities of IoT network activity and malicious behavior, containing 83 network features and three label features that capture various IoT attack forms and behaviors. Because of its precise details and real-world data qualities, it is an excellent proxy for analyzing the volatile nature of cloud data.

The IoT2020 dataset in this research enables an accurate simulation of the fluctuating and dynamic conditions found in cloud environments. The dataset's diversity in network traffic patterns and attack types accurately reflects the unpredictable and dynamic nature of cloud data, making it an excellent tool for researching concept drift. With its broad and increasing data properties, the IoT2020 dataset provides the appropriate context for exploring and learning about these drifts in a controlled yet realistic manner.

The IoT2020 dataset data was given as input into AWS Cloud 9 using Python code and then delivered to AWS Kinesis streams. This configuration enabled real-time data handling, similar to live cloud data streams. Following that, the data was processed by Kinesis Firehose and an AWS-managed Apache Flink Zeppelin notebook before being stored in parquet format in AWS S3 buckets. This process was handled by AWS SageMaker Studio Notebook, and comprised critical preparation procedures such as data cleaning, normalization, and feature selection, ensuring that the data was ready for the next stages of model creation and analysis.

3.2 Techniques and Scenario Set-Up

- **Model Development**: The research centered on the development of the ACE model, an ensemble learning framework. This model incorporates algorithms such as XGBoost, AdaptiveRandomForest, and SRP, which were chosen for their proven effectiveness in dealing with massive amounts of data and adaptability to concept drift.

- **Drift Detection:** To identify and respond to concept drift in data streams, the ACE model includes the DDM drift detector, ensuring the model's adaptability and accuracy over time.

- **Scenario Execution:**To evaluate the model's performance in real-world scenarios, it was tested in simulated cloud computing settings. To measure the model's responsiveness, multiple workload patterns were created and concept drifts were introduced.

3.3 Data Analysis and Statistical Techniques

- **Analysis Process:** The raw data from the IoT2020 dataset was thoroughly analyzed to assess the ACE model's performance in predicting cloud workloads, particularly under conditions of concept drift. This examination entailed applying the proposed and previously developed models to the preprocessed dataset and monitoring their responsiveness, adaptability to changes in data patterns over time, and changes in evaluation metrics.

- **Statistical Techniques:** Several primary statistical indicators Powers and Ailab (2011) were used to quantify and assess the performance of the ACE model:

  Accuracy: The proportion of true results (including true positives and true negatives) among the total number of instances studied is measured by this statistic. It is a key indicator of the model's overall accuracy. The accuracy formula is as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

  where TP, TN, FP, and FN stand for true positive, true negative, false positive, and false negative, respectively.
  Precision: The fraction of true positive predictions among all positive predictions made by the model is reflected in precision. It is especially important in situations where the cost of false positives is large. Precision is calculated as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

  Recall (Sensitivity): Recall assesses a model's ability to correctly detect actual positives in data. It is especially critical in instances where missing a true positive is costly. The recall formula is as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

  F1-Score: The F1-score is a harmonic mean of precision and recall that balances the two measures. It is especially effective when the distribution of classes is uneven. The F1-score is determined as follows:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

  These measures were chosen for their potential to provide a thorough picture of the model's performance in a variety of areas, ranging from accuracy to precision-to-recall balance.

4 Design Specification

The architecture of this project has been meticulously designed to handle cloud workload prediction, with an extensive flow that covers from initial data ingestion to final model building and evaluation. The architecture is based on two major data pipelines: the Upward Pipeline and the Downward Pipeline. The Upward Pipeline handles data ingestion and early storage, while the Downward Pipeline handles data transfer and final storage, completing the flow with data preparation and ACE model generation. This design, as indicated in the below figure 1, guarantees a continuous and efficient flow of data from initial collection to final processing and storage, laying the groundwork for advanced model creation and analysis.

