



**ASTES**

# **Advances in Science, Technology & Engineering Systems Journal**

---

**VOLUME 9-ISSUE 3 | MAY-JUN 2024**

**[www.astesj.com](http://www.astesj.com)**

**ISSN: 2415-6698**

## EDITORIAL BOARD

### Editor-in-Chief

**Prof. Passerini Kazmerski**  
University of Chicago, USA

### Editorial Board Members

**Dr. Jiantao Shi**  
Nanjing Research Institute  
of Electronic Technology,  
China

**Dr. Tariq Kamal**  
University of Nottingham, UK  
Sakarya University, Turkey

**Dr. Hongbo Du**  
Prairie View A&M University, USA

**Dr. Nguyen Tung Linh**  
Electric Power University,  
Vietnam

**Prof. Majida Ali Abed  
Meshari**  
Tikrit University Campus,  
Iraq

**Dr. Mohmaed Abdel Fattah Ashabrawy**  
Prince Sattam bin Abdulaziz University,  
Saudi Arabia

**Mohamed Mohamed  
Abdel-Daim**  
Suez Canal University,  
Egypt

**Dr. Omeje Maxwell**  
Covenant University, Nigeria

**Mr. Muhammad Tanveer Riaz**  
School of Electrical Engineering,  
Chongqing University, P.R. China

**Dr. Heba Afify**  
MTI University of Genoa,  
Italy

**Mr. Randhir Kumar**  
National University of  
Technology Raipur, India

**Dr. Serdar Sean Kalaycioglu**  
Toronto Metropolitan University, Canada

**Dr. Daniele Mestriner**  
University of Genoa, Italy

**Ms. Nasmin Jiwani**  
University of The  
Cumberlands, USA

**Dr. Umurzakova Dilnozaxon  
Maxamadjanovna**  
University of Information Technologies,  
Uzbekistan

**Dr. Pavel Todorov Stoyanov**  
Technical University of Sofia, Bulgaria

### Regional Editors

**Dr. Hung-Wei Wu**  
Kun Shan University,  
Taiwan

**Dr. Maryam Asghari**  
Shahid Ashrafi Esfahani,  
Iran

**Dr. Shakir Ali**  
Aligarh Muslim University, India

**Dr. Ahmet Kayabasi**  
Karamanoglu Mehmetbey  
University, Turkey

**Dr. Ebubekir Altuntas**  
Gaziosmanpasa University,  
Turkey

**Dr. Sabry Ali Abdallah El-Naggar**  
Tanta University, Egypt

**Mr. Aamir Nawaz**  
Gomal University, Pakistan

**Dr. Gomathi Periasamy**  
Mekelle University, Ethiopia

**Dr. Walid Wafik Mohamed Badawy**  
National Organization for Drug Control and  
Research, Egypt

**Dr. Abhishek Shukla**  
R.D. Engineering College,  
India

**Mr. Abdullah El-Bayoumi**  
Cairo University, Egypt

**Dr. Ayham Hassan Abazid** Jordan  
University of Science and Technology,  
Jordan

**Mr. Manu Mitra**  
University of Bridgeport, USA

**Mr. Manikant Roy**  
IIT Delhi, India

## Editorial

In the ever-evolving landscape of technological advancements, various sectors have leveraged innovations to address pertinent challenges and improve operational efficiency. This editorial summarizes eight research papers, each contributing significantly to their respective fields through novel methodologies and insightful findings.

The rise of internet economies has revolutionized business operations, notably in financial institutions leveraging credit card usage. However, this surge has also escalated cybercrime, leading to significant financial losses. This paper explores the development of robust credit card fraud detection algorithms, focusing on machine learning techniques like ensemble classifiers. Challenges such as data non-stationarity and class imbalance are addressed, proposing adaptive systems that enhance fraud detection accuracy amidst evolving data distributions [1].

The growth of solar photovoltaic (PV) power as a clean energy source is hindered by its variability due to weather conditions. This research investigates the use of light sensor networks and deep learning models (LSTM and GRU) to forecast solar PV output in Kotzebue, Alaska. Evaluating models on statistical and event-based error metrics, findings indicate the potential of GRU-based models, suggesting improvements with additional data, contributing to grid stability and reduced fuel costs [2].

Extending previous research on time-series clustering of vegetation indices for paddy rice, this study introduces UAVs for visualizing growth changes post-fertilization and optimizing fertilization amounts. Results show optimal yields with minimal fertilization, reducing environmental impacts. Random Forest-based analysis also optimizes monitoring periods, significantly cutting down workload and costs, promoting sustainable agricultural practices through advanced UAV technology [3].

Addressing the intermittent nature of solar energy, this study evaluates deep learning models (LSTM, GRU, and hybrid LSTM-GRU) for forecasting PV power output at the Zagtouli plant in Burkina Faso. Metrics like RMSE, MAE, and  $R^2$  demonstrate the superior performance of the hybrid model. Recommendations are made for transitioning to this advanced forecasting method to enhance the reliability and efficiency of solar energy management [4].

The bandwidth challenges of 360-degree video transmission over the internet are mitigated by Viewport Adaptive Streaming (VAS). This research introduces HEVEL, a deep learning-based method for predicting user viewing behaviour in VAS systems. Combining head and eye movement data, the model outperforms existing methods in precision and error metrics, offering significant advancements in efficient video streaming technologies [5].

Exploring the effects of PFAS on kidney function, this study employs machine learning to analyze chemical features and kidney parameters. Models like XGBoost and Random Forest achieve high accuracy in classifying kidney types and predicting PFAS accumulation, revealing significant correlations between PFAS exposure and kidney health. These findings contribute to understanding PFAS impacts and highlight machine learning's role in health studies [6].

Addressing biases and trust issues in machine learning predictions, this paper proposes a novel deployment methodology using public blockchain and smart contracts. Efficient algorithms ensure secure, immutable storage of model parameters, facilitating peer validation and feedback. A case study on vehicle price prediction models demonstrates the approach's effectiveness in enhancing model transparency and reliability [7].

Focusing on encryption and decryption times, this study evaluates three hybrid encryption schemes (AES-RSA, AES-ECC, AES-ElGamal) in terms of throughput. Results highlight Hybrid AES-RSA as the fastest and most efficient scheme, suitable for performance-critical applications. The research underscores the importance of balancing performance and security needs, recommending Hybrid AES-RSA for systems requiring optimal performance [8].

The research papers summarized herein underscore the diverse applications and potential of advanced technologies such as machine learning, deep learning, blockchain, and UAVs across various sectors. From enhancing cybersecurity and financial fraud detection to optimizing agricultural practices and improving renewable energy forecasting, these studies collectively highlight the transformative impact of innovative methodologies. As these technologies continue to evolve, their integration into practical solutions will likely drive further advancements and efficiencies, addressing contemporary challenges and paving the way for future developments.

## References:

- [1] T.M. Museba, K.V. Vanhoof, "An Adaptive Heterogeneous Ensemble Learning Model for Credit Card Fraud Detection," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 1–11, 2024, doi:10.25046/aj090301.
- [2] H. Toal, M. Wilber, G. Hailu, A.K. Das Kusum Das, "Evaluation of Various Deep Learning Models for Short-Term Solar Forecasting in the Arctic using a Distributed Sensor Network," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 2024, doi:10.25046/aj090302.
- [3] T. Ito, K. Minamino, S. Umeki, "Visualization of the Effect of Additional Fertilization on Paddy Rice by Time-Series Analysis of Vegetation Indices using UAV and Minimizing the Number of Monitoring Days for its Workload Reduction," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 29–40, 2024, doi:10.25046/aj090303.
- [4] S.F. Palm, S. Ezékiel Houénafa, Z. Boubakar, S. Waita, T. Nyachoti Nyangonda, A. Chebak, "Solar Photovoltaic Power Output Forecasting using Deep Learning Models: A Case Study of Zagtouli PV Power Plant," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 41–48, 2024, doi:10.25046/aj090304.
- [5] N. Viet Hung, T. Thanh Lam, T. Thanh Binh, A. Marshal, T. Thu Huong, "Efficient Deep Learning-Based Viewport Estimation for 360-Degree Video Streaming," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 49–61, 2024, doi:10.25046/aj090305.
- [6] A. Mazumder, K. Panda, "Leveraging Machine Learning for a Comprehensive Assessment of PFAS Nephrotoxicity," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 62–71, 2024, doi:10.25046/aj090306.
- [7] B. Wetzel, H. Xu, "Deploying Trusted and Immutable Predictive Models on a Public Blockchain Network," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 72–83, 2024, doi:10.25046/aj090307.
- [8] R.K. Muhammed, K.H. Ali Faraj, J.F.G. Mohammed, T.N. Ahmad Al Attar, S.J. Saydah, D.A. Rashid, "Automated Performance analysis E-services by AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC," *Advances in Science, Technology and Engineering Systems Journal*, **9**(3), 84–91, 2024, doi:10.25046/aj090308.

**Editor-in-chief**

**Prof. Passerini Kazmersk**

# ADVANCES IN SCIENCE, TECHNOLOGY AND ENGINEERING SYSTEMS JOURNAL

Volume 9 Issue 3

May-June 2024

## CONTENTS

<i>Adaptive Heterogeneous Ensemble Learning Model for Credit Card Fraud Detection</i>	01
Tinofirei Museba, Koenraad Vanhoof	
<i>Evaluation of Various Deep Learning Models for Short-Term Solar Forecasting in the Arctic using a Distributed Sensor Network</i>	12
Henry Toal, Michelle Wilber, Getu Hailu, Arghya Kusum Das	
<i>Visualization of the Effect of Additional Fertilization on Paddy Rice by Time-Series Analysis of Vegetation Indices using UAV and Minimizing the Number of Monitoring Days for its Workload Reduction</i>	29
Taichi Ito, Ken'ichi Minamino, Shintaro Umeki	
<i>Solar Photovoltaic Power Output Forecasting using Deep Learning Models: A Case Study of Zagtouli PV Power Plant</i>	41
Sami Florent Palm, Sianou Ezéckiel Houénafa , Zourkalaïni Boubakar, Sebastian Waita, Thomas Nyachoti Nyangonda, Ahmed Chebak	
<i>Efficient Deep Learning-Based Viewport Estimation for 360-Degree Video Streaming</i>	49
Nguyen Viet Hung, Tran Thanh Lam, Tran Thanh Binh, Alan Marshal, Truong Thu Huong	
<i>Leveraging Machine Learning for a Comprehensive Assessment of PFAS Nephrotoxicity</i>	62
Anirudh Mazumder, Kapil Panda	
<i>Deploying Trusted and Immutable Predictive Models on a Public Blockchain Network</i>	72
Anirudh Mazumder, Kapil Panda	
<i>Automated Performance Analysis E-services by AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC</i>	84
Rebwar Khalid Muhammed, Kamaran Hama Ali Farj, Jaza Faiq Gul-Mohammed, Tara Nawzad Ahmad Al Attar, Shaida Jumaah Saydah, Disoz Abdalkarim Rashid	

## An Adaptive Heterogeneous Ensemble Learning Model for Credit Card Fraud Detection

Tinofirei Museba<sup>1\*</sup>, Koenraad Vanhooft<sup>2</sup>

<sup>1</sup>Department of Information Systems, College of Business and Economics, University of Johannesburg, Johannesburg, 2006, South Africa

<sup>2</sup>Department of Quantitative Methods, Universiteit Hasselt, Campus Diepenbeek, Diepenbeek, Kantoort B10, Belgium

### ARTICLE INFO

Article history:

Received: 15 December 2023

Revised: 22 February 2024

Accepted: 23 February 2024

Online: 16 May, 2024

Keywords:

Credit Card Fraud

Machine learning

Concept drift

Ensemble selection

Class imbalance

### ABSTRACT

The proliferation of internet economies has given the corporate world manifold advantages to businesses, as they can now incorporate the latest innovations into their operations, thereby enhancing ease of doing business. For instance, financial institutions have leveraged credit card usage on the aforesaid proliferation. However, this exposes clients to cybercrime, as fraudsters always find ways to breach security measures and access customers' confidential information, which they then use to make fraudulent credit card transactions. As a result, financial institutions incur huge losses amounting to billions of United States dollars. To avert such losses, it is important to design efficient credit card fraud detection algorithms capable of generating accurate alerts. Recently, machine learning algorithms such as ensemble classifiers have emerged as the most effective and efficient algorithms in an effort to assist fraud investigators. There are many factors that hinder the financial sector from designing machine learning algorithms that can efficiently and effectively detect credit card fraud. Such factors include the non-stationarity of data related to concept drift. In addition, class distributions are extremely imbalanced, while there is scant information on transactions that would have been flagged by fraud investigators. This can be attributed to the fact that, owing to confidentiality regulations, it is difficult to access public data. In this article, the author designs and assesses a credit card fraud detection system that can adapt to the changes in data distribution and generate accurate alerts.

### 1. Introduction

The internet offers a great deal of convenience to people's daily routines. However, in the financial sector, which has been progressively adopting e-commerce, web-based technologies can also be a curse. The internet and related applications grow continually because they enable individuals and organisations to digitise and innovate their operations, thus, enabling them to perform more efficiently. With regard to commerce, the advent of the internet saw the birth of cashless transactions through the use of credit and debit cards, in addition to net banking and Unified Payments Interface (UPI) options [1, 2]. In [3], the author explains that fraudulent incidences have invariably risen in proportion with the rising volume of cashless commerce, thus, causing financial institutions to incur enormous losses.

In [4] the author defines credit card fraud as the use of stolen identity details to purchase goods and services or effect electronic cash transfers. Such fraud is not limited to online transactions as stolen or lost physical credit cards can also be used to transact [5]. The last ten years have seen a growing trend in credit card fraud, thus, prompting financial institutions globally to seek for techniques that can accurately and efficiently detect such crime. In this case, machine learning approaches are the most preferred techniques for detecting credit card fraud [6,7,8,9,10]. However, in [11] the author notes that machine learning has some inherent limitations, notably class imbalance, concept drift and verification latency. In [12] the author contends that these challenges are further compounded by lack of reliable up-to-date datasets, wherein hardly 1% of the samples that are labelled fraudulent are classified as such. The author in [13] refers to this phenomenon as

\*Corresponding Tinofirei Museba, Department of Information Systems, College of Business and Economics, University of Johannesburg, South Africa [tmuseba@uj.ac.za](mailto:tmuseba@uj.ac.za)

class imbalance and, according to the author in [9], it inhibits standard classifiers from optimally detecting credit card fraud.

Apart from class imbalance, credit card transactions are dynamic in nature, which then introduces the concept drift problem. This implies that there are no clear demarcations between normal and fraudulent transactions, given that there may be no variance between the fraudsters' and the legitimate cardholders' behaviour spending patterns. In any case, spending habits are dynamic, so non-fraudulent and fraudulent transactions have more or less equal chances of being flagged as suspicious. In attempting to solve the class imbalance problem, researchers have applied both oversampling and under sampling techniques, or combined these with the SMOTE technique [13]. Notwithstanding the aforementioned efforts, concept drift continues to be a hindrance in the detection of fraudulent card transactions using ensemble classifiers.

### *1.1 Credit Card Fraud*

Credit card fraud is a criminal case and occurs when a person uses someone else's personal credentials together with their credit standing to borrow money or use credit cards to transact with no intention of refunding the debt. Credit card is considered to be a form of identity that is most prevalent. When a credit is misappropriated, the victim typically accrues unpaid debts in their names. The process of identifying and detecting credit card flaws can be solved but the implementation process demands more effort and time and may lead to a temporary setback of credit scores temporarily and affect a person's credit score to get new credit for a time. To detect credit card transactional flaws and minimize identity theft, the implementation of machine learning in the design of models capable of addressing such problems has gained momentum due to the results obtained. This research article proposes a machine learning ensemble architecture capable of detecting credit card transaction flaws. For credit card fraud detection, machine learning algorithms can classify whether a credit card transaction is authentic or fraudulent. Machine learning algorithms can make a prediction to determine whether it is the cardholder or the fraudsters using the credit card through credit card profiling. In addition, machine learning algorithms can use outlier detection techniques to identify vast amounts of transactions for outliers from regular credit cards transactions to detect credit card fraud. When compared to other conventional fraud detection techniques, machine learning algorithms offer faster detection and adaptation to drifting concepts. A machine learning model has the capacity to quickly identify any drifts from regular transactions and user behaviour in real time. By detecting anomalies such as a sudden increase in transactional amount or location change, machine learning algorithms are capable of minimising the risk of fraud and ensure more secure transactions.

### *1.2 Ensemble Learning*

Machine Learning algorithms are capable of learning from data, find complex and noise patterns, and predict credit card theft. Ensemble Learning is a machine learning approach capable of optimising generalization performance and resilience in prediction tasks through the process of combining multiple models. Combining multiple prediction models enables the mitigation of errors or biases prevalent in classifier models by leveraging the combined outputs of the ensemble. Ensemble learning combines

the output of diverse models to generate an accurate prediction. Ensemble methods leverage the diversity and complementarity of their predictions to improve their generalization performance. As the underlying concept, ensemble learning considers multiple perspectives and utilizes the capacity of diverse models in an effort to optimize the overall generalization performance of the learning model. Ensemble learning optimises the accuracy of the learning model and also provides resilience against uncertainties in the data such as noise, skewed distributions and missing values. Effectively combining predictions from multiple diverse models has proven to be a powerful technique in various domains, providing more robust, accurate and reliable generalization performance for classification, clustering, regression tasks and anomaly detection by combining different types of base models and aggregation methods. For credit card transaction flaws, ensemble learning techniques are capable of detecting whether a transaction is a fraud or not given the historical spending patterns of a client. Ensemble learning techniques are capable of adapting to spending habits of clients and detect anomalies and alert the bank.

This study proposes a credit card fraud detection system that can be capable of accurately handling class imbalance and concept drift, and detecting credit card fraud. In addition to the foregoing section, there are six ensuing section that make up this paper. Section 2 provides a preview of precious work conducted on credit card transaction fraud. Section 3 describes the data processing process techniques, a description of the base learners, parameter optimisation techniques and feature selection strategy. Section 4 provides a description of the Dynamic Classifier Selection algorithm, performance metrics and the datasets used. Section 5 provides an outline of the experiments conducted and the performance comparisons with other state of the art algorithms. Section 6 provides the conclusion of the study.

## **2. Literature Review**

Scenarios associated with credit card fraud are not uncommon. Consequently, there are various contemporary approaches for detecting and generating accurate alerts that have been proposed. All these approaches are based on machine learning. To improve the detection performance of an ensemble classifier, in [14], the author suggests a machine learning approach-based credit card fraud detection engine that implements a genetic algorithm for feature selection. The prediction performance of the proposed credit card engine is validated with a dataset generated from European cardholders. The proposed engine disregards the relevance of class imbalance and concept drift on the performance of the engine. In order to optimise the efficiency and accuracy of machine learning algorithms for detecting credit card fraud, the author in [15] proposes that blockchain techniques and machine learning algorithms can be combined. Subsequent experiments showed that the approach outperformed all the other proposed algorithms of machine learning. The proposed approach does not consider the prevalence of class imbalance and the presence of concept drift. The author in [16] suggests the application of the Just-Add-Data (JAD) to automate the selection of machine learning algorithms, tune hyperparameter values and estimate prediction performance in the detection of illicit transactions. The proposed version of



JAD fails to detect data distribution drift and generate false alarms automatically. The authors in [17] propose a novel approach for detecting credit card fraud by analysing customers' past transactions details and extracting their behavioral patterns, before clustering them into different groups.

It is difficult to detect credit card fraud with accuracy, because there is need to process enormous streaming data. However, the learning models are not entirely capable of quickly adapting or responding to the fraud. This problem is exacerbated by concept drift, which introduces changes to the target concept. Optimum model performance is further inhibited by class imbalance, overlapping data, the dynamic nature of transactions, the scarcity of datasets and verification latency. These feedback mechanisms may cause delays in signaling fraudulent transactions, thus, not all of them are either caught or reported. Fraudsters design their own adaptive techniques against the existing detection models. This paper formulates a credit card fraud detection system, which incorporates class imbalance and concept drift. The proposed system is also robust against false alarms.

The authors in [18] proposed a machine learning model that implements random forest algorithm to predict and detect daily credit card fraud. The model lacks diversity thereby compromising its accuracy. In [19], the authors proposed a machine learning method with Hybrid feature selection technique consisting of filter and wrapper feature selection steps to ensure that only the most relevant features are used. For feature selection the approach uses Genetic Algorithm which can converge prematurely, and the problem of class imbalance was not addressed. To enhance the accuracy of machine learning algorithms in detecting credit fraud, the authors in [20] proposed a soft voting ensemble learning approach for detecting credit card fraud on imbalanced data. The impact of ensemble diversity and data standardization was not discussed. To address challenges such as class imbalance, concept drift, false positives/negatives, limited generalizability and challenges in real time processing, the authors in [21] proposed a novel ensemble model that integrates a Support Vector Machine, K-Nearest Neighbour, Random Forest, Bagging and Boosting classifiers. The approach is computationally inefficient. In credit card fraud detection, there is little time to perform any resampling of the data when training models, generally precluding the use of bagging, boosting or related methods that resample training data.

### 3. Data Preprocessing

This study uses standardized datasets, which are scaled within the 0-1 interval and all the missing values are approximated. The study focuses on the problem of class imbalance of credit card fraud datasets. The mean is removed and scaled to unit variance in an effort to standardize numeric features. The data are scaled using the 0-1 normalization method. If  $x$  is a given feature, then the normalized feature can be computed as follows:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}, \text{ where } x' \text{ expresses the standardised value.}$$

Feature normalization can significantly improve classifier accuracy. This especially applies to those classifiers that are based on distance or edge calculations and it results in the model being more assertive and accurate. All data related to credit card fraud are subject to class imbalance and their distribution is highly skewed. This can be attributed to the fact that the proportion of fraudulent transactions (minority class) is significantly lower than that of legitimate transactions (majority class). As a result, credit card fraud detection becomes an extremely challenging task. The level of difficulty is compounded by the prevalence of missing values in the relevant datasets. However, this gap can be closed by using XGBoost, which incorporates an algorithm for sparsity segmentation. The algorithm can approximate the missing values with significant accuracy. In addition, the use of standardization helps to subdue the influence of outliers. At the same time, the centralization process is used to curb extreme values. To solve the class imbalance problem, a resampling technique called Synthetic Minority Oversampling Technique (SMOTE) [22] is made use of to enhance the performance of the minority class classifiers recognition. The SMOTE technique generates artificial cases close to observed ones. In the process, it oversamples the minority class. In order to handle class imbalance with overlap, the imbalance ratio and the structure of the dataset are considered in an overlapping metric, which is known as degOver [23]. To handle different types of drifting concepts accurately, the Dynamic Ensemble Selection (DES) is applied, while verification latency is handled by employing the integrated Fraud Detection (FD) [24]. The FD is further integrated with Smooth Clustering based Boosting (SCBoost), which is a noise-robust boosting method and the k-Shortest Distance Ratio (k-SDR). The latter enables the full use of the labelled dataset and prevents interferences from the class imbalance therein. The key function of the k-SDR is to classify an instance by the ratio of its mean distance to k nearest instances in the positive class.

#### 3.1. Base Learners

From an intuitive point of view, the success of an ensemble of classifiers is dependent on the diverse performance of base classifiers [25]. To achieve this, the approach that was adopted in this study used two base learners namely: XGBoost and Support Vector Machines. In the process, these two can also effectively yield both accuracy and efficiency. In addition, Support Vector Machines have proved to be tremendously capable of handling regression and classification challenges in both static and dynamic domains. These machines are widely used to redress dimensionality, which is prevalent in classification problems. In such cases, SVMs find hyperplanes that can separate two classes of linear data in such ways that there can be large distances between the training instances. If the data are non-linear, the SVM kernel function can map them into high dimensional spaces.

For purposes of solving authentic classification problems, particularly the mitigation of model variances, the authors in [26] developed the eXtreme Gradient Boosting (XGBoost). It can also optimise the loss function by including regularisation in handling

sparse data and a weighted quantile sketch for tree learning. In terms of speed and accuracy, XGBoost is arguably the best machine learning algorithm, due to the superiority of its mechanisms and weights, such as Taylor's expansion, which approximates the loss function with remarkable promptness.

### 3.2. Parameter Optimization

The study employed SVMs and XGBoost as base learners, which are associated with a number of parameters. With regard to predicting how the credit card fraud detection system will perform, the parameters are significantly impactful. For the fraud detection system to perform optimally, the parameters have to be optimised using a number of existing optimisation algorithms, most of which are prone to dimensionality. Moreover, the resultant cost of computation tends to increase dramatically, in proportion to the number of hyperparameters or extended search space. Hyperparameter tuning for most applications is subjective and it relies on empirical judgement and trial and error approaches. To counter the limitations that are associated with existing optimisation algorithms, this study employed an adaptive heterogeneous Particle Swarm Optimiser (PSO), in order to optimise and generate an appropriately optimal subset of accurate parameters. This was also meant to improve the efficacy of XGBoost and SVMs for the classification problem. PSO is a heuristic algorithm and evolutionary computational method that is used extensively. The authors in [27] developed the algorithm and it can also be described as a heuristic population based iterative, global and stochastic optimisation system, which is inspired by the flocking and schooling social behaviours of birds and fish, respectively, for conducting intelligent searches for the optimal solutions [28]. Since it is derivative free, PSO does not require the optimisation problem to be differentiable, therefore, it does not require gradients. These characteristics enable PSO to be applicable to a variety of problems, including those that are discontinuous or non-convex and multimodal. In this study, the instantiation of the particles in the swarm was performed at individual level, thereby introducing heterogeneity. The individual instantiation of particles enables different search behaviours of particles in the swarm to be assumed, as they can randomly select the velocity and position update rules from the behaviour pool. The process also allows for the creation of a swarm composed of particles that are both explorative and exploitative in nature. This enables the optimisation algorithm to explore and exploit for the duration of the search process, thus, preventing premature convergence.

### 3.3. Feature Selection

XGBoost was employed for the purpose of performing feature selection. As a base learner, XGBoost is used to generate feature importance scores, which are used for measuring the average objective reduction. This is performed as soon as the specific variables have been selected for splitting. In the tree building process, a variable that is associated with a score that is high has a higher importance. This study used XGBoost as a joint base learner with SVM. The researcher followed propositions by the author in [29] when implementing the scores that were derived from feature importance. These propositions provided a guideline in terms of the sequential forward search (SFS) feature selection algorithm. SFS places all related features into a subset, after which it

iteratively adds the remaining features until the highest score is attained. This results in the generation of a series of candidate feature subsets. In this case, the feature subset that maximises the cross-validated accuracy is selected as the optimal feature set that is appropriate for the process of training the model in the subsequent steps.

## 4. Dynamic Classifier Selection

For ensembles of machine learning, there are two key features namely: diversity and accuracy. For the purpose of this study, the researcher selected classifiers based on their accuracy on the validation set, as well as diversity to accommodate both batch and incremental learning, given that transaction data vary with time. To select classifiers based on accuracy and diversity, the Selection by Accuracy and Diversity (SAD) algorithm in [30], which works as shown below, was used:

1. Train a set of heterogeneous classifiers from XGBoost and SVM
2. Determine the accuracy of each classifier on validation set
3. Select the most accurate classifiers
4. Measure the diversity between the most accurate classifiers
5. Select classifiers with strong diversity into the ensemble and repeat the process until the predetermined size of the ensemble is attained.
6. Use majority voting to combine classifiers into an ensemble
7. Evaluate the generalisation of the ensemble.

The Q static diversity measure in [31] was adopted in this study. The training dataset was used for generating and learning a pool of classifiers. Given a training dataset  $D_{train} = \{x, y\}$ , where  $x$  is an  $M \times N$  dimensional feature matrix and  $y \in \{0, 1\}^N$  denote the label. If  $y$  yields a value of 1, that would indicate a fraudulent transaction, while a value of 0 would imply a legitimate transaction. Here, the aforementioned two base learners, SVM and XGBoost, were used to create a heterogeneous ensemble architecture. SVM and XGBoost are highly renowned in credit card fraud detection, as they can generate highly efficient models. This can be attributed to the fact that the two base learners are frequently updated, in keeping with the behavior change of fraudsters.

### 4.1. Performance metrics

This paper presents a study that was modelled as a machine learning binary classification task, using five performance evaluation metrics. The main performance metric was the accuracy of the test data. Furthermore, for each model, the Precision, Recall, F1\_Score and the Area Under the Curve (AUC) were computed. The AUC provides a proper assessment of the quality of classification of each given model. The AUC metric measures the effectiveness of each classifier for a specific task. The value of AUC is within the interval 0 to 1 and an efficient classifier is identified with an AUC value that is almost close to 1. Accuracy is calculated by dividing the number of correct predictions by the sum of forecasts. Precision is the proportion of correct forecasts to the sum of correct guesses. On the other hand,

recall is the ratio of position predictions to the total of positive class values in the test data. The F1 score represents the balance between accuracy and recall. The performance metrics can be expressed mathematically, as shown below:

$$\text{Accuracy} = \frac{TN+PP}{TP+TN} \quad (1)$$

$$\text{Recall} = \frac{TP}{FN+TP} \quad (2)$$

$$\text{Precision} = \frac{TP}{FP+TP} \quad (3)$$

$$F1_{score} = \frac{PR \cdot RC}{PR+RC} \quad (4)$$

#### 4.2. Datasets

To conduct the research, two credit card fraud transaction datasets were used. The first dataset was from the Machine Learning group, ULB, which is obtained from Kaggle. The transactions were labelled as either legitimate or fraudulent and they were all made over two days in September 2013 by European cardholders. The dataset consisted of 284 807 transactions instances, of which 492 (0.172%) were fraudulent, thus, making it highly imbalanced. The dataset consisted of 30 features, which ranged from V1 to V28; Time and Amount.

The dataset comprised numerical attributes and its last column indicated the class type (type of transaction) and the value 1 represented a fraudulent transaction, while the value 0 denoted a legitimate transaction. The features V1 to V28 were not named for data security and integrity reasons [32]. The dataset was highly imbalanced and, to solve the class imbalance problem, the Synthetic Minority Oversampling Technique (SMOTE) was applied. A scikit\_learn library, called imblearn oversampling, was used to import SMOTE, whose function is to pick samples that are close to each other within the feature space, thereby drawing a line of separation between the data points in the feature space and creating a new instance of the minority at a point along the line. The Time column indicated the number of seconds that transpired between the initial and subsequent transactions. The Amount column featured the amount that was transacted, while the Class column had a values of 1 and 0, which denoted fraudulent and legitimate transactions, respectively.

The second dataset contained credit card transaction data that were sourced by an unnamed institute and were made available on Kaggle. The dataset featured five columns, the first of which was *distance from home*. As the labelling suggests, the first column showed the distance between the cardholders' registered home addresses and transactions locations.

The second column, *ratio\_to\_median\_purchase*, represented the ratio of the transaction to the median of the purchase price, while the third, *repeat\_retailer*, featured checks on whether the last two transactions were made at the same retailer. *Used\_chip* was the penultimate column, which contained verifications on whether or not a physical card was used to make the transaction.

Finally, the *used\_pin\_number* column featured verifications pertaining to whether online transactions were fraudulent or not. This dataset consisted of 912 597 legitimate and 87 403 fraudulent transactions, respectively, thereby rendering it exceedingly imbalanced.

The two datasets were loaded from the local host of the researcher's laptop and imported onto an online Python text editor, Google Colaboratory, in separate notebooks, as follows:

```
Df = pd.read_csv('/content/creditcard.csv')
```

```
Df = pd.read_csv('/content/card_trasdata.csv')
```

A scikit-learn module known as *train\_test\_split* was used to train and split the two datasets and the test size for each was set to 30 percent, while the random state was set to 42. These values are widely used for data training and splitting.

The creation process entails combining existing items by randomly selecting a point from the minority class and computing the k-nearest neighbours (default=5) for that point. Subsequently, each synthetic point is inserted between the chosen point and its neighbours by multiplying the distance by a value between 0 and 1.

The figures below illustrates the class imbalances of the two datasets. The figures show the differences between legitimate and fraudulent transactions.

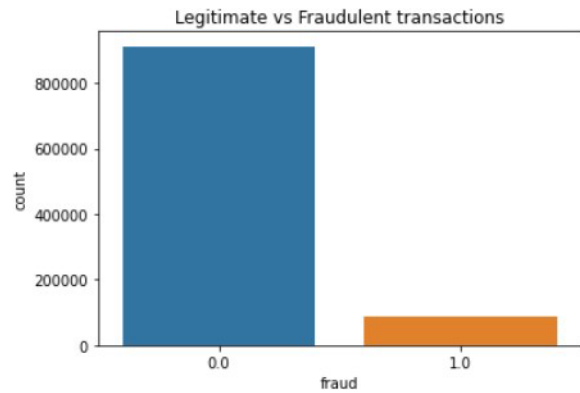


Figure 1: Class imbalance for dataset 1

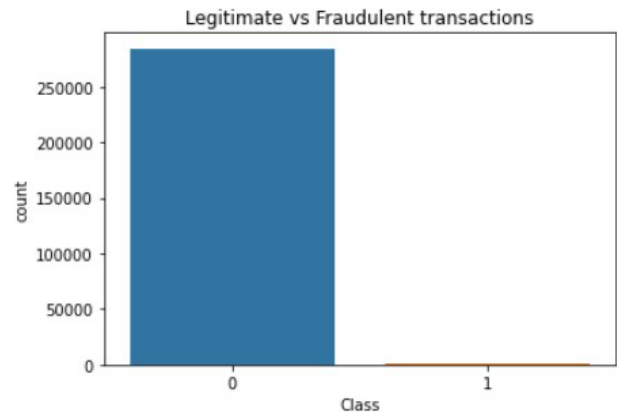


Figure.2: Class imbalance for dataset 2

The figures above show the class imbalances in the datasets. It is easy to deduce how the datasets would look like after the application of the SMOTE technique, which is used to create items for the minority class.

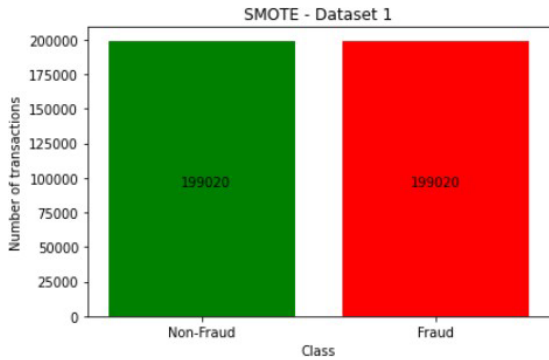


Figure 3: SMOTE dataset 1

For most applications, the difference between oversampling and SMOTE is insignificant, unlike in the data distribution, as illustrated in the figures below for dataset 1.

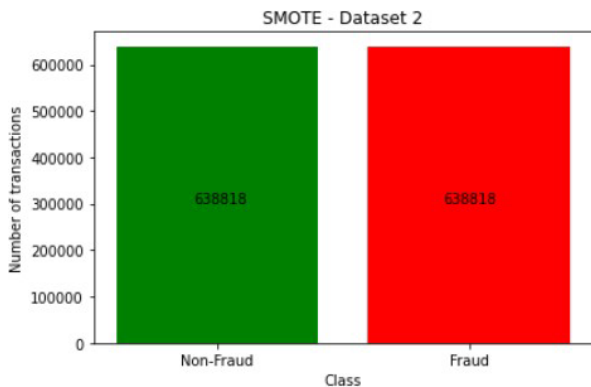


Figure 4: SMOTE dataset 2

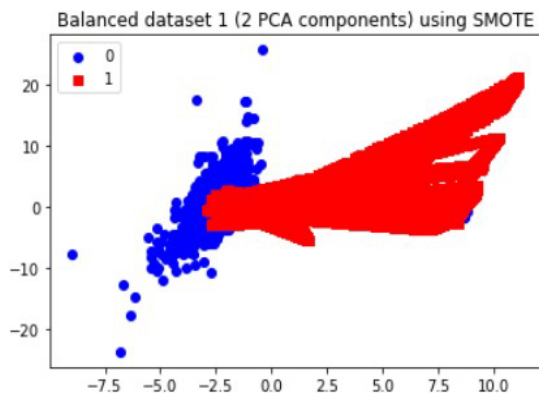


Figure 5: Dataset 1 PCA components SMOTE

The above graphs clearly show that new observations, which are labelled 1 (fraudulent transaction), are found in numerous locations, thus, demonstrating how the SMOTE and Oversampling algorithms can generate instances. Moreover, PCA can be used to display the data in a two-dimensional view, thereby vividly showing the unique differences between legitimate and

fraudulent transactions, as each class has a specified pattern and cluster.

### 5. Experiments

The experiments on the two datasets were conducted on Google Colab [33]. The specifications of the computer were as follows:

Intel Xeon Phi 7290 CPU with 72 cores at 1.5 GHz and 125 GB RAM on Ubuntu 18.04. The experiments were conducted on the Python 3.6.8 and the machine learning framework that was used was the Scikit-Learn [34]. The base learners XGBoost and SVM were implemented through Scikit-Learn. For each feature vector in the dataset, the following algorithms were trained and tested: Decision Tree, Random Forest, Neural XGBoost and an Ensemble of XGBoost and SVM. The results of the experiments, which were conducted without the application of DES on the first dataset are depicted in Table 1.

Table 1: Performance evaluation of the algorithms on dataset 1

Models	Accuracy	Precision	Recall	F1_Score	ROU AUC
Decision Tree	95.85%	3.45%	85.14%	6.63%	94.47%
Random Forest	99.95%	88.15%	80.41%	84.10%	94.93%
XGBoost	99.44%	21.7%	85.81%	34.07%	96.92%
Ensemble	99.94%	89.06%	77.03%	82.61%	95.9%

The Ensemble were composed of the XGBoost and SVM, while the Random Forest classifiers were the best performing algorithms with regard to Accuracy and F1\_Score, and these were followed.

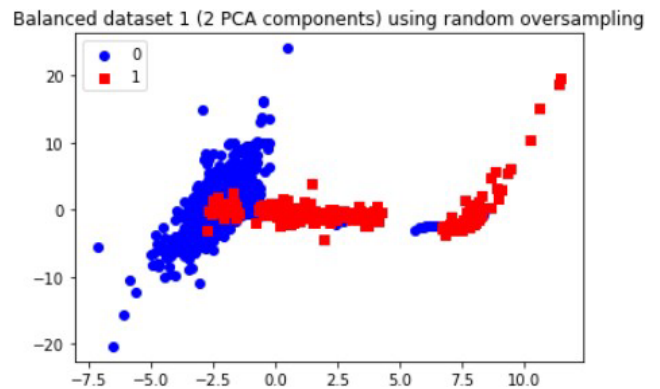


Figure 6: Dataset 2 PCA components ROS

Figure 7 to figure 10 shows the ROC Curve of each learning algorithm for dataset 1.

by the Decision Tree. The ensemble also scored highly on Precision. The Decision Tree classifier and XGBoost showed the worst performance given that they score a Precision of 3.45% and 21.75% as well as F1\_Score of 6.63% and 34.07% respectively

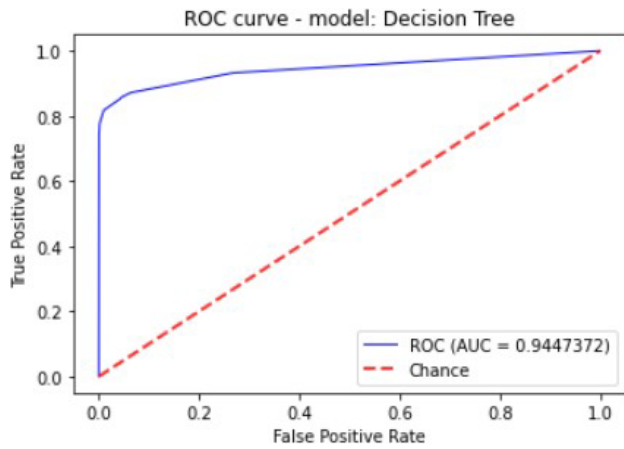


Figure 7: ROC Curve of the Decision Tree on dataset 1.

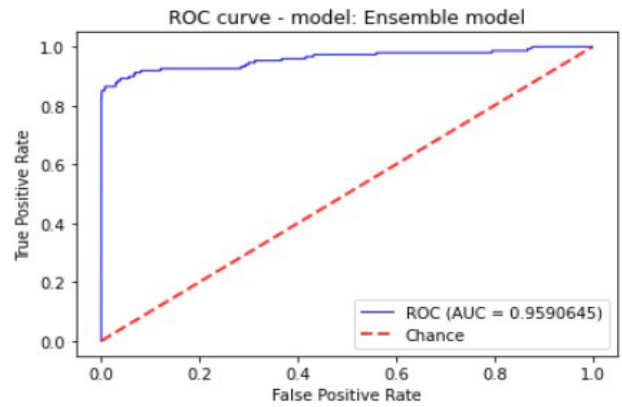


Figure 10: ROC Curve of the ensemble on dataset 1.

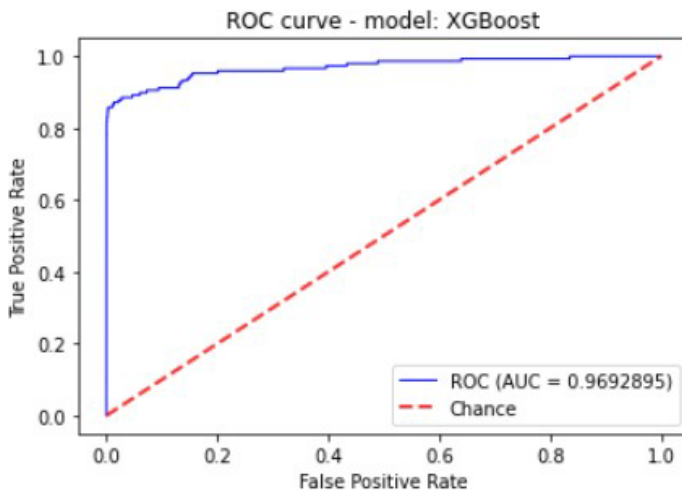


Figure 8: ROC curve of the Random Forest on dataset 1.

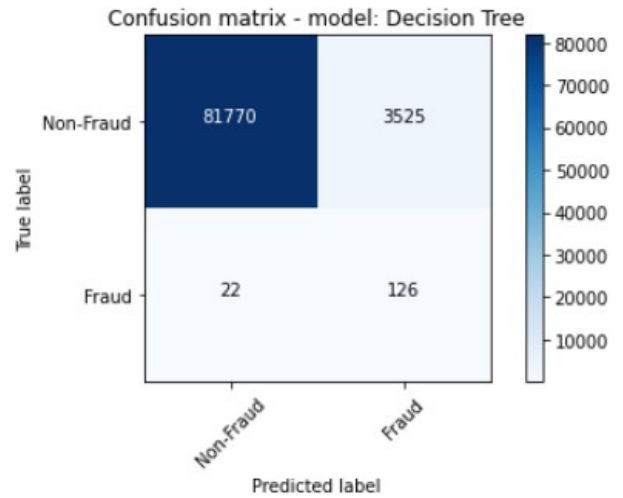


Figure 11: The Confusion matrix generated by the Decision Tree

The figures 11 to figure 14 show the ROC curves and confusion matrix for dataset 1.