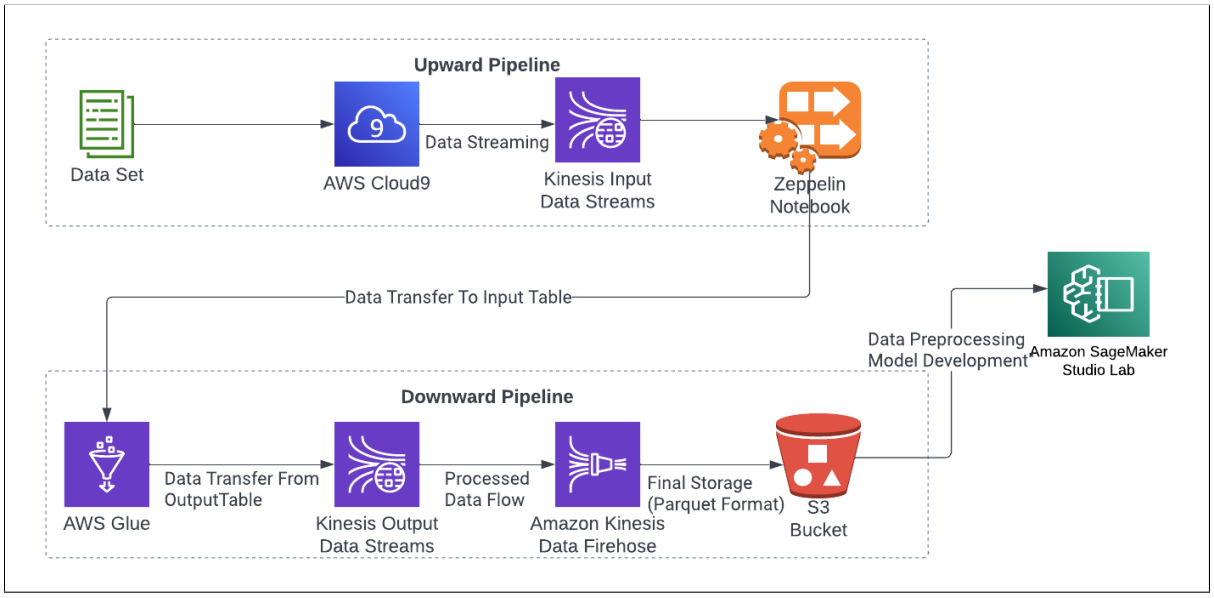

Figure 1: System Architecture

4.1 Upward Pipeline(Data Ingestion and Initial Storage):

- **Ingestion and Streaming**: The Upward Pipeline begins with data streaming from AWS Cloud 9, a cloud-based IDE, where a Python script feeds data into the AWS Kinesis Datastreams input stream. This configuration mimics real-time data collected from the IoT2020 dataset.

- **Temporary Storage in AWS Glue:** Data from the input stream is then routed to an AWS Glue table via the AWS Managed Apache Flink Zeppelin notebook, a web-based notebook that enables interactive data analytics with Flink, excellent for real-time processing and manipulation of streaming data. As a fully managed extract, transform, and load (ETL) service, AWS Glue streamlines data preparation for analytics. This step prepares the data for further processing by providing a structured and temporary storage solution.

4.2 Downward Pipeline(Data Transfer and Final Storage):

- **Data Transfer to Output Stream:** The data is transported from the input AWS Glue table to an output Glue table in the Downward Pipeline. This transfer is essential for separating and managing data flow within the system.

- **Delivery to AWS S3:** The data from the output Glue table is then routed into the AWS Kinesis Datastreams output stream. This output stream is linked to the AWS Kinesis Firehose delivery stream, which is a fully managed service for the reliable loading of streaming data into data stores and analytics tools. The data processing is completed by converting it to parquet format and transferring it to an AWS S3 bucket for long-term storage.

4.3 AWS SageMaker Studio Notebook for Data Preprocessing and Model Development:

Following data storage in AWS S3, the AWS SageMaker Studio Notebook is used for following data preparation and model building phases. This advanced tool efficiently manipulates and transforms stored data, including cleaning, normalization, and feature selection, preparing it for usage in machine learning models. SageMaker Studio Notebook offers a flexible environment for developing, training, and evaluating the ACE model, using the integrated algorithms and assuring the model's correctness and adaptability in anticipating cloud workloads and dealing with concept drift.

The project's dual-pipeline architecture efficiently manages real-time data streams by combining the Upward Pipeline for data ingestion with the Downward Pipeline for processing and storage. This structure connects AWS services from Cloud 9 to S3 using Kinesis Datastreams and Glue tables, allowing for model construction and analysis. The usage of AWS SageMaker Studio Notebook for data preprocessing and model construction after that ensures a streamlined workflow, allowing for a comprehensive approach to cloud workload prediction from initial data collecting to final model evaluation.

5 Implementation

5.1 Adaptive Cloud Ensemble(ACE) Model

In this research, the ACE model is a powerful ensemble learning system developed specifically for cloud workload prediction in dynamic settings with concept drift. The ensemble is composed of three fundamental machine learning algorithms: Adaptive Random Forest Classifier (ARF), Streaming Random Patches (SRP) Classifier, and XGBClassifier from XGBoost.

A. Ensemble Structure and Functionality:

- **Adaptive Random Forest Classifier(ARF):** ARF-ADWIN and ARF-DDM are two variants of the Adaptive Random Forest Classifier (ARF) Gomes et al. (2017). In this implementation, each variant has three models.ARF responds to changes in data streams effectively, making it suited for contexts with concept drift.The ARF-DDM variant incorporates the Drift Detection Method (DDM), which improves its capacity to respond to data drifts.

- **Streaming Random Patches (SRP) Classifier:** The SRP Classifier is implemented in two variants: SRP-ADWIN and SRP-DDM. Each variation is made up of three separate models, for a total of six SRP models in the ensemble Montiel et al. (2020). SRP-ADWIN detects changes using the ADWIN (ADaptive WINdowing) technique. ADWIN is a drift detection approach that automatically modifies the size of the sliding window, allowing the model to effectively respond to changes in the data streamBifet and Gavaldà (2007). SRP-DDM incorporates the Drift Detection Method (DDM), offering a tool for more accurately identifying and reacting to idea drift. This version is very good at detecting changes in data distribution, allowing for rapid adjustments in the ensemble's decision-making process. With its ensemble method, the SRP Classifier aggregates numerous weak learners to produce a more powerful predictive model. This aggregation improves the ACE model's overall forecast accuracy and stability.

- **XGBClassifier(XGBoost):** XGBoost excels in complicated prediction jobs by providing both speed and accuracy. Its powerful boosting techniques are especially useful for huge and complex datasets like those found in cloud workload predictions Chen and Guestrin (2016). The retraining technique is initiated by drift detection signals generated by the ARF-DDM and SRP-DDM models. When these models detect a major shift in data distribution, XGBoost is updated with novel data, allowing it to re-calibrate its forecasting method. This dynamic retraining procedure ensures that the XGBoost model remains relevant and accurate even when the input data characteristics change over time. XGBoost adds depth to the ensemble's predictions, particularly in finding and responding to complex data patterns. It adds to the adaptability of ARF and SRP models, resulting in a balanced and comprehensive prediction output.

B. Concept Drift Adaptation:

- **Real-Time Base Learner Monitoring:** The ACE approach involves continuous monitoring of each base learner's accuracy. This constant evaluation is critical for recognizing any differences in prediction performance that may suggest concept drift in the data stream.

- **Detecting and Responding to Drifts:**The DDM included inside the ARF-DDM and SRP-DDM models is largely used for drift detection. These models are sensitive to changes in data distribution, allowing for the early detection of concept drifts. When a drift is detected, the model initiates a retraining process for the XGBoost classifier. This retraining entails feeding the most recent data into the XGBoost model, allowing it to recalibrate its predictions based on the new data patterns.

- **Maintaining Predictive Accuracy:** The combination of real-time monitoring, drift detection, and adaptive retraining results in a comprehensive strategy for maintaining the accuracy of the ACE model over time. By combining these strategies, the ACE model can not only detect idea drifts but also dynamically adapt its predictive strategy to retain relevance and accuracy in a constantly changing data environment.