A

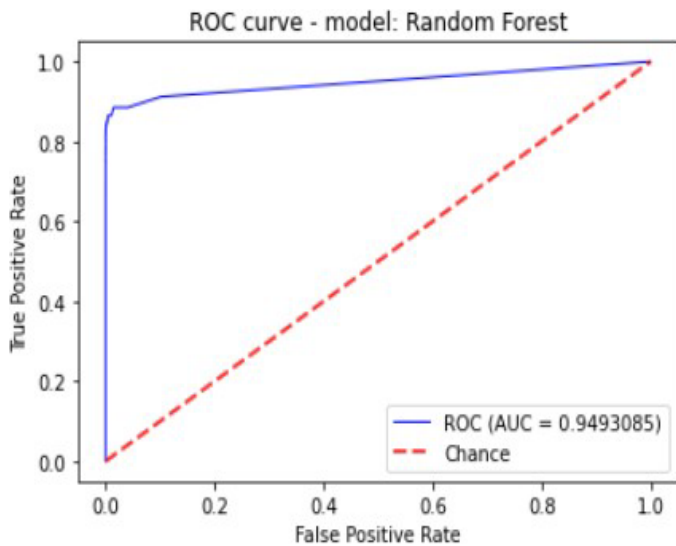


Figure 9: ROC Curve of the XGBoost on dataset 1.

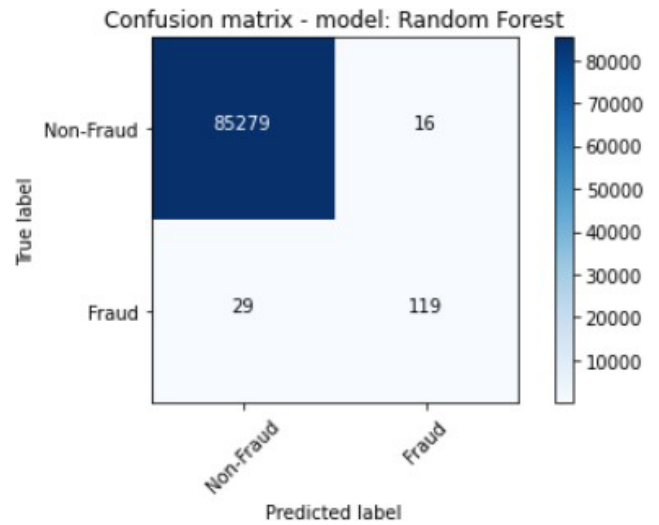


Figure 12: The Confusion Matrix generated by the Random Forest

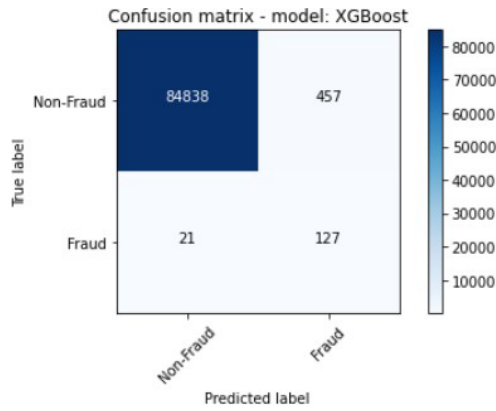


Figure 13: The Confusion Matrix generated by the ensemble.

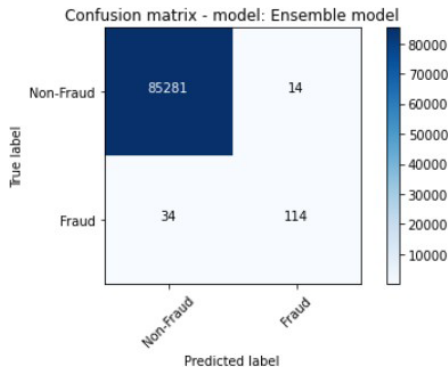


Figure 14: The Confusion Matrix generated by the ensemble.

We conducted further experiments using a second dataset to validate the efficiency and effectiveness of our proposed approach. The dataset consists of the following features: ratio\_to\_mean\_purchase, repeat\_retailer, Used\_chip, Used\_pin\_Number and fraud. The feature fraud denotes the target variable. Table 2 shows the details of the results that were obtained after the experiments were conducted.

The Ensemble were composed of the XGBoost and SVM, while the Random Forest classifiers were the best performing algorithms with regard to Accuracy and F1\_Score, and these were followed by the Decision Tree. The Ensemble also scored highly in Precision. The Decision Tree classifier and XGBoost showed the worst performance, given that they scored a Precision of 3.45% and 21.75%, as well as an F1\_Score of 6.63% and 34.07%, respectively

Table 2: Performance evaluation of the algorithms on dataset 2

Models	Accuracy	Precision	Recall	F1_Score	ROC AUC
Decision Tree	97.60%	79.77%	97.15%	87.61%	99.59%
Random Forest	99.75%	99.65%	99.98%	99.98%	99.89%
Neural Networks	99.95%	99.85%	99.73%	99.73%	99.96%
Ensemble	99.97%	99.76%	99.82%	99.83%	99.94%

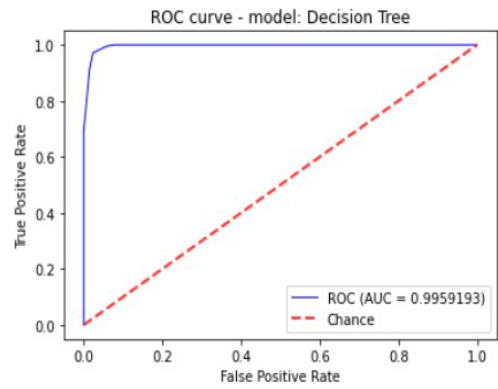


Figure 15: ROC Curve generated by the Decision Tree

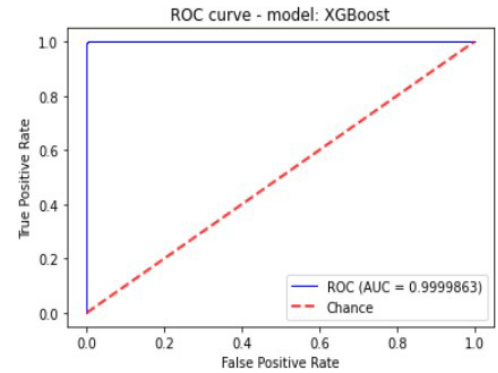


Figure 16: ROC Curve generated by the XGBoost

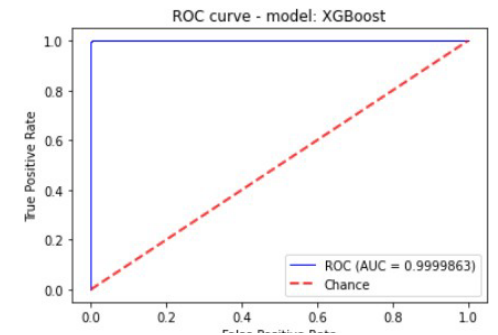


Figure 17: ROC Curve generated by the ensemble

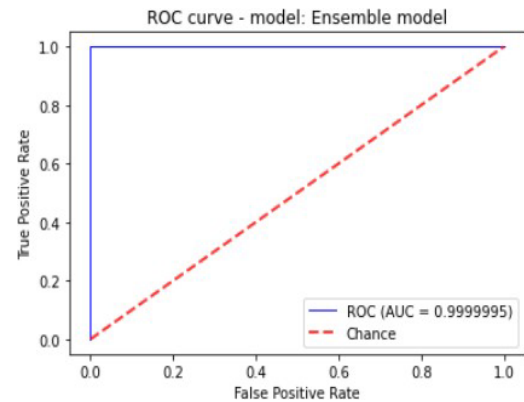


Figure 18: ROC Curve generated by Random Forest

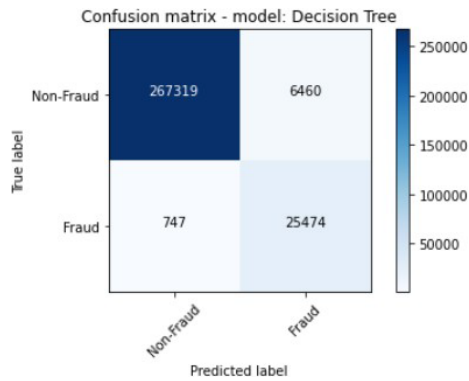


Figure 19: Confusion matrix generated by decision tree on dataset 2

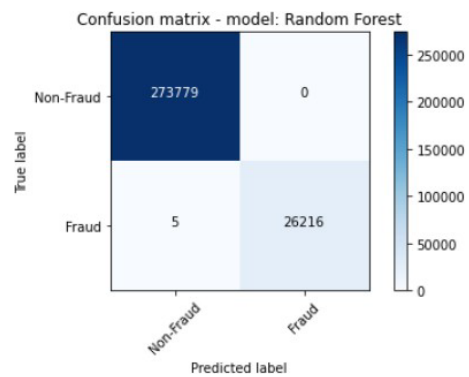


Figure 20: Confusion matrix generated by Random Forest on dataset 2.

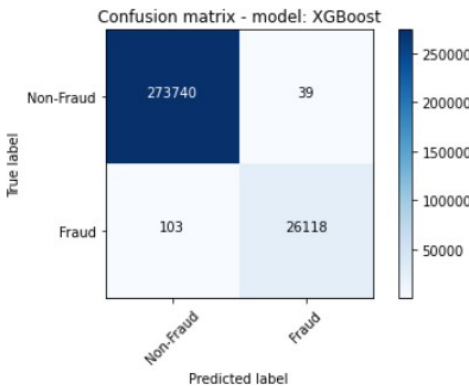


Figure 21: Confusion matrix generated by XGBoost on dataset 2.

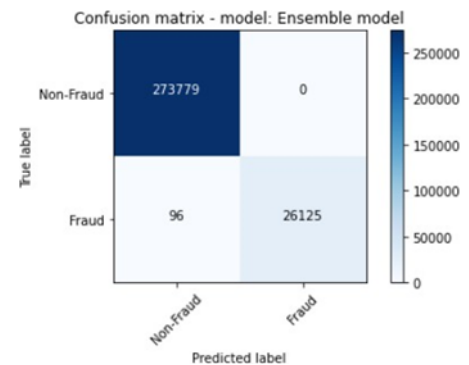


Figure 22: Confusion matrix generated by the ensemble on dataset 2.

### 5.1 Comparison with state-of-the-art algorithms

The European dataset, which is found on Kaggle, has been implemented in several researchers' proposed models. The comparisons that were made in this section were based on the results obtained from this very dataset, which is also the first dataset of this study. The proposed model was compared to the ensemble of models that were implemented on the European dataset, which the author found to have outperformed some state of the art models. Table 3 shows the comparative performances of the proposed ensemble. The comparison used five metrics against other state of the art algorithms.

Table 3: Performance evaluation of the DESP algorithms against state-of-the-art algorithms

Models	Accuracy	Precision	Recall	F1_Score	ROU AUC
DESP-ENSEMBLE	98.76%	72.65%	97.26%	61.49%	99.97%
KNORA-U	83.46%	61.24%	96.65%	70.25%	99.88%
KNORA-E	86.34%	84.87%	98.45%	80.76%	99.87%
GRU	81.63%	86.26%	72.08%	77.92%	86.02%
ℓ	79.85%	95.69%	66.74%	78.13%	83.37%
LSTM	80.54%	85.75%	74.08%	76.87%	87.02%

The proposed model achieved the best results in terms of Accuracy, Precision, F1\_Score, Recall, and AUC ROC. The proposed model used Dynamic Ensemble Selection Performance (DESP), K\_Nearest Neighbour Oracle Eliminate (KNORA-E), K\_Nearest Neighbor Union (KNORA-U). The results shown in the tables shows that this model has an overall better performance. The DESP technique picks all base classifiers that produce better classification performance than the random classifier in their domain of competence. The random classifier's performance is defined as  $RC=1/L$ , where L is the number of classes in the task.

If no base classifier is chosen, and the pool as a whole is utilized for classification.

If the KNORA-E technique looks for a local Oracle, which is a base classifier that properly classifies all samples in the test samples' zone of competence. All classifiers with faultless performance in their domain of expertise are chosen (local oracles). If no classifiers achieves complete accuracy, the size of the competence zone is lowered (by eliminating the farthest neighbor) and the classifier's performance is re-evaluated. The majority voting system is used to integrate the outputs of the selected ensemble of classifiers. If no base classifier is chosen, the pool as a whole is utilized for classification. In the KNORA-U technique, all classifiers are picked that accurately categorized at least one sample from the query sample's zone of competence. Each chosen classifier receives the same number of votes as the number of samples in the zone of competence for which it predicts

the correct label. The votes from all base classifiers are combined to get the final ensemble.

As shown in Table 3, GRU, LSTM and ensemble model  $\ell$  have the best precision performance with ensemble model  $\ell$  being number one. However, the DESP model proposed in this paper has the best overall performance, with KNORA-E. Together with ensemble  $\ell$ , GRU had the worst Recall performance. DESP and KNORA-U outperform LSTM in Recall and AUC ROC performance with KNORA-E is superior performing model.

With regard to Accuracy, Precision, F1\_Score, Recall, and AUC ROC, the proposed model produced the best results. The model employed Dynamic Ensemble Selection Performance (DESP), K\_Nearest Neighbour Oracle Eliminate (KNORA-E), and K\_Nearest Neighbour Union (KNORA-U). In terms of the competence domain, the DESP technique is better than the random classifier. This is because the former has a more superior capability to pick pick base classifiers that can yield more superior classification performance. The performance of the random classifier is defined as  $RC=1/L$ , where L is the number of classes in the task. If no base classifier were chosen and the pool as a whole were utilised for classification, the execution process would require more memory.

## 6. Conclusion

This study proposed and evaluated the performance of the Dynamic Ensemble Selection Performance (DESP) for credit card fraud detection. The study made use of two datasets, which were sourced from Kaggle, and both were found to be extremely imbalanced and associated with different types of concept drift. To address these challenges, the study used the SMOTE technique, which is broadly used for handling imbalance in the detection of credit card fraud. The technique yielded impressive results. To handle drifting concepts, the researcher employed an accuracy and diversity oriented algorithm. The aim of the research was to compare the performance of homogeneous ensembles with heterogeneous ensemble learning before performing a comparative study of the proposed approach against existing state of the art heterogeneous algorithms. The paper also sought to demonstrate how DESP handled class imbalance and concept drift in credit card fraud detection. The paper introduced diversity by combing two learning base algorithms namely: XGBoost and SVM and the Q Statistic diversity measure was used. Combining algorithms of different classifications, particularly in adaptive models, enables fraud detection models to be more efficient in picking the best classifiers for particular data in fraud transactions. Ensemble classifiers make it possible to build models that are capable of overcoming challenges like class imbalance, verification latency and concept drift, which are inherent in credit card fraud detection. The proposed DESP model outperformed existing state of the art techniques.

Several limitations were encountered in the process of conducting this research. The major challenge was unavailability of authentic contemporary datasets because of the attendant

privacy and security issues. For future work, there is need fo strike the balance between accuracy and computational efficiency. Machine learning algorithms exhibit distinct trade-offs between training and testing times as the prediction performance is evaluated using training and testing time and memory usage. In future, the efficiency of the learning model can further be improved by refining the training and testing durations. Streamlining computational overheads has the potential to develop fraud detections systems capable of real time operation ensuring swift responses to evolving fraud trends. For future work, the integration of deep learning can be explored in conjunction with traditional machine learning approaches with the potential of yielding more accurate and adaptable fraud detection solutions.

## Conflict of Interest

The authors declare no conflict of interest.

## References

- [1]. N. S. Alfaiz, S. M. Fati, "Enhanced credit card fraud detection model using machine learning," *Electronics*, **11**(4): 662, 2022, DOI: 10.3390/electronics.11040662.
- [2]. M. Z. Khan, A. Indian A., K. K. Mohbay K. K., "Credit card fraud prediction using XGBoost- An-Ensemble-Learning-Approach," *International Journal of Information Retrieval Research (IJIRR)*, **12**(2): 1-17, 2022, DOI: 10.4018/IJIRR.299940.
- [3]. E. Kim, J. Lee, H. Shin, H. Yang, S. Cho, S. Nam, Y. Song, J. Yoon, J. Kim, "Champion-Challenger analysis for credit card fraud detection: Hybrid ensemble and deep learning," *Expert Systems with Applications*, **128**: 214-224, 2019, doi: <https://doi.org/10.1016/j.eswa.2019.03.042>.
- [4]. Ross D. E. (2016). Credit card fraud. Retrieved from <https://www.britannica/topic/credit-card-fraud>
- [5]. P. Tomar, S. Shrivastara, U. Thakar, "Ensemble learning based credit card fraud detection system," 2021 5th Conference on Information and Communication Technology, CICT2021, 10-12 December 2021, Kurnool India, doi: 10.1109/CICT53865.2020.9672426.
- [6]. F. Carcillo, A. Dal Pozzolo, Y.-A. Le Borgne, O. Caelen, Y. Mazzer Y., G. Bontampi G., "SCARFF: A scalable framework for streaming credit card fraud detection with spark," *Information Fusion*, **41**: 182-194, May 2018, DOI: 10.1016/j.inffus.2017.09.005.
- [7]. J. Femila Roseline, GBSR Naidu, V. S. Pandi, S. A. Rajasree, D. N. Mageswari, "Autonomous credit card fraud detection using machine learning approach," *Computers and Electrical Engineering*, **102**: 108132, September2022, <https://doi.org/10.1016/j.compeleceng.2022.108132>.
- [8]. W. Liu, C. Wu, S. Ruan S, "CUS-RF-Based credit card fraud detection with imbalanced data," *Journal of Risk Analysis and Crisis Response*, **12**(3), <https://doi.org/10.54560/jracr.v12i3.332>.
- [9]. L. Gao, A. Li, Z. Liu, Y. Xie Y, "A heterogeneous ensemble learning model based on data distribution for credit card fraud detection," *Wireless Communications and Mobile Computing*, Volume 2021, Article ID: 2531210, <https://doi.org/10.1155/2021/2531210>.
- [10]. J. Forough, S. Momtazi, "Ensemble of deep sequential models for credit card fraud detection," *Applied Soft Computing*, **99**: 106883, February 2021, <https://doi.org/10.1016/j.asoc.2020.106883>.
- [11]. C. Alippi, G. Bontempi, O. Caelen, G. Boracchi, A. Dal Pozzolo A, "Credit card fraud detection: A Realistic Modeling and a Novel Learning Strategy," *IEEE Transactions on Neural Networks and Learning Systems*, **29**(8): 3784-3797, <https://doi.org/10.1109/tnnls.2017.2736643>.



- [12]. S. Bagga, A. Goyal, N. Gupta, "Credit card fraud detection using pipelining and ensemble learning," *Procedia Computer Science*, **173**: 104-112, 2020, <https://doi.org/10.1016/j.procs.2020.06.014>.
- [13]. U. Nambiar, R. Pratap, I. Sohony I, "Ensemble learning for credit card fraud detection," *ACM International Conference Proceeding Series* 2018, <https://doi.org/10.1145/3152494.3156815>.
- [14]. E. Iheberi, Y. Sun, Z. Wang, "A machine learning based credit card fraud detection for feature selection," *Journal of Big Data*, **9**: 24, 2022, <https://doi.org/10.1186/s40537-022-00573-8>.
- [15]. A. Maurya, A. Kumar A, "Credit card fraud detection system using machine learning technique," 2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), 16-18 June 2022, Malang, Indonesia, doi: 10.1109/cyberneticsCom55287.2022.9865466.
- [16]. V. Plakandaras, P. Gogas, T. Papadimitriou, I. Tsamardinos I, "Credit card fraud detection with automated machine learning systems," *Applied Artificial Intelligence International Journal*, **36**(1), 2022, <https://doi.org/10.1080/08839514.2022>.
- [17]. V. N. Dornadula, S. Geetha S, "Credit card fraud detection using machine learning algorithms," *Procedia Computer Science*, **165**: 631-641, 2019, <https://doi.org/10.1016/j.procs.2020.01.057>.
- [18]. J. K. Afrivie, K. Tawiah, W. A. Pels, S. Addai-Henne, H. A. Dwamena, E. O. Owiredo, S. A. Ayeh S, J. Eshun, "A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions," *Decision Analytics Journal*, Volume **6**, March 2023, <http://doi.org/10.1016/j.dajour.2023.100163>.
- [19]. M. Ienye, Y. Sun, "A machine learning method with hybrid feature selection for improved credit card fraud detection," *MDPI Applied Sciences*, 2023, <https://doi.org/10.3390/app13127254>.
- [20]. M. A. Mim, N. Majadi N, P. Mazumder, "A soft voting ensemble learning approach for credit card fraud detection," *Heliyon*, **10**(3): ee25466, 2024, PMID:38333818, <https://doi.org/10.1016/j.heliyon.2024.e25466>.
- [21]. A. R. Khalid, N. Owoh, M. A. Ashawah, J. Osmor J, J. Adejoh, "Enhancing credit card fraud detection: An Ensemble Machine Learning Approach," *Big Data and Cognitive Computing*, **8**(1): 6, 2024, <https://doi.org/10.3390/bdcc8010006>.
- [22]. Y. Xia, C. Li, N. Liu, "A boosted decision tree approach using Bayesian hyperparameter optimization for credit scoring," *Expert Systems with Applications*, **78**: 225-241, 2017, <https://doi.org/10.1016/j.eswa.2017.02.017>.
- [23]. M. Mercier, M. Santos, P. Henriques Abreu, C. Soares, J. Soares, J. Santos, "Analyzing the Footprint of classifiers in overlapped and imbalanced contexts," 17th International Symposium, IDA 2018, Hertogenbosch, The Netherlands, October 24-26, 2018, [https://doi.org/10.1007/978-3030-01768-2\\_17](https://doi.org/10.1007/978-3030-01768-2_17).
- [24]. R. Eberhart, J. Kennedy J, "Particle Swarm Optimization," In *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia, November 27-December 1, 1995, <https://doi.org/10.1109/ICNN.1995.488968>.
- [25]. L. L. Minku, X. Yao X, "DDD: A New Ensemble Approach for Dealing with Concept Drift," *IEEE Transactions on Knowledge and Data Engineering*, **24**(4), April 2012, doi: 10.1109/TKDE.2011.58.
- [26]. T. Chen, C. Guestrin C, "XGBoost: A scalable tree boosting system," In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pages 785-794, 2016, <https://doi.org/10.1145/2939072.2939785>.
- [27]. W. Rui, L. Guanjun, "Ensemble Method for credit card fraud detection," 2021 4th International Conference on Intelligent Autonomous Systems, 14-16 May 2021, Wuhan, China, doi: 10.1109/ico/AS53694.2021.00051.
- [28]. Google Colab [Online] Available on: <https://colab.research.google.com/>
- [29]. N. V. Chawla, K. W. Bowyer, L. O. Hall, W. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," *Journal of Artificial Intelligence Intelligence Research*, **16**(2002): 321-357, doi: 10.1613/jair.953.
- [30]. L. Yang L, "Classifier selection for ensemble learning based on accuracy and diversity," *Procedia Engineering*, **15**:4266-4277,2011, DOI: <https://doi.org/10.1016/j.poeng.2011.08.800>.
- [31]. G. Yule, "On the association of attribute in statistics," *Philosophical Transaction. Royal Society of London. Series A, Volume 194*, 1900, <https://doi.org/10.1098/rspi.1899.0067>.
- [32]. The credit card fraud [Online], <https://www.kaggle.com/mlg-ulb/creditcardfraud>.
- [33]. A. P. Engelbrecht, "Computational Intelligence: An Introduction," John Wiley and Sons, Chichester, December 2002.
- [34]. Scikit learn: machine learning in Python [Online]: <https://scikit-learn.org/stable>.

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

# Evaluation of Various Deep Learning Models for Short-Term Solar Forecasting in the Arctic using a Distributed Sensor Network

Henry Toal<sup>\*1</sup>, Michelle Wilber<sup>1</sup>, Getu Hailu<sup>2</sup>, Arghya Kusum Das<sup>3</sup>

<sup>1</sup>University of Alaska Fairbanks, Alaska Center for Energy and Power, Fairbanks, 99775, United States

<sup>2</sup>University of Alaska Anchorage, Mechanical Engineering Department, Anchorage, 99508, United States

<sup>3</sup>University of Alaska Fairbanks, Department of Computer Science, Fairbanks, 99775, United States

## ARTICLE INFO

Article history:

Received: 07 March, 2024

Revised: 02 May, 2024

Accepted: 03 May, 2024

Online: 22 May, 2024

Keywords:

Solar Photovoltaics

Sensor Network

Machine Learning

Deep Learning

## ABSTRACT

The solar photovoltaic (PV) power generation industry has experienced substantial, ongoing growth over the past decades as a clean, cost-effective energy source. As electric grids use ever-larger proportions of solar PV, the technology's inherent variability—primarily due to clouds—poses a challenge to maintaining grid stability. This is especially true for geographically dense, electrically isolated grids common in rural locations which must maintain substantial reserve generation capacity to account for sudden swings in PV power production. Short-term solar PV forecasting emerges as a solution, allowing excess generation to be kept offline until needed, reducing fuel costs and emissions. Recent studies have utilized networks of light sensors deployed around PV arrays which can preemptively detect incoming fluctuations in light levels caused by clouds. This research examines the potential of such a sensor network in providing short-term forecasting for a 575-kW solar PV array in the arctic community of Kotzebue, Alaska. Data from sensors deployed around the array were transformed into a forecast at a 2-minute time horizon using either long short-term memory (LSTM) or gated recurrent unit (GRU) as base models augmented with various combinations of 1-dimensional convolutional (Conv1D) and fully connected (Dense) model layers. These models were evaluated using a novel combination of statistical and event-based error metrics, including Precision, Recall, and  $F_\beta$ . It was found that GRU-based models generally outperformed their LSTM-based counterparts along statistical error metrics while showing lower relative event-based forecasting ability. This research demonstrates the potential efficacy of a novel combination of LSTM/GRU-based deep learning models and a distributed sensor network when forecasting the power generation of an actual solar PV array. Performance across the eight evaluated model combinations was mostly comparable to similar methods in the literature and is expected to improve with additional training data.

## 1. Introduction

Solar photovoltaic (PV) power generation is an increasingly attractive method for expanding global carbon-free power generation capacity, growing from around 39 GW in 2010 to over 1.2 TW in 2022 [1, 2]. Despite the technology's competitive cost and relative ease of installation, continued solar PV expansion is challenged by the fact that PV systems can experience significant variability in generation potential throughout the course of a day due to shading from weather events such as clouds, with sudden losses or gains in production of over 70% of maximum capacity per minute regularly

observed [3, 4, 5]. This variability poses a challenge from a grid-integration standpoint due to the potential for a mismatch between electrical generation and demand [6]. This is especially true for smaller, electrically isolated grids or "microgrids" with high proportions of geographically concentrated solar PV generation since a single cloud event could quickly and dramatically impact total grid production [7].

Communities with microgrids typically rely on diesel generators as a major component of their total generation capacity. These generators also serve as a convenient source of fast-response generation or

\*Corresponding Author: Henry Toal, 1764 Tanana Loop, Suite 420, Fairbanks, AK 99775, [ehtoal@alaska.edu](mailto:ehtoal@alaska.edu)

“spinning reserve” which allows the grid to absorb sudden changes in solar PV generation due to cloud events. Unfortunately, diesel generators are not an ideal solution to this problem [8]. For one, the grid-scale generators used by these communities typically require between 1 to 2 minutes to start and become grid-synchronized, meaning that they cannot be left off until needed to compensate for a loss in PV production. Additionally, these generators typically must be operated at or above 30% capacity. This means that, even if a grid’s total PV production exceeds demand, the generators must be kept running at minimum capacity to provide reserve, increasing costs from additional fuel burn and maintenance as well as increasing emissions [9, 10]. The relevance of this problem is not limited to small, rural communities without electrical connection to a wider grid. There is substantial and growing evidence that developing self-sufficient microgrid capabilities within existing grid networks could increase grid resiliency by reducing centralized dependence on electrical generation and also by allowing critical infrastructure to more easily transition to backup generation sources in the event of a grid disruption [11, 12]. Thus, the problem of sudden changes or “ramp” events in solar PV production within high solar penetration grids is likely to only grow in importance over the coming years.

A potential solution to this problem is to forecast short-term ramp events caused by clouds. If grid operators were to have access to accurate and reliable forewarning of incoming sharp ramp events at a 1-to-2-minute horizon, diesel generators could potentially be left off and started only when needed. Additionally, as energy storage systems become more widespread, both in microgrids and beyond, these forecasts have the potential to increase storage efficiency by providing grid control systems with advanced knowledge of when energy storage should start and stop.

A wide range of technologies exist for producing solar PV forecasts. These include numerical weather prediction (NWP) models, satellite imagery, total sky imager (TSI) systems (sometimes called “sky cameras”), and networks of distributed light sensors, with the latter two being most effective for the time horizon presented by this problem [13]. Deep learning neural network models have also seen substantial advancement in solar PV forecasting across a wide range of data collection methods and time horizons. While the majority of studies have utilized TSI systems, there is growing evidence that distributed sensor networks have a number of distinct advantages, including lower cost and improved forecast accuracy [10, 14]. A variety of models have been proposed for use in conjunction with distributed sensor networks, including matching local production peaks and examining covariance between sensors [15, 16], but there has been very limited exploration into sensor networks combined with deep learning models, especially of those deployed around an actual solar PV in the type of environment where this type of forecasting is likely to be particularly valuable. This study aims to fill the gap in the established body of work by developing and deploying a distributed sensor network around a utility-scale solar PV arraying a remote, electrically isolated community and using data from these sources to train a variety of deep learning model architectures. This study also aims to advance the field of solar PV forecasting by evaluating the performance of these models using both standard error metrics such as root-mean-square error (RMSE) but also quantifying their ability to detect individual ramp events and examining the impact of location-specific variability on forecast

quality.

This study was part of an ongoing collaboration with the Kotzebue Electric Association (KEA) in Kotzebue, Alaska, USA. Kotzebue, a community of just over three thousand people located just above the arctic circle at 66.9 °N, provides an excellent reference for a study examining the intersection of distributed sensor networks, deep learning models, and harsh, isolated environments. With over 1.2 MW of solar PV capacity across two co-located arrays, Kotzebue is an ideal example of a community which relies heavily on its diesel generators to absorb ramp events from high-penetration solar PV. This study uses high-resolution inverter data collected directly from one 72-kW sub-array of the 575-kW eastern KEA solar PV plant as a target forecast variable along with data from the sensors as model inputs.

In this paper, we review the current state of short-term solar PV forecasting and examine why the combination of deep learning models and distributed sensor network is of particular importance. We go on to provide an in-depth overview of our data collection and processing methodologies, the mechanism behind the various deep learning models used and our specific implementation as well as our chosen performance metrics and their relative advantages. We conclude with an analysis of model performance both in terms of the standard error metrics used throughout the literature as well as relative ramp detection performance across a range of ramp magnitudes.

## 2. Related Work

### 2.1. Solar PV Forecasting

The methodologies employed for solar forecasting are highly dependent on the amount of time into the future one wishes to predict (otherwise known as the “forecast horizon”) as well as the sources and availability of local weather data. Forecast horizons can generally be thought of in four timescale categories: (1) day-ahead (more than 1 day), (2) intra-day (1-24 hours), (3) intra-hour (10-60 minutes), and (4) very-short-term, sometimes called “nowcasting” (less than 10 minutes) [13]. There are also four primary types of data sources used for solar PV forecasting: NWP models, satellite images, TSI systems, and distributed sensor networks. NWP uses regional weather data and complex weather simulations to forecast cloud cover over wide areas and, due to its computational intensity, is primarily used for day-ahead forecasts. Prediction using satellite images is generally best for intra-day horizons since the images typically lack the spatial and temporal resolution for more refined forecasting [17]. TSI systems use a specialized camera to take hemispheric images of the sky around the solar PV array from which cloud coverage and relative motion information can be transformed into a power production forecast. The majority of the research into intra-hour and nowcasting forecasts uses TSI systems but there is evidence that, for horizons shorter than 5 minutes, these systems may suffer from reduced accuracy due to a protective element inside the imager called a “shadow band” which shields the sensor from direct sunlight but also obscures cloud cover information in the region directly around the sun, the exact zone needed for quality very-short-term forecasting [18, 19]. In an attempt to produce more accurate forecasts in the 1-to-5-minute range, a handful of studies over the

past decade have investigated the use of networks of inexpensive solar irradiance sensors placed far enough around the PV array to detect ramp events before they impact the array [10, 15, 16, 20]. Results from these systems are promising and have the advantage of being more affordable than TSI systems, a particular concern for small communities lacking significant funds for additional power infrastructure. This encouraging performance as well as the lack of studies examining the technique in high-latitude locations is why a distributed sensor network was selected as the basis for this study.

Studies on solar PV nowcasting have used a variety of model structures for generating time series forecasts from input data. These can include peak-matching algorithms, wavelet decomposition models, computer vision algorithms and, more recently, deep learning and/or recurrent neural networks (RNNs) such as long short-term memory (LSTM), gated recurrent unit (GRU), and 1-dimensional convolutional (Conv1D) models which show a substantial ability to learn complex relationships between their input and target data sets.

## 2.2. Distributed Sensor Networks

Despite their lower cost and potential for improved nowcasting performance over TSI systems [14], distributed sensor networks have seen limited use in the literature. A study presented in [15] state that in 2013 utilized 80 rooftop PV installations across a 2500 km<sup>2</sup> area in Tucson, Arizona as a network of sensors and a centralized PV installation as the target for forecasting. The installations were located at various distances from the central array. Data were recorded at 15-minute intervals and a forecast was generated by calculating the covariance between pairs of installations to determine the speed and direction of clouds as they passed over the array. The measurement installation most closely correlated with the central PV array at any given time step was used as that time step's forecast by lagging the data point by the calculated time-of-arrival. Forecasts were evaluated on horizons of 15 to 90 minutes using mean-square error (MSE). In [16], the authors developed a network of five irradiance sensors located within a large PV plant to extract cloud speed and direction information using the cross-correlation between sensors. This was done by assuming that each cloud had a linear edge and then using geometry to find the orientation of the edge and its estimated speed. Similarly to [15], data from the sensor most correlated to the PV array output was lagged by the calculated time-of-arrival at each time step to create the forecast. Performance was evaluated using RMSE. In [10], the authors used 19 low-cost irradiance sensors localized to the north-west of a PV array in combination with a peak-matching algorithm to find the lag between each sensor and the PV array output, find the most correlated sensor, then time-shift the sensor data stream by the time lag to generate a forecast. The novel peak-matching algorithm was compared to simple autoregressive neural networks with and without inputs from the sensor data. The peak-matching algorithm outperformed both neural network models on an RMSE basis. A 2015 study presented in [20] used the US National Renewable Energy Laboratory's (NREL) Horizontal Irradiance Grid, a network of 17 irradiance sensors distributed over approximately 1 km<sup>2</sup> in Oahu, HI as a data source with a central sensor used as a PV array analog. A regression forecasting model known as Least Absolute Shrinkage and Selection Operator (LASSO) was used with good results when compared to ordinary

least squares and auto-regressive models. Models were compared using normalized mean absolute error (nMSE) and a metric known as "forecast skill" which compares model RMSE to the RMSE of a "naïve" or "persistence" forecast (which simply assumes the next data point will be the same as the current).

## 2.3. Recurrent Neural Networks for Solar Forecasting

A number of studies over the past decade have explored RNNs as a model structure for intra-hour and nowcasting irradiance forecasting using TSI systems, sensor networks, and single-source irradiance combined with local meteorological data [21, 22, 23]. For example, the authors of [24] explored the use of various hybrid LSTM, GRU, and Conv1D models for generating forecasts at horizons between 5 and 60 minutes. The researchers used a large database of solar PV production data from a microgrid at the University of Trieste, Italy and were able to produce very accurate forecasts, particularly at a 1-minute, 1-step-ahead time horizon. The study also demonstrated the feasibility of multi-step ahead forecasting, albeit with slightly lower accuracy. The authors did find potential for further improvements, especially during periods of heavy cloud cover and for multi-step-ahead forecasts. Overall, they determined that their methodology was sufficient to aid in the design of small-scale energy storage systems. In [25], the authors utilized a LSTM-Conv1D hybrid deep learning model to forecast 15-minute resolution data at various horizon using real-world solar data from Rabat, Morocco. Their hybrid model showed improved performance when compared to simpler machine learning techniques and non-hybrid models in terms of mean absolute error (MAE) and RMSE as well as enhanced stability in training. The researchers noted accurate performance up to a horizon of 90 minutes and concluded that their hybrid model could prove useful for real-time microgrid energy management and that there was potential for further forecast integration with wind power and load forecasting. A 2021 study [26] also used a Conv1D-LSTM hybrid model but this time in conjunction with the same NREL Horizontal Irradiance grid data as was used in [20]. The model was used to first extract localized dependencies between sensors which were then compared to long-term patterns learned by the LSTM component. The hybrid model outperformed both simple LSTM and GRU models on a RMSE and MAE basis. The work presented in [26], to the best of these authors' knowledge, is the only example in the literature of an RNN model used to generate a forecast using a distributed sensor network and the lack of an actual solar PV array in the data set represents a gap in the established literature which this paper is intended to fill.

## 3. Methodology

### 3.1. Sensor Network

In order to fulfill the research objectives of this study, it was important to consider the extreme environment in which the sensor network would operate. The sensors needed to be survivable in harsh, arctic conditions under high winds and temperatures below -40 °C while also being cost-effective for a small, rural electric utility to install. Additionally, the remote location of the KEA PV array necessitated sensors which could communicate wirelessly and

operate under their own power. To meet these requirements, a simple design was formulated which uses a 1-watt PV panel mounted at a 30-degree tilt angle both as a power source and measurement instrument.

The sensors' data collection system was based around the RocketStream Mini Ultra™ Arduino-compatible microcontroller, a low-power specialty microcontroller based on the Arduino ATmega4808, which used a sensing resistor and its built-in voltage sensor to record the PV panel voltage and translate it into irradiance measurements. For energy storage, the sensors used a 4.7-farad supercapacitor as well as a non-rechargeable 3.6-volt lithium thionyl chloride battery for use during times when PV power was not available. A circuit diagram for the sensors is shown in Figure 1 and an exhaustive list of the materials used in the sensor design is presented in Table 1.

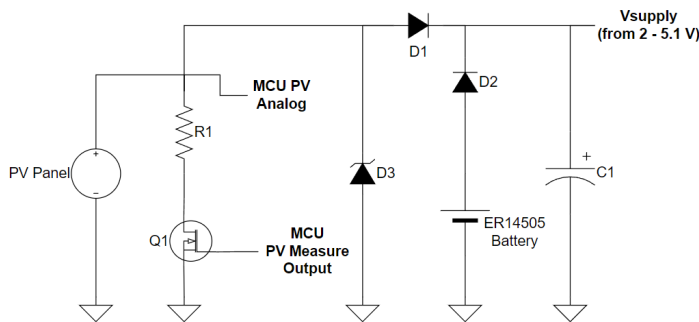


Figure 1: Internal sensor wiring schematic. "MCU" refers to the microcontroller. Component labels are referenced in Table 1.

In total, ten sensors were constructed and deployed in an approximately circular pattern around the 575-kW eastern portion of the KEA PV array. The locations of the sensors were selected based on the minimum distance required to detect a ramp event caused by a cloud 2 minutes before the event would impact the PV array. This was calculated by using historical wind speed data from the site as a surrogate for cloud speed. A cloud moving at the 99.9th percentile wind speed (21.1 m/s) would require a distance of approximately 2500 m to meet the desired 2-minute delay, so each sensor was placed as close to 2500 m from the PV array as possible given the physical accessibility of the site as well as local land use agreements. The locations of the ten sensors in reference to the KEA 575-kW PV array are shown in Figure 2.

### 3.1.1. Data Transmission and Limitations

Cellular modems are commonly employed for similar remote data collection tasks, however the extremely limited cellular coverage in the area surrounding the KEA PV array made this an infeasible option. While satellite communications do not have this geographic restriction, the high cost and power consumption of satellite modems would have necessitated a much more expensive and cumbersome sensor design. For these reasons, this study made use of short-range radio transmissions using the Long-Range Wide Area Network (LoRaWAN) protocol [27]. LoRaWAN uses a system of centralized nodes, often called "gateways," which serve as connection points between LoRaWAN-capable devices and the wider internet. For this study, one Dragino™ LG16 Indoor LoRaWAN gateway was installed at the KEA PV site (where a wired internet connection

was available) and was fitted with a Dragino™ ABL-AN-040 fiberglass extended-range outdoor antenna for improved signal reception. The sensors themselves each utilized a Seeed Grove™ LoRa-E5 LoRaWAN module to interface with the gateway and transmit data.

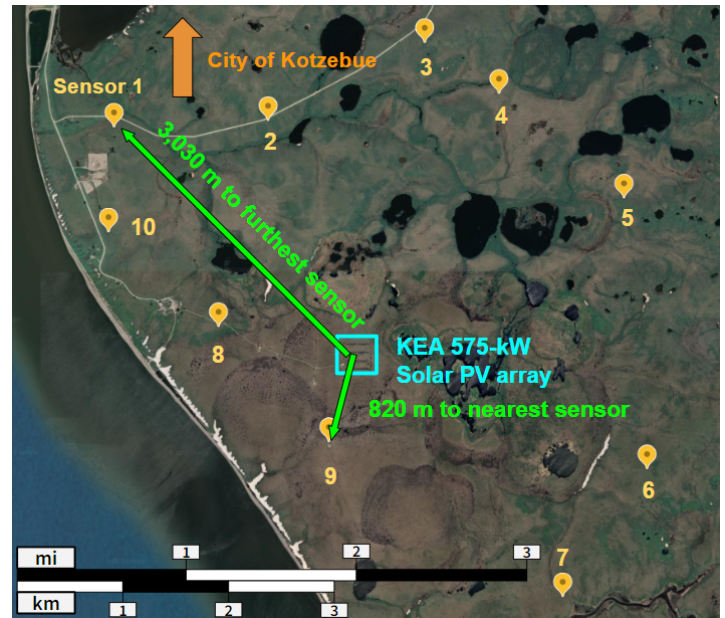


Figure 2: Map showing the locations of the sensors (labeled "1" through "10") relative to the KEA solar PV array.

While the LoRaWAN protocol does generally require much less power than competing technologies, transmitting data still accounted for the vast majority of energy used by the sensors. To ensure a reasonable service life of at least 2 years, limits were placed on the frequency of data collection and transmission. The sensors were programmed to record and store a rolling window of 10 irradiance measurements, each 2 seconds apart but only transmitted the 10 data points currently in memory when certain threshold values were met:

- Absolute mean difference between the last 10 data points is greater than  $6.5 \text{ W/m}^2$ .
- Absolute median difference between the last 10 data points is greater than  $10 \text{ W/m}^2$

These values were determined by calculating a sensor's maximum average energy usage which would still allow it to operate and transmit data using only its PV panel and supercapacitor during the day. A maximum number of transmissions per day was then calculated from this value and found to be approximately 1 transmission every 2 minutes. The maximum measured irradiance ramp rate which would result in an average of 1 ramp every 2 minutes during daylight hours was determined empirically using global horizontal irradiance (GHI) data collected from the KEA PV site. The first 10 data points (20 seconds) of each of these qualifying ramp events as well as the mean and median absolute differences between each of the first 10 data points were recorded for each ramp and the averages of these values were used to set the thresholds listed above. Transmissions were also halted entirely when the mean of the 10-measurement window was less than  $15 \text{ W/m}^2$ .