C. Weighted Aggregation of Predictions: The ACE model modifies the weighting of each base learner's predictions in response to concept drift. The weights are inversely

```
  # Function to check for drift and retrain XGBoost if needed
  def check_and_retrain_xgboost():
      nonlocal hat5
      in_drift1, in_warning1 = hat3.drift_detector.update(metric3.get())
      in_drift2, in_warning2 = hat4.drift_detector.update(metric4.get())

      if in_drift1 or in_drift2:
          # Retrain XGBoost on recent data
          X_recent, y_recent = np.array(recent_data['X']), np.
↪array(recent_data['y'])
          hat5.fit(X_recent, y_recent)

  # Initialize a buffer for recent data to retrain XGBoost
  recent_data = {'X': [], 'y': []}
  recent_data_size = 500  # Adjust the size based on your data and␣
↪requirements
```

Figure 2: Retrain XgBoost)

proportional to their real-time error rates, guaranteeing that more accurate models have a higher influence on the ensemble's ultimate forecast at any given time. This dynamic weighting method ensures that the ensemble's forecasts stay balanced and accurate as the data stream evolves.

```
      # Calculate the real-time error rates of the base learners
      e1 = 1 - metric1.get()
      e2 = 1 - metric2.get()
      e3 = 1 - metric3.get()
      e4 = 1 - metric4.get()

      ep = 0.001  # The epsilon used to avoid dividing by 0
      # Calculate the weight of each base learner by the reciprocal of its␣
↪real-time error rate
      ea = 1 / (e1 + ep) + 1 / (e2 + ep) + 1 / (e3 + ep) + 1 / (e4 + ep)
      w1 = 1 / (e1 + ep) / ea
      w2 = 1 / (e2 + ep) / ea
      w3 = 1 / (e3 + ep) / ea
      w4 = 1 / (e4 + ep) / ea

      # Calculate the final probabilities of classes 0 & 1 to make predictions
      y_prob_0 = w1 * y_prob1.get(0, 0) + w2 * y_prob2.get(0, 0) + w3 *␣
↪y_prob3.get(0, 0) + w4 * y_prob4.get(0, 0)
      y_prob_1 = w1 * y_prob1.get(1, 0) + w2 * y_prob2.get(1, 0) + w3 *␣
↪y_prob3.get(1, 0) + w4 * y_prob4.get(1, 0)

      y_pred = 0 if y_prob_0 > y_prob_1 else 1

      # Update the real-time accuracy of the ensemble model
      metric = metric.update(yi, y_pred)

      t.append(1)
```

Figure 3: Weighted Aggregation of Predictions)

5.2 Implementation Process

The ACE model's implementation was a multifaceted procedure that included not only the coding of each base learner but also their integration into an effective ensemble.

A. AWS Resource Setup & Configuration:

- **AWS Resource Setup:** The first stage was to configure several AWS services such as Cloud 9, Kinesis Datastreams, Firehose, and S3. This configuration was crucial for establishing a continuous pipeline from data streaming to storage.

- **Connecting AWS Services**: To ensure a smooth data flow, each service was methodically connected. For example, connect Cloud 9 to Kinesis Datastreams for real-time data streaming and then to S3 via Kinesis Firehose for data storage.

- **Permissions and IAM Policies:** Adequate permissions were granted using IAM (Identity and Access Management) policies. This step was critical for ensuring secure and efficient access to AWS resources, allowing each service to interact as needed without exposing itself to security threats.

B. Ensemble Integration and Coding:

- **Base Learner Development:** Each base learner - ARF, SRP, and XGBoost - was coded individually. Their configuration was given special consideration to optimize them for the task of cloud workload prediction.

- **Integrating Base Learners in the Ensemble:** The integration of these key learners was at the center of the ACE approach. To ensure that each learner effectively contributed to the ensemble's overall performance, careful coding for weighted aggregation of predictions was required.

C. Addressing Challenges:

- **Tuning parameters:** Fine-tuning the parameters for each base learner was a major issue. Iterative testing and modifications were used to get the appropriate settings for the best predictive performance.

- **Drift Detection and Response:** Another key challenge was ensuring adequate drift detection and response. This necessitated not only the right implementation of drift detection algorithms within the ARF and SRP models but also the efficient retraining of the XGBoost model upon drift detection.

D. Data Integration and Workflow Execution:

- **Streaming Data to AWS Kinesis Datastreams:** A A Python script was written to stream the IoT2020 dataset to AWS Kinesis Datastreams using AWS Cloud 9. This script connected to the iot-input-stream in the eu-west-1 region, received data from 'IoT_2020_b_0.01_fs.csv', converted each row to JSON, and then transmitted these JSON records to the Kinesis stream as individual entries. For effective distribution within the stream, each record was allocated a partition key, and the script offered feedback by displaying the sequence number of each transmitted record.