Table 1: Sensor components.

Component	Manufacturer	Model
Microcontroller	RocketScream	Mini Ultra™
LoRaWAN module	Seeed	Grove LoRa-E5
Solar panel	Voltaic Systems	1 Watt 6 Volt Solar Panel
Solar panel mounting bracket	Voltaic Systems	Solar Panel Bracket - Small
Solar panel cable extension	Voltaic Systems	Cable Extension with Exposed Leads
Supercapacitor (C1)	Eaton	PHV-5R4V474-R
Diode (D1, D2)	Diodes Incorporated	1N5817-T
Zener diode (D3)	Vishay General Semiconductor	BZX85B5V1-TR
Sensing resistor (R1)	Vishay General Semiconductor	MRS25000C6808FC100
MOSFET transistor	Infineon Technologies	IRLZ44NPBF
ER14505 battery	EVE Energy Co.	ER14505 STD
Antenna for LoRaWAN module	SparkFun Electronics	915MHz LoRa Antenna
LoRaWAN antenna adapter	Padarsey Electronics	RF U.FL(IPEX/IPX) Mini PCI to SMA Female Pigtail
Breadboard (for mounting internal components)	Adafruit	Perma-Proto Quarter-sized Breadboard
Cable tension relief glands	Lokman	PG7
Electronics housing	Polycase	An-16F
Tripod mounting bracket	Liuhé	Universal Stainless Steel Vertical Pole Mount
Mounting tripod	Davis Instruments	Weather Station Mounting Tripod

As a final precaution against excessive power draw, the sensors were programmed to be unable to transmit more than once in any 2-minute period. This impacted the ultimate quality of the data since, if another ramp event were to occur within that 2-minute waiting period, it would not be recorded. It was ultimately decided that the detection of ramp events large enough to warrant the activation of a diesel generator at a resolution greater than 2 minutes would likely not be significantly more valuable to grid operators, since the generators required to ensure grid stability would already be in the process of starting.

### 3.2. Data

Data used in this research were sourced from two distinct locations: (1) the Kotzebue solar PV site and (2) the NREL 1-second Global Horizontal Irradiance Measurement Grid in Oahu, Hawaii.

The data from the Kotzebue site can be broken down into three categories:

- 1-minute resolution alternating current (AC) power production data measured from the inverter one of eight 72-kW sub-arrays that form the eastern, 575-kW segment of the KEA PV array.
- 1-minute resolution meteorological data recorded from a meteorological station immediately adjacent to the sub-array.
- 2-second, non-continuous irradiance data from each of the 10 sensors.

PV and meteorological data at the Kotzebue site were measured every 5 seconds via the same data logger and averaged over each minute at the time of recording, leaving only the 1-minute resolution data available. Figure 3 shows an example period of approximately 70 minutes of AC solar PV power data as well as several ramp events from sensors 4 and 5, demonstrating the difference in temporal resolution and continuity.

Figure 3 shows that, while the sensors were not able to capture the entire irradiance time series, they do transmit clear upwards or downwards ramp event data in the minutes preceding the ramp experienced by the PV array.

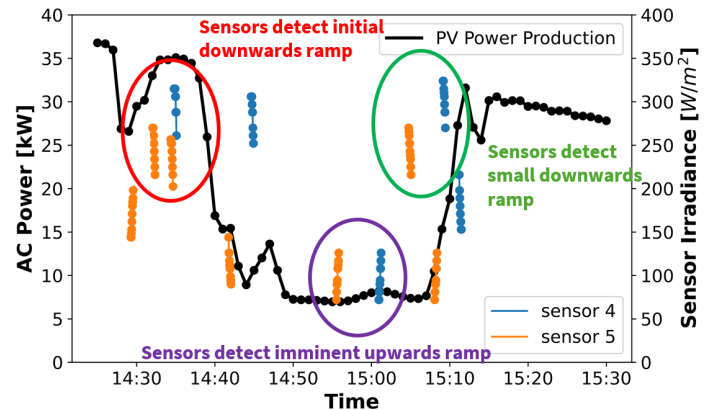


Figure 3: A section of data taken from September 13th, 2022 showing how ramp events detected by upwind sensors correspond to upcoming ramp events in PV power production.

Data from the meteorological station and KEA sub-array were collected from late 2021 through March of 2023 with 486 days of data deemed usable. While the sensor network was installed in July of 2022, only 71 days of data between August and November of 2022 were deemed to be usable due to persistent snow cover of the sensors' 1-watt reference PV panels as well as low light levels during the winter months of the 2022-2023 season.

The 1-Second Global Horizontal Irradiance Grid data set provided by NREL consists of 17 GHI sensors distributed across approximately 1 km<sup>2</sup> in Oahu, Hawaii. Data were collected from these sensors from March of 2010 through October of 2011 [28]. A number of studies have used this data set as training data for various solar nowcasting models [20, 26, 29] and it was used in this

study to validate the results from the Kotzebue site as well as to investigate the impact of differing ramp rate profiles on the efficacy of various models. The NREL Oahu data set is not a perfect analog to the Kotzebue data set as it lacks wind speed and direction data. Additionally, the sensors were placed much closer together than those deployed in Kotzebue for this study.

Due to the different sample rates between the Oahu and Kotzebue data sets, it was necessary to modify the Oahu data set to simulate the limitations of the Kotzebue sensors. First, the data were resampled to a 2-second resolution to match the internal sampling rate of the Kotzebue sensors then the ramp detection thresholds and 2-minute transmission limit described in Section 3.1 were also applied. This resulted in a discontinuous data set comparable to the Kotzebue sensor data. Additionally, since the Oahu sensor network did not include a centrally located PV array, the center-most sensor was chosen as a surrogate and its data were resampled to a 1-minute resolution to match the data from the KEA PV array.

### 3.2.1. Quality Control

For the solar PV power production data, the capacity of the sub-array inverter (67 kW) was used as a maximum reference value. All data points exceeding 67 kW were discarded.

Data were also removed based on their associated solar elevation (angle of the sun above the horizon). There is evidence to show that irradiance data with solar elevation angles less than 5° can be unreliable due to low incident angles on reference surfaces [30, 31]. In this data set, due to Kotzebue’s extreme latitude, solar elevation did not exceed 5° at all from January 1st to January 28th or from November 13th to December 31st but did not overlap with the available data period.

### 3.2.2. Data Restructuring

Data from the sensors were transmitted at a 2-second resolution with gaps between of at least 2 minutes while data from the PV sub-array and meteorological data were logged as 1-minute averaged data. A truncated example of some of the collated data from a day in September, 2022 is shown in Table 2 below.

Note that data from multiple sensors may overlap but the majority of each sensor column is empty due to the 2-minute transmission restriction. Data from the meteorological station and solar PV sub-array (the examples of power production and ambient air temperature are used in Table 2) is only timestamped each minute, leaving substantial data gaps in this unmodified state.

Neural networks of the kind used in this study cannot be trained with missing data, so data manipulation was required to produce a continuous data set. One solution considered was to resample the solar PV sub-array and meteorological data to a 2-second resolution but this proved impractical due to the gaps in the sensor data caused by the 2-minute transmission limit. It was instead decided to alter the sensor data to fit the 1-minute resolution of the remaining data fields. A diagram of this manipulation is shown in Figure 4 below.

The key insight for this manipulation is that, due to the 2-minute hard cap on sensor transmissions, any given minute can only contain, at most, one transmission per sensor. This allowed for the sensor data to be pivoted, with each of the 10 data points from each

transmission assigned their own column. In this structure, assuming an arbitrary sensor transmission is a vector of 10 data points  $[t_1, t_2, \dots, t_{10}]$ , then column "sensor 1, data point 1" contains every  $t_1$  data point from sensor 1, "sensor 1, data point 2" contains every  $t_2$  data point from sensor 1, and so on with column "sensor  $m$ , data point  $n$ " containing all  $n$ th (1-10) data points from sensor  $m$  (1-10).

Table 2: Example of collected data.

Time	Sensor 1	Sensor 2	...	PV Pwr. (kW)	Air Temp. (C)
0:00	-	-	...	52.55	13.33
0:10	288	-	...	-	-
0:12	225	-	...	-	-
0:14	225	-	...	-	-
0:16	252	-	...	-	-
0:18	243	-	...	-	-
0:20	243	-	...	-	-
0:22	234	378	...	-	-
0:24	225	387	...	-	-
0:26	225	400	...	-	-
0:28	225	414	...	-	-
0:30	-	423	...	-	-
0:32	-	432	...	-	-
0:34	-	432	...	-	-
0:36	-	432	...	-	-
0:38	-	387	...	-	-
0:40	-	378	...	-	-
0:42	-	-	...	-	-
0:44	-	-	...	-	-
0:46	-	-	...	-	-
0:48	-	-	...	-	-
0:50	-	-	...	-	-
0:52	-	-	...	-	-
0:54	-	-	...	-	-
0:56	-	-	...	-	-
0:58	-	-	...	-	-
1:00	-	-	...	54.51	13.21

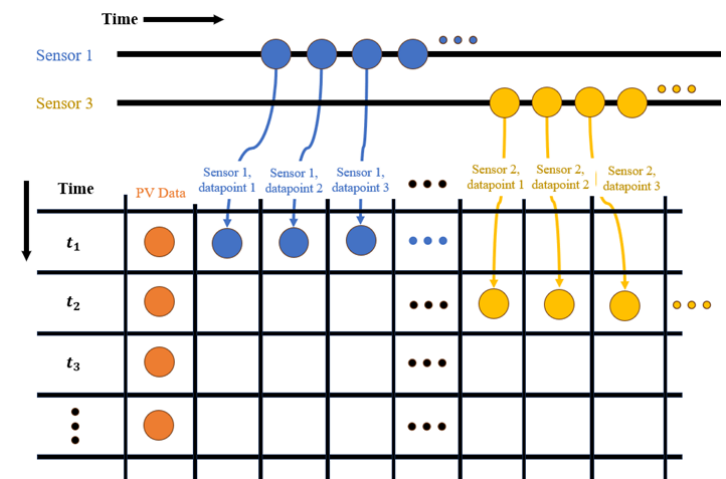


Figure 4: Example of how the sensor data is re-organized to fit into the 1-minute timestamps of the solar PV power production data set. The ramp detected by sensor 1 took place during minute 1 while the ramp detected by sensor 3 occurred in minute 2 and was thus assigned accordingly.

To preserve some additional time resolution information, the time from the beginning of the ramp event until the end of its corresponding minute was recorded and assigned its own data field for each sensor. To aid the models' ability to isolate ramp events, a binary flag field was created for each sensor to indicate if that minute contained a transmission (1 for transmission, 0 for no transmission). If a sensor did not transmit a ramp event in a given minute, these fields for that sensor were set to 0.

In total, 125 fields were used as inputs to the models:

- 10 data-point fields for each of the 10 sensors.
- 1 time-to-end-of-minute field for each sensor.
- 1 ramp-event binary flag for each sensor.
- 2 for wind direction sine and cosine components.
- 1 for wind speed.
- 1 for solar elevation to aid the models in tracking the time of day.
- 1 for the solar PV production.

Of the 71 days of available data, 47 were used for training data, 16 were used for validation, and eight days with a notable diversity of ramp profiles were not used in the training process at all and instead selected for testing.

### 3.3. Solar Variability

Forecasting weather events in general is a difficult task. Solar irradiance, and thus PV production, undergoes long-term seasonal variation due to the solar cycle and short-term variability from weather effects such as clouds. Figure 5 shows an example of four different days of measured solar PV AC power production under dramatically different cloud coverage conditions.

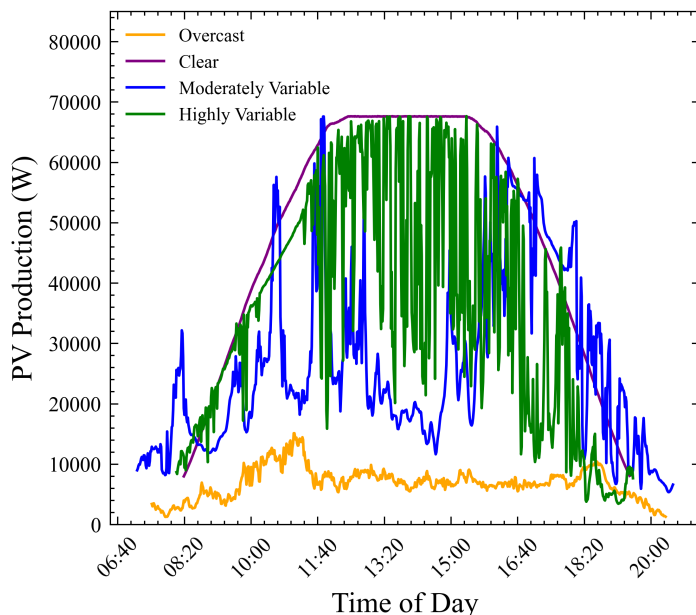


Figure 5: Examples of different daily PV production profiles from Kotzebue showing the clear difference in frequency of sudden ramp events across the four categories [32]

Assessing the variability of solar PV production on both timescales is another common tool apart from deterministic forecasts to which can help improve PV integration efficiency. While a number of metrics exist for evaluating the variability of time series data, a simple one used often when assessing PV production specifically is the Variability Index (VI) [33]. This metric was used in previous work by these authors in assessing the variability of the Kotzebue PV production data set [32].

$$VI = \frac{\sqrt{\sum_{k=2}^n (IRR_k - IRR_{k-1})^2 + \Delta t^2}}{\sqrt{\sum_{k=2}^n (CSI_k - CSI_{k-1})^2 + \Delta t^2}} \quad (1)$$

VI characterizes the variability of solar PV data over a given period of  $n$  time steps by comparing it to a corresponding clear-sky (synthetic, unobstructed PV data) over the same period. A value of 1 indicates a perfectly clear day while larger values correspond with increasing variability. References [33] and [32] used the VI metric to categorize daily variability over the course of a year. In (1) above,  $IRR_k$  is the measured irradiance at time step  $k$ ,  $CSI_k$  is the clear-sky irradiance at time step  $k$ ,  $n$  is the number of time steps, and  $\Delta t$  is the length of time between time steps.

This metric is of particular interest in this work because it provides a potential explanation for the differences in forecasting accuracy depending of the data used to train models. As outlined in Section 3.2, a comparison was done between the Kotzebue PV and sensor data and data from the NREL sensor network in Oahu. These two locations have significantly different daily and yearly weather patterns and, since the objective of this research is to develop methodologies for detecting sudden ramp events, it was deemed useful to evaluate the frequency of such events across the two data sets.

### 3.4. Artificial Neural Networks

Artificial neural networks (ANNs) refer to a wide classification of computational tools which approximate the function of a biological brain. They are composed of complex collections of simple arithmetic operations which can be optimized to produce a desired output from a given input [34]. Because they can be adapted to arbitrary problems, ANNs have found widespread use across many data-driven fields, including solar PV forecasting [35, 36, 37, 38, 39, 40, 41, 42].

ANNs are composed of interconnected structures called "neurons" which intake the weighted sum of arbitrary numerical inputs, add an arbitrary bias value, and apply a activation function (usually a function bounding between 0 and 1 or restricting to positive values). The output  $y$  of a neuron with  $n$  inputs can be written as  $y = f(\mathbf{X} \cdot \mathbf{W} + \mathbf{b})$  where  $\mathbf{X}$  is the vector of input values to the neuron  $[x_1, x_2, x_3, \dots, x_n]$ ,  $\mathbf{W}$  is the vector of weights associated with each of the input values  $[w_1, w_2, w_3, \dots, w_n]$ ,  $\mathbf{b}$  is the bias, and  $f$  is the activation function.

The values which a neuron intakes can either be raw input data such as temperature, wind speed, etc. or the output of another neuron. The ability to stack neurons into layers gives ANNs the ability to model complex tasks efficiently [43]. The layers of neurons between the input and output layers are often called "hidden layers" as their inputs and outputs are not directly captured. Figure 6 below



shows a diagram of an arbitrary ANN with its interconnected layers. For solar PV forecasting, examples of input fields might include ambient temperature, time-of-day, current solar irradiance, etc. while the output layer would necessarily be future PV production.

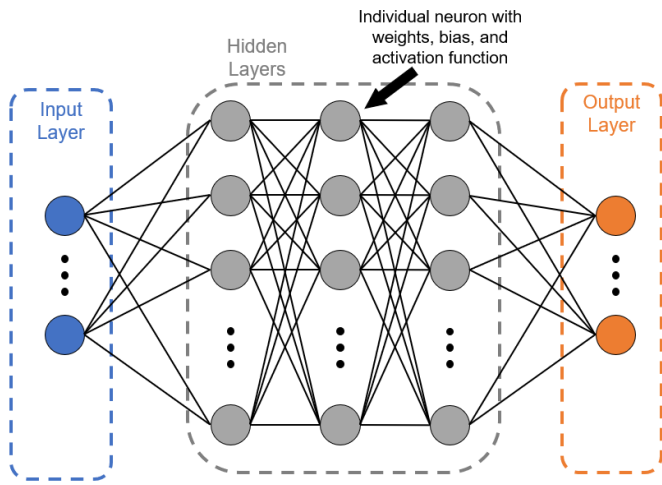


Figure 6: A diagram showing the interconnected structure of an ANN with 3 hidden layers and an arbitrary number of neurons per layer.

ANNs are optimized to learn input/output relationships in a given data set through a process called "training" during which the weights and bias of each neuron are iteratively adjusted until a stable solution is found. Training is performed through a process called "backpropagation" where, starting from the output end of the network, a gradient descent algorithm is used to find the value of each weight and bias for which some user-defined error function is minimized [44]. This procedure uses a loss function  $Q(w)$  of the form:

$$Q(w) = \frac{1}{n} \sum_{i=1}^n Q_i(w) \quad (2)$$

where  $n$  is the length of the data set and  $w$  is the weight or bias value to be optimized. An initial guess for  $w$  is selected then iteratively updated using the process:  $w = w - \alpha \nabla Q(w)$  where  $\alpha$  is a scalar factor often called the "learning rate" which modulates the size of each step down the gradient and  $\nabla$  is the gradient operator. This iteration repeats until  $w$  stabilizes at some minimum value of  $Q(w)$ , at which point the next weight or bias is adjusted. This repeats until the entire network is optimized [45].

### 3.4.1. Long Short-Term Memory

While the general ANN structure (sometimes called a dense ANN, since each neuron is connected to every neuron in the next layer) is still widely used for numerous forecasting applications, other variants exist which are specialized for specific tasks. One such variant is the RNN which is designed for sequential applications, such as time series forecasting. RNNs work by adding the weighted output of each neuron's activation function together with the weighted input from the next time step. Each neuron can be thought of as a chain of "cells" which each share the same weights and biases. Figure 7 below shows the structure of a simple RNN neuron [46].

Since each cell in the chain shares weights and a bias, an RNN which intakes  $n$  time steps will have its output values scaled by  $w_2^n$ . For even small values of  $n$ , this can quickly lead to an unstable gradient and is often referred to as the "exploding/vanishing gradient problem." This flaw makes simple RNN structures like that shown in Figure 7 unfeasible for learning the longer-term, yearly dependencies required for accurately forecasting solar PV data [47].

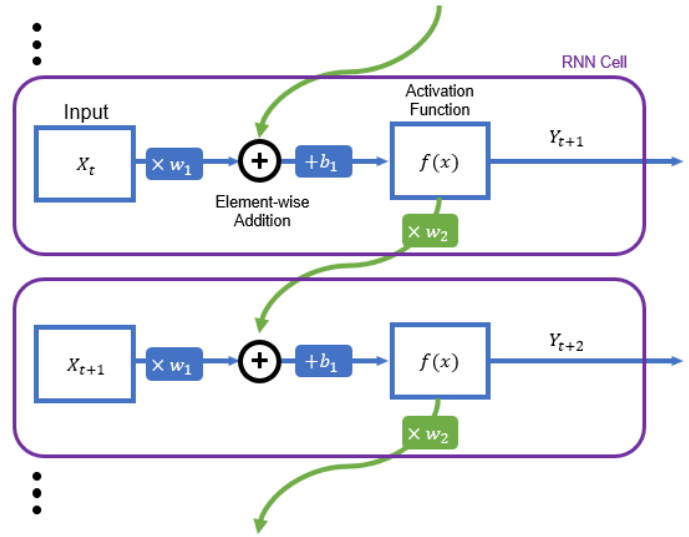


Figure 7: The chain of cells which make up an individual RNN neuron.  $X$  is the vector of input values at time  $t$ ,  $w_{1,2}$  are the weights, and  $b_1$  is the bias.

LSTM-based models have become an extremely popular choice for time series forecasting due to its ability to overcome the exploding/vanishing gradient problem [48, 49, 50]. This is achieved by a more complex cell structure which contains three sub-units called "gates," each of which contains either one or two sets of their own weights, biases, and activation functions. This structure allows the LSTM cell to maintain two internal states: the short-term and long-term memory. The short-term memory retains information about the previous time steps in the chain while the long-term memory is used to learn dependencies across much wider timescales, allowing long-term pattern information to be transferred without the gradient growing too large/small [51, 52]. The structure of an LSTM cell can be seen in Figure 8 below.

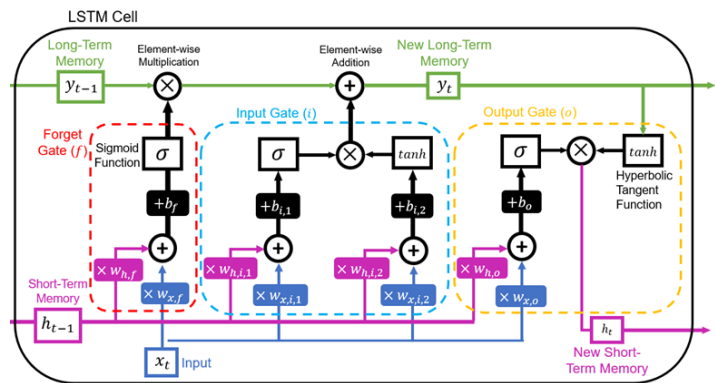


Figure 8: LSTM internal structure showing the three gates, "forget" ( $f$ ), "input" ( $i$ ), and "output" ( $o$ ) as well as their internal weights and biases.

The output of the three gates at time  $t$  can be expressed as follows:

$$f_t = \sigma(h_{t-1}w_{h,f} + x_t w_{x,f} + b_f) \quad (3)$$

$$i_t = \sigma(h_{t-1}w_{h,i,1} + x_t w_{i,1} + b_{i,1}) \cdot \tanh(h_{t-1}w_{h,i,2} + x_t w_{i,2} + b_{i,2}) \quad (4)$$

$$h_t = \sigma(h_{t-1}w_{h,o} + x_t w_{x,o} + b_o) \cdot \tanh(y_t) \quad (5)$$

Here,  $x_t$  can either be a scalar (in the case of a single variable input) or a vector of length equal to the number of input variables and the weights  $w$  and biases  $b$  are the same shape as  $x_t$ . All operations are also assumed to be element-wise across the vector. The current timestamp's long-term memory state,  $y_t$ , can be expressed as:

$$y_t = y_{t-1} \cdot f_t + i_t \quad (6)$$

Like with simple RNNs, the weights and biases inside each LSTM cell are shared between all cells in the neuron's chain. LSTM cells can also be interconnected into layers, processing multiple time steps simultaneously, and allowing for even deeper pattern recognition (see Figure 9 below).

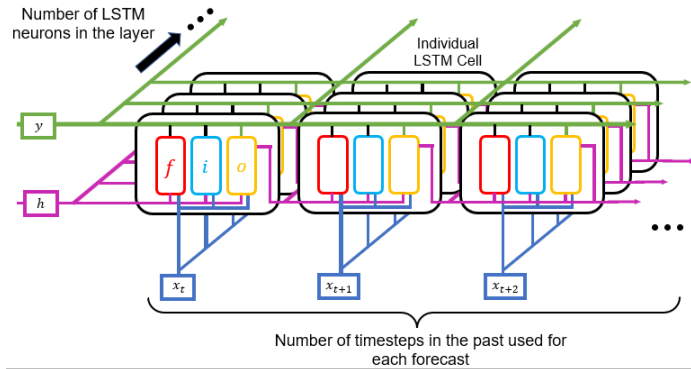


Figure 9: One layer of LSTM of arbitrary size, with the length of the chain (the number of previous time steps to be used in the next time step prediction) extending to the right. Note how the input vector  $x$  is connected to each neuron at every time step and likewise with the long-term memory  $y$  and short-term memory  $h$ .

### 3.4.2. Gated Recurrent Unit

The Gated Recurrent Unit (GRU) model is an advancement on the LSTM and is fundamentally very similar in its application with some notable differences. As opposed to the three gates present in the LSTM cell, GRU cells only have two. Additionally, the GRU design opts to forgo the two separate memory states of the LSTM in favor of one, long-term memory [53, 54]. The structure of the GRU cell is shown in Figure 10.

Here, the output states of the reset and update gates can be expressed as:

$$r_t = \sigma(x_t w_{x,r} + y_{t-1} w_{y,r} + b_r) \cdot y_{t-1} \quad (7)$$

$$z_t = \sigma(x_t w_{x,z} + y_{t-1} w_{y,z} + b_z) \cdot \tanh(r_t w_r + x_t w_x + b) \quad (8)$$

Finally, the new singular long-term state,  $y_t$  can be written as:

$$y_t = y_{t-1} \cdot (1 - \sigma(x_t w_{x,z} + y_{t-1} w_{y,z} + b_z)) + z_t \quad (9)$$

This highlights the fundamental differences between the LSTM and GRU models, primarily the fact that the GRU model cannot control the amount the previous long-term state  $y_{t-1}$  to intake into the current calculation (which LSTM is able to do with the forget gate). This potentially allows for a more nuanced ability to learn across multiple time steps. There is also the lack of short-term memory state in the GRU formulation. This affords the model notably more computational efficiency, allowing for larger models to be practical while also potentially limiting its ability to extract more detailed patterns from training data [55].

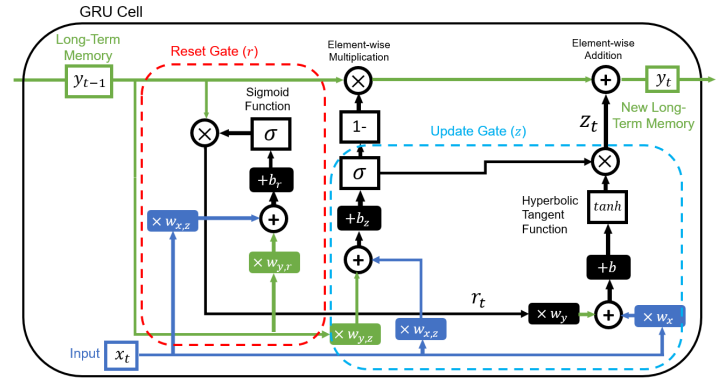


Figure 10: The internal structure of a GRU cell showing the two gates, "reset" ( $r$ ) and "update" ( $z$ ) as well as their internal weights and biases.

### 3.4.3. 1-Dimensional Convolutional

Convolutional Neural Networks (CNNs) are a subset of ANNs most commonly used in the field of image processing. They use a 2-dimensional sliding window over a data set and then feeding that window into a set of connected neurons. This same methodology can be applied, with some adjustment, to 1-dimensional sequential data, such as a time series. These 1-dimensional CNNs are often referred to as Conv1D models when used in this context. Conv1D models are particularly useful for reducing the number of input variables in large, multivariate time series as well as for determining which patterns and combinations of variables have the largest impact of accuracy. As mentioned in Section 2, they are often layered before LSTM and/or GRU layers in order to reduce the input size and improve training efficiency [56]. The structure of a Conv1D neuron is shown

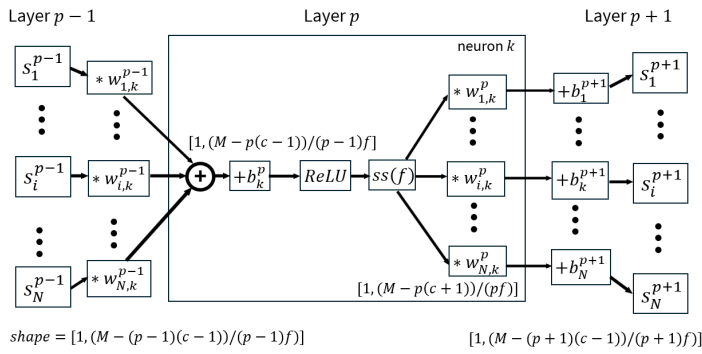


Figure 11: Diagram of three consecutive Conv1D layers showing one neuron ( $k$ ) of layer  $p$ .

Figure 11 shows the Conv1D data flow process over a data set with  $N$  data columns and an initial window of size  $M$ . The output vectors of the previous layer,  $s_i^{p-1}$  convoluted ( $*$  being the convolution operator) with "kernels" or vectors of shape  $[1, c]$  (denoted by  $w_i^{p-1}$ ). These resulting vectors are now reduced in size by  $c - 1$ , since the kernel must fit entirely within the rolling window  $M$ . A bias  $b_k^p$  is then applied followed by a rectified linear unit (ReLU) activation function. The vector is then sub-sampled (denoted by the  $ss$  operator) by a factor of  $f$  before being convoluted with the kernels of layer  $p$  and interconnected with the neurons of layer  $p + 1$ . The output of layer  $p$  for each data column  $i$  can be expressed as:

$$s_i^p = \sum_{k=1}^L (ss_f(\text{ReLU}(\sum_{i=1}^N (s_i^{p-1} * w_{i,k}^{p-1}) + b_k^p)) * w_{i,k}^p) \quad (10)$$

where  $L$  is the number of neurons in layer  $p$  and each subsequent layer reduces the vector size, thus limiting the possible combinations of kernel and window size [57].

### 3.5. Model Structures and Implementation

For this study, eight combinations of LSTM, GRU, Conv1D, and Dense layers were tested. These include: a base LSTM and GRU model, an LSTM and GRU model preceded by a Dense model, an LSTM and GRU model preceded by a Conv1D model, and a "hybrid" model, which combines the Conv1D, LSTM/GRU, and Dense models. The naming convention for these eight models is presented in Table 3 below.

Table 3: Base model structure naming conventions.

Model	LSTM	GRU
N/A	LSTM	GRU
Dense	Dense-LSTM	Dense-GRU
Conv1D	Conv1D-LSTM	Conv1D-GRU
Dense and Conv1D	Hybrid-LSTM	Hybrid-GRU

Another useful addition to machine learning models is the "dropout" layer. Not a computational layer like the Conv1D, Dense, or CNN layers, the dropout instead randomly sets a specified fraction of the weights between connected layers in a model to zero

on each training iteration. This is highly valuable to combat a phenomenon known as "overfitting" whereby a model will optimize its weights and biases to fit its training data set but will perform poorly on an as-yet unseen (or "validation") data set. This happens because the model learns patterns which are specific to the training data but cannot be generalized to future data. The dropout layer forces the model to adapt to variation in the input data set as, with the dropout layer, the model now cannot rely on any one neural pathway between layers. The generalized model structure with dropout layers is shown in Figure 12. The specific number of Conv1D, LSTM/GRU and Dense layers, as well as the fraction of each dropout layer, is specified in Section 3.7 below.

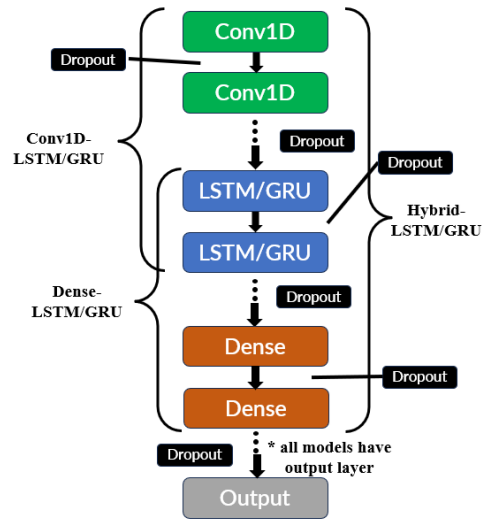


Figure 12: Illustration of the base LSTM/GRU layers and how additional layers are stacked to form a full model.

These eight models were constructed, trained, tuned, and validated using the Python library "Keras" which provides a wide variety of tools for implementing machine learning models of all kinds including neural networks. Keras provides a framework for creating hybrid neural networks in a highly modular fashion and is ideal for quickly prototyping models while also including a robust system for refining and validating model performance.

### 3.6. Performance Metrics

The metrics used to evaluate the efficacy of the various model structures were divided into two categories: (1) standard error metrics and (2) event-based metrics.

#### 3.6.1. Standard Error Metrics

Previous research into solar PV forecasting across all time horizons used some combination of the standard statistical error metrics MAE, MSE, RMSE, and the coefficient of determination,  $r^2$ .

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i| \quad (11)$$

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (12)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2} \quad (13)$$

$$r^2 = 1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2} \quad (14)$$

In the above equations,  $y$  is the actual data set,  $\hat{y}$  is the forecast,  $\bar{y}$  is the mean of the actual data set, and  $N$  is the length of both  $y$  and  $\hat{y}$ . Each of these metrics quantifies the error of a forecast differently. MAE treats all individual errors proportionally while MSE penalized larger absolute differences more severely. RMSE preserves the units of the input data and is also used to calculate another metric ("skill"), which is discussed below.  $r^2$  is bounded between 0 and 1 and shows the degree to which changes in  $\hat{y}$  are determined by changed in  $y$ . In order to more directly compare results with previous work, variants of MAE and MSE, normalized Mean Absolute Error (nMAE) and normalized Mean Square Error (nMSE) were used. These simply scale the error by the maximum value of the actual data set.

$$nMAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i| / \max(y) \quad (15)$$

$$nMSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 / \max(y) \quad (16)$$

Forecast skill compares the RMSE score of  $\hat{y}$  to that of a persistence model. A persistence model, sometimes referred to as a "naive" model, is the simplest prediction algorithm possible. It simply assumes that the next data point will be the same as the previously measured data point.

$$\hat{y}_{t+H} = y_t \quad (17)$$

In (17),  $H$  is the forecast horizon, or the number of steps ahead to be forecast. The most commonly used skill definition uses RMSE as its metric.

$$skill = 1 - \frac{RMSE_{forecast}}{RMSE_{persistence}} \quad (18)$$

A score of 0 indicates that the model in question performed no better than a persistence model while negative and positive scores demonstrate worse and better relative performance respectively. While not itself useful for generating meaningful forecasts, persistence models and forecast skill are important tools for establishing baseline model performance. Data sets with large periods of little change, such as extremely sunny or extremely overcast days in the case of solar irradiance data, can lead to surprisingly good performance from a persistence model in terms of the standard error metrics.

### 3.6.2. Event-based Metrics

While the standard error metrics are useful for understanding the aggregate performance of a model, the ultimate goal of this research is to develop a deterministic forecasting method which can accurately predict individual large ramp events which would threaten the stability of a high-solar-penetration electric grid. In order to more directly measure this capability, the models presented in this paper were also evaluated on an individual event basis. This was accomplished using three standard event-based metrics: Precision (PR), Recall (RE), and  $F_\beta$ .

$$PR = \frac{hits}{hits + false\ alarms} \quad (19)$$

$$RE = \frac{hits}{hits + misses} \quad (20)$$

$$F_\beta = (1 + \beta^2) \cdot \frac{PR}{\beta^2 \cdot PR + RE} \quad (21)$$

Hits, misses, and false alarms are binary classifications of data points. PR represents a model's ability to only forecast events which actually happen while RE reflects its success at actually forecasting events in the first place. In (21),  $\beta$  is a scaling factor used to weight PR and RE differently. These metrics are commonly used in machine learning classification applications, but are rarely applied to time series forecasting. This makes sense for situations in which one is primarily concerned with cumulative performance but, considering the highly event-based nature of this research, these metrics were warranted. Additionally, while it may be that those using such a model would place more importance on either PR or RE in their calculation of  $F_\beta$ , this research evaluated all models at  $\beta = 1$ , which is written as  $F_1$ .

Applying these metrics to a time series data set first required defining hits, misses, and false alarms. Since all ramp events are relative, it was necessary to define a minimum ramp size threshold  $h$ . Additionally, while both upward and downward ramp events effect grid stability, upwards ramps can be dealt with more easily through dispatchable loads. For this reason, only downward ramp events were considered in this paper. Misses were calculated by first locating points at time  $t$  in the actual data set  $y$  where  $y_{t-1} - y_t \geq h$  and then evaluating against the forecasted data set  $\hat{y}$  to find if  $\hat{y}_t - y_t \geq h$ . False alarms were calculated in the opposite way, where all of the data points in the forecasted data set  $\hat{y}$  where  $\hat{y}_{t-1} - \hat{y}_t \geq h$  and then checking if  $y_t - \hat{y}_t \geq h$ . A hit was defined as a data point for which both  $\hat{y}_{t-1} - \hat{y}_t \geq h$  and  $y_{t-1} - y_t \geq h$ . Figure 13 shows a visual representation of how each event is classified.

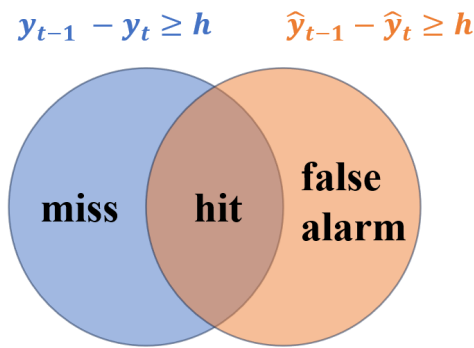


Figure 13: Venn diagram of the definition of misses, hits, and false alarms.

Under real-world conditions, the minimum downward ramp magnitude for which a forecast would be useful to a grid operator is extremely dependent on both their specific grid's configuration and current state. A small grid on a clear day with a large proportion of its power coming from solar PV is able to absorb a different relative ramp magnitude compared to a larger grid with battery storage, alternative generation sources, and low PV penetration. For this reason, PR, RE, and  $F_1$  were evaluated over a range of  $h$  values in order to explore how the models performed at predicting different ramp sizes. Plotting RE, PR, and  $F_1$  versus  $h$  can give a visual sense of model performance, but it was still deemed useful to select a consistent baseline value of  $h$  at which to evaluate the models. The 99th percentile downward ramp rate was selected, which corresponded to approximately 25.3% of maximum PV capacity per minute, and is referred to as  $h_{99}$  throughout the paper. Values of RE, PR, and  $F_1$  evaluated at  $h_{99}$  are labeled  $PR_{h_{99}}$ ,  $RE_{h_{99}}$ , and  $F_{1,h_{99}}$  respectively.

A version of forecast skill to be used with  $F_1$  was also devised in order to use a persistence model as a baseline. The  $F_1$  forecast skill,  $F_1^*$  is presented in (22) below and is similar to the standard RMSE definition shown in (18), although modified to preserve the convention of positive values indicating better performance over the persistence model.

$$F_1^* = \frac{F_{1,model}}{F_{1,persistence}} - 1 \quad (22)$$

### 3.7. Hyperparameter Tuning

The term "parameter" is often used to refer to the trainable internal weights and biases within a neural network. Similarly, the term "hyperparameter" refers to the values which define the overall architecture of the model itself, including the number of layers of each type, the number of neurons (or "units" as referred to in Keras), and learning rate  $\alpha$ . The ideal values for these hyperparameters is highly dependent on the task and there is no set rule for determining them. The model design process often involves significant iterative trial and error where one specifies a range of values for each hyperparameter, models are constructed using various combinations of those values, then tested to determine their relative accuracy. This process, often referred to as "tuning," is easily automated using Keras. The ranges of the hyperparameters tuned for each type of model layer are presented in Table 4 below.

Table 4: Hyperparameter ranges for each layer.

Parameter	Min.	Max	Step
LSTM/GRU Layers	1	5	1
LSTM/GRU Units per Layer	4	128	4
Conv1D Layers	1	5	1
Conv1D Units per Layer	4	128	4
Dense Layers	1	5	1
Dense Units per Layer	4	128	4
Dropout % Between Layers	0.0	0.3	0.05
Learning Rate ( $\alpha$ )	$1 \times 10^{-5}$	$1 \times 10^{-2}$	10

The results of the hyperparameter tuning, and thus the architectures of the eight models used in this study, are presented in Table 5. The number of values in parentheses denote the number of layers in each category and are listed in order of how the data flows through the layers of the model (i.e. the Dense-GRU model has three GRU layers, the first with 24 units, the second with 24, and the third with 12 a well as 2 Dense layers, the first with 20 units and the second with four). Each layer also has a dropout layer preceding it, the associated dropout fraction of which is also ordered sequentially in the "Dropout Layers" column.

## 4. Results and Discussion

### 4.1. Performance of the Eight Base Models

Models were constructed using Keras based on the hyperparameter tuning results and each was evaluated using the performance metrics outlined in Section 3.6. As with the hyperparameter tuning, the loss function used for all eight models was MSE. To eliminate the effect of random variation in the initial parameter guesses during training, each model was trained a total of 10 times, after which all internal parameters were reset. The models were not restricted in terms of how many full passes through the training data, or "epochs" each training iteration used, but instead a "patience" threshold was set which automatically ended the training process if three epochs occurred without any improvement in the models' MSE score. Once trained, each of the 10 iterations of the eight model architectures was used to create a 2-minute-ahead forecast for the eight test days and the performance of the resulting forecasts was evaluated using the standard and event-based metrics. The mean values of the performance metrics were calculated across the 10 iterations for each model and are presented in Table 6 below with  $h_{99}$  again equal to 25.3% of maximum PV capacity per minute.

Overall, it is apparent from Table 6 that the standard error metrics and event based metrics are not strictly linked. The basic LSTM/GRU models performed poorly across all metrics, showing a severe inability to detect ramp events at or above  $h_{99}$  and generally having high error scores across the standard metrics. The Dense-LSTM fared little better, although the Dense-GRU did show the highest  $PR_{h_{99}}$  of any model and improved standard error scores. The Conv1D-LSTM/GRU models displayed particularly interesting results, with the Conv1D-GRU showing underwhelming event-based scores and middling standard errors but the Conv1D-LSTM architecture having a substantially higher  $F_{1,h_{99}}^*$  than any other model,

Table 5: Hyperparameter tuning results.

Model	RNN Layers	Conv1D Layers	Dense Layers	Dropout Layers	Learning Rate ( $\alpha$ )
LSTM	(20, 8, 24)	N/A	N/A	(0.15, 0.15, 0.1)	$1 \times 10^{-3}$
GRU	(16, 8, 8)	N/A	N/A	(0.2, 0.2, 0.25)	$1 \times 10^{-3}$
Dense-LSTM	(28, 8)	N/A	(16, 16, 16)	(0.15, 0.15, 0.15, 0.2, 0.2)	$1 \times 10^{-3}$
Dense-GRU	(24, 24, 12)	N/A	(20, 4)	(0.25, 0.1, 0.15, 0.15, 0.15)	$1 \times 10^{-3}$
Conv1D-LSTM	(28, 24)	(36)	N/A	(0.2, 0.1, 0.2)	$1 \times 10^{-4}$
Conv1D-GRU	(32, 16, 16)	(64)	N/A	(0.15, 0.2, 0.15, 0.1)	$1 \times 10^{-4}$
Hybrid-LSTM	(32, 16)	(36)	(8, 8)	(0.1, 0.1, 0.2, 0.1, 0.15)	$1 \times 10^{-4}$
Hybrid-GRU	(28, 4, 20)	(28)	(12, 4)	(0.1, 0.15, 0.15, 0.25, 0.15, 0.2)	$1 \times 10^{-4}$

Table 6: Model performance across all metrics over the eight test days.

Model	nMAE	nMSE	$r^2$	Skill	$PR_{h99}$	$RE_{h99}$	$F_{1,h99}$	$F_{1,h99}^*$
LSTM	0.215	0.412	0.825	-0.165	0.000	0.000	0.000	-1.000
GRU	0.211	0.370	0.843	-0.104	0.000	0.000	0.000	-1.000
Dense-LSTM	0.212	0.401	0.842	-0.100	0.000	0.000	0.000	-1.000
Dense-GRU	0.181	0.331	0.869	0.000	0.895	0.009	0.017	-0.630
Conv1D-LSTM	0.210	0.397	0.832	-0.143	0.242	0.069	0.102	1.220
Conv1D-GRU	0.211	0.396	0.872	0.008	0.000	0.000	0.000	-1.000
Hybrid-LSTM	0.186	0.365	0.856	-0.050	0.006	0.028	0.041	-0.107
Hybrid-GRU	0.169	0.330	0.870	0.001	0.030	0.022	0.015	-0.673

although with poor results in other metrics. Finally, the Hybrid-LSTM/GRU models showed improved standard error scores but generally had skill scores around 0, indicating rough parity with persistence models.

The results in Table 6 indicate a rather large gap in performance across the models in terms of the event-based metrics. This was likely due to the relatively small size of the available data set. A downwards ramp event with a magnitude of at least  $h_{99}$  only occurred 64 times across the eight test days. This meant that missing only a few more events than another model lead to drastically lower event-based scores. The relative performance of the eight models in terms of  $F_1$  is better illustrated in Figure 14.

els'  $F_1$  score across the 10 trials along with a persistence model for downwards ramp events  $h$  of 5% to 40% of the maximum PV array capacity per minute at increments of 1% with  $h_{99}$  plotted as a reference. No ramp events over 40% per minute were recorded during any of the eight testing days. This plot shows a number of surprising results. The basic LSTM/GRU as well as the Dense-LSTM did not approach the persistence model's performance for any value of  $h$  while the Dense-GRU, Hybrid-LSTM/GRU, and Conv1D-GRU models all performed relatively similarly. Interestingly, the Conv1D-LSTM was able to consistently outperform persistence across all values of  $h$ . It is apparent from Figure 14 that there is significant variability within each model's  $F_1$  score, especially for larger values of  $h$  where there are fewer ramp events of that size.

The fact that only one of the models was able to barely outperform the persistence model across multiple values of  $h$  shows that these models are struggling to accurately forecasting significant ramp events when they happen. The standard error metrics are also somewhat high compared to similar studies in the literature [10, 20, 58]. One potential reason for the former is the use of MSE as the loss function for the model training process. While MSE does penalize larger errors disproportionately, it does not specifically incentivize models to train in a way which produces good event-based metric scores. This could also be impacted by the hard-coded 2-minute window between sensor transmissions, during which ramp events could be missed. This is most likely due to a general lack of quality training data.

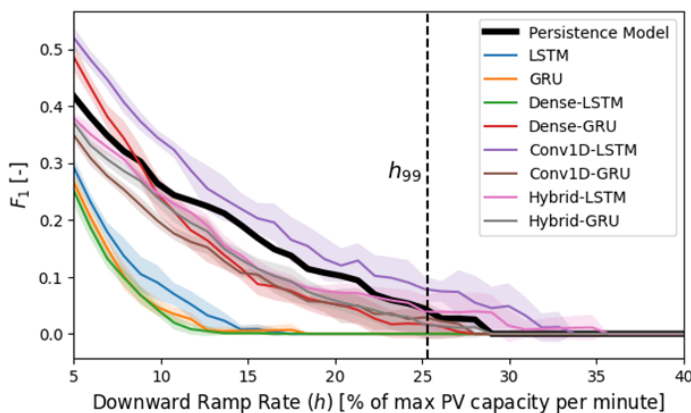


Figure 14: Mean  $F_1$  score versus downward ramp rate magnitude ( $h$ ) distribution for the eight base models over 10 trials. Shaded region around the mean line represents 1 standard deviation.

Figure 14 shows the mean and variance of each of the 10 mod-

#### 4.2. Comparison to Alternative Data Set

While not the best performing model across all metrics, the Conv1D-LSTM's notable improvement over other models in terms of  $F_1$  warranted further testing. As mentioned in Section 3.2, data from

NREL’s Global Horizontal Irradiance Grid sensor network in Oahu, HI was used as a comparison to the Kotzebue PV production data. The same Conv1D-LSTM architecture was used with the 570 days of available data from Oahu, with similar proportions of training, validation, and testing data. This is not an ideal comparison due to the Oahu data set (1) not having associated wind speed and direction data, which was an input into the Kotzebue models, and (2) the data from Kotzebue being production data from a PV array compared to the pyranometer measurements from Oahu. Since it takes time for a cloud to move across a PV array, ramp events happen slower when compared to a point-source measurement, like a pyranometer, where the cloud edge impact happens all at once. This spatial smoothing effect has been observed across a wide variety of locations, including Kotzebue [32, 59, 60].

To create a more fair comparison between the Oahu and Kotzebue data sets, the Conv1D-LSTM model architecture was trained on the same Kotzebue data as before minus the wind speed and direction variables. The models were again fully re-trained a total of 10 times and the results across all metrics were averaged. The  $F_1$  plot of these is shown below in Figure 15.

It is apparent in Figure 15 that the Conv1D-LSTM trained on and targeting the Oahu data was much less successful in terms of  $F_1$  compared to its respective persistence model but also showed significantly less variance between trials than the Kotzebue Conv1D-LSTM. The lower variance was likely due to the proportionally larger data set from Oahu while the reduced relative performance can likely be explained using the variability index  $VI$  outlined in Section 3.3. The daily  $VI$  was calculated for both the Kotzebue PV production and the central Oahu PV-surrogate pyranometer. To obtain a better sense of the variability in Kotzebue, 286 days within calendar year 2022 were used. These included the 71 days for which there was also usable sensor array data.

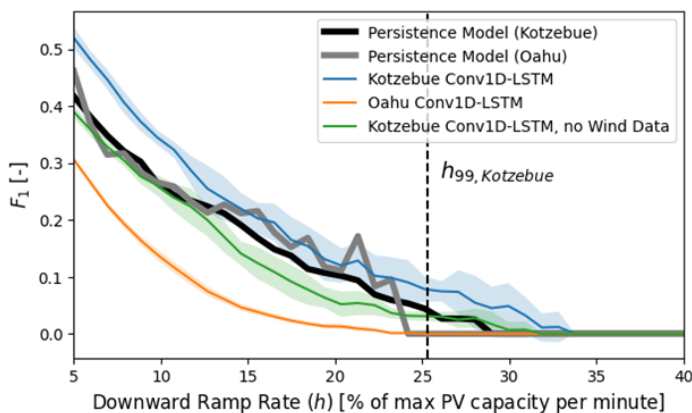


Figure 15: Mean  $F_1$  score versus downward ramp rate magnitude ( $h$ ) distribution for the base Kotzebue Conv1D- and no-wind Conv1D-LSTM models from both Kotzebue and Oahu. Shaded region around the mean line represents 1 standard deviation.

Figure 16 also includes  $VI$  values for a plane-of-array (POA) reference cell located at the Kotzebue PV array over the same 286-day time span. This reference cell, like a pyranometer, takes single point measurements and thus should show more variability than the PV array production due to spatial smoothing. This is evident in Figure 16 but it is also clear that the Oahu data set contains significantly

more days of higher variability than either of the Kotzebue data sets. This indicates that the increased variability in Oahu is not simply due to a lack of spatial smoothing and is instead a result of local weather patterns.

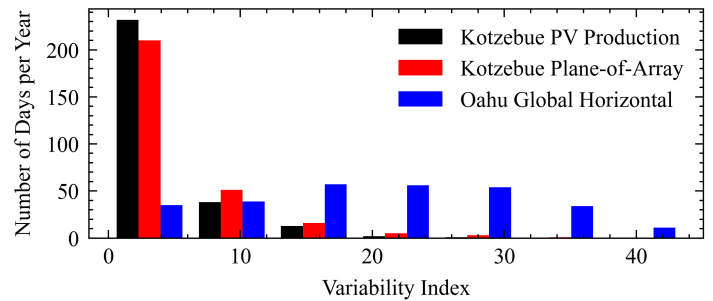


Figure 16: Histogram showing the relative frequency of various  $VI$  values between the Kotzebue and Oahu data sets.

It is possible that there are so many sudden ramp events in the Oahu data set that a persistence model is able to achieve a relatively high  $F_1$  score simply because ramp events happen to occur within the 2-minute time horizon of each other very frequently. It is also likely that the Conv1D-LSTM architecture model learned that optimizing for MSE could be achieved without trying to match every local minima/maxima. This is supported by the standard error metrics presented in Table 7 below which show that the Oahu Conv1D-LSTM model performed significantly better than either of the Kotzebue models and even achieved a positive skill.

Table 7: Mean statistical metrics over 10 trials for Kotzebue Conv1D-LSTM and no-wind Conv1D-LSTM models from both Kotzebue and Oahu.

Model	nMAE	nMSE	$r^2$	Skill
Base Conv1D-LSTM	0.210	0.397	0.832	-0.143
Oahu Conv1D-LSTM	0.115	0.274	0.88	0.142
No-Wind Kotzebue Conv1D-LSTM	0.178	0.348	0.863	-0.025

## 5. Conclusion

The aim of this research was to develop and test a method for predicting sudden ramp events in solar PV production data at a 2-minute time horizon. To this end, a network of 10 solar irradiance sensors was deployed around a utility-scale solar PV array in Kotzebue, Alaska, just above the Arctic Circle. Data from these sensors was used to detect incoming changes caused by passing clouds via a number of deep neural network models, all of which were evaluated using both standard error metrics such as MSE, MAE, and forecast skill as well as an application of the event-based metrics PR, RE, and  $F_1$ . A total of eight models were tested, combining the widely used RNN models LSTM and GRU with 1-dimensional convolutional (Conv1D) and fully connected (dense) networks into various hybrid models. While most of the models showed middling results in terms of the standard error metrics and largely failed to outperform a persistence model in terms of  $F_1$  across a range of minimum ramp magnitudes, the combined Conv1D-LSTM model show notable improvement over the others in the  $F_1$  case, although its

mediocre standard error metric scores are not enough to definitively conclude that it is the best model for this specific application.

It is likely that a model's ability to forecast ramp events is related to the frequency and severity of ramp events. When the  $VI$  metric was applied to both the Kotzebue and Oahu data sets, it is apparent that the Oahu site experiences significantly more extreme ramp events over a similar time period when compared to the Kotzebue site. This likely both impedes the models' ability to accurately learn what constitutes a ramp event as well as gives the persistence models an advantage since shifting the data set by 2 data points is more likely to coincidentally line up with existing ramp events. This performance difference is reflected in the event-based metrics for the Oahu data set as compared to Kotzebue.

There were two notable limitations with this research: (1) the minimum 2-minute window between sensor transmissions, during which time ramp events could be missed, and (2) the general lack of available training data. It is possible that the models were simply unable to reliably learn patterns between the sensor, meteorological, and PV production data sets but would see improved performance with additional data. It is also likely that the failure of most of the models to outperform persistence on an  $F_1$  basis was due to the use of MSE (or any standard error metric for that matter) as the loss function. MSE and functions like it are not designed to optimize a model for event-based forecasting. Unfortunately, RE, PR, and  $F_1$  use discrete values and as such are non-differentiable, meaning they cannot be used as a loss function.

This research provides two main contributions to the field of solar PV nowcasting. For one, it represents by far the highest-latitude deployment of a distributed irradiance sensor network yet tested. It also marks a step forward in evaluating the ability of nowcasting models to accurately and reliably predict sudden changes which might threaten grid stability. While there has been great success in the past two decades in PV forecasting of all kinds according to the standard error metrics, very little analysis has been presented which quantifies the number of critical events captured by models. The few studies which do either evaluate these metrics over too long of a time horizon for this specific application, only evaluate them at one minimum ramp magnitude  $h$ , or both [61, 62, 63].

Lastly, assessing the capabilities of artificial intelligence (AI) across various domains has emerged as a vibrant research area. Particularly in the Arctic, AI is proving to be a vital tool for exploring the region's future and supporting its communities amidst climate change [64, 65]. Further research is essential to fully harness AI's potential in this context. By evaluating multiple AI models towards Arctic energy challenges, this paper has laid the groundwork for numerous research endeavors in this direction, paving the way for a deeper understanding of AI's applications in the Arctic and beyond.

**Conflict of Interest** The authors declare no conflict of interest.

**Acknowledgment** This work is part of the Arctic Regional Collaboration for Technology Innovation and Commercialization (ARCTIC) Program, an initiative supported by the Office of Naval Research (ONR) Award #N00014-18-S-B001. This paper is also supported in part by U.S. National Science Foundation's Major Research Instrumentation grant (Award #2320196) and EPSCoR Research Infrastructure Improvement Track 4 grant (Award #2327456).

The authors of this paper extend our deepest thanks to the Kikiktagruk Inupiat Corporation who generously allowed the sensor network to be installed on their land, to the Kotzebue Electric Association (KEA) for the use of their PV production data, and to the community of Kotzebue as a whole for making this project possible. We also thank Alan Mitchell of Analysis North for his invaluable help in designing the sensors as well as Christopher Pike of the Alaska Center for Energy and Power (University of Alaska Fairbanks) for installing and managing the KEA PV data collection system.

## References

- [1] G. Masson, E. Bosch, A. V. Rechem, M. de l'Epine, "Snapshot of Global PV Markets 2023," 2023.
- [2] R.-E. Precup, T. Kamal, S. Z. Hassan, editors, *Solar Photovoltaic Power Plants: Advanced Control and Optimization Techniques*, Power Systems, Springer Singapore, Singapore, 1st edition, 2019, doi:10.1007/978-981-13-6151-7, published: 07 February 2019 (eBook), 20 February 2019 (Hardcover).
- [3] J. Marcos, L. Marroyo, E. Lorenzo, D. Alvira, E. Izco, "Power output fluctuations in large scale pv plants: One year observations with one second resolution and a derived analytic model," *Progress in Photovoltaics: Research and Applications*, **19**(2), 218–227, 2010, doi:10.1002/pip.1016.
- [4] S. Abdollahy, A. Mammoli, F. Cheng, A. Ellis, J. Johnson, "Distributed compensation of a large intermittent energy resource in a distribution feeder," in 2013 IEEE PES Innovative Smart Grid Technologies Conference (ISGT), 1–6, 2013, doi:10.1109/ISGT.2013.6497911.
- [5] R. van Haaren, M. Morjaria, V. Fthenakis, "Empirical assessment of short-term variability from utility-scale solar PV plants," *Progress in Photovoltaics: Research and Applications*, **22**(5), 548–559, 2012, doi:10.1002/pip.2302.
- [6] H. M. Diagne, P. Lauret, M. David, "Solar irradiation forecasting: state-of-the-art and proposition for future developments for small-scale insular grids," in WREF 2012 - World Renewable Energy Forum, Denver, United States, 2012.
- [7] P. Nikolaidis, A. Poullikkas, "A novel cluster-based spinning reserve dynamic model for wind and PV power reinforcement," *Energy*, **234**, 121270, 2021, doi:https://doi.org/10.1016/j.energy.2021.121270.
- [8] N. Green, M. Mueller-Stoffels, E. Whitney, "An Alaska case study: Diesel generator technologies," *Journal of Renewable and Sustainable Energy*, **9**(6), 061701, 2017, doi:10.1063/1.4986585.
- [9] C. S. McCallum, N. Kumar, R. Curry, K. McBride, J. Doran, "Renewable electricity generation for off grid remote communities; Life Cycle Assessment Study in Alaska, USA," *Applied Energy*, **299**, 117325, 2021, doi:https://doi.org/10.1016/j.apenergy.2021.117325.
- [10] S. Achleitner, A. Kamthe, T. Liu, A. E. Cerpa, "SIPS: Solar Irradiance Prediction System," in IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks, 225–236, 2014, doi:10.1109/IPSIN.2014.6846755.
- [11] A. Hussain, V.-H. Bui, H.-M. Kim, "Microgrids as a resilience resource and strategies used by microgrids for enhancing resilience," *Applied energy*, **240**, 56–72, 2019, doi:10.1016/j.apenergy.2019.02.055.
- [12] A. Lagrange, M. de Simón-Martín, A. González-Martínez, S. Bracco, E. Rosales-Asensio, "Sustainable microgrids with energy storage as a means to increase power resilience in critical facilities: An application to a hospital," *International Journal of Electrical Power & Energy Systems*, **119**, 105865, 2020, doi:10.1016/j.ijepes.2020.105865.
- [13] D. S. Kumar, G. M. Yagli, M. Kashyap, D. Srinivasan, "Solar irradiance resource and forecasting: a comprehensive review," *IET Renewable Power Generation*, **14**(10), 1641–1656, 2020, doi:10.1049/iet-rpg.2019.1227.



- [14] P. Sukič, G. Štumberger, "Intra-minute cloud passing forecasting based on a low cost IoT sensor—A solution for smoothing the output power of PV power plants," *Sensors*, **17**(5), 1116, 2017, doi:10.3390/s17051116.
- [15] V. P. Lonij, A. E. Brooks, A. D. Cronin, M. Leuthold, K. Koch, "Intra-hour forecasts of solar power production using measurements from a network of irradiance sensors," *Solar energy*, **97**, 58–66, 2013, doi:10.1016/j.solener.2013.08.002.
- [16] J. L. Bosch, J. Kleissl, "Cloud motion vectors from a network of ground sensors in a solar power plant," *Solar Energy*, **95**, 13–20, 2013, doi:10.1016/j.solener.2013.05.027.
- [17] R. H. Inman, H. T. Pedro, C. F. Coimbra, "Solar forecasting methods for renewable energy integration," *Progress in Energy and Combustion Science*, **39**(6), 535–576, 2013, doi:10.1016/j.peccs.2013.06.002.
- [18] C. W. Chow, B. Urquhart, M. Lave, A. Dominguez, J. Kleissl, J. Shields, B. Washom, "Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed," *Solar Energy*, **85**(11), 2881–2893, 2011, doi:10.1016/j.solener.2011.08.025.
- [19] R. Marquez, C. F. Coimbra, "Intra-hour DNI forecasting based on cloud tracking image analysis," *Solar Energy*, **91**, 327–336, 2013, doi:10.1016/j.solener.2012.09.018.
- [20] D. Yang, Z. Ye, L. H. I. Lim, Z. Dong, "Very short term irradiance forecasting using the lasso," *Solar Energy*, **114**, 314–326, 2015, doi:10.1016/j.solener.2015.01.016.
- [21] M. Jaihuni, J. K. Basak, F. Khan, F. G. Okyere, T. Sihalath, A. Bhujel, J. Park, D. H. Lee, H. T. Kim, "A novel recurrent neural network approach in forecasting short term solar irradiance," *ISA Transactions*, **121**, 63–74, 2022, doi:https://doi.org/10.1016/j.isatra.2021.03.043.
- [22] P. Kumari, D. Toshniwal, "Deep learning models for solar irradiance forecasting: A comprehensive review," *Journal of Cleaner Production*, **318**, 128566, 2021, doi:10.1016/j.jclepro.2021.128566.
- [23] S. Mishra, P. Palanisamy, "An Integrated Multi-Time-Scale Modeling for Solar Irradiance Forecasting Using Deep Learning," *CoRR*, **abs/1905.02616**, 2019.
- [24] A. Mellit, A. M. Pavan, V. Lughi, "Deep learning neural networks for short-term photovoltaic power forecasting," *Renewable Energy*, **172**, 276–288, 2021, doi:10.1016/j.renene.2021.02.166.
- [25] A. Bhatt, W. Ongsakul, J. G. Singh, et al., "Sliding window approach with first-order differencing for very short-term solar irradiance forecasting using deep learning models," *Sustainable Energy Technologies and Assessments*, **50**, 101864, 2022, doi:10.1016/j.seta.2021.101864.
- [26] X. Jiao, X. Li, D. Lin, W. Xiao, "A graph neural network based deep learning predictor for spatio-temporal group solar irradiance forecasting," *IEEE Transactions on Industrial Informatics*, **18**(9), 6142–6149, 2022, doi:10.1109/TII.2021.3133289.
- [27] J. Haxhibeqiri, E. De Poorter, I. Moerman, J. Hoebeke, "A survey of Lo-RaWAN for IoT: From technology to application," *Sensors*, **18**(11), 3995, 2018, doi:10.3390/s18113995.
- [28] M. Sengupta, A. Andreas, "Oahu solar measurement grid (1-year archive): 1-second solar irradiance; Oahu, Hawaii (data)," 2010.
- [29] A. W. Aryaputera, D. Yang, L. Zhao, W. M. Walsh, "Very short-term irradiance forecasting at unobserved locations using spatio-temporal kriging," *Solar Energy*, **122**, 1266–1278, 2015, doi:10.1016/j.solener.2015.10.023.
- [30] J. S. Stein, C. W. Hansen, M. J. Reno, "Global horizontal irradiance clear sky models: implementation and analysis," Technical report, Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA, 2012.
- [31] V. Lara-Fanego, J. Ruiz-Arias, D. Pozo-Vázquez, F. Santos-Alamillos, J. Tovar-Pescador, "Evaluation of the WRF model solar irradiance forecasts in Andalusia (southern Spain)," *Solar Energy*, **86**(8), 2200–2217, 2012, doi:10.1016/j.solener.2011.02.014.
- [32] H. Toal, A. K. Das, "Variability and Trend Analysis of a Grid-Scale Solar Photovoltaic Array above the Arctic Circle," in 2023 IEEE 24th International Conference on Information Reuse and Integration for Data Science (IRI), 242–247, 2023, doi:10.1109/IRI58017.2023.00049.
- [33] J. Stein, C. Hansen, M. J. Reno, "The variability index: A new and novel metric for quantifying irradiance and PV output variability," Technical report, Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA, 2012.
- [34] B. Yegnanarayana, *Artificial neural networks*, PHI Learning Pvt. Ltd., 2009.
- [35] V. Z. Antonopoulos, D. M. Papamichail, V. G. Aschonitis, A. V. Antonopoulos, "Solar radiation estimation methods using ANN and empirical models," *Computers and Electronics in Agriculture*, **160**, 160–167, 2019, doi:10.1016/j.compag.2019.03.022.
- [36] V. Srikrishnan, G. S. Young, L. T. Witmer, J. R. Brownson, "Using multipyranometer arrays and neural networks to estimate direct normal irradiance," *Solar Energy*, **119**, 531–542, 2015, doi:10.1016/j.solener.2015.06.004.
- [37] F.-V. Gutierrez-Corea, M.-A. Manso-Callejo, M.-P. Moreno-Regidor, M.-T. Manrique-Sancho, "Forecasting short-term solar irradiance based on artificial neural networks and data from neighboring meteorological stations," *Solar Energy*, **134**, 119–131, 2016, doi:10.1016/j.solener.2016.04.020.
- [38] H. T. Pedro, C. F. Coimbra, "Assessment of forecasting techniques for solar power production with no exogenous inputs," *Solar Energy*, **86**(7), 2017–2028, 2012, doi:10.1016/j.solener.2012.04.004.
- [39] S. Z. Hassan, H. Li, T. Kamal, M. Nadarajah, F. Mehmood, "Fuzzy embedded MPPT modeling and control of PV system in a hybrid power system," in 2016 International Conference on Emerging Technologies (ICET), 1–6, 2016, doi:10.1109/ICET.2016.7813236.
- [40] A. Abbas, N. Mughees, A. Mughees, A. Mughees, S. Yousaf, S. Z. Hassan, F. Sohail, H. Rehman, T. Kamal, M. A. Khan, "Maximum Power Harvesting using Fuzzy Logic MPPT Controller," in 2020 IEEE 23rd International Multitopic Conference (INMIC), 1–6, 2020, doi:10.1109/INMIC50486.2020.9318188.
- [41] S. Z. Hassan, H. Li, T. Kamal, U. Arifoğlu, S. Mumtaz, L. Khan, "Neuro-Fuzzy Wavelet Based Adaptive MPPT Algorithm for Photovoltaic Systems," *Energies*, **10**(3), 2017, doi:10.3390/en10030394.
- [42] S. Mumtaz, S. Ahmad, L. Khan, S. Ali, T. Kamal, S. Z. Hassan, "Adaptive Feedback Linearization Based NeuroFuzzy Maximum Power Point Tracking for a Photovoltaic System," *Energies*, **11**(3), 2018, doi:10.3390/en11030606.
- [43] C. M. Bishop, *Neural networks for pattern recognition*, Oxford university press, 1995.
- [44] A. Goh, "Back-propagation neural networks for modeling complex systems," *Artificial Intelligence in Engineering*, **9**(3), 143–151, 1995, doi:10.1016/0954-1810(94)00011-S.
- [45] P. Baldi, "Gradient descent learning algorithm overview: A general dynamical systems perspective," *IEEE Transactions on neural networks*, **6**(1), 182–195, 1995, doi:10.1109/72.363438.
- [46] I. Sutskever, J. Martens, G. E. Hinton, "Generating text with recurrent neural networks," in Proceedings of the 28th international conference on machine learning (ICML-11), 1017–1024, 2011.
- [47] Y. Bengio, P. Simard, P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, **5**(2), 157–166, 1994, doi:10.1109/72.279181.
- [48] S. Siami-Namini, N. Tavakoli, A. S. Namin, "The performance of LSTM and BiLSTM in forecasting time series," in 2019 IEEE International conference on big data (Big Data), 3285–3292, IEEE, 2019, doi:10.1109/BigData47090.2019.9005997.
- [49] A. Yadav, C. Jha, A. Sharan, "Optimizing LSTM for time series prediction in Indian stock market," *Procedia Computer Science*, **167**, 2091–2100, 2020, doi:10.1016/j.procs.2020.03.257.

- [50] X. Qing, Y. Niu, "Hourly day-ahead solar irradiance prediction using weather forecasts by LSTM," *Energy*, **148**, 461–468, 2018, doi:10.1016/j.energy.2018.01.177.
- [51] S. Hochreiter, J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, **9**(8), 1735–1780, 1997, doi:10.1162/neco.1997.9.8.1735.
- [52] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, **404**, 132306, 2020, doi:10.1016/j.physd.2019.132306.
- [53] R. Dey, F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," in 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS), 1597–1600, IEEE, 2017, doi:10.1109/MWSCAS.2017.8053243.
- [54] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, J. Wang, "Machine health monitoring using local feature-based gated recurrent unit networks," *IEEE Transactions on Industrial Electronics*, **65**(2), 1539–1548, 2018, doi:10.1109/TIE.2017.2733438.
- [55] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv preprint arXiv:1412.3555, 2014.
- [56] S. Kiranyaz, A. Gastli, L. Ben-Brahim, N. Al-Emadi, M. Gabbouj, "Real-time fault detection and identification for MMC using 1-D convolutional neural networks," *IEEE Transactions on Industrial Electronics*, **66**(11), 8760–8771, 2019, doi:10.1109/TIE.2018.2833045.
- [57] S. Kiranyaz, T. Ince, M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE transactions on biomedical engineering*, **63**(3), 664–675, 2016, doi:10.1109/TBME.2015.2468589.
- [58] H. Zhou, Y. Zhang, L. Yang, Q. Liu, K. Yan, Y. Du, "Short-Term Photovoltaic Power Forecasting Based on Long Short Term Memory Neural Network and Attention Mechanism," *IEEE Access*, **7**, 78063–78074, 2019, doi:10.1109/ACCESS.2019.2923006.
- [59] R. Van Haaren, M. Morjaria, V. Fthenakis, "Empirical assessment of short-term variability from utility-scale solar PV plants," *Progress in Photovoltaics: Research and Applications*, **22**(5), 548–559, 2014, doi:10.1002/pip.2302.
- [60] R. Perez, M. David, T. E. Hoff, M. Jamaly, S. Kivalov, J. Kleissl, P. Lauret, M. Perez, et al., "Spatial and temporal variability of solar energy," *Foundations and Trends® in Renewable Energy*, **1**(1), 1–44, 2016, doi:10.1561/27000000006.
- [61] M. Abuella, B. Chowdhury, "Forecasting of solar power ramp events: A post-processing approach," *Renewable Energy*, **133**, 1380–1392, 2019, doi:10.1016/j.renene.2018.09.005.
- [62] A. Sanfilippo, L. Martin-Pomares, N. Mohandes, D. Perez-Astudillo, D. Bachour, "An adaptive multi-modeling approach to solar nowcasting," *Solar Energy*, **125**, 77–85, 2016, doi:10.1016/j.solener.2015.11.041.
- [63] M. Abuella, B. Chowdhury, "Forecasting solar power ramp events using machine learning classification techniques," in 2018 9th IEEE International Symposium on Power Electronics for Distributed Generation Systems (PEDG), 1–6, IEEE, 2018, doi:10.1109/PEDG.2018.8447599.
- [64] Y. Wang, C. Purev, H. Barndt, H. Toal, J. Kim, L. Underwood, L. Avalo, A. K. Das, "Toward Energy-Efficient Deep Neural Networks for Forest Fire Detection in an Image," *The Geographical Bulletin*, **64**(2), 13, 2023.
- [65] F. Huettmann, P. Andrews, M. Steiner, A. K. Das, J. Philip, C. Mi, N. Bryans, B. Barker, "A super SDM (species distribution model) 'in the cloud' for better habitat-association inference with a 'big data' application of the Great Gray Owl for Alaska," *Scientific Reports*, **14**(1), 7213, 2024, doi:10.1038/s41598-024-57588-9.

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

## Visualization of the Effect of Additional Fertilization on Paddy Rice by Time-Series Analysis of Vegetation Indices using UAV and Minimizing the Number of Monitoring Days for its Workload Reduction

Taichi Ito<sup>1,\*</sup>, Ken'ichi Minamino<sup>1</sup>, Shintaro Umeki<sup>2</sup>

<sup>1</sup>Graduate School of Software and Information Science, Iwate Prefectural University, Takizawa, 0200611, Japan

<sup>2</sup>Hanamaki Satellite, Research Center for Industrial Science and Technology, Iwate University, Hanamaki, 0250312, Japan

### ARTICLE INFO

#### Article history:

Received: 14 March, 2024

Revised: 2 May, 2024

Accepted: 4 May, 2024

Online: 25 May, 2024

#### Keywords:

Additional Fertilization

Machine Learning

Paddy Rice

Time-Series Clustering

Unmanned Aerial Vehicle

Vegetation Index

### ABSTRACT

*This research is an extension of the research (ISEEIE 2023), which dealt with Time-Series Clustering (TSC) of Vegetation Index (VI) for paddy rice. The novelty of this research is "Visualization of growth changes before and after additional fertilization," "Analyzing the appropriate amount of additional fertilizer," and "Optimization of monitoring period to minimize the number of monitoring days for its workload reduction using Unmanned Aerial Vehicle (UAV)." For visualization of growth changes before and after fertilization, a UAV was used to obtain VIs for each mesh and make its time-series data during the monitoring period. Then, TSC was performed on the data. As a result of clustering, NDVI and NDRE increased with additional fertilizer, making it possible to visualize the fertilizer effects. For analyzing the appropriate amount of fertilizer, the its amount applied was changed for each paddy field (2.8, 3.5, 4.2g/m<sup>2</sup>). In a field experiment conducted, both the TSC results and the crop estimates by unit acreage sampling for each paddy field revealed no difference in yield among fields, indicating that the paddy field with the least fertilizer amount (2.8g/m<sup>2</sup>) is optimal. It was estimated that this would reduce nitrate nitrogen, which is harmful to soil and the human body, by 0.070mg/L. In addition, for optimization of the monitoring period, the importance of each independent variable outputted by Random Forest (RF) was used to find a subset of monitoring dates. In any VI, there is a period, determined by the range of effective accumulated temperature, when the clustering result does not change even if the number of monitoring dates is reduced (The period could be reduced from 30 to 40 days, which is particularly important for three vegetation indices). From these results, the technologies can help reduce fertilizer costs, excessive fertilization and environmental impacts, and promote the use of UAV.*

## 1. Introduction

This paper is an extension of work originally presented at the 3rd International Symposium on Electrical, Electronics, and Information Engineering (ISEEIE 2023) [1].

A long-standing problem in Japanese agriculture is the aging and decline of farmers. According to a survey, from 2015 to 2020, the number of farmers decreased by approximately 22% to 1,363,000, and the percentage of those aged 65 years or older increased by 5% to 69.6% [2]. The main causes of this are that

appropriate farm work requires tacit knowledge and much human labor. Regarding the former, appropriate work is based on the heuristics of experts; therefore, it is difficult to share. Regarding the latter, Japan's agricultural fields are small, and mechanization has not yet progressed. These problems create barriers for new farmers and lead older farmers to quit farming. To solve this problem, "Smart Agriculture" is being promoted. Specifically, production management systems and the automatic operation of agricultural machinery are expected to clarify and save labor in farm work. However, these technologies are expensive and difficult to implement by farmers. Therefore, in this study, crop monitoring using unmanned aerial vehicles (UAV) was adopted, which is more reasonable than other technologies.

Corresponding Author: Taichi Ito, Iwate Prefectural University, Takizawa, 0200611, Japan, [s236w001@s.iwate-pu.ac.jp](mailto:s236w001@s.iwate-pu.ac.jp)

In UAV monitoring, multispectral cameras are often used to obtain the sunlight reflectance of crop leaves in various bands. Generally, healthy crops have high reflectance in the near-infrared band and low reflectance in the red band. However, in stressed crops, the difference in reflectance between the two bands decreases, as shown in Figure 1 [3], [4]. Based on this reflection characteristic, many vegetation indices (VI) have been developed using calculation formulas to understand vegetation conditions [5]. For example, using VIs, studies have created rice yield maps using the Normalized Difference Vegetation Index (NDVI) and the Normalized Difference Red Edge Index (NDRE) [6], [7]. These studies have shown that the VI value in the young panicle formation stage and the heading stage are correlated with yield; therefore, yield prediction is possible. However, these predictive models cannot be applied everywhere but within the research environment. Furthermore, despite multiple monitoring sessions, the model predicted using only the VI value input during the heading stage. In Japan, yield is important for planning rice demand and supply. Therefore, early and highly accurate predictions with numerous features are required.

This study focused on additional fertilization during farming. This study is important because it directly influences growth and yield. Generally, this work is conducted between the young panicle-forming and heading stages to supply nitrogen and potassium and can be expected to increase yield by enriching the contents of ears [8]. However, excessive fertilizer causes lodging owing to overgrowth, releasing water from the roots due to osmotic pressure, and health hazards owing to nitrate-nitrogen dissolution into groundwater [9], [10]. For nitrate-nitrogen, the Japanese standard value for water pollution is set at 10 mg/L [11]. Based on

these negative effects, it is necessary to determine the appropriate amount of fertilizer and apply variable fertilization. Because the effect of farm work appears gradually, it is important to compare it before and after work. Therefore, in this study, we use a new method of time-series analysis of VIs obtained from a UAV to visualize the vegetation before and after additional fertilization and analyze the appropriate fertilization amount. Furthermore, to reduce the workload of UAV monitoring, we use random forest (RF) to minimize the monitoring period.

## 2. Method

### 2.1. Monitoring

The monitoring targeted paddy fields owned by private farmers in Hanamaki City in Iwate Prefecture, Japan, and yielded 10 analysis results for six sites from 2021 to 2023 [1], [12], [13]. Several sites were used in the experiment, and the additional fertilizer amount varied for each field. This is because growth differences were observed for each fertilizer amount.

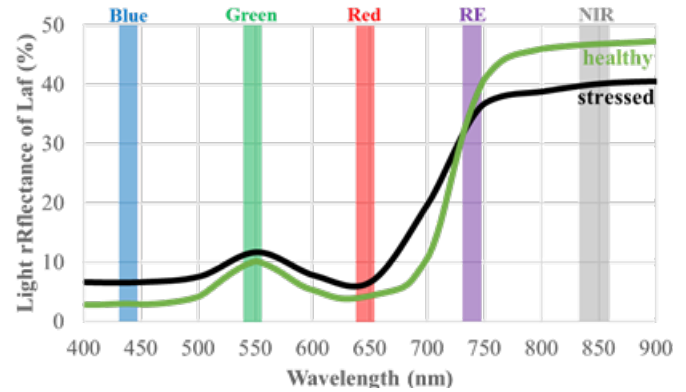


Figure 1: Spectral reflection characteristics of plants [1], [12], [13]

Table 1: Site and Cropping Information

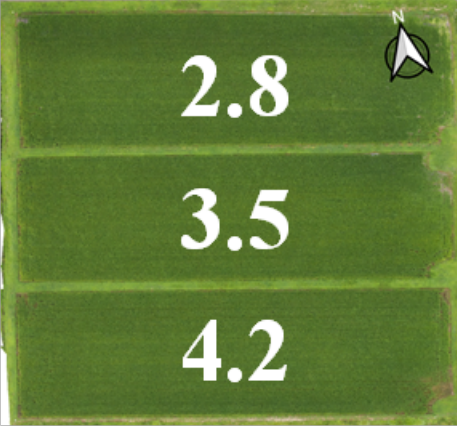
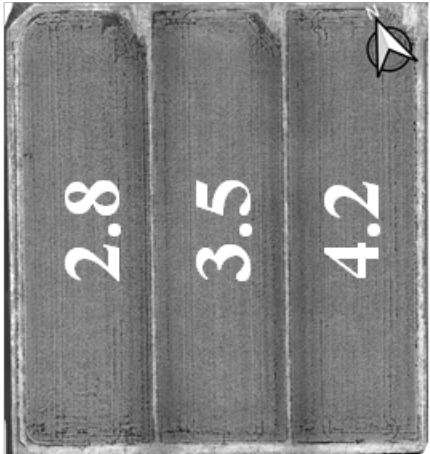
		Yuguchi-8 (2022)	Yuguchi (2023)
Variety		Yumi-azusa	Homusume-mai
Date	Number of Meshes	819	857
	Transplanting	May 7	May 5
	Additional Fertilization	Jul. 20	Middle Jul.
	Heading	Aug. 3	Late Jul.
	Harvest	Oct. 6	Late Sep.
Aerial photo. & Fertilizer Amount [ $gN/m^2$ ]			



Figure 2: UAV used for monitoring (Phantom 4 Multispectral)

Table 1 shows the sites and cropping information where the comparative fertilizer application experiments were conducted (Yuguchi-8 in 2022 and Yuguchi in 2023). The distance between the two fields was approximately 1 km; therefore, the weather conditions were similar. Both fields are adjacent to each other and can be divided into North, Center, and South in Yuguchi-8, and West, Center, and East in Yuguchi. Fertilizer amounting to 2.8, 3.5, and 4.2 gN/m<sup>2</sup> was applied in the middle of July using a pesticide spraying UAV (AGRAS MG-1, DJI Co., Ltd. [14]).

To calculate the VI, a UAV equipped with a multispectral camera is required. Thus, a UAV (Phantom 4 Multispectral, DJI Co., Ltd.), shown in Figure 2 was used. It weighs 1,487 g, has a maximum flight time of 27 min, and has a multispectral camera with one RGB sensor for visible light and five monochrome sensors for multispectral light. The bands of the monochrome sensor were 450 nm (blue), 560 nm (green), 650 nm (red), 730 nm (red edge: RE), and 840 nm (near-infrared: NIR) [15]. The flight altitude was set to 30 m, and orthogonal images of each band were created after monitoring. The VI values were then calculated from these orthoimages. The VIs calculated in this study were (1), (2), and (3).

$$NDVI = (R_{NIR} - R_{Red}) / (R_{NIR} + R_{Red}) \quad (1)$$

$$NDRE = (R_{NIR} - R_{RE}) / (R_{NIR} + R_{RE}) \quad (2)$$

$$sNDRE = (NDRE - \mu_{NDRE}^d) / \sigma_{NDRE}^d \quad (3)$$

where

$R_b$  : Reflectance of band "b"

$\mu_{NDRE}^d$  : Average of NDRE on date "d"

$\sigma_{NDRE}^d$  : Standard Deviation of NDRE on date "d"

The NDVI was used for vegetation diagnosis. This was calculated by normalizing the NIR and Red bands. It is widely used; however, it has been reported that the soil easily affects it, and the index becomes saturated during the peak vegetation season [6], [16], [17]. The NDRE was used to diagnose crop stress and nitrogen content. The RE band reaches deeper into the canopy than the Red band; therefore, stress can be detected earlier. Additionally, it is not easily saturated; therefore, depending on the season, it may be more appropriate than NDVI for diagnosing vegetation and yield [7], [17], [18]. A Standardized Normalized Difference Red Edge Index (sNDRE) was developed to ensure that the NDRE did not change unless it was under significant stress. This is the relative value for each date and is used for a relative comparison within a field [1], [12], [13]. In addition, it uses only NDRE for calculations,

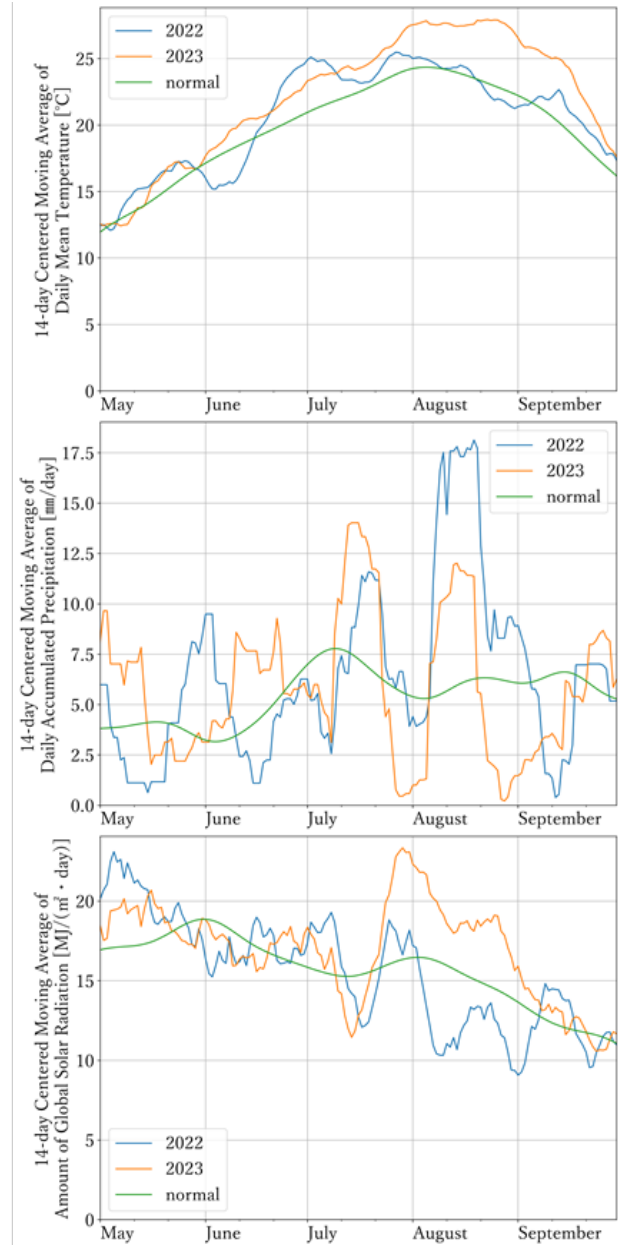


Figure 3: Meteorological data at the experimental field (Daily mean temperature, Precipitation, and Solar radiation)

and it is possible to diagnose the relative stress and nitrogen content levels in the field.