- **Data Processing and Preprocessing in AWS SageMaker Studio Notebook:** SageMaker Studio Notebook was used to access data stored in S3. Dask, which is effective for handling massive datasets, was used to read the data, which was initially in parquet format in the S3 bucket (iotdestbucket). For easy handling and preprocessing, the Dask dataframe was transformed into a Pandas dataframe. Preprocessing methods included normalization and feature selection to prepare the data for model training.

E. Iterative Model Training Using Stream Data:
The ACE model was trained using a real-time, iterative approach suitable for streaming data. Data was formatted appropriately for the model, with X_train as features and

y_train as labels. The training involved iterating over the dataset using the stream.iter_pandas method from the river library Montiel et al. (2020), processing data in a streaming fashion akin to real-world environments. Each iteration involved feeding a single data point (xi1) and its label (yi1) into the model using the learn_one method. This online learning approach allowed the model to adjust and learn incrementally, continuously updating its strategy to effectively handle concept drift and maintain accuracy.

F. Tools and Technologies Used

- **Programming Language and Libraries:**

  Python: Python was chosen as the primary programming language due to its versatility and depth of support in data science and machine learning.

  River Library: This Python library was critical for implementing the ACE model's ensemble learning components, particularly its skills in online learning and drift detection.

  Scikit-learn (sklearn.metrics): This library was used for performance evaluation and provided crucial tools for calculating metrics such as accuracy, precision, recall, and F1-score, which provided insights into the model's usefulness.

- **AWS Tools Suite:**

  AWS Cloud 9: Used as an integrated development environment for building and testing the data streaming Python application.

  AWS Kinesis Datastreams: Played an important role in real-time data handling, allowing the IoT2020 dataset to be streamed into the system.

  AWS Kinesis Firehose: Made it easier to send streaming data to AWS S3 for storage.

  AWS S3: Served as the primary data storage option, holding processed and transformed data.

  AWS Glue: Used for temporary data storage and management, notably between streaming and eventual storage.

  AWS SageMaker Studio Notebook: Used to preprocess data and train the ACE model, as well as to provide a powerful environment for machine learning activities.

These tools and technologies served as the project's backbone, each contributing to different parts of data management, model creation, and performance evaluation. The entire development process was iterative, with cycles of testing, feedback, and refinement repeated. This method enabled the detection and correction of errors in real-time, resulting in a robust and efficient ACE model.

6 Evaluation

The purpose of the research project is to determine the efficacy of combining several forecasting ensemble models into the AdaptiveCloudEnsemble (ACE) approach. The research specifically seeks to analyze the ACE model's capabilities in improving predictive resource management for dynamic cloud workloads, as well as its ability to address the

issues posed by concept drift. By comparing the ACE model's performance to that of established forecasting models, the project aims to validate its improvements in adaptability and predicted accuracy under changing data situations.

6.1 Analysis of Results

The Analysis of Results shown in the below Table highlights the ACE Model's outstanding performance highlighted by its dynamic adaptability and refined algorithm integration. It outperformed the ARF-ADWIN model by achieving a more precise balance between precision and recall, demonstrating its ability to reduce false positives while maintaining high accuracy. While commendable, the ARF-DDM was unable to match the ACE Model's precision, which is crucial in avoiding false alarms and retaining trust in forecast accuracy.

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
ARF-ADWIN	98.05%	98.59%	99.43%	98.97%
ARF-DDM	98.68%	98.97%	99.64%	99.3%
HT	95.45%	95.91%	99.42%	97.63%
LB	97.8%	98.25%	99.43%	98.83%
PWPAE	99.02%	98.99%	99.98%	99.48%
ACE	**99.2%**	**99.2%**	**99.96%**	**99.58%**

Table 1: Comparative Performance of Ensemble Models

The Hoeffding Tree model, while strong in recall, lagged in precision and accuracy, indicating that the ACE Model handles various data patterns better. Similarly, despite its robustness, the Leveraging Bagging model could not match the ACE Model's high recall rates, owing to the latter's complete detection of true positives. The PWPAE model performed well in terms of accuracy and recall, but it was outscored by the ACE Model in terms of precision and F1 score. This minor advantage reflects the ACE Model's improved ability to provide reliable and consistent predictions.