After computing these three VIs, meshes were created in the rice field portion of the orthoimage. The mesh size was 3 m square.

The VI of each mesh was determined as the average VI value of pixels within the mesh. After computing the VIs of all meshes on all monitoring dates, VI time-series data were generated for each mesh.

Figure 3 shows the 14-day centered moving average of the daily mean temperature, accumulated precipitation, and global solar radiation at Yuguchi during the rice-growing period (May to September) in 2022, 2023, and a normal year. As a characteristic point, there was heavy rainfall in August 2022 and high

temperatures in 2023. To adjust for these weather differences, monitoring dates were converted to growth stages using the Effective Accumulated Temperature (EAT) from the transplantation date and paddy rice growth model [19], [20]. Table 2 shows the correspondence between the EAT and growth stages, and Figure 4 shows the relationship between the growth stages and dates for both years.

### 2.2. Evaluation of Additional Fertilization

To visualize the fertilization effect, this research used time-series clustering (TSC) with the K-Means++ method on the VI time-series data presented in Section 2.1. Clustering usually classifies 2 or 3-dimensional data which can be shown in scatter diagrams [21]. However, this analysis aimed to classify time-series data. Therefore, a program was developed to handle time-series

data with several missing monitoring dates [1]. This program is an improved version of the function “TimeSeriesKMeans” [22] in the Python library “tslearn.” This program was used in this research to conduct TSC. The elbow method [1], [23] was used to determine the number of clusters. From the TSC results, the VI time-series transition of each cluster's centroid, cluster distribution, and cluster ratio for each paddy field were obtained. Based on these results, we determined the growth differences according to the fertilizer amount and the appropriate fertilizer amount. Furthermore, a crop estimate by unit acreage sampling was conducted immediately before harvest to confirm the consistency between the appropriate fertilizer amount determined by TSC analysis and the actual yield. The sampling method was to select five meshes per field so that the heading-stage NDVI was scattered, and five stubbles per mesh were collected. The selected meshes are listed in Table 3.

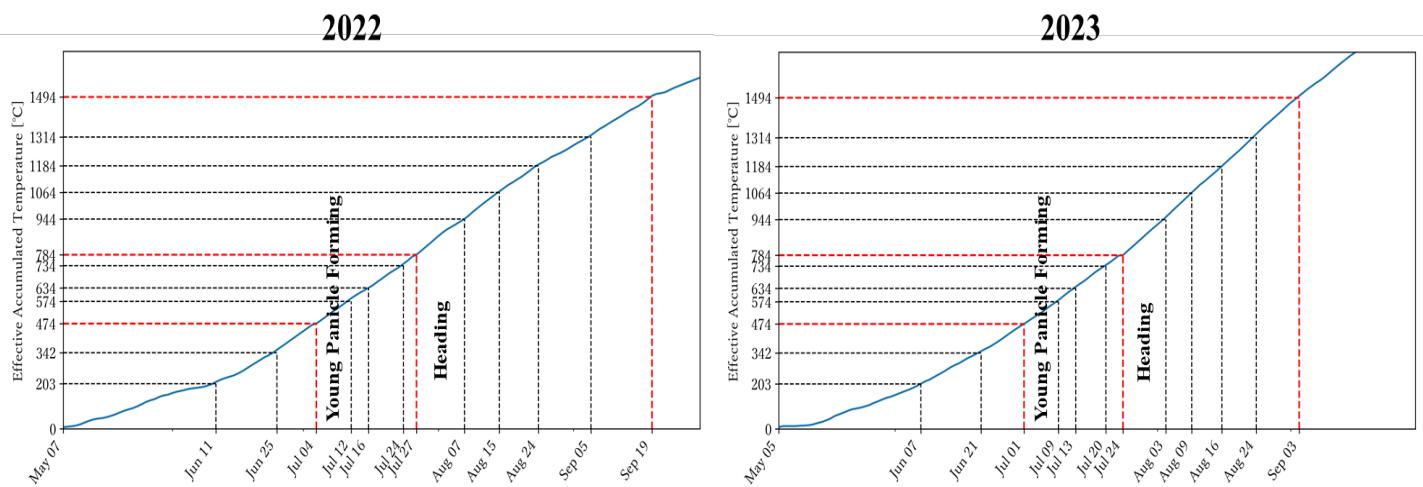


Figure 4: Relationship between the growth stages and dates in 2022 and 2023

Table 2: Correspondence between growth stage and EAT

Model	Effective & Upper limit temperature	Growth Stage	Required EAT [°C]
Leaf number on the main culm	Effective: 10°C Upper limit: 24°C	Slow leaf emergence	0
		Turning leaf emergence rate	203
		Fast leaf emergence	342
Young panicle development	Effective: 10°C	Young panicle forming	474
		Pollen mother cell differentiation	574
		Meiosis	634
		Pollen accumulation	734
		Heading	784
Brown Rice Development	Effective: 10°C	Milk Ripe	944
		Dough Ripe	1064
		Yellow Ripe	1184
		Full Ripe	1314
		Harvest	1494

### 2.3. Optimization of Monitoring Period

Previous studies have analyzed healthy vegetation growth, stress, etc. in paddy rice using the TSC of VI time-series data [12], [13]. However, UAV monitoring, such as moving to sites, preparing for flight, and computing VI values from aerial photographs, takes time. Therefore, a survey was conducted to identify the subset of monitoring dates for which the feature quantity was not reduced significantly [1]. RF was used in this survey. This was adopted because it can handle the categorical values that are the output results of TSC (i.e., cluster ID) and it outputs the importance of each independent variable (monitoring date). The execution conditions for RF are listed in Table 4 [1]. From the RF, a classifier model was created that outputs the cluster ID when given the VI time-series data of each mesh. The optimization method is shown in Figure 5. First, a classifier was created using the training data of the VI time series over the entire period (Entire RF). Second, the optimized period was determined based on the importance of the test data given to the classifier. Third, another classifier was created using the training data of the VI time series values in the optimized period (Optimized RF). Fourth, the cluster ID match percentages were calculated when classifying the test data between the TSC and Entire RF ( $pppp_{RRReee}$ ) and between the TSC and Optimized RF ( $pppp_{ooooee}$ ). Finally, the McNemar's test was performed with the null hypothesis as " $pp_{RRReee} = ppp_{ooooee}$ "

and it was evaluated that optimization was successful if the null hypothesis was not rejected. Using this method, the optimized period was determined from the monitoring data of eight sites (excluding the two sites in Table 1 from the monitoring data of all ten sites). This confirmed the consistency between the appropriate fertilizer amount using TSC analysis results in the optimized period and the actual yield.

### 3. Results

#### 3.1. Comparison of Additional Fertilizer Amounts

Table 5 presents the yields of the five sampled meshes at both sites. Sample 5 from West of Yuguchi was missing because the survey was discontinued. The sampled yield was calculated as the weight of five stubbles. These units were converted to SI units, and the average for each rice field is shown in the second line from the bottom. The bottom line shows the p-value of the Kruskal-Wallis test (non-parametric, unpaired, differences between three or more groups) with the null hypothesis set as "the average of sampled yields of all rice fields are equal." As a result, the null hypothesis was adopted even at the 10% significance level, as the p-value for both sites significantly exceeded 0.1.

For the entire TSC period, the time-series transitions in each cluster centroid, cluster ID distribution map, and percentage of

Table 3: Sampled meshes

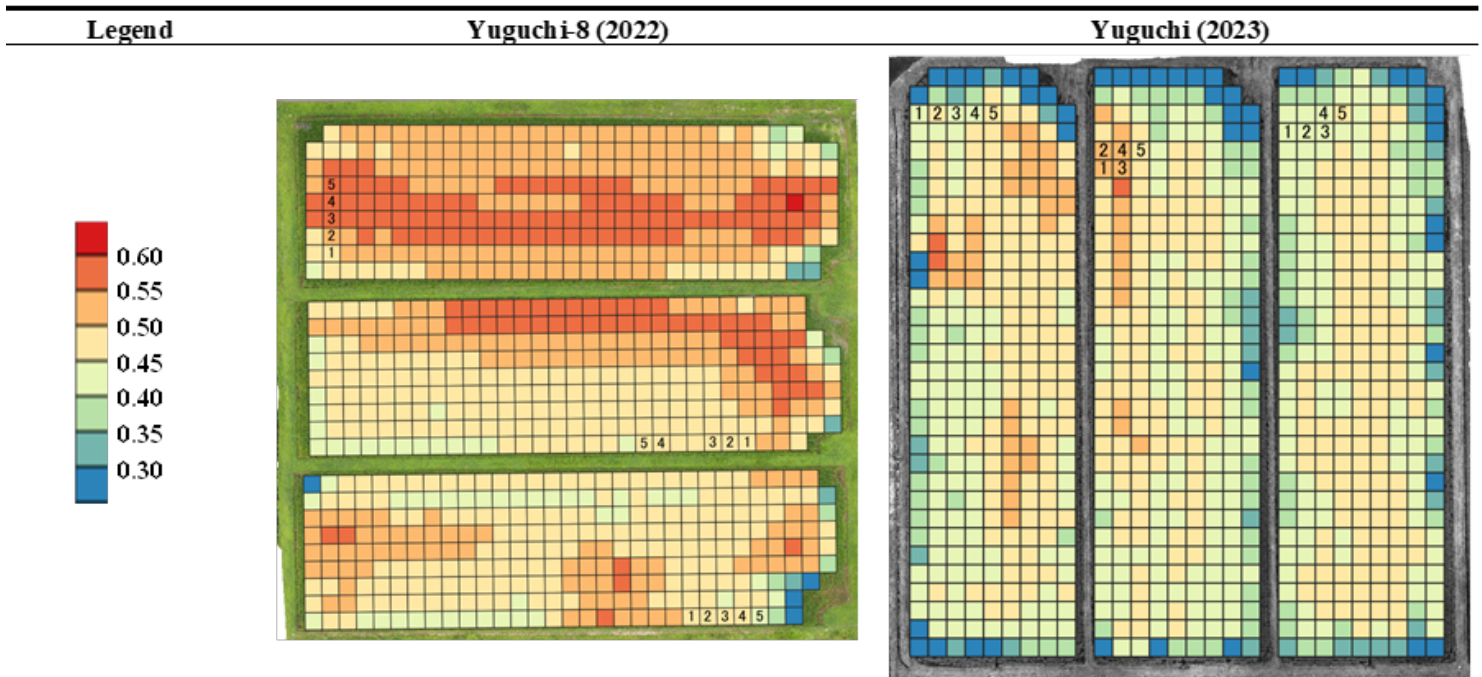


Table 4: Execution conditions of RF

Item	Condition
Independent Valuable	Time-series VI value per monitoring date of each mesh
Dependent Valuable	Cluster ID of each mesh
Training Data	Sampling by Stratified Sampling with Cluster ID as a layer (25% of total meshes)
Test Data	Remaining 75% of total meshes

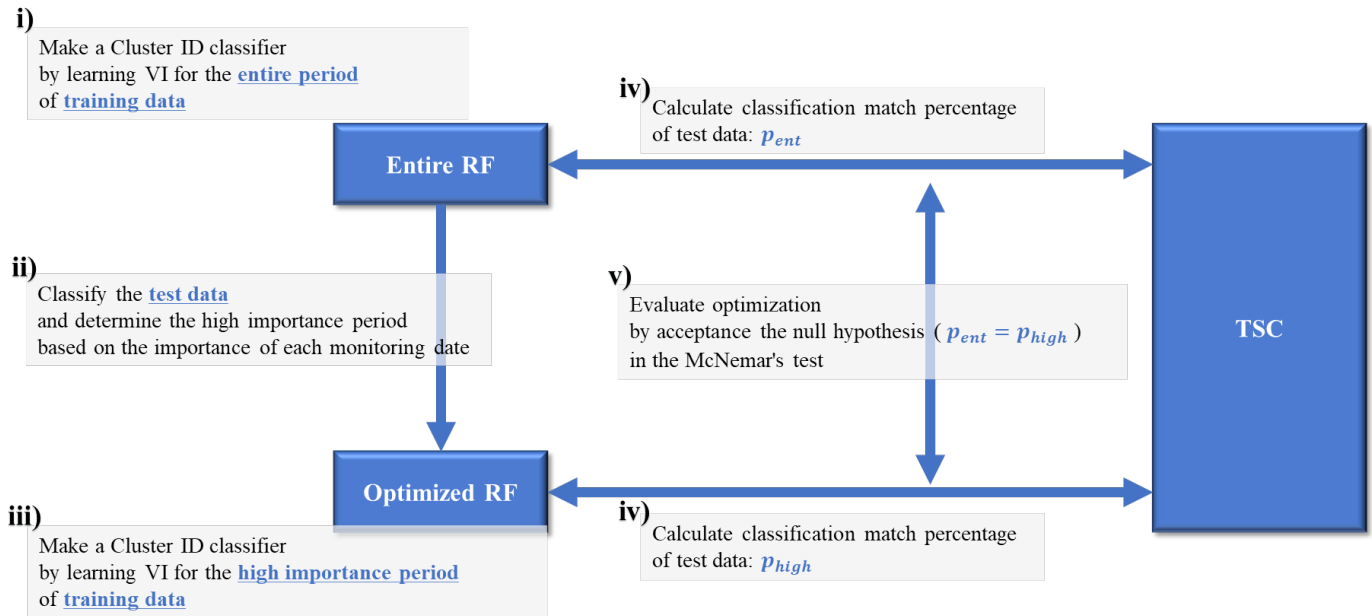


Figure 5: Method of monitoring period optimization

Table 5: Yields of 5 sampled meshes and results of Kruskal-Wallis test

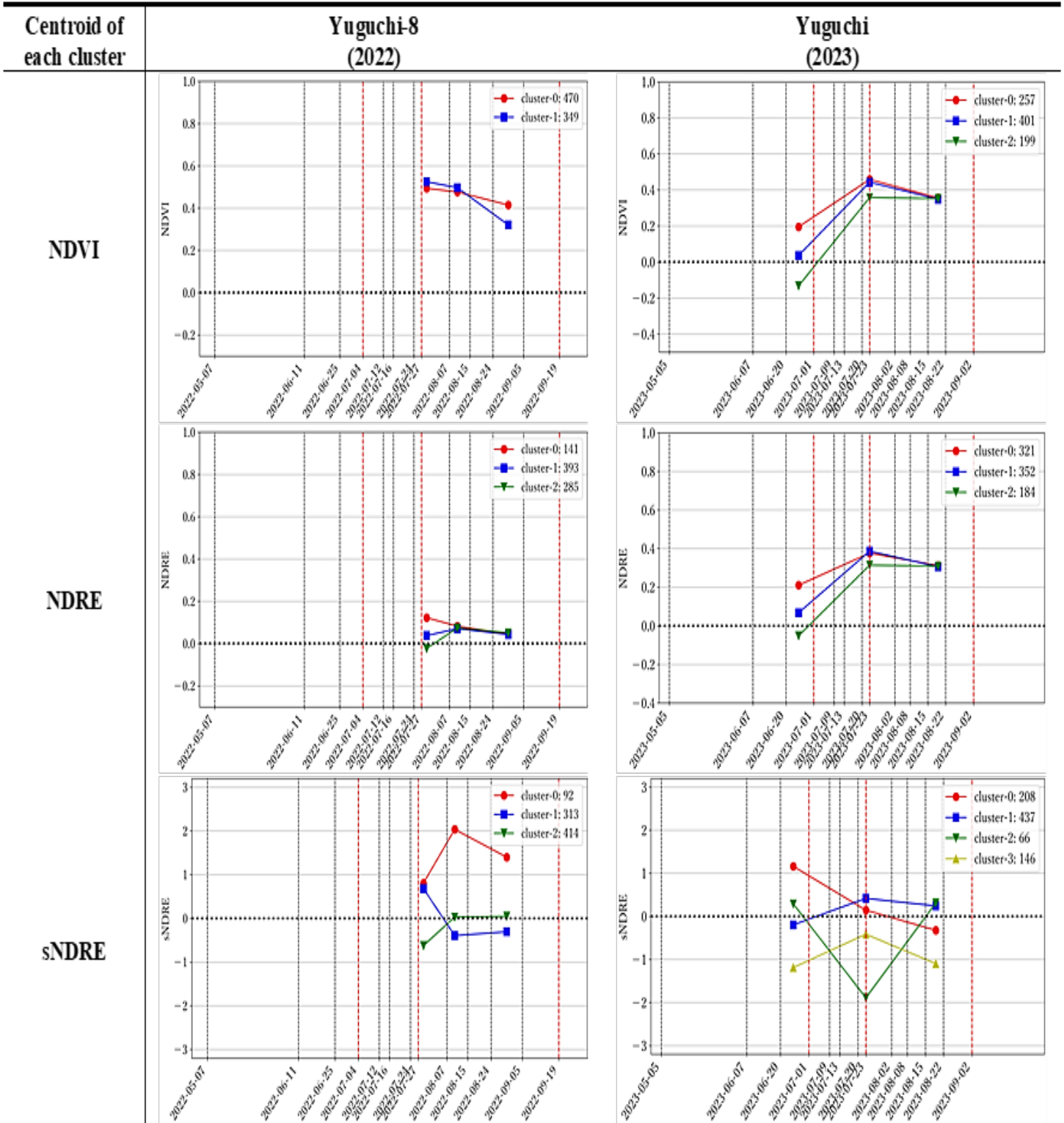
g / 5stubbles	Yuguchi-8 (2022)			Yuguchi (2023)		
	North	Center	South	West	Center	East
Sample1	176.0	180.8	220.6	136.4	161.8	221.9
Sample2	239.0	172.2	188.4	203.0	66.6	175.7
Sample3	189.6	192.0	164.0	190.5	204.8	142.0
Sample4	180.2	178.2	155.6	177.6	214.3	206.9
Sample5	213.0	175.8	139.2	N/A	181.5	178.0
Average	199.6	179.8	173.6	176.9	165.8	184.9
Standard Deviation	26.3	7.5	31.7	28.9	59.1	30.9
Average [g/mmm <sup>2</sup> ]	724	653	630	642	602	671
p-value of Kruskal-Wallis Test	0.249			0.873		

each cluster are shown in Table 6, Table 7, and Table 8. For Yuguchi-8 in 2022, the meshes were classified into two, three, and three clusters for NDVI, NDRE, and sNDRE, respectively. For the NDVI, there was no difference between clusters in the heading stage, and the yields were expected to be similar in all rice fields. In the NDRE, differences were observed at the heading stage, but no differences were observed thereafter. Cluster-0, which had relatively low stress, was found near the ridges. In sNDRE, Cluster-0 was high and often observed near the ridge. The central meshes of each paddy field were many Clusters-1 and Cluster-2, but after heading, the value did not differ by more than 0.5, with no difference in the stress level. The same can be said for all three rice fields, and no growth differences were observed between them. Therefore, the appropriate fertilizer amount was analyzed as the north field (2.8 gN/m<sup>2</sup>), where the amount is the lowest, considering fertilizer costs etc.

At Yuguchi in 2023, the meshes were classified into three, three, and four clusters for NDVI, NDRE, and sNDRE, respectively. In the NDVI, there was a difference of approximately 0.15 between clusters at the first monitoring. However, the low value (Cluster-2) improved because of the fertilizer. Thus, it was concluded that there was no difference in growth between the fields because the value of each cluster was similar at the heading stage. There was a large difference in sNDRE between the four clusters. Similar to other VI, high-value (Cluster-0 and Cluster-1) were observed at the center of the field. These two clusters appeared slightly more frequently in the West and Center. Similar



Table 6: Centroid of each cluster (entire period)



to Yuguchi-8, the same was observed for all three fields, but no growth differences were observed. Therefore, the appropriate amount of fertilizer was analyzed in the western field (2.8 gN/m<sup>2</sup>).

### 3.2. Optimization of Monitoring Period

The importance results for the six sites (excluding Yuguchi-8

in 2022 and all sites in 2023) are shown in Figure 6. The vertical axis in this figure represents the Relative Importance. The Importance output by the RF is the percentage contribution [1], [24]. That is, the fewer the monitoring dates, the higher the value. To adjust for this, Relative Importance was calculated by dividing the usual output importance by the reciprocal of the number of monitoring dates [1]

Table 7: Cluster ID distribution map (entire period)

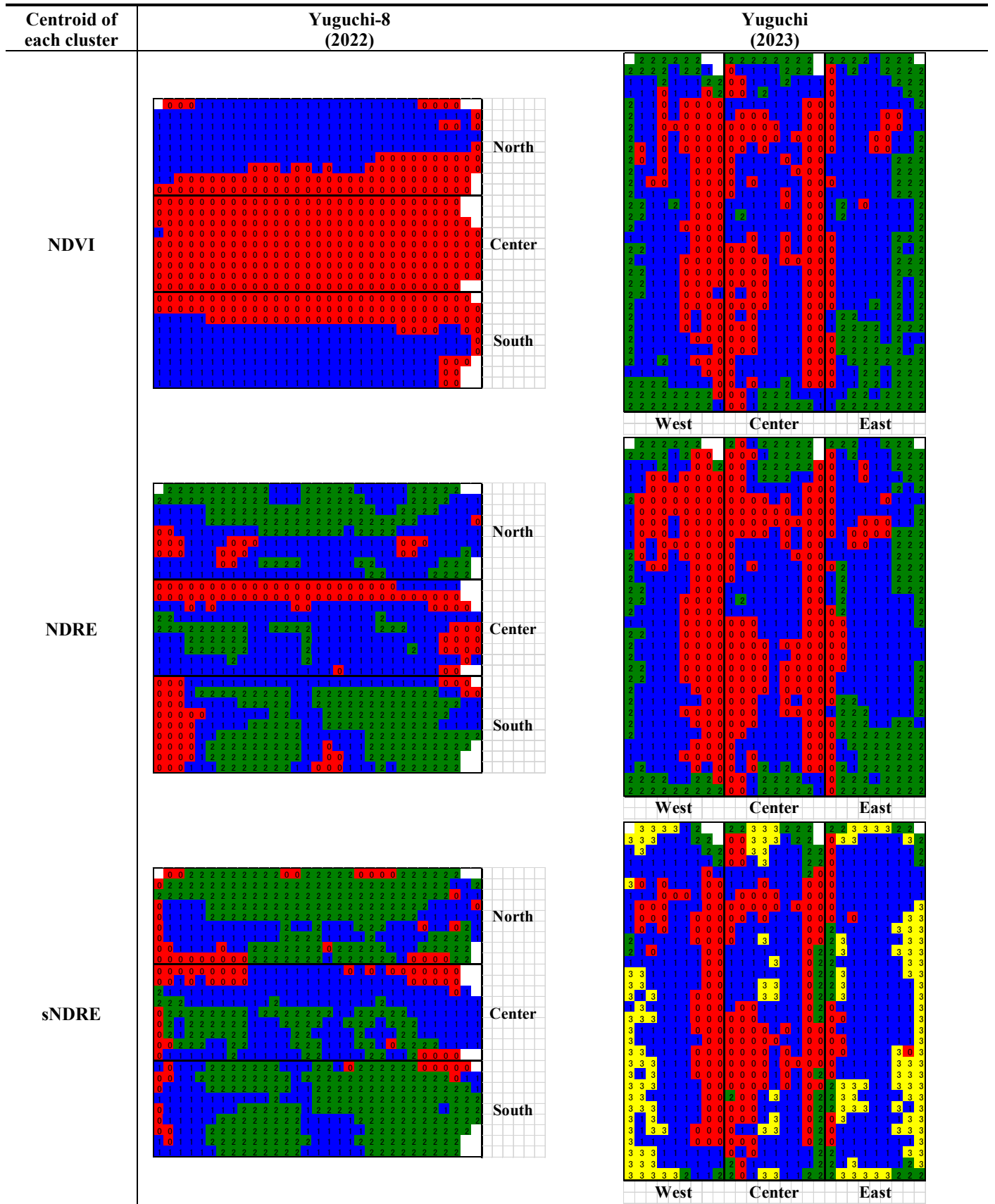
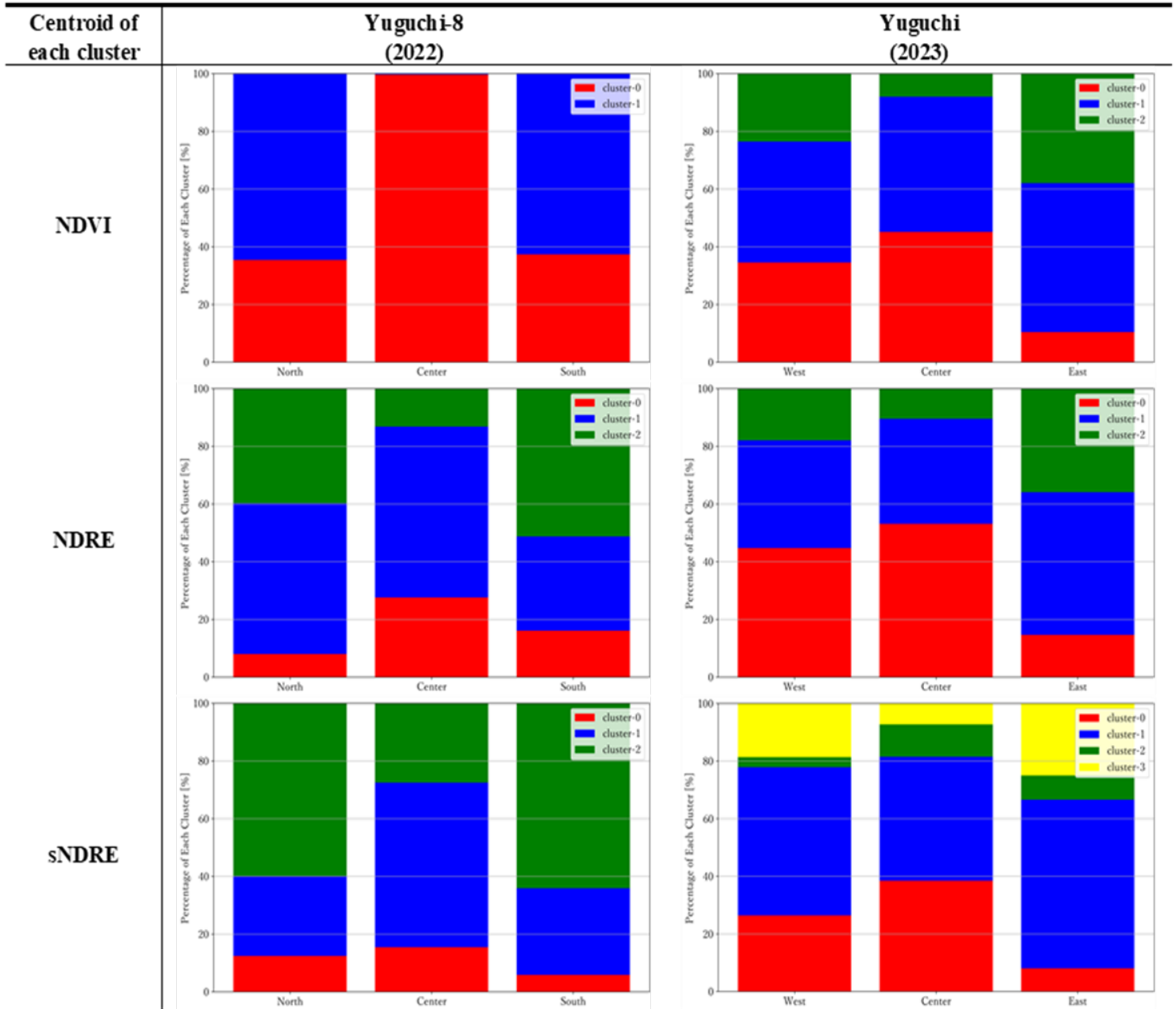


Table 8: Percentage of each cluster (entire period)



[24]. To optimize the optimization method of monitoring period, first, the initial period was extracted, for which the minimum period included all data with a Relative Importance of one or more. Next, the period was expanded or deduced to the sum of the Relative Importance of 49% to 51%, in contrast to the entire period shown in Figure 6.

The optimized periods of the EAT determined using this method are listed in Table 9. The rightmost column shows the EATs of the optimized period converted to dates using the EAT from May 7 in a normal year. The optimal period for all VIs was approximately one month. This period was from the young panicle forming to the milk ripe for NDVI and NDRE, and from heading to the full ripe for sNDRE. In Table 10, the results are applied to two sites that were excluded when calculating the optimized period (i.e., Yuguchi -8 in 2023 and Yuguchi in 2022). For the match rate

of the cluster ID classification of the test data between the Entire RF and TSC ( $p_{ent}$ ), and the Optimized RF and TSC ( $p_{opt}$ ),  $p_{opt}$  was greater than  $p_{ent}$ . They are not necessarily 100% because the training data comprise 25% of all the meshes and the remaining 75% of the test data. The rightmost column shows the p-value of the test for the difference between two percentages. No VI was significant at the 5% level. In other words, the monitoring period can be optimized without significantly reducing the classification accuracy. In addition, although the 2023 data were not used to derive the optimized period,  $p_{opt}$  of Yuguchi had high accuracy (over 98%). This indicates that the optimized year can be derived using the EAT without depending on the year.

4. Discussion

4.1. Visualization of Paddy Rice Growth

In the TSC during the optimized period, most results showed that growth was different in the center of the fields and near the ridge. In the NDVI and NDRE, the area near the ridge was classified into low-value clusters, so it is thought that the vegetation growth and stress states were not better than those in the center of the field. In contrast, for the sNDRE at Yuguchi-8, the near-ridge meshes were classified into high-value clusters. It is thought that the amount of nitrogen was high because the fertilizer was washed away by heavy rain in August, (Figure 3, Table 7).

However, at Yuguchi-8, all monitoring occurred after heading, and at Yuguchi, the first time was before young panicle formation and the second time during heading. In short, no monitoring was conducted between the start date of the optimized period and the fertilization date at either site. From the results of the optimized TSC period, it was not possible to visualize the growth before fertilization. Thus, vegetation differences are discussed using the results for the entire period (Table 6). From the time-series trends of each cluster, it can be seen that NDVI and NDRE increased between the first and second monitoring days. This suggests that nutrient supplementation with fertilizers improves vegetation and stress conditions. In addition, sNDRE is not suitable for comparing these conditions with other days because of the standardization of each day.

Table 9: Optimized period of EAT of each VI

VI	Optimized period of EAT [°C]	Date for a Normal Year
NDVI	509 – 913	Jul. 10 – Aug. 10
NDRE	588 – 1063	Jul. 16 – Aug. 21
sNDRE	860 – 1332	Aug. 6 – Sep. 11

As described above, TSC of the VIs was able to visualize the health status of paddy rice and predict the appropriate fertilizer amount. On the other hand, in order to further improve predictive.

Table 10: Results of mesh classification match percentage between RF and TSC and McNemar’s test

VI	Site	$p_{ent}$	$p_{opt}$	p-value
NDVI	Yuguchi-8	96.3%	100.0%	< 0.001
	Yuguchi	96.5%	99.4%	0.002
NDRE	Yuguchi-8	94.3%	99.0%	< 0.001
	Yuguchi	96.0%	98.1%	0.050
sNDRE	Yuguchi-8	91.0%	95.9%	0.003
	Yuguchi	93.4%	99.4%	< 0.001

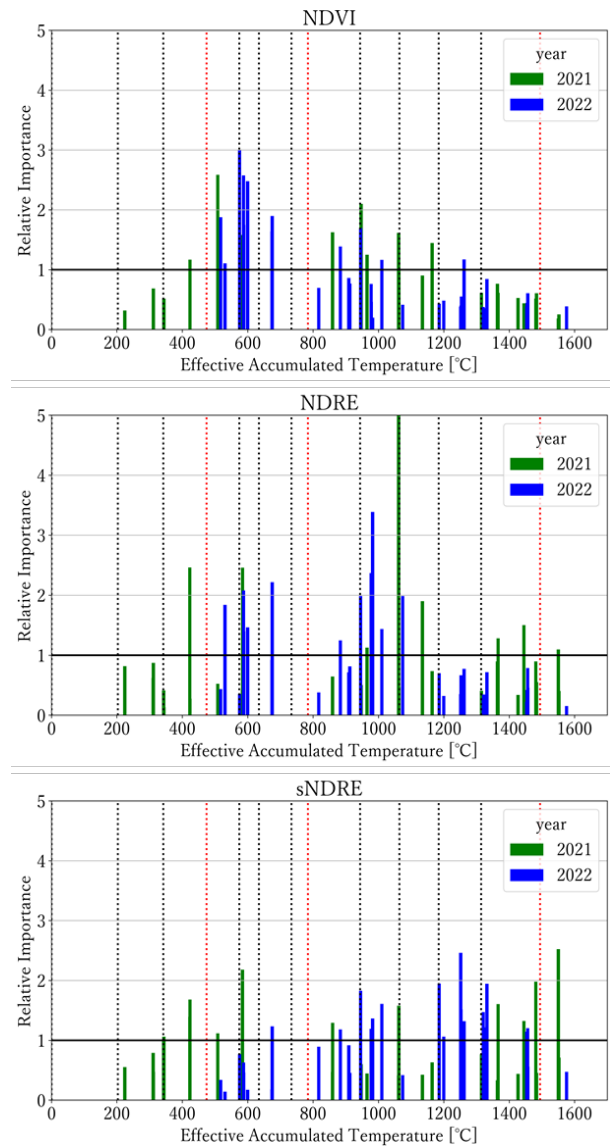


Figure 6: Relative Importance

For vegetation growth changes before and after fertilization, there were no monitoring data before fertilization included in the optimized period for either field (Table 6, Table 9).

Usually, additional fertilization is conducted on a day when EAT is around 500°C to 650°C (about 2 to 4 weeks before heading). Although sNDRE was not included in this period, NDVI and power, the following two things need to be considered. The first is the use of a broader range of wavelengths beyond visible light. Some of the VIs that have been developed so far use these bands (e.g., NDNI, CAI, etc.) [5], [25]. By using these VIs after confirming their effectiveness with ground truth data such as RGB images and site inspections, it becomes possible to visualize water stress, nutritional deficiencies, pest infestation, and so on. For ensuring the accuracy and reliability, it is necessary to verify them using detailed measurement data obtained from sensors such as air temperature and water temperature in mesh unit. Future research should utilize such ground truth data for TSC to observe various aspects of crop conditions. The second is to perform accurate spatial analysis. In this paper, for fast processing, the data used in the TSC was reduced in size by determining the mesh size

(approximately 3m x 3m) based on the resolution. It is desirable to perform TSC with various mesh sizes and verify the mesh size that increases the accuracy of the optimal fertilizer amount prediction and yield prediction. Also, the spatial variability of vegetation indices within the rice field and its implications for fertilizer management may be caused by the evenness of the rice field, location of water outlets and water flow within the rice field, and so forth. We hope to identify the causes of this spatial heterogeneity using spatial analysis techniques and the analytical methods of this study.

#### 4.2. Evaluation of Optimization

In the previous section 4.1, it was analyzed in both fields that the appropriate fertilizer amount was the lowest amount field by the TSC using the optimized period. The bottom panel of Table 5 shows the evaluation of this analysis from the viewpoint of yield. As mentioned in the previous chapter, this is the p-value of the test for significant differences in yield between fields when the null hypothesis is set as "the average of sampled yields of all rice fields are equal." Based on this test, the null hypothesis was adopted, even at the 10% significance level. In other words, it cannot be said that there is a difference in yield between paddy fields, and it also cannot be said that changing the fertilizer amount will change the yield. Therefore, the field with the least amount of fertilizer is appropriate from the viewpoint of avoiding fertilizer costs, excessive fertilization, and environmental impacts. This result agrees with the TSC analysis results.

In local practice, the fertilizer applied to all fields is 4.2 gN/m<sup>2</sup>. Assuming that this was changed to an appropriate 2.8 gN/m<sup>2</sup>, the fertilizer was reduced to 87 and 91 kg, and the amount of leached nitrogen was reduced to 0.070mg/L for both Yuguchi-8 and Yuguchi. It was small compared to the water pollution standards. This is the case when fertilizer is applied during irrigation, and the leaching amount is only approximately 3% of that during surface drainage [26], [27][28]. Therefore, TSC method reduces the amount of excess fertilizer used, so it is expected to reduce the environmental impact. Furthermore, the development of such appropriate fertilization techniques will contribute to the sustainability of agriculture by creating soil that can produce food over the long-term without degrading the environment [29]. And continued analysis of this study will promote long-term soil health by reducing chemical fertilizers and increasing organic ones without reducing yields. This is a topic for future work.

As described above, the amount of excess fertilizer was estimated by using TSC for VI from UAV. In order to ensure accuracy and reliability, this research will need to address the following three things in the future. First is the use of data fusion technology. This is important to enhance the predictive power of the model and provide a more comprehensive understanding of factors influencing crop response to fertilization. Future research would like to try to analyze differences in soil and past crop management practices for each rice field.

#### 5. Conclusions

In this research, the VIs time-series data were obtained using a UAV. Additionally, the monitoring dates were converted into

growth stages (EAT) using a paddy rice growth model from related research [19], [20]. On this data, three analyzes were performed. In the first analysis, it was possible to visualize healthy vegetation growth and stress in paddy rice using the TSC. We also confirmed that healthy vegetation growth and stress of crops were improved by fertilizer. In the second analysis, the Kruskal–Wallis test confirmed the appropriate amount from the results of TSC. In the third analysis, the optimized period was determined based on the importance of each independent variable using RF. Based on these findings, our work have discovered the technologies can help reduce fertilizer costs, excessive fertilization and environmental impacts, and promote the use of UAV.

A future challenge is the development of a growth-prediction model. This study deals with ex-post evaluations, such as the visualization and classification of crop conditions. Also it is necessary to make future predictions for crop yields, early disease detection, and variable fertilization using VIs. Currently, there are no absolute standards for VIs, and no study has been found that generalizes the desirable values at each growth stage. However, a research has established a target NDVI for each growth stage using a portable device for NDVI measurement [30], [31]. This study collects and analyzes VI data via UAVs to improve productivity and reduce costs. For this purpose, we will develop a system that uses deep learning to provide decision support for fertilization in natural language based on the features of monitoring images [32] and methods such as farm income prediction based on yield prediction using regression, and so on. Then, it would be desirable to be able to connect UAVs with these systems and provide information to farmers in real time. As the IoT advances, security issues are a concern, so it will be important to develop autonomous decentralized systems that increase data tamper resistance using blockchain technology [33], [34], [35], [36].

#### Conflict of Interest

The authors declare no conflict of interest.

#### Acknowledgment

The authors would like to thank the Research Center for Industrial Science and Technology, Iwate University, and the farmers in the experimental field for providing monitoring and yield survey data. We would also like to thank Editage ([www.editage.jp](http://www.editage.jp)) for English language editing. Finally, we are grateful to the referees for their helpful comments.