The ACE Model's improved efficiency can be explained by its dynamic retraining capabilities in response to concept drift, which ensures it stays updated with changing data trends. The ACE Model's versatility, along with a weighted methodology that leverages the capabilities of various algorithms, enables it to maintain high accuracy and precision even in changing data settings. Its outstanding performance highlights the model's potential as a dependable, accurate tool for predictive resource management in cloud computing, establishing a new benchmark in the field.

6.2 Graphical Analysis of the Results

The graph 4 depicts the comparative performance of various ensemble models over a series of samples in an appealing visual manner. It measures the accuracy percentage as a function of sample count, providing a clear picture of each model's behavior in response to concept drift, as demonstrated by the vertical lines labeled "Drift 1" and "Drift 2."
Initial Performance Dip: At the beginning of the graph, all models have a drop in accuracy, which corresponds to the first concept drift. This is a normal occurrence as models react to the abrupt shift in the statistical features of the data.
Sustained Performance Post-Drift: The ACE Model continuously outperforms the

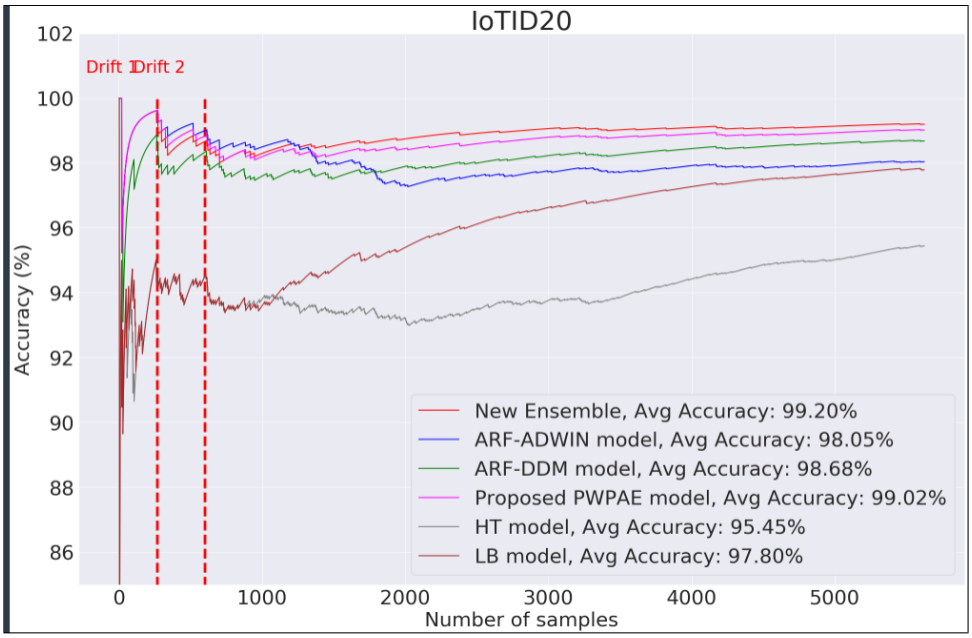

Figure 4: Graphical Representation of Results

other models in terms of post-drift accuracy. Notably, the ACE Model's performance stays consistent even after the second drift, demonstrating its ability to effectively handle concept drift.

Comparison to Other Models: The proposed PWPAE model has the same accuracy as the ACE Model. The ACE Model, on the other hand, edges ahead slightly, implying a more nuanced treatment of concept drift. The ARF-DDM model closely follows, suggesting efficacy but with slightly lower average accuracy. The HT and LB models lag behind, showing that while they are successful, they may not be as adept at dealing with quick changes in data.

6.3 Discussion

The ACE Model represents a considerable advance in the management of rapidly changing cloud computing workloads, a concept that is well supported by current academic research. The model's design cleverly blends numerous forecasting methodologies, which has been proved to outperform single prediction models. Interestingly, when new data trends arise, the ACE Model's performance drops marginally, most likely because the model's complex structure needs time to respond to these changes. However, after adapting to new conditions, it immediately outperforms previous models, maintaining superior accuracy over time.

While the ACE Model represents a breakthrough in predictive modeling, the architecture of the system from data ingestion to analysis needs particular concern. The seamless integration of AWS components - from pushing data to Kinesis Streams to storing it in S3, and finally analyzing it in SageMaker Studio - demonstrates a well-executed information flow. However, there is a possibility for improvement in the process. For example, automating model parameter tuning could improve efficiency, and broadening the range of data scenarios could improve the model's robustness. Automation in the early stages of data streaming and preprocessing could reduce time-to-insight while diversifying datasets

in SageMaker could test and perhaps improve the model's adaptability to diverse cloud environments. Such improvements could strengthen the system's design, ensuring that the model not only learns from data but also does so with greater speed and precision. In conclusion, the ACE Model represents a significant advancement in predictive modeling for dynamic cloud systems, with future modifications highlighted to improve adaptability and accuracy.

7 Conclusion and Future Work

This research project aimed to investigate the integration of several forecasting ensemble models into the AdaptiveCloudEnsemble (ACE) approach, to improve predictive resource management in dynamic cloud workloads and successfully deal with concept drift. The research was met with a thorough design and rigorous testing, demonstrating the ACE Model's superior performance, notably in terms of adaptability and response to concept drift when compared to established models. The major findings show that the ACE Model not only manages the initial impact of concept drift well but also excels at sustaining high accuracy over time, outperforming other models in sustained predictive performance. This demonstrates the model's efficacy and the project's success in meeting its goals.

However, the research recognizes the constraints of relying on a single dataset and manually adjusting the model's parameters. An investigation into automated hyperparameter optimization would be a significant next step in future research, potentially improving the model's efficiency and performance. Furthermore, expanding the current work to incorporate a broader range of datasets with various features could give a more thorough validation of the model's applicability across multiple cloud environments. Introducing AWS Lambda functions could enable seamless interconnection of services, automating data stream flow and better simulating a live streaming environment. Such automation would not only improve the system's efficiency and responsiveness, but it would also potentially raise the model's responsiveness and efficiency.

In the future, the ACE Model holds significant commercial potential for real-time cloud workload management. Deploying the model into live cloud systems to evaluate real-world performance and scalability could reinforce its practical applicability and pave the way for innovative industry applications. The transition from a research prototype to a commercial solution has the potential to transform cloud computing resource management, marking a significant advancement in the area.

References

Barros, R., Cabral, D., Alves, P. and Santos, S. (2017). Rddm: Reactive drift detection method, *Expert Systems with Applications* **90**: 344–355.

Bifet, A. and Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing, Vol. 7.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, Association for Computing Machinery, New York, NY, USA, p. 785–794.
URL: *https://doi.org/10.1145/2939672.2939785*

D, P. and Prem M, V. (2023). A novel ensemble classifier framework to preprocess, learn and predict imbalanced heterogeneous drifted data stream, *2023 Second International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT)*, pp. 1–7.

Gama, J., Medas, P., Castillo, G. and Rodrigues, P. (2004). Learning with drift detection, Vol. 8, pp. 286–295.

Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfahringer, B., Holmes, G. and Abdessalem, T. (2017). Adaptive random forests for evolving data stream classification, *Machine Learning* **106**: 1–27.

Hameed, A., Khoshkbarforoushha, A., Ranjan, R., Jayaraman, P. P., Kolodziej, J., Balaji, P., Zeadally, S., Malluhi, Q. M., Tziritas, N., Vishnu, A., Khan, S. U. and Zomaya, A. (2016). A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems, *Computing* **98**: 751–774.
URL: *https://doi.org/10.1007/s00607-014-0407-8*

Jin, D., Hao, L. and Guoyin, W. (2023). Sample selection method for concept drift, *2023 8th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, pp. 402–406.

Kardani-Moghaddam, S., Buyya, R. and Ramamohanarao, K. (2021). Adrl: A hybrid anomaly-aware deep reinforcement learning-based resource scaling in clouds, *IEEE Transactions on Parallel and Distributed Systems* **32**(3): 514–526.