#### References

- [1]. T. Ito, K. Minamino and S. Umeki, "Analysis of Vegetation Indices by Time Series Clustering of Drone Rice Monitoring Data," 3rd International Symposium on Electrical, Electronics and Information Engineering (ISEEIE 2023), 2023, DOI: 10.1049/icp.2023.1853.
- [2]. Ministry of Agriculture, Forestry and Fisheries, "2020 nen nou-rin-gyuu census kekka no gaiyou (kakuteichi)," URL: [https://www.maff.go.jp/j/tokei/kekka\\_gaiyou/noucen/2020/index.html](https://www.maff.go.jp/j/tokei/kekka_gaiyou/noucen/2020/index.html), [Accessed 5 February 2024], (article in Japanese).
- [3]. K. Miyama, "Nou-sakumotsu · dozyou no bunkou-hansya-tokusei - nou-you-chi he no remote sensing gizyutsu no tekiyou-sei (I)," *Journal of the Agricultural Engineering Society, Japan*, **56**(12):1197-1202, 1988, (article in Japanese).
- [4]. J. L. Hatfield and J. H. Prueger, "Value of Using Different Vegetative Indices to Quantify Agricultural Crop Characteristics at Different Growth

- Stages under Varying Management Practices," *Remote Sensing*, **2**(2):562-578, 2010, DOI: 10.3390/rs2020562.
- [5]. J. Xue and B. Su, "Significant Remote Sensing Vegetation Indices: A Review of Developments and Applications," *Journal of Sensors*, **2017**:1-17, 2017, DOI: 10.1155/2017/1353691
- [6]. K. Tanaka and A. Kondoh, "Mapping of Rice Growth Using Low Altitude Remote Sensing by Multicopter," *Journal of The Remote Sensing Society of Japan*, **36**(4):373-387, 2016, DOI: 10.11440/rssj.39.S1.
- [7]. K. Harashina, K. Yamamoto, M. Maki, Y. Muto and E. Kurashima, "Monitoring Growth of Water-seeded Rice in Tsunami-stricken Paddy Fields Using UAV-mounted Multispectral Sensor," *Journal of the Japanese Society of Irrigation, Drainage and Rural Engineering*, **87**(2):121-126, 2019, (article in Japanese with a title in English).
- [8]. A. Matsuzaki, Inasaku dai hyakka dai 2 han dai 3 kan saibai no zissai/sehigizyutsu, Rural Culture Association, 2004, (article in Japanese).
- [9]. Kubota Co., Ltd., "Mizu to hiryou ni yoru control | tanbo no kanri to higaitaisaku | okome ga dekirumade | Kubota no tanbo [manannde tanoshii! tanbo no sougou zyouhou site]," URL: <https://www.kubota.co.jp/kubotatanbo/rice/management/manure.html>, [Accessed 21 December 2023], (article in Japanese).
- [10]. National Institute of Science and Technology Policy, "NISTEP PDF cover 200612," URL: <https://nistep.repo.nii.ac.jp/record/5536/files/NISTEP-STT069.pdf>, [Accessed 21 December 2023], (article in Japanese).
- [11]. Ministry of the Environment, "Suishitsu-osen ni kakawaru kankyo-kizyun," URL: <https://www.env.go.jp/kijun/mizu.html>, [Accessed 7 February 2024], (article in Japanese).
- [12]. T. Ito, K. Minamino and S. Umeki, "Analysis of Vegetation Indexes by Time Series Clustering of Drone Rice Monitoring Data," *The 21st Forum on Information Technology (FIT2022)*, **21**(4):105-110, 2022, (article in Japanese with a title in English).
- [13]. T. Ito, K. Minamino and S. Umeki, "Analysis of Fertilization Effects on Rice and Wheat by Time-Series Clustering of Vegetation Index Data," *Proceedings of the 5th International Symposium on Advanced Technologies and Applications in the Internet of Things (ATAIT 2023)*, 2023.
- [14]. Da-Jiang Innovations Science and Technology Co., Ltd., "AGRAS MG-1S Series - DJI," URL: <https://www.dji.com/mg-1s>, [Accessed 8 February 2024].
- [15]. Da-Jiang Innovations Science and Technology Co., Ltd., "P4 MultispectralDJI," URL: <https://www.dji.com/p4-multispectral>, [Accessed 8 February 2024].
- [16]. A. Karnieli, N. Agam, R. T. Pinker, M. Anderson, M. L. Imhoff, G. G. Gutman, N. Panov and A. Goldberg, "Use of NDVI and Land Surface Temperature for Drought Assessment: Merits and Limitations," *Journal of Climate*, **23**(3):618-633, 2010, DOI: 10.1175/2009JCLI2900.1
- [17]. H. Zheng, W. Ji, W. Wang, J. Lu, D. Li, C. Guo, X. Yao, Y. Tian, W. Cao, Y. Zhu and T. Cheng, "Transferability of Models for Predicting Rice Grain Yield from Unmanned Aerial Vehicle (UAV) Multispectral Imagery across Years, Cultivars and Sensors," *Drones*, **6**(12):423:1-19, 2022, DOI: 10.3390/drones6120423.
- [18]. L. Osborn, "NDVI vs. NDRE: What's the Difference? - Sentera," URL: <https://sentera.com/resources/articles/ndvi-vs-ndre-whats-the-difference/>, [Accessed 8 February 2024].
- [19]. E. Kanda, Y. Torigoe and T. Kobayashi, "A Model to Estimate the Increase of Leaf Number on the Main Culm of the Rice Plant," *Japanese Journal of Crop Science*, **69**(4):540-546, 2000, DOI: 10.1626/jcs.69.540, (article in Japanese with an abstract in English).
- [20]. E. Kanda, Y. Torigoe and T. Kobayashi, "A Simple Model to Predict the Developmental Stages of Rice Panicles Using the Effective Accumulative Temperature," *Japanese Journal of Crop Science*, **71**(3):394-402, 2002, DOI: 10.1626/jcs.71.394, (article in Japanese with an abstract in English).
- [21]. A. K. Jain, M. N. Murty and P. J. Flynn, "Data Clustering: A Review," *ACM Computing Surveys*, **31**(3):264-323, 1999, DOI:10.1145/331499.331504.
- [22]. R. Tavenard, "tslearn.clustering.TimeSeriesKMeans — tslearn 0.6.3 documentation," URL: [https://tslearn.readthedocs.io/en/stable/gen\\_modules/clus\\_tering/tslearn.clustering.TimeSeriesKMeans.html](https://tslearn.readthedocs.io/en/stable/gen_modules/clus_tering/tslearn.clustering.TimeSeriesKMeans.html),
- [23]. C. Shi, B. Wei, S. Wei, W. Wang, H. Liu and J. Liu, "A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm," *EURASIP Journal on Wireless Communications and Networking*, **31**:1-16, 2021, DOI: 10.1186/s13638-021-01910-w.
- [24]. S. Ronaghan, "The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark," URL: <https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3>, [Accessed 11 February 2024].
- [25]. The IDB Project, "Index DataBase," URL: <https://www.indexdatabase.de/>, [Accessed 23 April 2024].
- [26]. Toyama Prefecture, "Chika-sui kanyou no suishin ni mukete ," URL: <https://www.pref.toyama.jp/documents/7739/00745883.pdf>, [Accessed 17 February 2024], (article in Japanese).
- [27]. C. Imagawa, "Nitrogen Dynamics Modeling to Support Land and Water Management in Rice Paddy Watershed," *Soil/Water Research Group Materials*, **30**, 2013, (article in Japanese with a title in English).
- [28]. M. Tsuji and M. Takei, "Degrees of Water Purification in Cultivated Rice Fields and Fallow Fields," *Research bulletin of the Aichi-ken Agricultural Research Center*, **43**:1-6, 2011, (article in Japanese with an abstract in English).
- [29]. M. M. Tahat, K. M. Alananbeh, Y. A. Othman and D. I. Leskovar, "Soil Health and Sustainable Agriculture," *Sustainability*, **12**(12), 2020, DOI: 10.3390/su12124859.
- [30]. NIKON-TRIMBLE CO., LTD., "GreenSeeker 2," URL: [https://www.nikon-trimble.co.jp/products/product\\_detail.html?tid=428](https://www.nikon-trimble.co.jp/products/product_detail.html?tid=428), [Accessed 15 February 2024].
- [31]. Y. Kaneta, M. Nishida, F. Takakai and T. Sato, "New growth diagnosis standards of high-yielding rice and demonstration of high yielding using NDVI by GreenSeeker handheld crop sensor," *Japanese Journal of Soil Science and Plant Nutrition*, **91**(6):417-425, 2020, DOI: 10.20710/dojo.91.6\_417, (article in Japanese with an abstract in English).
- [32]. S. Ayoub, Y. Gulzar, F. A. Reegu and S. Turaev, "Generating Image Captions Using Bahdanau Attention Mechanism and Transfer Learning," *Symmetry*, **14**(12), 2022, DOI: 10.3390/sym14122681.
- [33]. F. A. Reegu, H. Abas, Y. Gulzar, Q. Xin, A. A. Alwan, A. Jabbari, R. G. Sonkamble and R. A. Dziauddin, "Blockchain-Based Framework for Interoperable Electronic Health Records for an Improved Healthcare System," *Sustainability*, **15**(8), 2023, DOI: 10.3390/su15086337.
- [34]. A. A. Dar, M. Z. Alam, A. Ahmad, F. A. Reegu and S. A. Rahin, "Blockchain Framework for Secure COVID-19 Pandemic Data Handling and Protection," *Computational Intelligence and Neuroscience*, **2022**(7025485):1-11, 2022, DOI: 10.1155/2022/7025485.
- [35]. M. Z. Alam, F. Reegu, A. A. Dar and W. A. Bhat, "Recent Privacy and Security Issues in Internet of Things Network Layer: A Systematic Review," *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, 2022, DOI: 10.1109/ICSCDS53736.2022.9760927.
- [36]. F. A. Reegu, W. A. Bhat, A. Ahmad and M. Z. Alam, "A review of importance of blockchain in IOT security," *International Conference on Emerging Trends in Materials, Computing and Communication Technologies (ICETMCCT 2021)*, **2587**(1), 2023, DOI: 10.1063/5.0150432.

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

## Solar Photovoltaic Power Output Forecasting using Deep Learning Models: A Case Study of Zagtouli PV Power Plant

Sami Florent Palm<sup>1</sup>, Sianou Ezéckiel Houénafa<sup>2</sup>, Zourkalaïni Boubakar<sup>3</sup>, Sebastian Waita<sup>1</sup>, Thomas Nyachoti Nyangonda<sup>1</sup>, Ahmed Chebak<sup>4</sup>

<sup>1</sup> Condensed Matter Research Group, Department of Physics, University of Nairobi, Nairobi, Kenya

<sup>2</sup> Institute for Basic Science, Technology and Innovation, Pan African University, Nairobi, Kenya

<sup>3</sup> Ecole Doctorale Informatique, Télécommunication et Electronique, Sorbonne Université, Paris, France

<sup>4</sup> Green Tech Institute, Mohammed VI Polytechnic University, Benguerir, Morocco

### ARTICLE INFO

Article history:

Received: 22 March, 2024

Revised: 6 May, 2024

Accepted: 7 May, 2024

Online: 25 May, 2024

Keywords:

Deep learning

LSTM

GRU

Solar PV Power

Zagtouli

### ABSTRACT

Forecasting solar PV power output holds significant importance in the realm of energy management, particularly due to the intermittent nature of solar irradiation. Currently, most forecasting studies employ statistical methods. However, deep learning models have the potential for better forecasting. This study utilises Long Short-Term Memory (LSTM), Gate Recurrent Unit (GRU) and hybrid LSTM-GRU deep learning techniques to analyse, train, validate, and test data from the Zagtouli Solar Photovoltaic (PV) plant located in Ouagadougou (longitude:12.30702° and latitude:1.63548°), Burkina Faso. The study involved three evaluation metrics: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and coefficient of determination ( $R^2$ ). The RMSE evaluation criteria gave 10.799(LSTM), 11.695(GRU) and 10.629(LSTM-GRU) giving the LSTM-GRU model as the best for RMSE evaluation. The MAE evaluation provided 2.09, 2.1 and 2.0 for the LSTM, GRU and LSTM-GRU models respectively, showing that the LSTM-GRU model is superior for MAE evaluation. The  $R^2$  criteria similarly showed the LSTM-GRU model to be best with 0.999 compared to 0.998 for LSTM and 0.997 for GRU. It becomes evident that the hybrid LSTM-GRU model exhibits superior predictive capabilities compared to the other two models. These results indicate that the hybrid LSTM-GRU model has the potential to reliably predict the solar PV power output. It is therefore recommended that the authorities in charge of the solar PV Plant in Ouagadougou should consider switching to the deep learning LSTM-GRU model.

### 1. Introduction

In the pursuit of sustainable and renewable energy sources, solar photovoltaic (PV) systems have emerged as a leading solution for harnessing the abundant energy provided by the sun. A critical factor in optimizing the efficiency and reliability of solar PV installations is the accurate forecasting of power output [1,2]. This is particularly vital for ensuring the seamless integration of solar energy into the existing power grid and effectively managing energy resources.

The application of deep learning techniques has gained considerable attention in this context due to its capacity to model complex relationships within large datasets, offering a promising tool to enhance the precision of solar PV power output forecasting [3].

Researchers in [4] employed techniques to improve the performance of grid-connected PV systems. Financial and technical limitations emerge as hindrances to the development of PV systems, prompting a recommendation for the utilization of artificial intelligence to enhance power generation. Comparison between time series methods and artificial intelligence-based methods for power output prediction in a large grid-connected PV

\*Corresponding Author: Sami Florent Palm, University of Nairobi, Nairobi, Kenya palm@students.uonbi.ac.ke

plant in China indicated the efficiency of neural network models over statistical models for PV power output prediction, particularly for short-term forecasts [5]. In a solar power prediction study in India, the efficiency of Long Short-Term Memory (LSTM) and Backpropagation Neural Network (BPNN) models was compared, confirming the effectiveness of the LSTM model [3]. LSTM and Multi-layer Perception (MLP) techniques were employed to forecast short-term solar PV power. A comparison of their performance based on parameters like Mean Absolute Errors (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and  $R^2$  revealed LSTM as the superior model [6]. A hybrid deep learning model for short-term PV power forecasting, integrating Wavelet Packet Decomposition (WPD) and LSTM, exhibits remarkable accuracy and stability. However, additional investigation is needed for long-term forecasting, especially during cloudy and rainy periods [7]. A method is proposed that combines LSTM-RNN and temporal correlation principles for PV power prediction, showcasing enhanced predictive capabilities and emphasizing the interplay between climate and electricity production [8]. The hybrid model (VMD-ISSA-GRU), integrating Variational Mode Decomposition (VMD), Improved Sparrow Search Algorithm (ISSA), and GRU, was utilized to improve PV power prediction. Results showed strong performance with an MAE of 1.0128 kW, RMSE of 1.5511 kW, and R-squared value of 0.9993 [9]. The study in [10] presents a new forecasting method for a large grid-connected PV plant in Vietnam, emphasizing climate uncertainty and employing the LSTM algorithm. The result underscored the impact of climate data on prediction accuracy, emphasizing the need for careful model configuration. The study suggests the importance of using LSTM configurations tailored to specific climatic and operational conditions. PV power generation, inherently linked to unpredictable weather conditions, poses challenges in prediction. The case studies presented reflect the ongoing efforts to develop more accurate forecasting methods to address the intermittency and instability of PV systems connected to the power grid. As per the existing literature, the LSTM model proves highly proficient in forecasting solar PV power. Moreover, its amalgamation with other models demonstrates superior effectiveness compared to the individual performance of each model.

Limited literature has been undertaken in Burkina Faso and Sahel countries in general regarding the forecasting of solar PV power using deep learning methods. This study underscores the necessity of advanced forecasting techniques using LSTM, GRU and hybrid LSTM-GRU models in predicting the output of the Zagtouli PV power. Forecasting in Zagtouli is important since more energy is needed to meet regional power demands. By delving into the challenges associated with traditional forecasting methods, the study aims to contribute valuable insights to the broader field of renewable energy research, paving the way for improved efficiency and reliability in the integration of solar power. The findings hold the potential to deepen our understanding of the dynamics influencing solar energy production and inform future developments in sustainable energy planning and management. The remainder of this paper will be structured as follows: first, the methodology, followed by the results and discussions, and finally, the conclusion and perspectives.

## 2. Methodology

### 2.1. Zagtouli PV Power Plant

The Zagtouli Solar PV Power Plant sits in Ouagadougou, the capital of Burkina Faso, positioned at a longitude of 12.30702° and latitude -1.63548°. Figure 1 depicts the location of the Zagtouli PV facility in Burkina Faso. This PV Plant has a total installed capacity of 33.696 MWp composed of 16 subsystems. In this study, one subsystem of 2.3 MW is considered.

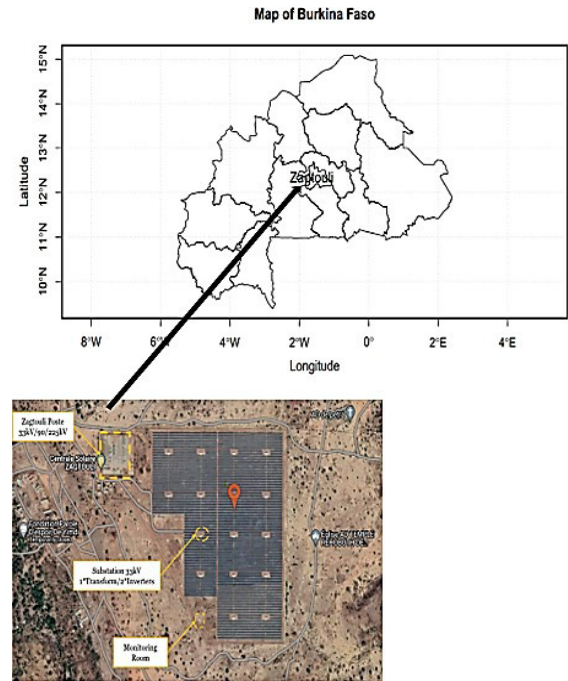


Figure 1: The Zagtouli PV Power Plant Location

### 2.2. Deep Learning Models

In brief, Artificial Intelligence (AI) is the endeavour to automate cognitive tasks typically executed by humans [11]. Consequently, AI constitutes a broad domain that encompasses not only Machine Learning (ML) and Deep Learning (DL) but also various other approaches that may not necessitate any form of learning [12]. ML is the art of studying algorithms that learn from examples and experiences [13]. The difference from hardcoding is that the machine learns on its own to find such rules [13]. Figure 2 below shows the difference between classical programming and ML.

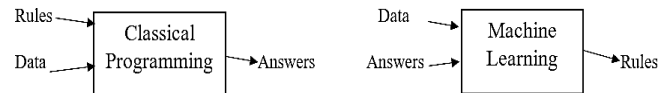


Figure 2: Difference between Classical Programming and Machine Learning.

Deep Learning (DL), a subset of Machine Learning (ML), represents a novel approach to deriving meaningful representations from data by prioritizing the acquisition of progressively more meaningful representations across successive layers [12]. Figure 3 explains the difference between AI, ML and DL.



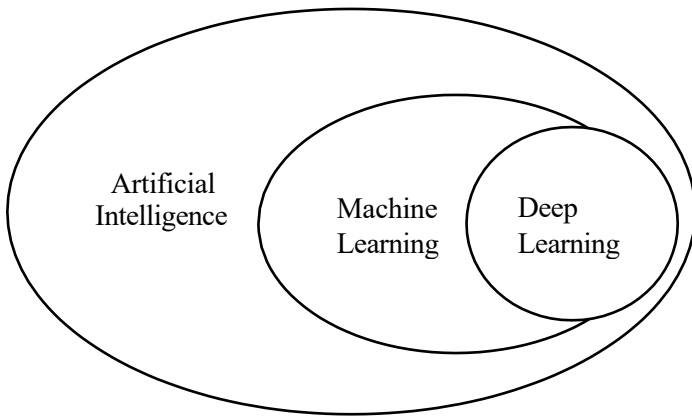


Figure 3: Artificial Intelligence, Machine Learning and Deep Learning

The DL technique is fundamentally based on the Artificial Neuron Network (ANN) which is ML method known as the artificial intelligence system which reflects the human brain. To comprehend the fundamental structure of ANN, it is essential to first grasp the concept of a 'node.' The general configuration of a node is depicted in Figure 4 below:

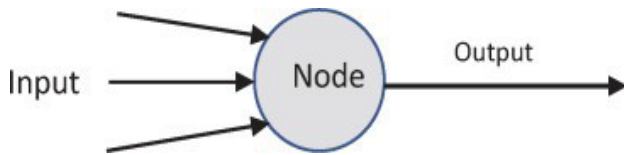


Figure 4: Basic Node Model [13]

Every node receives an array of inputs through connections and transmits them to neighbouring nodes [14]. Figure 5 depicts the overall model of an ANN, inspired by the functioning of a biological neuron [15]. Nodes are organized into linear networks referred to as layers. The ANN comprises three layers: the input layer, the output layer, and the hidden layer [16]. In the input layer,  $X_1, X_2, X_3, \dots, X_n$  represent multiple inputs to the network. Meanwhile,  $W_1, W_2, W_3, \dots, W_n$  are referred to as connection weights, indicating the strength associated with a specific node. In ANN, weights are regarded as crucial factors since they are numerical parameters that influence the interactions among neurons, playing a significant role in shaping the output by transforming the input [17]. Within the ANN, the processing component takes place in the hidden layer [17]-[19]. The hidden layer carries out two operational functions, specifically, the summation function and the transfer function, also recognized as an activation function [17,20]. The summation function serves as the initial step, where each input ( $X_i$ ) to the ANN undergoes multiplication by its corresponding weight ( $W_i$ ). Subsequently, the products  $W_i.X_i$  are accumulated into the summation function represented as  $\xi = \sum W_i.X_i$ . The parameter 'B,' denoting bias, is employed to control the neuron's output in conjunction with the weighted sum of the inputs. This process is denoted by equation (1) below:

$$Output = \sum(Weights \times Input) + Bias \quad (1)$$

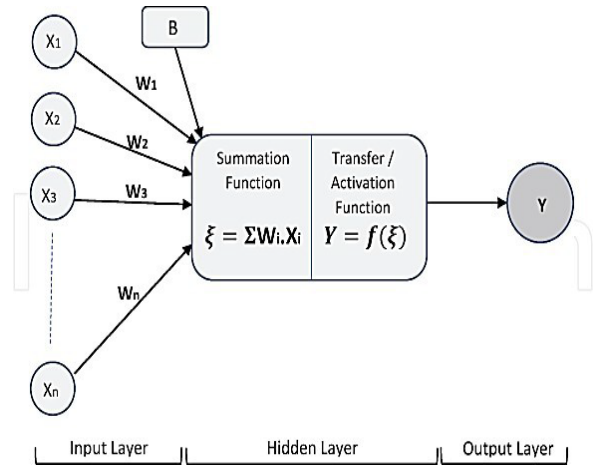


Figure 5: Generic Artificial Neural Network (ANN) model [13]

The activation function constitutes the second phase, wherein it takes the input signal from the summation function module and transforms it into the output of a node within an ANN model [14]. In general, each ANN comprises three fundamental components: node characteristics, network topology, and learning rules. Node characteristics govern signal processing by determining the number of associated inputs and outputs, the weights assigned to each input and output, and the activation function for each node. Learning rules dictate the initiation and adjustment of weights. Meanwhile, network topology defines the connectivity and organization of nodes. The operation of the ANN model involves computing the output of all neurons, representing a wholly deterministic calculation.

In this study, we will focus on two DL techniques: Long Short-Term Memory and Gate Recurrent Unit (GRU).

### 2.3. Long Short-Term Memory (LSTM)

LSTM stands for Long Short-Term Memory, and it functions as a network composed of interconnected neurons, each responsible for retaining previous state information [21]. With enough network elements, an LSTM network can conduct computations. The structure of an LSTM cell, depicted in Figure 6, comprises three key gates: the forget gate, the input gate, and the output gate. A distinctive aspect of LSTM networks is the memory cell, which serves as a repository for state information. Opening the input gate allows new information to be gathered into the cell, whereas opening the output gate leads to the erasure of past information. Within the feedback loop, the sigmoid function determines whether information should be preserved or deleted in the memory cell, while the hyperbolic tangent function manages the cell's input and output. This amalgamation of functions empowers the LSTM to selectively retain or discard information, significantly enhancing its performance. This capability makes LSTMs particularly valuable for handling temporal datasets and making predictions [22]. Notably, in LSTM networks, the final cell is transmitted to the concluding stage solely when the output gate is opened. This specific behaviour unique to LSTMs prevents gradients from dissipating rapidly within the cell, resulting in improved performance in processing time series data and generating predictions compared to other approaches.

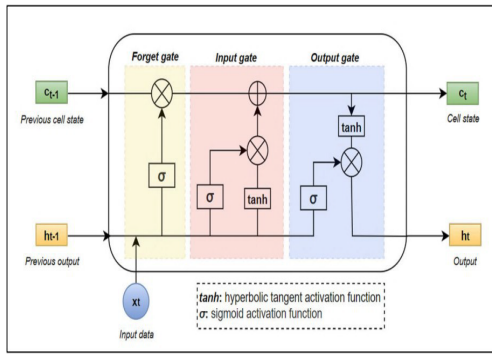


Figure 6: Long Short-Term Memory Structure [22]

#### 2.4. Gated Recurrent Unit (GRU)

The GRU shares similarities with LSTM as it represents a more simplified and streamlined variant of the LSTM architecture. It was presented [23] by in 2014, at a time when the interest in recurrent networks was resurging within the relatively small research community. Just like the LSTM unit, the GRU possesses gating units that regulate information flow within the unit, yet it operates without distinct memory cells. The GRU lacks a mechanism to manage the extent of its state exposure, invariably revealing its entire state with each occurrence [24], [25]. Figure 7 represents the structure of the GRU model.

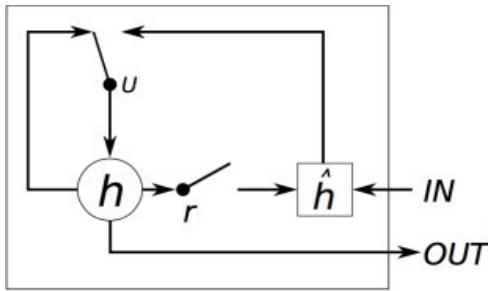


Figure 7: Gated Recurrent Unit Structure [23]

#### 2.5. Forecasting Time

The prediction time is called the forecasting horizon [25]. Before designing the model, it is necessary to choose the appropriate forecasting horizon because the quality of the prediction is sensitive to the forecasting horizon. Prediction accuracy is influenced by the change in the forecast horizon, even with similar parameters in the same model. Figure 8 below explains the classification of PV power forecasting based on time.

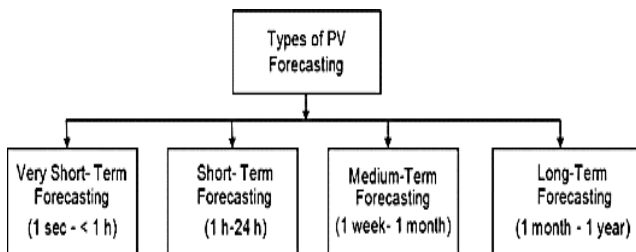


Figure 8: Classification of PV Power Forecasting Based on Time [24]

Very short-term (1sec - 1 h) forecasting is useful for real-time electricity transmission, optimal reserves, and power smoothing, while short-term (1h - 24 h) forecasting is useful for improving network security. By estimating the available electric power shortly, medium-term forecasting (1 week to 1 month) keeps the power system planning and maintenance schedule on track. Long-term forecasts (from one month to one year) aid in the planning of electricity generation, transmission, and distribution, as well as tendering and security operations.

#### 2.6. Performance Evaluation of Forecasting Methods

Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination ( $R^2$ ) are widely utilized performance metrics in PV power forecasting using machine learning approaches. RMSE quantifies the average error magnitude by taking the square root of the mean of squared differences between predicted values and observed outcomes [1]. On the other hand, MAE assesses the average significance of errors in a forecast dataset by averaging the differences between actual observations and predicted outcomes across the entire test sample, with each discrepancy assigned equal weight [1].  $R^2$  provides a quantitative measure of the model's predictive accuracy and its capability to offer reliable estimates of future PV power output. It is important to highlight that a predictive model demonstrates increased accuracy when both MAE and RMSE are minimized, and its efficiency is enhanced when  $R^2$  approaches a value of 1 [26]. The equations (2), (3) and (4) represent the expressions of RMSE, MAE and  $R^2$  respectively:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - t_i)^2} \quad (2)$$

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - t_i| \quad (3)$$

$$R^2 = 1 - \frac{\sum_{i=1}^N (y_i - t_i)^2}{\sum_{i=1}^N (y_i - \frac{1}{N} (\sum_{i=1}^N y_i))^2} \quad (4)$$

where  $y_i$  and  $t_i$  are the measured and corresponding predicted values of PV power and  $N$  is the number of test samples.

#### 2.7. Data Analysis

##### 2.7.1. Problem Framing

The endeavour involves the application of deep learning techniques to develop grid-connected photovoltaic solar power facilities customized for the Sahelian climate. This segment focuses on accurately predicting solar PV power generation in Burkina Faso, as it plays a crucial role in effectively managing the intermittency of solar resources to enhance grid injection. Solar forecasting emerges as a highly cost-effective method for the seamless integration of solar energy. The process entails gathering historical data from a 2.3 MWp solar PV system at the Zagtouli PV Power plant and transforming it into a spreadsheet using Microsoft Excel. System coding will be implemented using

Python, and data will be uploaded into the machine learning Toolbox for analysis.

### 2.7.2. Forecasting Input Variables

A set of variables were selected to perform the multivariate time series forecasting task. These variables are:

- Irradiance on an inclined plane (Irr);
- Global Horizontal Irradiance (GHI);
- Air temperature (Tair);
- Module temperature (Tm);
- PV Current (I<sub>pv</sub>);
- PV Voltage (V<sub>pv</sub>);
- Relative Humidity (RH);
- PV Power (P<sub>pv</sub>)
- Title angle
- Wind direction (W<sub>dire</sub>)
- Wind speed.

Thus, a strong correlation can be observed with the following variables: I<sub>pv</sub>, P<sub>pv</sub>, Irr, GHI, T<sub>m</sub>, T<sub>inv</sub> and V<sub>pv</sub>. These variables will be used as inputs.

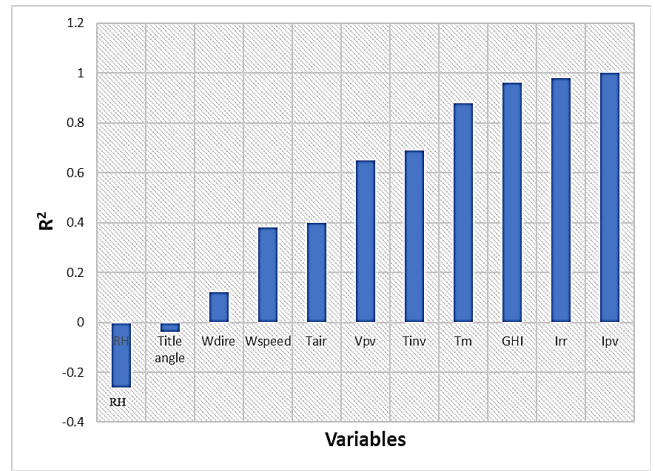


Figure 10: Correlation between Power and other Variables

Figure 11 illustrates the daily PV power trend from the 2.3 MW solar PV system at the Zagtoui power plant in May, June, July, and August 2019. Some days, notably May 4th, July 1st, July 4th, and August 3rd, saw reduced output due to cloudy conditions.

### 2.7.3. Data Normalization

The data's time series are all on distinct scales. To ensure that all of the features take small values on a similar scale, each feature was therefore independently normalized to have a mean of 0 and a standard deviation of 1. Because of their large range of values, target data were also normalized, just like input data. The following formula expressed by equation (5) was used to normalize the data:

$$X^k = \frac{x^k - \text{mean}}{\text{Std}} \tag{5}$$

where  $X^k$  is the normalized value of series k,  $x^{kk}$  is the original input data value of series k,  $\text{mean}$  is the mean of the input data value of series k and  $\text{Std}$  is the standard deviation of the input data of series k.

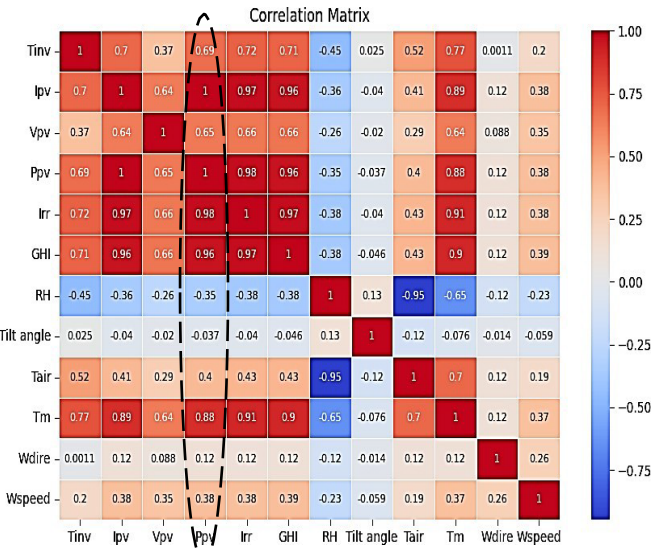


Figure 9: Correlation Matrix

The Figure 9 clearly illustrates the correlation that exists between these various variables. The output variable is the output power of the PV system (P<sub>pv</sub>), and we will focus on the correlation between this variable and others depicted in Figure 10.

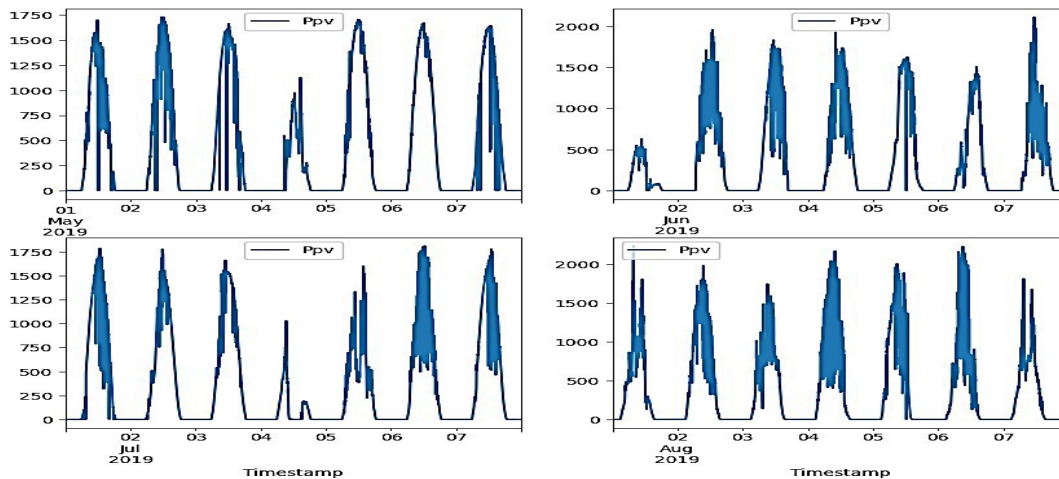


Figure 11: PV Power Evolution during the first week of May, June, July and August respectively

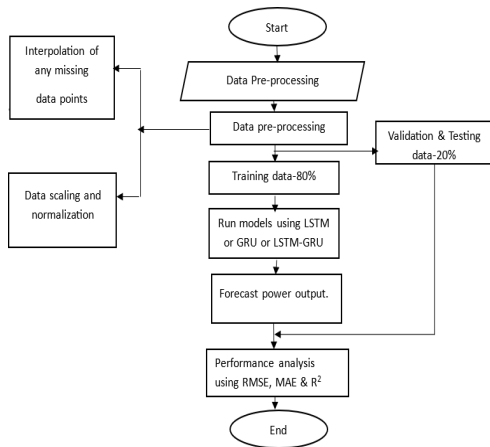


Figure 12: Flowchart of the Method

### 3. Results and Discussions

#### 3.1. Models Training and Testing

The Zagtouli Solar PV System's power output was analysed using three distinct models. These models underwent training and evaluation using identical datasets for training, validation, and testing. Figures 13-15 depict the training and validation results of the LSTM, GRU, and LSTM-GRU models, respectively. The results demonstrate that the training loss curve is higher than the validation loss curve. That means the training data is more difficult to model than the validation data. The outcomes presented were achieved with the optimal hyperparameters detailed in Table 1. All models demonstrated commendable performance on the training and validation sets because lower training loss and validation loss indicate that the model fits the data correctly. Following the training phase, the finalized models were assessed using test sets, comprising data unfamiliar to the models. Unlike validation sets, test sets are utilized to gauge a trained model's performance on previously unseen data. Table 2 provides a summary of the models' test performances, measured with the RMSE, MAE and  $R^2$  metrics, after the training. Figure 16 clearly illustrates that the hybrid model (LSTM-GRU) outperforms, followed by the LSTM and GRU models, respectively.

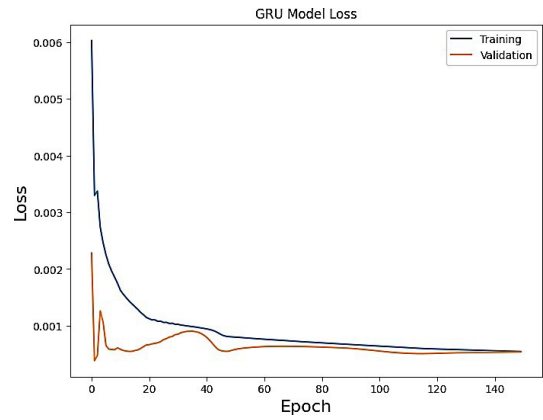


Figure 14: GRU Model Training and Validation

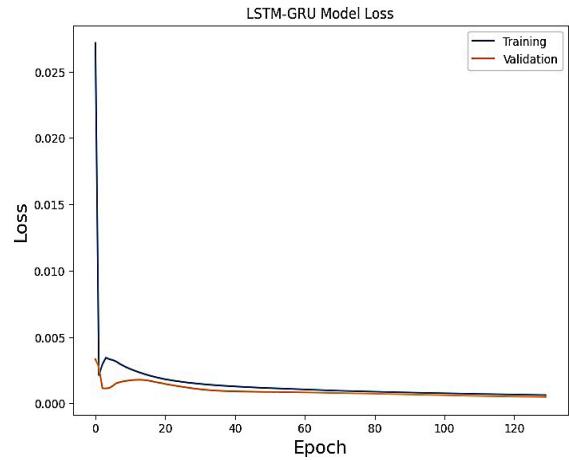


Figure 15: LSTM-GRU Model Training and Validation

Table 1. Hyper-parameters

Models	parameters	Epochs	Total Units
LSTM	37531	120	91
GRU	34601	150	101
LSTM-GRU	85626	130	181

Table 2: Test Performances using LSTM, GRU and LSTM-GRU models

Models	RMSE	MAE	$R^2$
LSTM	10.799	2.09	0.998
GRU	11.695	2.1	0.997
LSTM-GRU	10.629	2.0	0.999

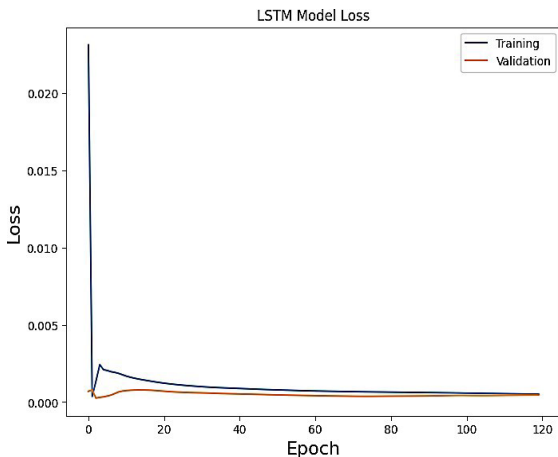


Figure 13: LSTM Model Training and Validation

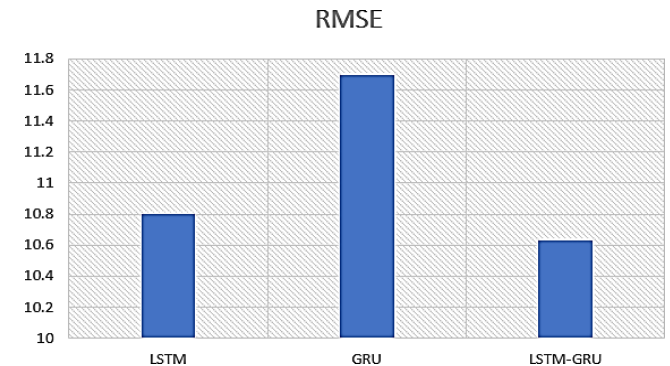


Figure 16: Root Mean Square Errors of LSTM, GRU and LSTM-GRU models

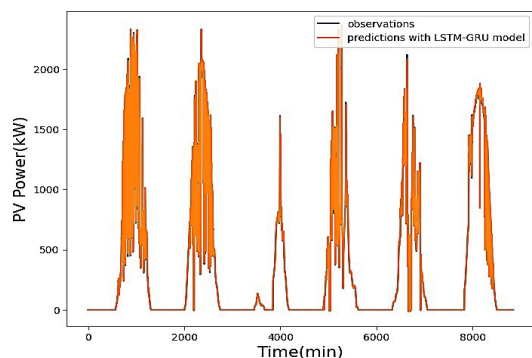


Figure 17: PV Power Forecasting vs Actual Values with LSTM-GRU

### 3.2. Solar PV Power Forecasting Using the Hybrid Model (LSTM-GRU)

To implement the proposed models, a dataset comprising 176,940 records was gathered within the period of 00:00 to 23:59, with minute intervals, covering the period from May 1, 2019, to August 31, 2019. The training phase utilized 80% of the data, amounting to 141,552 data points, and spanned a total of 98 days. Validation involved 15% of the data, equivalent to 26,541 data points, spanning a potential 18-day period. Testing utilized 5% of the data corresponding to 8,847 data points, covering 6 days, specifically modelled for predictive analysis. These six days were employed to predict solar PV power, and the comparison between observed and predicted values was illustrated in Figure 17 using the LSTM-GRU model. Throughout this period, a noticeable overlap between the two curves indicates the model's effectiveness in predicting the PV output power of Zagtouli's solar power plant. Notably, deep learning models showcased superior performance compared to traditional models based on statistical series.

Table 3 below shows some of the results of predicting photovoltaic power output using hybrid deep learning models around the world. The results of predictions are very sensitive to the nature of the input data used, the hidden layers number as well as the duration and period during which data were collected [27]. Given the results of Table 3, we can conclude that the LSTM-GRU prediction model for Zagtouli's solar power plant performs well.

Table 3. Comparison of some Hybrid deep learning models for PV Power Forecasting

Year	Location	Horizon	Model	RMSE	Reference
2019	Alice Springs (Australia)	1 day	LSTM-CNN	13.82	[28]
2020	Nevada (USA)	1 day	LSTM-RNN	6.29	[8]
2020	Limberg, (Belgium)	45 min	LSTM-CNN	6.404	[26]
2021	China	1day	VMD-ISSA-GRU	1.5511	[9]
2021	China	1 day	GRUP	6.83	[29]
2022	Rabat (Morocco)	1 day	LSTM-CNN	6.65	[30]
2023	Iran	12 hours	GSA-LSTM	10	[31]
2024	Zagtouli (Burkina Faso)	1 day	LSTM-GRU	10.629	Actual work

## 4. Conclusion and Perspectives

Predicting solar PV power effectiveness presents a viable alternative for overseeing grid-connected PV solar plants. In this investigation, we employed two deep learning techniques and their combination to forecast a system at the Zagtouli PV plant site. The hybrid model (LSTM-GRU) demonstrated superior results compared to LSTM and GRU with the RMSE metric, recording values of 10.799, 11.695 and 10.629 respectively. The data utilized for this analysis were gathered from May 2019 to August 2019, corresponding to the rainy season. In the future, data from other seasons could be employed to compare performance outcomes. This research lays the groundwork for developing an efficient and intelligent digital platform for managing the inflow of injected solar PV power into Burkina Faso's national electrical grid, aiming to secure the electrical network and optimize energy lost during continuous disconnections of power plants from the grid.

### Conflict of Interest

The authors declare no conflict of interest.

### Acknowledgements

The authors extend their gratitude to the University of Nairobi and Mohammed VI Polytechnic University for their valuable support, as well as to SONABEL for assisting in data collection at the Zagtouli PV Plant site. A special acknowledgement is also due to PASET RSIF for their financial contributions to this research endeavour.

### References

- [1] R. Ahmed, V. Sreeram, Y. Mishra, M.D. Arif, A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization, *Renewable and Sustainable Energy Reviews*, **124**, 2020, doi:10.1016/j.rser.2020.109792.
- [2] R.-E. Precup, T. Kamal, S.Z. Hassan, *Solar Photovoltaic Power Plants*, Springer Singapore, Singapore, 2019, doi:10.1007/978-981-13-6151-7.
- [3] D.K. Dhaked, S. Dadhich, D. Birla, "Power output forecasting of solar photovoltaic plant using LSTM," *Green Energy and Intelligent Transportation*, **2(5)**, 2023, doi:10.1016/j.geits.2023.100113.
- [4] S. Sattenapalli, V.J. Manohar, "Research on Single-Phase Grid Connected PV Systems," *International Journal of Engineering and Advanced Technology*, **9(2)**, 5549–5555, 2019, doi:10.35940/ijeat.b5159.129219.
- [5] H. Sharadga, S. Hajimirza, R.S. Balog, "Time series forecasting of solar power generation for large-scale photovoltaic plants," *Renewable Energy*, **150**, 797–807, 2020, doi:10.1016/j.renene.2019.12.131.
- [6] M. Elsaraiti, A. Merabet, "Solar Power Forecasting Using Deep Learning Techniques," *IEEE Access*, **10**, 31692–31698, 2022, doi:10.1109/ACCESS.2022.3160484.
- [7] P. Li, K. Zhou, X. Lu, S. Yang, "A hybrid deep learning model for short-term PV power forecasting," *Applied Energy*, **259**(November), 114216, 2020, doi:10.1016/j.apenergy.2019.114216.
- [8] F. Wang, Z. Xuan, Z. Zhen, K. Li, T. Wang, M. Shi, "A day-ahead PV power forecasting method based on LSTM-RNN model and time correlation modification under partial daily pattern prediction framework," *Energy Conversion and Management*, **212**, 2020, doi:10.1016/j.enconman.2020.112766.
- [9] P. Jia, H. Zhang, X. Liu, X. Gong, "Short-Term Photovoltaic Power Forecasting Based on VMD and ISSA-GRU," *IEEE Access*, **9**, 105939–105950, 2021, doi:10.1109/ACCESS.2021.3099169.

- [10] N.Q. Nguyen, L.D. Bui, B. Van Doan, E.R. Sanseverino, D. Di Cara, Q.D. Nguyen, "A new method for forecasting energy output of a large-scale solar power plant based on long short-term memory networks a case study in Vietnam," *Electric Power Systems Research*, **199**(June), 107427, 2021, doi:10.1016/j.epr.2021.107427.
- [11] A.P. Casares, "The brain of the future and the viability of democratic governance: The role of artificial intelligence, cognitive machines, and viable systems," *Futures*, **103**, 5–16, 2018, doi:10.1016/j.futures.2018.05.002.
- [12] F. Chollet, *Deep Learning with Python*, 2nd Edition, Manning Publications Co, 2021.
- [13] Dheeraj Mehrotra, *Basics of Artificial Intelligence & Machine Learning*, Notion Press, 2019.
- [14] W. and A.H.Q. Salah Alaloul, *Data Processing Using Artificial Neural Networks*, Intechopen, 2020.
- [15] R.C. Staudemeyer, E.R. Morris, "Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks," 2019.
- [16] M. Hussain, M. Dhimish, S. Titarenko, P. Mather, "Artificial neural network based photovoltaic fault detection algorithm integrating two bi-directional input parameters," *Renewable Energy*, **155**, 1272–1292, 2020, doi:10.1016/j.renene.2020.04.023.
- [17] R. Derakhshani, M. Zaresefat, V. Nikpeyman, A. GhasemiNejad, S. Shafieibafiti, A. Rashidi, M. Nemati, A. Raouf, "Machine Learning-Based Assessment of Watershed Morphometry in Makran," *Land*, **12**(4), 2023, doi:10.3390/land12040776.
- [18] A. Shah, M. Shah, A. Pandya, R. Sushra, R. Sushra, M. Mehta, K. Patel, K. Patel, "A comprehensive study on skin cancer detection using artificial neural network (ANN) and convolutional neural network (CNN)," *Clinical EHealth*, **6**, 76–84, 2023, doi:10.1016/j.ceh.2023.08.002.
- [19] N. V. Ranade, V. V. Ranade, "ANN based surrogate model for key Physico-chemical effects of cavitation," *Ultrasonics Sonochemistry*, **94**, 2023, doi:10.1016/j.ulsonch.2023.106327.
- [20] R. Langbauer, G. Nunner, T. Zmek, J. Klarnner, R. Prieler, C. Hochenaue, "Modelling of thermal shrinkage of seamless steel pipes using artificial neural networks (ANN) focussing on the influence of the ANN architecture," *Results in Engineering*, **17**, 2023, doi:10.1016/j.rineng.2023.100999.
- [21] C.H. Liu, J.C. Gu, M.T. Yang, "A Simplified LSTM Neural Networks for One Day-Ahead Solar Power Forecasting," *IEEE Access*, **9**, 17174–17195, 2021, doi:10.1109/ACCESS.2021.3053638.
- [22] N.L.M. Jailani, J.K. Dhanasegaran, G. Alkaws, A.A. Alkahtani, C.C. Phing, Y. Baashar, L.F. Capretz, A.Q. Al-Shetwi, S.K. Tiong, "Investigating the Power of LSTM-Based Models in Solar Energy Forecasting," *Processes*, **11**(5), 2023, doi:10.3390/pr11051382.
- [23] K. Cho, B. van Merriënboer, D. Bahdanau, Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," 2014.
- [24] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," 2014.
- [25] M.N. Akhter, S. Mekhilef, H. Mokhlis, N.M. Shah, "Review on forecasting of photovoltaic power generation based on machine learning and metaheuristic techniques," *IET Renewable Power Generation*, **13**(7), 1009–1023, 2019, doi:10.1049/iet-rpg.2018.5649.
- [26] G. Li, S. Xie, B. Wang, J. Xin, Y. Li, S. Du, "Photovoltaic Power Forecasting with a Hybrid Deep Learning Approach," *IEEE Access*, **8**, 175871–175880, 2020, doi:10.1109/ACCESS.2020.3025860.
- [27] S. Theocharides, G. Makrides, A. Livera, M. Theristis, P. Kaimakis, G.E. Georghiou, "Day-ahead photovoltaic power production forecasting methodology based on machine learning and statistical post-processing," *Applied Energy*, **268**, 2020, doi:10.1016/j.apenergy.2020.115023.
- [28] K. Wang, X. Qi, H. Liu, "A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network," *Applied Energy*, **251**, 2019, doi:10.1016/j.apenergy.2019.113315.
- [29] Y. Qu, J. Xu, Y. Sun, D. Liu, "A temporal distributed hybrid deep learning model for day-ahead distributed PV power forecasting," *Applied Energy*, **304**, 2021, doi:10.1016/j.apenergy.2021.117704.
- [30] A. Agga, A. Abbou, M. Labbadi, Y. El Houm, I.H. Ou Ali, "CNN-LSTM: An efficient hybrid deep learning architecture for predicting short-term photovoltaic power production," *Electric Power Systems Research*, **208**, 2022, doi:10.1016/j.epr.2022.107908.
- [31] D. Sadeghi, A. Golshanfard, S. Eslami, K. Rahbar, R. Kari, "Improving PV power plant forecast accuracy: A hybrid deep learning approach compared across short, medium, and long-term horizons," *Renewable Energy Focus*, **45**, 242–258, 2023, doi:10.1016/j.ref.2023.04.010.

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

# Efficient Deep Learning-Based Viewport Estimation for 360-Degree Video Streaming

Nguyen Viet Hung<sup>\*1,2</sup>, Tran Thanh Lam<sup>2</sup>, Tran Thanh Binh<sup>1</sup>, Alan Marshal<sup>3</sup>, Truong Thu Huong<sup>2</sup>

<sup>1</sup>School of Electrical and Electronic Engineering, Hanoi University of Science and Technology, Hanoi, Vietnam

<sup>2</sup>Faculty of Information Technology, East Asia University of Technology, Bacninh, Vietnam

<sup>3</sup>Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, United Kingdom

## ARTICLE INFO

Article history:

Received: 16 March, 2024

Revised: 14 May, 2024

Accepted: 29 May, 2024

Online: 12 June, 2024

Keywords:

Video Streaming

360-degree Video

QoE

VR

Deep Learning

## ABSTRACT

While Virtual reality is becoming more popular, 360-degree video transmission over the Internet is challenging due to the video bandwidth. Viewport Adaptive Streaming (VAS) was proposed to reduce the network capacity demand of 360-degree video by transmitting lower quality video for the parts of the video that are not in the current viewport. Understanding how to forecast future user viewing behavior is therefore a crucial VAS concern. This study presents a new deep learning-based method for predicting the typical view for VAS systems. Our proposed solution is termed Head Eye Movement oriented Viewport Estimation based on Deep Learning (HEVEL). Our proposed model seeks to enhance the comprehension of visual attention dynamics by combining information from two modalities. Through rigorous experimental evaluations, we illustrate the efficacy of our approach versus existing models across a range of attention-based tasks. Specifically, viewport prediction performance is proven to outperform four reference methods in terms of precision, RMSE, and MAE.

## 1. Introduction

The proliferation of mobile head-mounted display (HMD) devices has led to the widespread adoption of 360-degree video streaming within the consumer video industry [1]. Since 360-degree videos offer users immersive settings and improve the Quality of Experience (QoE) of both video-on-demand and live-video streaming, they have seen a growing amount of public interest [2]. Compared to standard videos, 360-degree videos provide a more engaging viewing experience [3]. Therefore, VR streaming has begun to be used for live broadcasts of events like sporting contests and breaking news, giving viewers access to instantly immersive experiences [1].

In comparison to traditional flat videos, 360-degree videos necessitate significantly higher resolution and frame rates. For instance, to deliver an authentic experience to consumers, 360-degree videos are expected to have resolutions up to 24K and frame rates of 60fps [4, 5]. Consequently, streaming 360-degree videos demands considerable network bandwidth.

In contrast to the approximate 25 Mb/s bandwidth requirement for traditional 4K videos, streaming a 360-degree video requires about 400 Mb/s to deliver a 4K resolution while offering a complete 360-degree viewing range [6]. However, traditional 360-degree

panoramic video streaming solutions are ineffective because they download the complete picture, while the viewer only sees a small portion of 360-degree video known as the viewport. This approach can result in the wastage of over 80% of network bandwidth [7].

Cutting-edge 360-degree video streaming techniques aim at decreasing video streaming bandwidth while retaining a good user experience. One of these is the Viewport Adaptive Streaming (VAS), which is a technique used in 360-degree video streaming to dynamically adjust the parameters based on the user's viewport. Instead of streaming the entire panoramic video, VAS delivers only the portion of the content that is currently visible to the user, optimizing bandwidth usage and improving streaming quality. The fundamental concept involves delivering the viewport (i.e., the portion of the video visible to a user) at a high bitrate (ensuring high quality), while the remaining parts are provided at a lower bitrate (ensuring lower quality) [8].

In Viewport Adaptive Streaming (VAS), these videos are segmented into tiles, each assigned weights corresponding to the user's viewport, and then transmitted accordingly. We can reduce the bandwidth of tiles that users overlook while maintaining the quality of tiles that users pay attention to by allocating the weights among the most likely tiles. However, as users view a 360-degree video,

\*Corresponding Author: Truong Thu Huong, Address: Dai Co Viet, Bach Khoa, Hai Ba Trung, Hanoi, Vietnam & Email: [huong.truongthu@hust.edu.vn](mailto:huong.truongthu@hust.edu.vn)

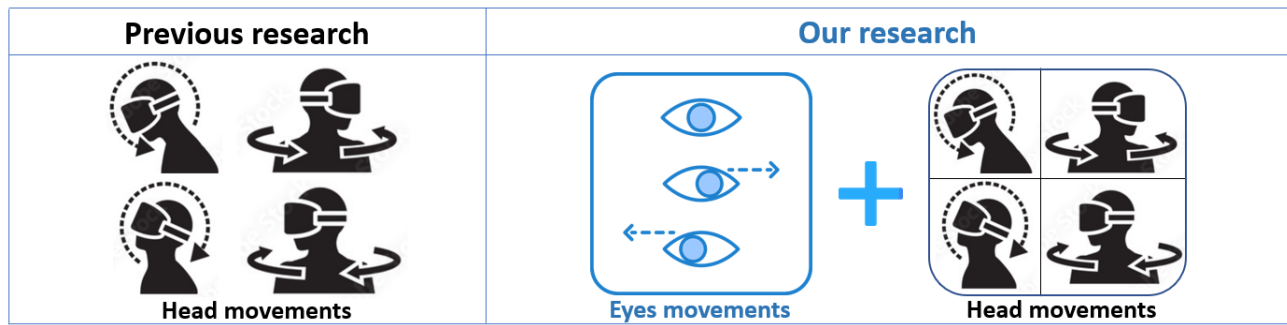


Figure 1: Head movements and eyes movements

their viewport positions frequently change. This dynamic nature underscores the necessity of viewport prediction. Viewport prediction refers to the process of forecasting where a user's viewport will be located in the near future. It is a fundamental component of delivering immersive 360-degree video experiences. By intelligently anticipating where the viewer's attention will be focused, streaming services can optimize resource usage, improve quality, reduce latency, enhance interactivity, and ultimately provide a more satisfying and immersive viewing experience.

Thus, a wrong viewport prediction will drastically lower the quality of the VAS system due to wasted bandwidth on unnoticed areas. Therefore, viewport prediction is a crucial requirement for VAS, which consequently requires a high level of precision in viewport prediction [9].

Nonetheless, forecasting the user's point of view with intensive accuracy is challenging. Users can shift their viewpoints as they rotate their heads in different directions. Additionally, the user's viewpoint may vary due to eye movements with or without moving their heads. Therefore, eye movements should be an important component to be considered. As Figure 1 illustrates, regarding the user's movements when watching 360 videos, while the recent methods only take the head movement into account, while our research considers both eye movements and head movements to determine the viewport more accurately.

Regarding viewport prediction, recently machine learning and deep learning-based solutions have been widely used for viewport prediction applications to enhance user experience [10, 11, 12, 13]. Based on past behavior and other characteristics, these solutions can learn and forecast the possibility that a user would pay attention to a particular area of the screen. Therefore, deployment of a proper Deep learning model that includes more actual behaviors of users potentially provides a more accurate predictive ability.

In this paper, we propose an accurate viewport prediction for 360-degree video viewport adaptive streaming that is based on a deep learning technique - LSTM and takes into account both the head and eye movements of viewers. Our solution is proven to outperform some current models like RNN [14], GRU [15], AVEE [16], and GLVP [9] in terms of Precision by 2.65%.

The remainder of this paper is structured as follows: The related work is discussed in Section 2. The proposed viewport estimation method is described in Section 3. The performance evaluation can be found in Section 4. Finally a discussion of our findings and pending issues is presented in 5.

## 2. Related work

Viewport adaptive streaming techniques have been proposed to address the bandwidth limitations of 360-degree videos, as evidenced by studies [5, 8, 17, 18, 19, 20]. These methods involve dividing the video into tiles and encoding each tile in multiple quality versions. The tiles within the user's viewport are transmitted in high quality, while the remaining tiles are delivered in lower quality [8]. This approach reduces redundancy in transmission and allows for flexible adaptation of tile quality based on the predicted viewport, taking into account network capacity constraints [21]. In the realm of Machine learning, researchers in [22] have employed machine learning techniques to anticipate future head movements, indirectly inferring future viewport positions. By analyzing past head movement data and leveraging statistical patterns observed in other viewers of spherical videos, the author utilizes regression models, including linear regression, to estimate the head orientation at each time point during the video. Similarly, linear regression has been used to predict motion-based fixation [13]. However, the authors in [23, 3] have shown that linear regression may be less impressive compared to alternative algorithms. In [23], algorithms such as average (Avg), linear regression (LR), and weighted LR (WLR) are compared, with the weighted LR yielding the best results. Furthermore, [3] demonstrates the performance of three machine learning models, including linear regression, Bayesian regression, and random forest, with random forest achieving the best results. Moreover, researchers in [7] proposed an advanced machine learning approach utilizing 3DCNN (3D-Convolutional Neural Networks) and LSTM-based RNN (Long Short-Term Memory) to extract spatiotemporal features from videos. This approach is proven to outperform other strategies in terms of accuracy and prediction [7].

However, it is important to note that all of the above models only consider the head movements to estimate the user's viewport positions, which could be a limitation since other factors, such as eye movement, also contribute to the viewport and may result in non-generalizable predictions. Thus, there is an overlooked potential for accuracy enhancement by incorporating both head and eye movements to the viewport prediction task.

Therefore, in this article, we design a solution that uses a deep-learning machine to forecast the viewport for the VAS system, using Long-Short Term Memory (LSTM) cells in Figure 2. This deep learning solution with LSTM can achieve the ideal balance between accuracy and redundancy [9]. During the learning process, head and eye movements are taken into account to increase the accuracy of



the precognition of future viewpoints.

This deep learning solution with LSTM can achieve the ideal balance between accuracy and redundancy [9]. During the learning process, head and eye movements are taken into account to increase the accuracy of the precognition of future viewpoints.

### 3. Proposed viewport estimation method - HEVEL

The proposed solution is named HEVEL as an acronym for **H**ead **E**ye Movement oriented **V**iewport **E**stimation based on Deep **L**earning. Before the HEVEL design is described, we will formulate the video streaming problem with the assumptions and constraints in Section 3.

#### 3.1. Problem Formulation

On the one hand, we define several concepts below to address the viewport prediction problem. Assume  $P(t_0)$  is the position of the viewport at time  $t_0$ . A viewport's center point can be located by using the longitude and latitude values of the viewport [24, 25]. Figure 3 illustrates how spherical video captures a scene from every angle. It is the primary form of content used in Virtual Reality, giving viewers an immersive experience. The viewport is the portion of the video that a user can currently watch due to their field of vision.

On the other hand, predicting where the viewport  $P(t_0 + m)$  will be in time  $t_0$  in the future is the responsibility of the viewport predictor. At the same time, the letter  $m$  is used to indicate the forecast horizon. As shown in Figure 4, the predictor must offer a prediction for the interval  $[t_0 + m, t_0 + m + s]$ , where  $s$  stands for the segment duration because 360-video streaming is commonly done on a segment/adaptation interval basis [26, 27].

#### 3.2. Viewport prediction and selection framework

In our proposed HEVEL framework, we employ the LSTM model because it can capture and model long-term dependencies within input data. By maintaining an internal state capable of retaining information from previous time steps, LSTM helps the HEVEL framework effectively model and utilize historical information of head and eyes movements, thereby enhancing its predictive or analytical capabilities.

Modeling these long-term dependencies is achieved through gating mechanisms that selectively enable the network to hold or discard information from past time steps [28]. In addition, HEVEL also helps to save memory usage since it eliminates unnecessary positions by adapting to both head and eye movements. As a result, our approach requires less memory while ensuring accurate viewport predictions. Each of the  $n$  inputs in the Long Short-Term Memory (LSTM) model corresponds to  $n$  video frames or images that have been taken out of a video.  $\{x_1, x_2, \dots, x_{n-1}, x_n\}$  is a representation of this sequence. Each input  $x_i$  represents the visual information present in the video sequence's  $i$ -th frame. The LSTM analyzes this series of inputs in time steps while taking into consideration their temporal relationship.

The LSTM creates an output known as  $y_t$  at each time step  $t$ . This output is the anticipated viewport, which is the precise region of interest within the video frame that a viewer is most likely to

pay attention to at that exact time step. The geographical position and visual information that are thought to be most important to the viewer's attention at the given time instant are essentially encapsulated by  $y_t$ .

In LSTM, the cell represents the long-term memory component. It is responsible for storing and updating information over time. The cell maintains a state vector that carries information from previous time steps and is passed along through the recurrent connections. In contrast, the hidden state represents the output of the LSTM unit at a given time step. It carries information relevant for making predictions at the current time step stored in the cell. Moreover, in LSTM network design, gates play an important role in regulating the flow of information through the network. Its purpose is to determine how much of the new information should be stored in the cell state. As Figure 5 illustrates,  $c_{t-1}, h_{t-1}$  are the cell state and hidden cell state respectively at time  $t - 1$ . While  $c_t, h_t$  are the cell state and hidden state at time  $t$  that will be used as input for the next cell. At the end,  $h_t$  represents output  $y_t$  - the predicted viewport.

Next,  $W_{i\alpha}, W_{i\beta}, W_f, W_{o\alpha}, W_{o\beta}$  are the recurrent weight matrices,  $b_{i\alpha}, b_{i\beta}, b_f, b_{o\alpha}, b_{o\beta}$  are the bias terms.

Additionally, we employ  $\tanh$  as the tanh function and  $\sigma$  as the logistic sigmoid function. The gates used in the cell are defined as follows:

- **Input gate** -  $i_t, i_{\alpha t}, i_{\beta t}$ : to decide which values should be stored in the cell state.
- **Forget gate** -  $f_t$ : to decide which information from the long-term memory should be kept or discarded.
- **Output gate** -  $o_{\alpha t}, o_{\beta t}, y_t$ : to decide which data from the current cell will be managed as output.

In more detail, the working mechanism of those gates can be described as follows:

**Input gate:** The input gate helps regulating the process of deciding what to forget and what new information to store in the cell state. The input gate with data from the current input  $x_t$  and short-term memory from the last time step  $h_{t-1}$  to determine what brand-new data should be kept. In this part, we have designed the input interface in two layers:

- **First layer:** The sigmoid activation function is used in the first layer to help an LSTM cell decide which information from the input and previous cell state is important to retain or forget. Because the Sigmoid Activation Function turns any input into a number between 0 and 1, the output determines what percentage of the Long-term Memory is remembered. Output 0 indicates that part of the information is unimportant, akin to forgetting that piece of information. In contrast, 1 suggests that the information will be retained. This is crucial for managing the memory of past viewport positions and deciding what new information to incorporate. Formula 1 encapsulates this process.

$$i_{\alpha t} = \sigma(W_{i\alpha} \otimes (h_{t-1}, x_t) + b_{i\alpha}) \quad (1)$$

- **Second layer:** The tanh activation function is used in the second layer to regulate the output gate of LSTM cells. To normalize the output values  $i_{\beta t}$  obtained from equation (2) in the LSTM architecture, we normalize the values from the

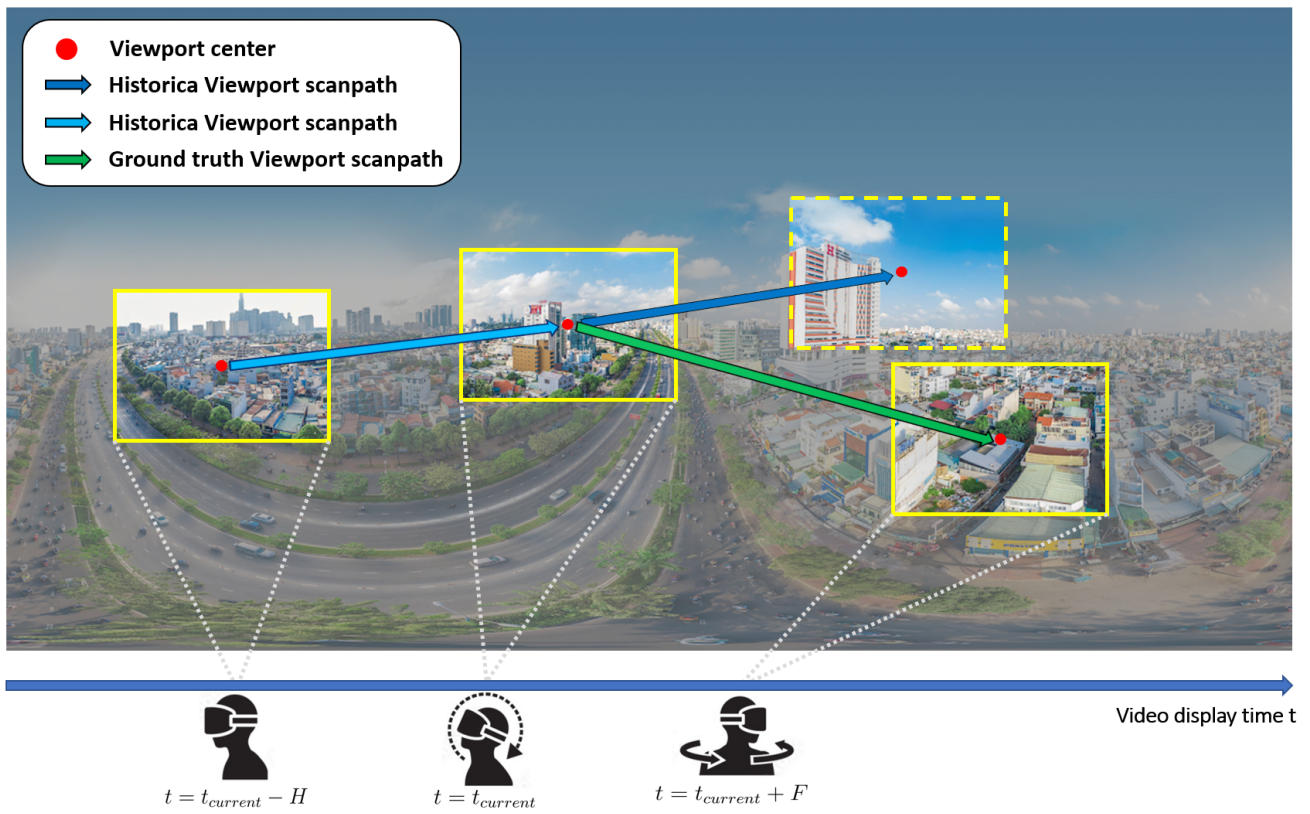


Figure 2: The Proposed model compares the viewport sweep in the past H-seconds to predict the viewport sweep in the future F seconds.

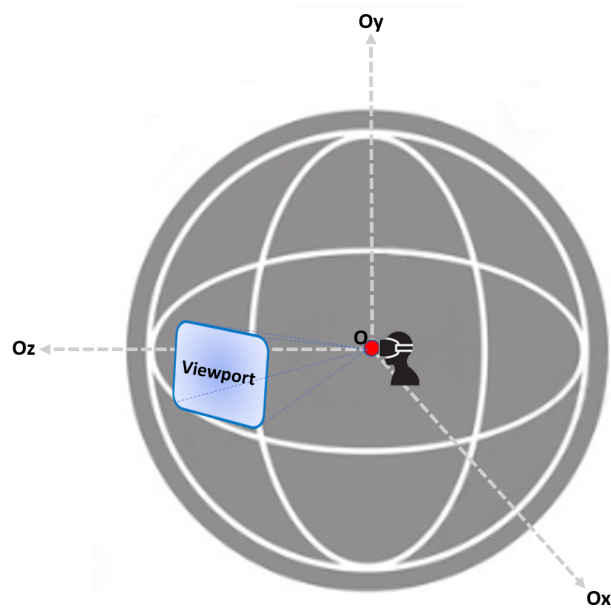


Figure 3: The viewport of a user at a time

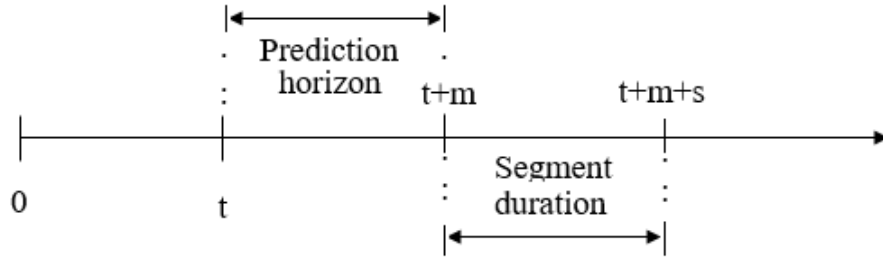


Figure 4: Problem formulation of Viewport Prediction

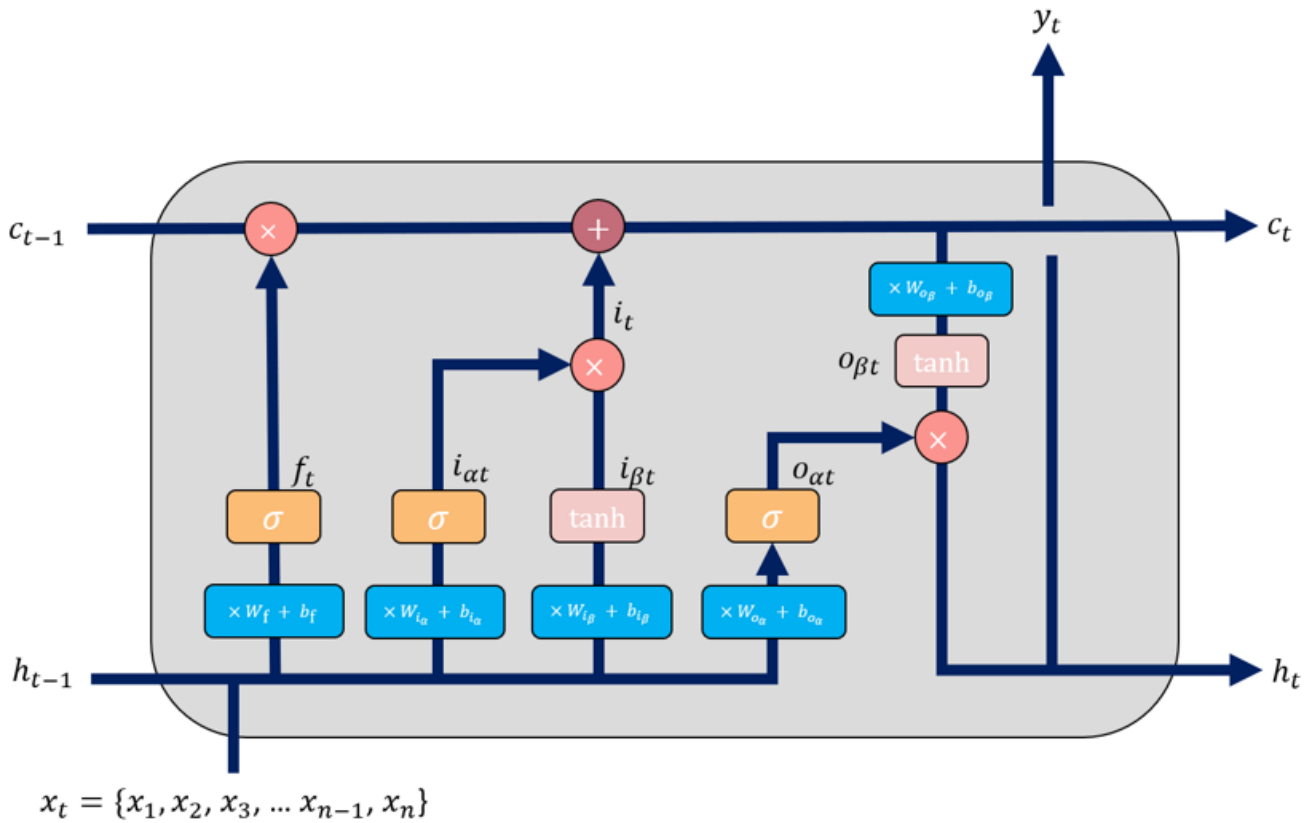


Figure 5: LSTM model for viewport estimation

Table 1: Characteristics of the five experimental videos

Video	Description
Turtle	People are releasing baby turtles into the sea on the beach during the day
Bar	Light, users moving, bartender at work
Ocean	Under the ocean, people are going underwater to see whales.
Sofa	People are sitting on sofas in the living room to talk
Po. Riverside	Riverside, outdoor, during the day, with human activities

range [-1, 1] to [0, 1] by the formula:  $no_v = (ot_v + 1) / 2$ , with  $ot_v$  represents the value obtained from the LSTM cell within the range [-1, 1], and  $no_v$  represents the corresponding normalized value within the range [0, 1]. Layer 2 combines the Short-term memory and the input to create a potential long-term memory. Formula 2 shows the process.

$$i_{\beta t} = \tanh(W_{i\beta} \otimes (h_{t-1}, x_t) + b_{i\beta}) \quad (2)$$

Note that, placing the tanh activation function in Layer 1 might disrupt the gating mechanism of the LSTM cell. The purpose of using sigmoid in Layer 1 is to regulate the flow of information, deciding what information to keep and discard. Tanh activation in this context might not provide the necessary gating behavior required for effective memory management.

Finally, we calculate  $i_t$  by multiplying the above two layers, and obtain the input gate by Formula 3:

$$i_t = i_{\alpha t} * i_{\beta t} \quad (3)$$

**Forget gate:** The forget gate  $f_t$  (Formula 4) selects which data from the long-term memory should be retained or erased. This is calculated using the previously hidden state  $h_{t-1}$  and the current input  $x_t$  with the *sigmoid* function, similarly to the first input layer  $i_{\alpha t}$  but with different weights.

$$f_t = \sigma(W_f \otimes (h_{t-1}, x_t) + b_f) \quad (4)$$

The forget gate  $f_t$  is a scalar value between 0 and 1 used to weigh the previous cell state when combined with the current input to update the cell state. A value of 0 indicates the last cell state has been completely forgotten. Whereas a value of 1 indicates that have should be retained entirely. The LSTM cell can simulate long-term dependencies and recall pertinent information over time thanks to the forget gate's assistance in selectively maintaining and forgetting information from the input sequence.

**Cell state:** The cell state  $c_t$  (Formula 5) is a new feature of LSTM based on RNN to overcome the memory limitation of the old algorithm. It is a "memory" in an LSTM cell that stores information from the input sequence over time. The update at each time step is based on the input  $x_t$ , forget  $f_t$ , and the previous cell state  $c_{t-1}$ . The outcomes of the **Input** and **Forget gates** will be added pointwise to create a new version of the long-term memory that will be sent to the next cell.

$$c_t = c_{t-1} * f_t + i_t \quad (5)$$

Furthermore, the new long-term memory will be employed in the output gate, which is the last gate.

**Output gate:** The following hidden state's value is determined. This state stores information from previous inputs. Two layers can be described as follows:

- **First layer:** Once again, the current input and the hidden state are passed into a sigmoid function to generate the last filter  $o_{\alpha t}$  (Formula 6):

$$o_{\alpha t} = \sigma(W_{o\alpha} \otimes (h_{t-1}, x_t) + b_{o\alpha}) \quad (6)$$

- **Second layer:** an activation tanh function is implemented on the newly created long-term memory from the cell state in Formula 7:

$$o_{\beta t} = \tanh(W_{o\beta} \otimes c_t + b_{o\beta}) \quad (7)$$

Finally, using Formula 6 and 7, new short-term memory is calculated according to the formula below:

$$h_t, y_t = o_{\alpha t} * o_{\beta t} \quad (8)$$

The new short-term memory refers to the hidden state  $h_t$ , and long-term memory  $c_t$  will be moved to the next cell, where the process will repeat. The outcome of each step will also derive from  $h_t$ , which we define as  $y_t$ . The Pseudo code in Algorithm 1 summarizes the entire process of estimating viewpoints.

## 4. Performance Evaluation

### 4.1. Experimental Settings

In our experiment, we employ five 360-degree videos with different contexts, as shown in Table 1, and Figure 6. The videos depict a variety of situations and environments. Using such a diverse set of videos, we can assess the model's ability to deal with diversified situations. For example, video Turtle possesses turtle movement that may cause a user's eyes and head to continuously move to track this object location. While video Bar presents a complex scenario with its varying lighting and conditions, human presence, and sound. This intricate mix of factors may cause a user to focus at some random spots from one time to another time. In video Ocean, space, height, frequency, and direction of waves and marine life will affect observations through water clarity and visibility. Video Sofa offers a static setting, with only small movements and tends to be quiet in sound. Video Po. Riverside Video shows vivid and exciting portraits of the plant world, surrounding trees and birds, insects, and other animals. Each video contains a corresponding head-eye motion trace. Even while the head position is fixed, the eyes can still move to different positions and change the viewport. The video's head-eye movement traces are obtained from the dataset in [29] and illustrated in Figure 7. Each 60-second video is encoded and projected using equirectangular projection with a 4K quality (3840 × 1920). The video is divided into 24 tiles, each with a resolution of 480 × 480. In this context, "pitch" refers to the user's head-eye position's longitude, which ranges from -180 to 180 degrees. "Yaw" represents the latitude and has a range of -90 to 90 degrees.

On the other hand, we conduct the experiments using the same five videos to facilitate an unbiased comparison of the performance of HEVEL with the reference models.

### 4.2. Viewport prediction performance

The viewport prediction performance of HEVEL is compared with the current reference models such as GLVP [9], RNN [14], GRU [15], and A EVE [16] in terms of Precision, RMSE (Root Mean Square Error), MAE (Mean Absolute Error).



(a) Bar



(b) Ocean



(c) Porto Riverside



(d) Sofa



(e) Turtle

Figure 6: Experimental 360-Degree videos

---

**Algorithm 1: Viewport Estimation**

---

**Input:**  $c_{t-1}, h_{t-1}, x_t$

**Output:**  $c_t, h_t, y_t$

```

1 for  $t = 1$  to  $N$  do
2   Calculate  $i_{\alpha t} = \sigma(W_{i\alpha} \otimes (h_{t-1}, x_t) + b_{i\alpha})$ 
3   Calculate  $i_{\beta t} = \tanh(W_{i\beta} \otimes (h_{t-1}, x_t) + b_{i\beta})$ 
4   Calculate  $i_t = i_{\alpha t} \odot i_{\beta t}$ 
5   Calculate  $f_t = \sigma(W_f \otimes (h_{t-1}, x_t) + b_f)$ 
6   Calculate  $c_t = c_{t-1} \odot f_t + i_t$ 
7   Calculate  $o_{\alpha t} = \sigma(W_{o\alpha} \otimes (h_{t-1}, x_t) + b_{o\alpha})$ 
8   Calculate  $o_{\beta t} = \tanh(W_{o\beta} \otimes (c_t) + b_{o\beta})$ 
9   Calculate  $h_t = o_{\alpha t} \odot o_{\beta t}$ 
10  Calculate  $y_t = h_t$ 
11 end
    
```

---

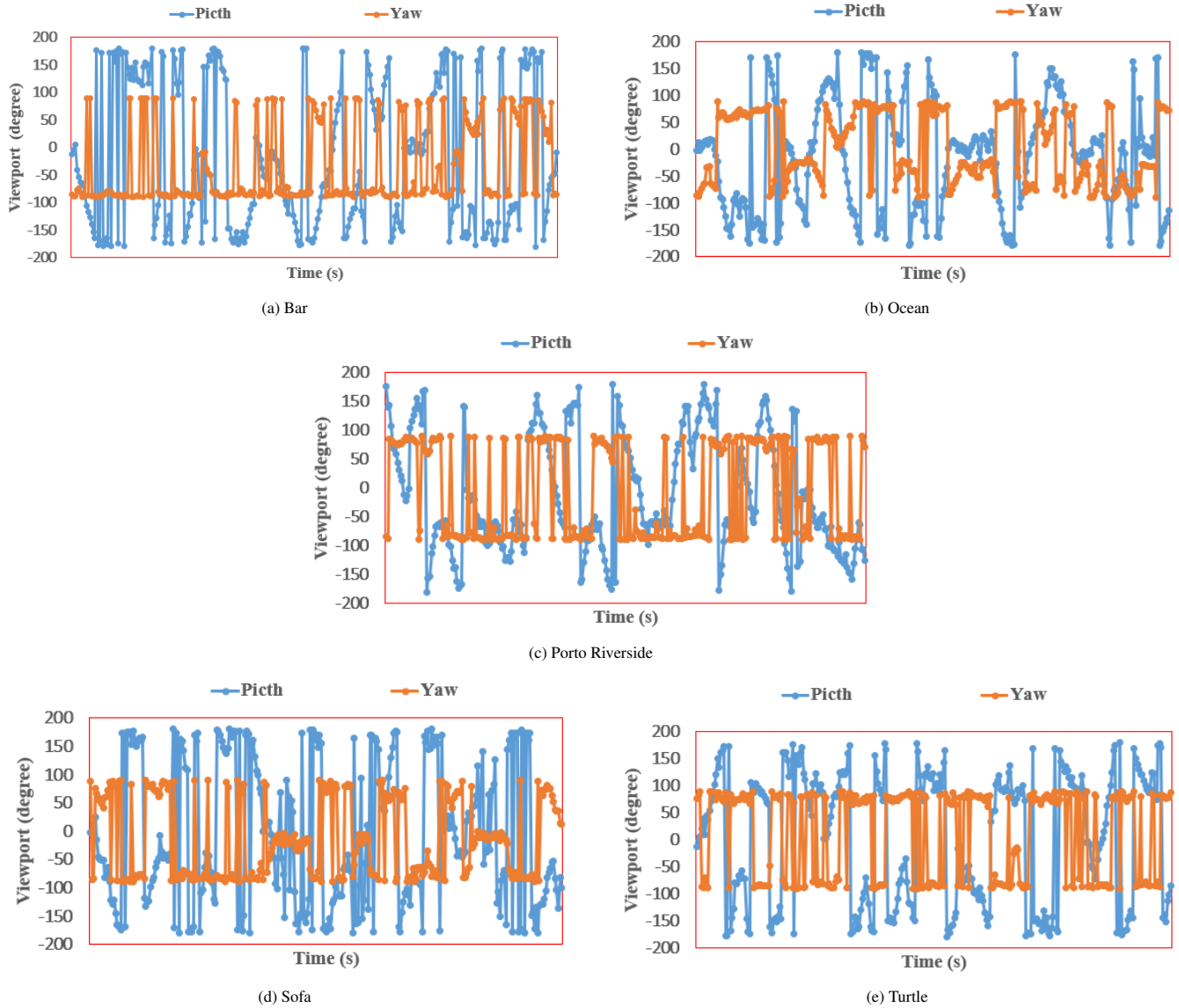


Figure 7: Head and eye movements of experimental videos over time

**Precision:** a fraction of correctly-predicted viewport positions relative to all the viewport positions predicted.

$$Precision = \frac{TP}{TP + FP} \quad (9)$$

Where:

- TP is the number of correctly predicted viewport positions.
- FP is the number of incorrectly predicted viewport positions.

In addition, to evaluate the precision of our prediction solution, each VR video is divided into small cells as illustrated in Fig. 8. Within Figure 8, you can observe two regions depicting the actual and predicted regions evolving over time. When the expected and actual regions overlap, as demonstrated in Figure 8a, the prediction is classified as a TP (True Positive). If the predicted and actual regions fail to overlap, the prediction is considered an FP (False Positive), as shown in Fig. 8b. If the overlapped area of the actual

and estimated viewport exceeds a threshold of 50%, it is considered a TP; otherwise, it is counted as an FP.

**RMSE:** the average magnitude of the errors between predicted and observed viewport positions.

$$RMSE = \sqrt{\frac{\sum_{t=1}^N (Act_t - Pre_t)^2}{N}} \quad (10)$$

**MAE:** the average absolute magnitude of the errors between predicted and observed (true) viewport positions.

$$MAE = \frac{1}{N} \sum_{i=1}^n |Act_i - Pre_i| \quad (11)$$

Where:

- $Pre_t$  is the predicted viewport position for data point  $i$
- $Act_t$  is the actual viewport position in the testing data set
- $N$  is the number of data points.