Kaur, G., Bala, A. and Chana, I. (2019). An intelligent regressive ensemble approach for predicting resource usage in cloud computing, *Journal of Parallel and Distributed Computing* **123**: 1–12.
URL: *https://www.sciencedirect.com/science/article/pii/S0743731518306063*

Korycki, L. and Krawczyk, B. (2021). Concept drift detection from multi-class imbalanced data streams, *2021 IEEE 37th International Conference on Data Engineering (ICDE)* pp. 1068–1079.
URL: *https://api.semanticscholar.org/CorpusID:233324515*

Kumar, J. and Singh, A. K. (2018). Workload prediction in cloud using artificial neural network and adaptive differential evolution, *Future Generation Computer Systems* **81**: 41–52.
URL: *https://www.sciencedirect.com/science/article/pii/S0167739X17300444*

Kumar, J., Singh, A. K. and Buyya, R. (2020). Ensemble learning based predictive framework for virtual machine resource request prediction, *Neurocomputing* **397**: 20–30.
URL: *https://www.sciencedirect.com/science/article/pii/S0925231220301892*

Lu, J., Liu, A., Dong, F., Gu, F., Gama, J. and Zhang, G. (2019). Learning under concept drift: A review, *IEEE Transactions on Knowledge and Data Engineering* **31**(12): 2346–2363.

Martínez Pérez, J. L., Palomino Mariño, L. M. and Maior De Barros, R. S. (2021). Improving diversity in concept drift ensembles, *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8.

Montiel, J., Halford, M., Mastelini, S. M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H. M., Read, J., Abdessalem, T. and Bifet, A. (2020). River: machine learning for streaming data in python, *ArXiv* **abs/2012.04740**.
URL: *https://api.semanticscholar.org/CorpusID:228063845*

Powers, D. and Ailab (2011). Evaluation: From precision, recall and f-measure to roc, informedness, markedness correlation, *J. Mach. Learn. Technol* **2**: 2229–3981.

Rahmanian, A. A., Ghobaei-Arani, M. and Tofighy, S. (2018). A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment, *Future Generation Computer Systems* **79**: 54–71.
URL: *https://www.sciencedirect.com/science/article/pii/S0167739X17309378*

Saxena, D., Kumar, J., Singh, A. K. and Schmid, S. (2023). Performance analysis of machine learning centered workload prediction models for cloud, *IEEE Transactions on Parallel and Distributed Systems* **34**: 1313–1330.
URL: *https://doi.org/10.1109/TPDS.2023.3240567*

Shao, J., Ahmadi, Z. and Kramer, S. (2014). Prototype-based learning on concept-drifting data streams, *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, Association for Computing Machinery, New York, NY, USA, p. 412–421.
URL: *https://doi.org/10.1145/2623330.2623609*

Singh, A. K., Saxena, D., Kumar, J. and Gupta, V. (2021). A quantum approach towards the adaptive prediction of cloud workloads, *IEEE Transactions on Parallel and Distributed Systems* **32**(12): 2893–2905.

Suárez-Cetrulo, A. L., Quintana, D. and Cervantes, A. (2023). A survey on machine learning for recurring concept drifting data streams, *Expert Systems with Applications* **213**: 118934.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417422019522*

Tuli, S., Gill, S. S., Garraghan, P., Buyya, R., Casale, G. and Jennings, N. R. (2023). Start: Straggler prediction and mitigation for cloud computing environments using encoder lstm networks, *IEEE Transactions on Services Computing* **16**(01): 615–627.

Ullah, I. and Mahmoud, Q. (2020). A scheme for generating a dataset for anomalous activity detection in iot networks, pp. 508–520.

Wang, S., Zhu, F., Yao, Y., Tang, W., Xiao, Y. and Xiong, S. (2021). A computing resources prediction approach based on ensemble learning for complex system simulation in cloud environment, *Simulation Modelling Practice and Theory* **107**: 102202.
URL: *https://www.sciencedirect.com/science/article/pii/S1569190X20301416*

Wu, Y., Liu, L., Yu, Y., Chen, G. and Hu, J. (2023). An adaptive ensemble framework for addressing concept drift in iot data streams.

Xiao, Y., Yao, Y., Chen, K., Tang, W. and Zhu, F. (2022). A simulation task partition method based on cloud computing resource prediction using ensemble learning, *Simulation Modelling Practice and Theory* **119**: 102595.
URL: *https://www.sciencedirect.com/science/article/pii/S1569190X22000818*