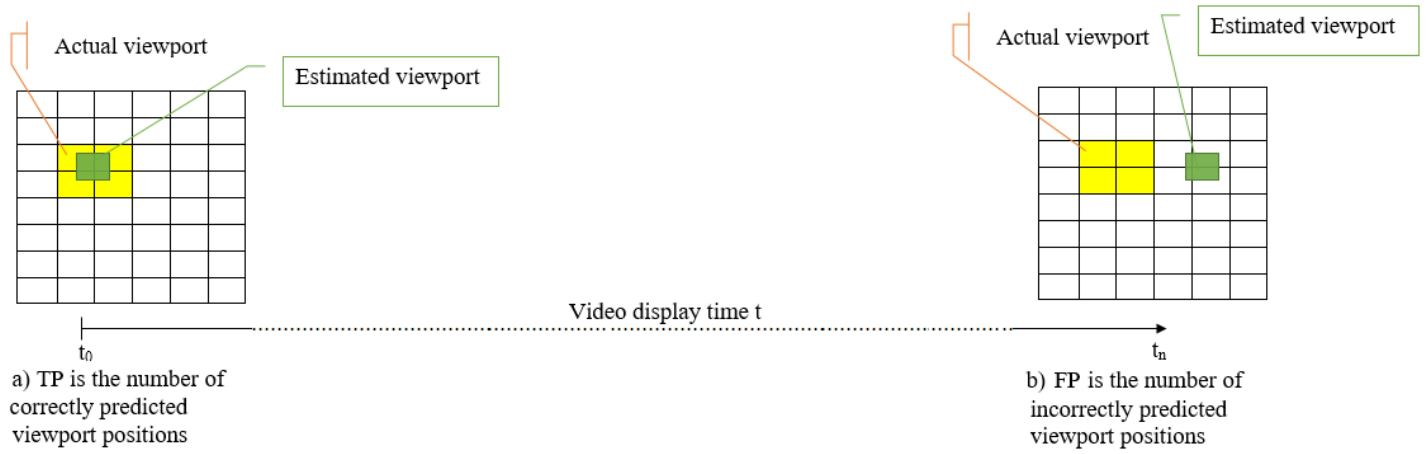


Figure 8: Calculation of TP and FP

Table 2: Accuracy (%) of HEVEL compared to the reference methods

Videos	GRU	RNN	GLVP	HEVEL
<b>BAR</b>	61.65	62.12	74.21	<b>84.54</b>
<b>Ocean</b>	60.52	71.30	73.21	<b>85.86</b>
<b>Po. Riverside</b>	88.65	87.23	90.11	<b>91.62</b>
<b>Sofa</b>	74.21	58.16	74.21	<b>85.86</b>
<b>Turtle</b>	72.57	71.62	74.21	<b>75.58</b>
<b>Average</b>	<b>70.09</b>	<b>68.62</b>	<b>76.64</b>	<b>84.54</b>

For the RMSE metric, as shown in Figure 9, the CDF curves for HEVEL is the steepest curve among the curves for RNN, A EVE, GRU, GLVP, this means that HEVEL generally exhibits smaller RMSE values compared to those reference methods across a range of data points. This suggests that HEVEL exhibits better prediction performance than the reference methods.

For the MAE metric, as shown in Figure 10, the CDF for HEVEL is also steeper than the curves for RNN, A EVE, GRU, and GLVP. The results demonstrate that those reference methods generally have larger MAE values compared to HEVEL across a range of data points. This suggests that HEVEL is better than those reference methods in terms of MAE.

The result is shown in Figure 11 in which HEVEL has the steepest CDF curve. It indicates that HEVEL consistently achieves higher precision across various decision thresholds. Therefore, HEVEL is proven to be more effective at correctly identifying correct viewport positions and avoiding false viewport prediction compared to the other reference methods.

As Table 2 shows, HEVEL improve accuracy in comparison with to the reference methods by from 1.37% to 27.71%. It consistently exhibits higher accuracy, with the lowest recorded accuracy in the Turtle video being approximately 75.58%. In the Ocean video, the proposed method surpasses GRU by 25.35%, RNN by 14.56%, and GLVP by 12.65% in terms of accuracy.

Therefore, HEVEL is proven to be more effective at correctly identifying correct viewport positions and avoiding false viewport prediction compared to the other reference methods.

The findings indicated above can come from the facts that LSTM, leveraging its capacity to grasp long-term relationships

in sequential data, effectively preserves and transmits information across extended sequences. This makes it particularly adept for tasks such as viewport prediction, which heavily relies on prolonged dependencies, consequently enhancing overall performance. Moreover, LSTM allows it to selectively update and forget unnecessary information, allowing the model to focus on relevant information while ignoring noise input. This helps the LSTM process be more effective.

LSTM’s gating mechanism and memory cell provide a more stable and efficient training process than other models like GLVP, GRU, RNN, and A EVE. This stability and efficiency ensure more reliable convergence and improved performance, even in challenging scenarios like shaky videos.

Moreover, since HEVEL incorporates eye and head movements, it gains a more holistic perspective on the user’s focus and intention. This input understanding allows the model to make more accurate prediction and adapt to various scenarios, leading to improved performance.

In summary, our prediction model has been proved to guarantee users to be consistently provided the most relevant and engaging content at any given moment.

### 4.3. Training time evaluation

In general, for such a video application, once a viewport prediction model has been built, we can use it for a long time. Because the data pattern and user’s behavior of such an application do not change rapidly over time. However, in some specific case, it might be necessary to re-train the learning model regularly in real-time. In that case is good to have more insight into the training time of the

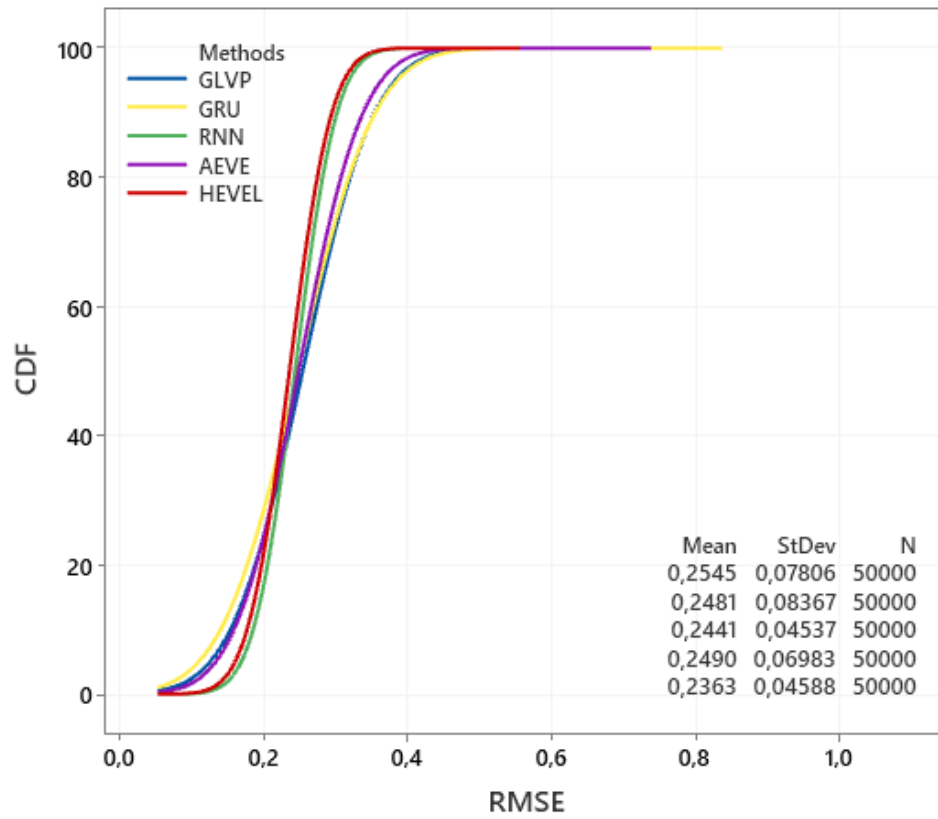


Figure 9: CDF of RMSE of HEVEL vs. the reference methods

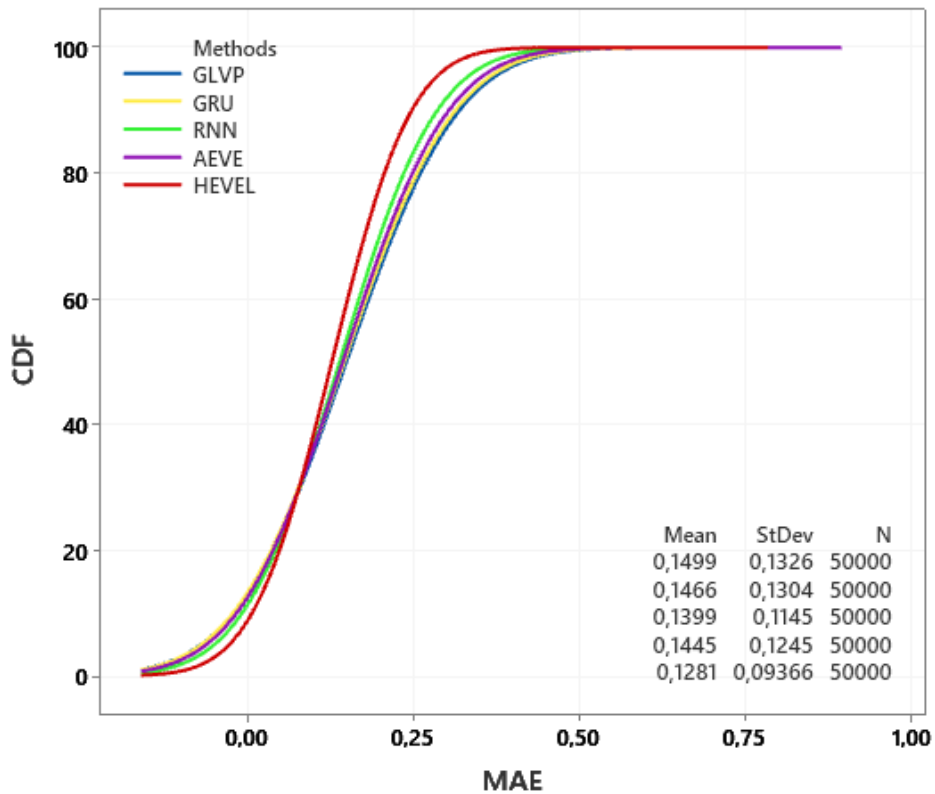


Figure 10: CDF of MAE of HEVEL vs. the reference methods



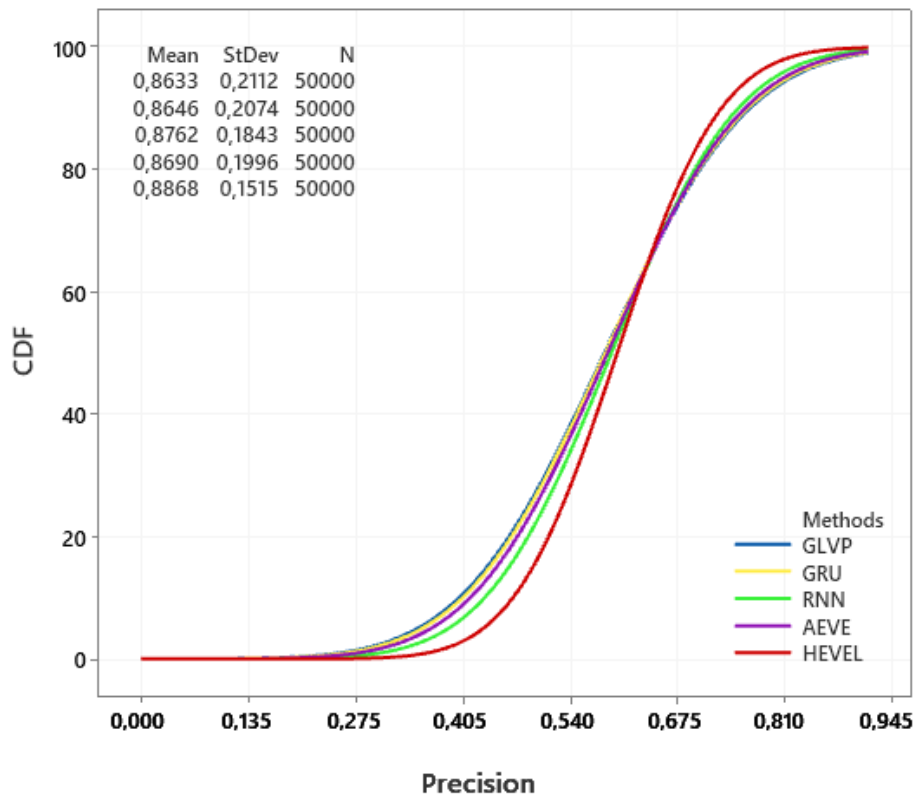


Figure 11: CDF of Precision of HEVEL vs. the reference methods

current prediction model solutions.

Figure 12 shows that the reference methods GRU, A EVE, GLVP, and our proposed HEVEL take a training time of more or less than 0.1s. In summary, our approach addresses the dual concerns of training efficiency and real-time responsiveness. The experiment’s outcomes firmly position our proposed HEVEL method as a promising solution for practical applications on devices with inherent resource constraints.

### 5. Conclusions, Discussion and Future work

In this research, we have delved into a specific focus on addressing the challenging viewport prediction task. The proposed solution has been proven to outperform the 4 reference methods in several critical evaluation metrics, including Precision, RMSE, and MAE. By accurately predicting the user’s viewport, VR video streaming has taken one more step toward reducing the latency and improving the quality of VR content delivery, ultimately enhancing user satisfaction on a more immersive and enjoyable VR services in multiple domains, including gaming, education, training.

For the future work, there are opportunities for further exploration and refinement of our approach such as adapting our solution to different VR hardware configurations. It would entail tailoring our approach to work seamlessly with the various VR devices that are currently available. This could include popular headsets like the Oculus Rift, HTC Vive, or PlayStation VR, as well as emerging technologies like augmented reality (AR) glasses. Each hardware con-

figuration may have distinct technical specifications and interaction mechanisms, necessitating adjustments to improve our solution’s performance and usability. In addition, we need understanding and modeling more user behaviors in VR environments to improve the overall VR experience. Our approach could concentrate on specific user behaviors, like locomotion and object manipulation. However, many other aspects of user interaction and engagement in VR can be investigated and integrated. This could entail researching and incorporating interaction patterns such as more behaviors, communication cues, and collaborative actions to create more immersive and realistic virtual worlds.

Furthermore, more research could be done to investigate user preferences, comfort levels, and physiological responses in VR. Understanding how users perceive and interact with virtual environments allows us to tailor our approach to better meet their needs and preferences. This could include conducting user studies, gathering feedback, and iteratively optimizing our solution based on the findings.

Another space for exploration can be investigation on incremental or online learning paradigm where the model is trained continuously as new data becomes available, without needing to retrain on the entire dataset.

The investigation can be extended to combining LSTM with other deep learning model such as GRU in order to boost up the training and processing data like VR services faster.

Last but not least, using information about the head and eyes movement may pose some challenges in capturing long-term depen-

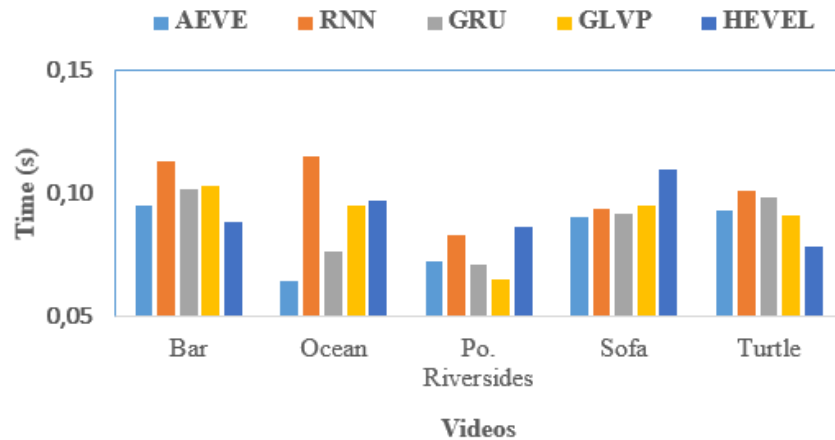


Figure 12: Training time overview

dependencies. That is, if there is a long period between when the person looks at an object and then moves their head to look at a new location. This creates a significant time delay between related events. LSTM is designed to understand and remember long-term data patterns through its LSTM gate mechanism, helping it retain important information and forget unnecessary information. However, if the delay between events is too large, LSTM may struggle to maintain long-term dependencies and retain complete information. Thus, our next work will involve around the question: how to preprocess the input data to cope with the large delay between 2 events.

## Acknowledgment

This research is funded by the Hanoi University of Science and Technology (HUST) under Project number T2022-PC-012; and partly funded by Project 2395 of the Ministry of Science and Technology of Vietnam.

## References

- [1] X. Feng, Y. Liu, S. Wei, "LiveDeep: Online Viewport Prediction for Live Virtual Reality Streaming Using Lifelong Deep Learning," in 2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), 800–808, 2020, doi:10.1109/VR46266.2020.00104.
- [2] Y. Zhang, et al, "DRL360: 360-degree Video Streaming with Deep Reinforcement Learning," in IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 1252–1260, 2019, doi:10.1109/INFOCOM.2019.8737361.
- [3] M. M. Uddin, J. Park, "Machine learning model evaluation for 360° video caching," in 2022 IEEE World AI IoT Congress (AIIoT), 238–244, 2022, doi:10.1109/AIIoT54504.2022.9817292.
- [4] N. V. Hung, B. D. Tien, T. T. T. Anh, P. N. Nam, T. T. Huong, "An efficient approach to terminate 360-video stream on HTTP/3," in AIP Conference Proceedings, volume 2909, AIP Publishing, 2023.
- [5] D. V. Nguyen, et al, "Scalable 360 Video Streaming using HTTP/2," in 2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP), 1–6, 2019, doi:10.1109/MMSP.2019.8901805.
- [6] M. Zink, et al, "Scalable 360° Video Stream Delivery: Challenges, Solutions, and Opportunities," Proceedings of the IEEE, **107**(4), 639–650, 2019, doi:10.1109/JPROC.2019.2894817.
- [7] S. Park, A. Bhattacharya, Z. Yang, S. R. Das, D. Samaras, "Mosaic: Advancing user quality of experience in 360-degree video streaming with machine learning," IEEE Transactions on Network and Service Management, **18**(1), 1000–1015, 2021, doi:10.1109/TNSM.2021.3053183.
- [8] H. L. Dieu Huong, et al, "Smooth Viewport Bitrate Adaptation for 360 Video Streaming," in 2019 6th NAFOSTED Conference on Information and Computer Science (NICS), 512–517, 2019, doi:10.1109/NICS48868.2019.9023807.
- [9] H. Nguyen, et al, "An Accurate Viewport Estimation Method for 360 Video Streaming using Deep Learning," EAI Endorsed Transactions on Industrial Networks and Intelligent Systems, **9**(4), e2, 2022, doi:10.4108/eetinis.v9i4.2218.
- [10] Y. Jiang, et al, "Robust and Resource-efficient Machine Learning Aided Viewport Prediction in Virtual Reality," arXiv preprint arXiv:2212.09945, 2022.
- [11] J. Adhuran, et al, "Deep Learning and Bidirectional Optical Flow Based Viewport Predictions for 360° Video Coding," IEEE Access, **10**, 118380–118396, 2022, doi:10.1109/ACCESS.2022.3219861.
- [12] L. Zhang, et al, "MFVP: Mobile-Friendly Viewport Prediction for Live 360-Degree Video Streaming," in 2022 IEEE International Conference on Multimedia and Expo (ICME), 1–6, 2022, doi:10.1109/ICME52920.2022.9859789.
- [13] Y. Ban, et al, "Exploiting Cross-Users Behaviors for Viewport Prediction in 360 Video Adaptive Streaming," in 2018 IEEE International Conference on Multimedia and Expo (ICME), 1–6, 2018, doi:10.1109/ICME.2018.8486606.
- [14] C.-L. Fan, et al, "Optimizing Fixation Prediction Using Recurrent Neural Networks for 360° Video Streaming in Head-Mounted Virtual Reality," IEEE Transactions on Multimedia, **22**(3), 744–759, 2020, doi:10.1109/TMM.2019.2931807.
- [15] C. Wu, R. Zhang, Z. Wang, L. Sun, "A Spherical Convolution Approach for Learning Long Term Viewport Prediction in 360 Immersive Video," Proceedings of the AAAI Conference on Artificial Intelligence, **34**(01), 14003–14040, 2020, doi:10.1609/aaai.v34i01.7377.
- [16] D. Nguyen, "An evaluation of viewport estimation methods in 360-degree video streaming," in 2022 7th International Conference on Business and Industrial Research (ICBIR), 161–166, IEEE, 2022, doi:10.1109/ICBIR54589.2022.9786513.
- [17] D. V. Nguyen, et al, "An Optimal Tile-Based Approach for Viewport-Adaptive 360-Degree Video Streaming," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, **9**(1), 29–42, 2019, doi:10.1109/JETCAS.2019.2899488.
- [18] D. Nguyen, et al, "Scalable multicast for live 360-degree video streaming over mobile networks," IEEE Access, **10**, 38802–38812, 2022, doi:10.1109/ACCESS.2022.3165657.

- [19] N. V. Hung, P. H. Thinh, N. H. Thanh, T. T. Lam, T. T. Hien, V. T. Ninh, T. T. Huong, "LVSUM-Optimized Live 360 Degree Video Streaming in Unicast and Multicast Over Mobile Networks," in 2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN), 29–34, IEEE, 2023, doi:[10.1109/CICN59264.2023.10402136](https://doi.org/10.1109/CICN59264.2023.10402136).
- [20] N. V. Hung, et al, "Flexible HTTP-based Video Adaptive Streaming for good QoE during sudden bandwidth drops," EAI Endorsed Transactions on Industrial Networks and Intelligent Systems, **10**(2), e3–e3, 2023, doi:[10.4108/eetinis.v10i2.2994](https://doi.org/10.4108/eetinis.v10i2.2994).
- [21] N. Kan, et al, "RAP360: Reinforcement Learning-Based Rate Adaptation for 360-Degree Video Streaming With Adaptive Prediction and Tiling," IEEE Transactions on Circuits and Systems for Video Technology, **32**(3), 1607–1623, 2022, doi:[10.1109/TCSVT.2021.3076585](https://doi.org/10.1109/TCSVT.2021.3076585).
- [22] J. Vielhaben, et al, "Viewport Forecasting in 360° Virtual Reality Videos with Machine Learning," in 2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), 74–747, 2019, doi:[10.1109/AIVR46125.2019.00020](https://doi.org/10.1109/AIVR46125.2019.00020).
- [23] F. Qian, et al, "Optimizing 360 Video Delivery over Cellular Networks," in Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges, ATC '16, 1–6, Association for Computing Machinery, New York, NY, USA, 2016, doi:[10.1145/2980055.2980056](https://doi.org/10.1145/2980055.2980056).
- [24] D. V. Nguyen, et al, "An Evaluation of Tile Selection Methods for Viewport-Adaptive Streaming of 360-Degree Video," ACM Trans. Multimedia Comput. Commun. Appl., **16**(1), 2020, doi:[10.1145/3373359](https://doi.org/10.1145/3373359).
- [25] V. H. Nguyen, et al, "Retina-based quality assessment of tile-coded 360-degree videos," EAI Endorsed Transactions on Industrial Networks and Intelligent Systems, **9**(32), 2022.
- [26] T. C. Thang, et al, "An Evaluation of Bitrate Adaptation Methods for HTTP Live Streaming," IEEE Journal on Selected Areas in Communications, **32**(4), 693–705, 2014, doi:[10.1109/JSAC.2014.140403](https://doi.org/10.1109/JSAC.2014.140403).
- [27] V. H. Nguyen, D. T. Bui, T. L. Tran, C. T. Truong, T. H. Truong, "Scalable and resilient 360-degree-video adaptive streaming over HTTP/2 against sudden network drops," Computer Communications, **216**, 1–15, 2024, doi:[10.1016/j.comcom.2024.01.001](https://doi.org/10.1016/j.comcom.2024.01.001).
- [28] T. T. Huong, et al, "Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach," Computers in Industry, **132**, 103509, 2021, doi:[10.1016/j.compind.2021.103509](https://doi.org/10.1016/j.compind.2021.103509).
- [29] E. J. David, et al, "A dataset of head and eye movements for 360 videos," in Proceedings of the 9th ACM Multimedia Systems Conference, 432–437, 2018, doi:[10.1145/3204949.3208139](https://doi.org/10.1145/3204949.3208139).

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

# Leveraging Machine Learning for a Comprehensive Assessment of PFAS Nephrotoxicity

Anirudh Mazumder, Kapil Panda\*

University of North Texas, Texas Academy of Mathematics and Science, Denton, 76203, United States of America

## ARTICLE INFO

### Article history:

Received: 04 March, 2024

Revised: 18 May, 2024

Accepted: 19 May, 2024

Online: 12 June, 2024

### Keywords:

Machine Learning

Kidneys

Polyfluoro-Alkyl Substances

Toxicokinetics

## ABSTRACT

Polyfluoroalkyl substances (PFAS) are persistent chemicals that accumulate in the body and environment. Although recent studies have indicated that PFAS may disrupt kidney function, the underlying mechanisms and overall effects on the organ remain unclear. Therefore, this study aims to elucidate the impact of PFAS on kidney health using machine learning techniques. Utilizing a dataset containing PFAS chemical features and kidney parameters, dimensionality reduction and clustering were performed to identify patterns. Machine learning models, including XGBoost classifier, regressor, and Random Forest regressor, were then developed to predict kidney type from PFAS descriptors, estimate PFAS accumulation in the body, and predict the ratio of glomerular surface area to proximal tubule volume, which indicates kidney filtration efficiency. The kidney type classifier achieved 100% accuracy, confirming that PFAS exposure alters kidney morphology. The PFAS accumulation model attained an  $R^2$  of 1.00, providing a tool to identify at-risk individuals. The ratio prediction model reached an  $R^2$  of 1.00, offering insights into PFAS effects on kidney function. Furthermore, PFAS descriptors and anatomical variables were identified through analyses using feature importance, demonstrating discernible links between PFAS and kidney health, offering further biological significance. Overall, this study can significantly contribute to the current findings on the effect of PFAS while offering machine learning as a contributive tool for similar studies.

## 1. Introduction

Polyfluoro-alkyl substances (PFAS) have garnered significant attention in recent years due to their presence in a wide range of consumer and industrial products and their existence in the environment [1]. However, the characteristics that make PFAS so prevalent in our society also underscore their challenges to human and environmental health [2].

PFAS are a group of synthetic organic compounds characterized by their perfluoroalkyl chains, which consist of carbon atoms fully saturated with fluorine atoms [3]. This unique chemical structure results in one of the most robust and most stable bonds in organic chemistry, the carbon-fluorine (C-F) bond, which is responsible for PFAS's exceptional resistance to heat, chemical degradation, and biological breakdown processes, earning it the name of the "forever chemical" [4]. The fluorine atoms in PFAS molecules form a protective shield around the carbon backbone, rendering these compounds highly hydrophobic, which contributes to their utility in various industrial applications, such as the production of non-stick coatings and water-resistant textiles [5]. However, this is also the reason behind their persistence in the environment and their ability

to bioaccumulate in organisms [6].

PFAS's hydrophobic properties make it highly insoluble in water, which prevents the chemical from dissolving into aqueous environments, allowing it to persist in the soil, water, and sediment for extended periods [7]. Furthermore, this hydrophobicity disrupts normal metabolic pathways, as it partitions into fatty tissues rather than remaining in aqueous solutions [8]. This phenomenon leads to bioaccumulation in organisms, as PFAS are absorbed through ingestion or absorption and accumulate in fatty tissues over time, resulting in elevated concentrations within organisms throughout the food chain [9].

Over the past few years, various research has revealed the growing dangers associated with PFAS on the body, encompassing concerns such as cancer, thyroid disorders, developmental anomalies in children, and immune system dysfunction [10, 11]. It is well known that the carcinogenic potential of PFAS has been associated with various cancers due to the PFAS-induced oxidative stress that plays a role in cellular damage and DNA mutations, contributing to cancer development [12, 13]. Furthermore, studies have shown that by interfering with hormonal regulation systems and suppressing immune system function, PFAS can lead to disorders like hypothyroidism,

\*Corresponding Author: Kapil Panda, 1155 Union Cir, kapilpanda@my.unt.edu

developmental anomalies, such as stunted growth and delayed cognitive development in children, and neurotoxic effects [14, 15].

However, recent studies have uncovered the notion that PFAS, in fact, also has the potential to infiltrate further and disrupt fundamental physiological processes, namely the kidneys [16]. The kidneys, often called the body's natural filtration and waste management system, play a pivotal role in maintaining homeostasis. Their intricate network of nephrons and tubules ensures the efficient removal of waste products, excess fluids, and electrolytes from the bloodstream [17]. Consequently, the exploration of the relationship between PFAS exposure and kidney function has assumed a position of paramount importance in the realm of health research [18]. Extensive research in recent years has shed light on the profound impact of PFAS on kidney health. When introduced into the body, PFAS compounds can infiltrate renal tissues, interacting with various cellular components and initiating molecular responses [19]. These interactions can lead to structural changes in the kidneys, potentially altering the distribution of kidney types, which has been identified in previous studies. Such changes in kidney morphology have significant implications for kidney function and overall health [20]. Moreover, studies have indicated that PFAS exposure can disrupt the finely tuned balance of hormonal regulation systems, potentially leading to disorders such as hypothyroidism, which can further affect kidney health [21].

Therefore, this research aims to gain further insight into the relationship between PFAS and kidney function [22]. Utilizing machine learning techniques, we aim to unravel the complexities of this relationship and shed more light on how PFAS accumulation may impact the intricate structure and function of the kidneys [23]. Utilizing a dataset containing PFAS chemical features and kidney parameters, exploratory data analysis and dimensionality reduction were performed using PCA to identify patterns and correlations within the data. To ensure and verify the correlation between PFAS and kidneys, an XGBoost Classifier was used to predict kidney type from PFAS descriptors. Next, an XGBoost Regressor was used to estimate PFAS accumulation in the organ, assessing the impact of PFAS on the kidneys. Finally, a Random Forest Regressor was developed to determine the ratio of Glomerular Total Surface Area to Proximal Tubule Volume to offer insights into kidney function. The models were trained on 70% of the dataset and evaluated using metrics such as R-squared, confusion matrices, Mean Absolute Error, and residual analysis. Hyperparameter tuning through methods like Grid Search and Cross-Validation was also conducted.

## 2. Materials and Methods

### 2.1. Dataset

A dataset found at [24] was used. The dataset is critical to providing a few key pieces of information in depth. The first key piece of information that it provides is the chemical features of PFAS. For example, the dataset has information about the PFAS inside of the body's lipophilicity, the vapor pressure of PFAS, the water solubility of PFAS, and more critical descriptors of PFAS chemically. Additionally, the dataset contains information about the actual animal being looked at; for example, it has information about the species type and the gender of the animal. Furthermore, some critical physi-

ological features of the animal were kept within the dataset, such as information about the animal's body mass, while also keeping physiological characteristics of the kidney inside of the body, which has vital information about things like the diameter of the proximal tubules. Utilizing this entire feature set, we conducted some methods to see if the data could be used to create important machine learning algorithms proving correlations between different parts of PFAS and kidney functions.

### 2.2. Data Analysis

Before the machine learning algorithms were created, exploratory data analysis was undertaken to see the key patterns and statistical backing behind the data before any machine learning was run, ensuring that the algorithms depend on there being learnable patterns. Additionally, conducting the analysis would allow for essential pieces of data that would mess with the accuracy of the machine learning models in the future, like outliers or potential data errors that may be found.

The following data analysis technique was a dimensionality reduction utilizing a Principal Component Analysis (PCA). PCA is perfect for finding the key correlations and patterns in the data, which directly leads to variation. Additionally, Figure 1 shows how the higher dimensionality is reduced to lower dimensionality spaces, making it easier to visualize data patterns while also allowing us to understand the explained variance ratio for the principal components. This allows us to see the primary source of variation in the data and how the first few components are vital in depicting the first few data points. Additionally, we can find clusters and patterns in the data, allowing us to find relationships that are not very noticeable or subgroups of data.

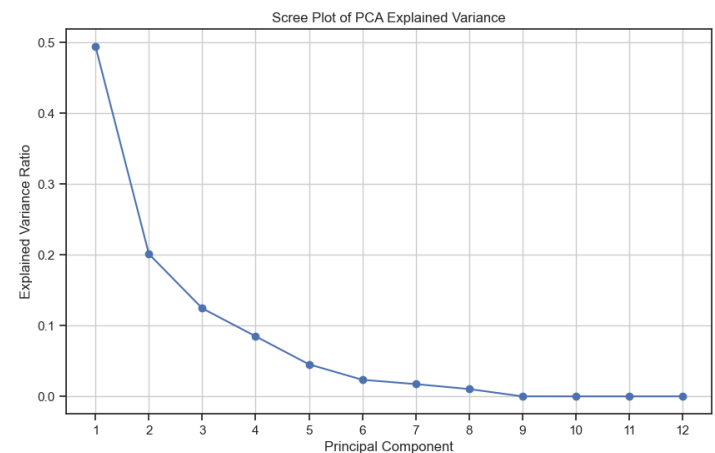


Figure 1: Principal Component Analysis and Explained Variance Ratio Showing a key change at The Principal Component 2

Furthermore, using methods like clustering on top of PCAs, we can better understand the patterns inside of the dataset [25]. This is a crucial method because clustering while using PCA allows us to find the natural groups of the patterns within the data, as PCA simplifies the data while keeping the essential characteristics, so the clusters of this allow us to understand the groupings of these characteristics allowing us to find these specific groups within the

data [26]. Additionally, K-Means can cluster the points based on their similarity, allowing us to find the similarity between the different characteristics delineated as essential due to the PCA algorithm. As seen in Figure 2, there seems to be an essential characteristic within the dataset that can show how some parts of the principal component analysis are very close. However, there are sections in the middle where multiple clusters are near one another.

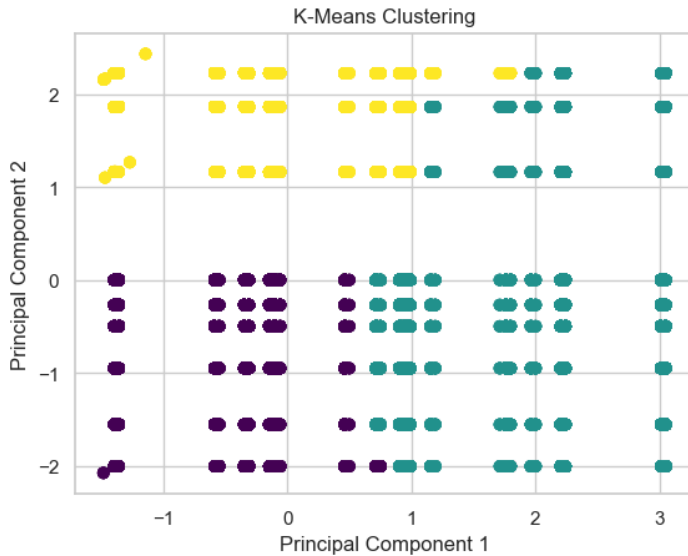


Figure 2: Clustering Analysis Using PCA and K-Means Algorithm to Identify Natural Data Patterns and Similarity Showing the Clusters of the First and Second Principal Components

### 2.3. Kidney Type Prediction

Utilizing our data, we attempted first to establish and confirm a correlation between PFAS and kidneys, which we would later conduct the rest of our analysis on. This correlation can describe if there is a relationship between the information about the PFAS and precisely how much it affects the kidney while providing the actual characteristics of the organ. To do this, we created a classifier that would classify what type of kidney was being affected by the PFAS inside of the species, which could allow for early measures and precautions to be taken based on the predicted effects of PFAS on the organ.

#### 2.3.1. Feature Selection

The data was split up into all of the pieces of data which actively described any property or action of the PFAS chemical was used as a X variable, while just the kidney type was left as the Y variable.

#### 2.3.2. Model Selection & Hyperparameter Tuning

The machine learning algorithm utilized an XGBoost Classifier (XGBC) to classify the PFAS data into the two different types of kidneys in the dataset. XGBC is critical to creating multiple weaker decision trees, which are then compiled up and added into a singular, more robust predictor while correcting previous tree errors by selectively creating new trees, allowing the algorithm to make

the predictions more accurate [27]. As a result, the ensemble-based algorithm is critical to creating a robust classifier, as it uses the concept that if each tree has a vital understanding of the data, then when they all are combined, the final decision that the model is going to be outputting will understand all of the dimensionalities of the dataset. Additionally, the algorithm has inbuilt regularization techniques, which is critical because the dataset that we were working with had a ton of data points, meaning that the algorithm was very prone to just learning the pattern in training data without having a resemblance to the output data because it would be overfitting.

Furthermore, XGBC allows us to look into the specific features with the highest correlations, especially within the context of the machine learning algorithm's trained understanding of the patterns within the data [28]. Utilizing the XGBC feature importances, we can see which columns are the most important for the training of the algorithm and which columns may potentially be negatively affecting the understanding of the data, which would allow us to go back to the feature selection step and ensure that they are removed from the training dataset. Additionally, we can see which features have the highest correlation with the target output that we are looking for using this feature selection to understand better the patterns within the data, which otherwise would not be understandable.

Then, after the model had been trained and the feature selector had been used to see which features were key, hyperparameter tuning was used to create the most optimal machine learning model, especially in the context of XGBC where there are multiple parts to the actual algorithm including important pieces like loss functions which could change. Thus, essential concepts like Grid Search and Cross-Validation were used to conduct hyperparameter tuning so the classifier could achieve the best accuracy possible.

#### 2.3.3. Model Training & Evaluation

The model was trained using a majority of the data, but a train test split of 70% - 30% was used to evaluate the model and prevent overfitting of the data. Then, to understand the relationship between the different pieces of data, a few metrics were used, including a confusion matrix to understand how the model was doing when predicting the testing data, and an accuracy score was used to understand the difference between the predicted and the actual outputs.

### 2.4. PFAS Accumulation Model

After establishing if there is a correlation between the kidney type and the information about PFAS, it is vital to see the amount of PFAS that is going to accumulate in the kidney based on the PFAS descriptors. We must see this information because it is critical to assessing the impact of PFAS on the kidney. Additionally, it is essential to understand if there is a correlation between the chemical descriptors of PFAS and the actual effect of PFAS when inside the human body. As a result, a machine learning model was created to make these assessments and understand if we can predict the accumulation of PFAS inside the human body.

#### 2.4.1. Feature Selection

A fundamental difference between this algorithm and the kidney-type prediction algorithm is that this one needs to output a numerical

rather than a categorical response. This delineation is critical to understanding how the data will be used, as many features exist in the datasets. However, some of them are categorical, so the data must be encoded into numerical data. After the data was imputed, the data was split into all the PFAS descriptors: the X data. At the same time, the singular Y data was the amount of PFAS accumulating inside the body.

#### 2.4.2. Model Selection & Hyperparameter Tuning

Similarly to the Kidney Type Prediction model, an XGBoost Regression algorithm was used to predict the amount of PFAS that would accumulate in the body. XGBoost regression was used for this algorithm due to its innate ability to understand nonlinear data through its use of gradient-boosted decision trees [29]. Using these trees, the algorithm can perform an ensemble method, reducing overfitting inside the algorithm. Additionally, since it is a gradient-boosting algorithm, it can optimize its performance through an iterative process where it continuously improves itself [30]. Also, it can understand larger datasets with many features, delineate which ones are the most essential parts for it to learn, and then create the most optimal outputs after it has achieved the response it is attempting to get [31].

#### 2.4.3. Model Training & Evaluation

The model was trained on a 70%-30% train test split. Using the train test split, the model could be evaluated on its ability to truly learn the training data and still apply it to the testing data without just outputting it based on the answers it was already given. On this, an  $R^2$  and MAE were calculated to output the model's accuracy to see how closely the model truly understood and predicted the data.

### 2.5. Glomerular Total Surface Area vs Proximal Tubule Predictor

Following the prediction of PFAS accumulation in kidneys, we aimed to discern the effects of PFAS on the kidneys' function. To do this, we sought to estimate the ratio of Glomerular Total Surface Area (GlomTotSA) to the Volume of the Proximal Tubule (ProxTubTotVol) within the kidneys, which provides insight into the structural dynamics of the organ. A higher ratio may indicate efficient filtration and reabsorption processes, suggesting healthier kidney function. In comparison, a lower ratio might suggest potential kidney morphology and function alterations, providing early indicators of kidney health issues.

#### 2.5.1. Feature Selection

The feature selection process involved carefully examining the relevance and significance of each feature in the dataset. Using Recursive Feature Elimination (RFE) and correlation analysis, we identified the most informative attributes for our algorithm. Furthermore, domain expertise was crucial in selecting pertinent features, ensuring that the model captured the essence of PFAS accumulation within the kidneys.

Feature engineering, however, was not confined to feature selection alone. It extended to creating engineered features, such

as interaction terms between PFAS accumulation descriptors and anatomical variables, enabling the model to capture nuanced relationships within the data. These engineered features served as the basis for the algorithm to make precise predictions regarding the GlomTotSA/ProxTubTotVol ratio.

#### 2.5.2. Model Selection & Hyperparameter Tuning

The choice of a Random Forest Regressor was deliberate, given its aptitude for handling complex datasets and capturing linear and nonlinear relationships within the data. Our dataset encompassed a multitude of variables, including PFAS exposure descriptors, anatomical features, and the target variable of GlomTotSA/ProxTubTotVol ratio. These variables interact in intricate ways that may need to be more linear and straightforward. The ensemble nature of Random Forest, comprising multiple decision trees, allows the algorithm to capture these complex relationships effectively. Each decision tree contributes its unique perspective, and the ensemble aggregates these insights to yield robust and accurate predictions. This robustness is particularly advantageous when dealing with biological data, which often exhibits intricate and nonlinear interactions. The ensemble nature of the Random Forest algorithm, comprising multiple decision trees, enhances its robustness and predictive power [32]. Another compelling aspect of the Random Forest Regressor is its innate ability to provide insights into feature importance [33]. Understanding which features are most influential in predicting the GlomTotSA/ProxTubTotVol ratio is essential for gaining mechanistic insights into the relationship between PFAS exposure and kidney morphology [34]. Random Forest calculates feature importance scores, enabling us to identify which PFAS descriptors and anatomical variables are pivotal in predicting the target variable [35].

On the other hand, hyperparameter tuning, a pivotal aspect of model development, involved an exhaustive search for optimal parameters. We employed techniques such as grid search and cross-validation to find the parameter configuration that yielded the best predictive accuracy and generalization performance.

#### 2.5.3. Model Training & Evaluation

The Random Forest Regressor was trained on the training data, utilizing a 70/30 train test split, allowing the selected features and engineered descriptors to learn the intricate relationships between PFAS accumulation and the GlomTotSA/ProxTubTotVol ratio.

To gauge the model's performance, we employed several evaluation metrics, including Mean Absolute Error (MAE) and  $R^2$ , which quantified the model's predictive accuracy. Additionally, we assessed the model's ability to generalize to unseen data through k-fold cross-validation, ensuring robustness and mitigating overfitting concerns.

### 3. Results

#### 3.1. Accuracy

##### 3.1.1. Kidney Type Prediction

To assess the performance of the model, the accuracy of the model had to be calculated by comparing the model’s prediction of the training dataset against the actual values of the training dataset. Using this metric to calculate the model’s accuracy, the model achieved an accuracy of 100%. The accuracy of this model can be depicted through the confusion matrix as seen in Figure 3. The figure shows that the predictions give the same output as is expected from the actual data, allowing for the machine learning model to be associated with perfect accuracy. Additionally, other performance metrics were used to measure the machine learning algorithm’s performance, which can be seen in Table 1. These metrics can depict the fact that the accuracy can perfectly classify what type of kidney is affected by PFAS and the spread of PFAS throughout the body.

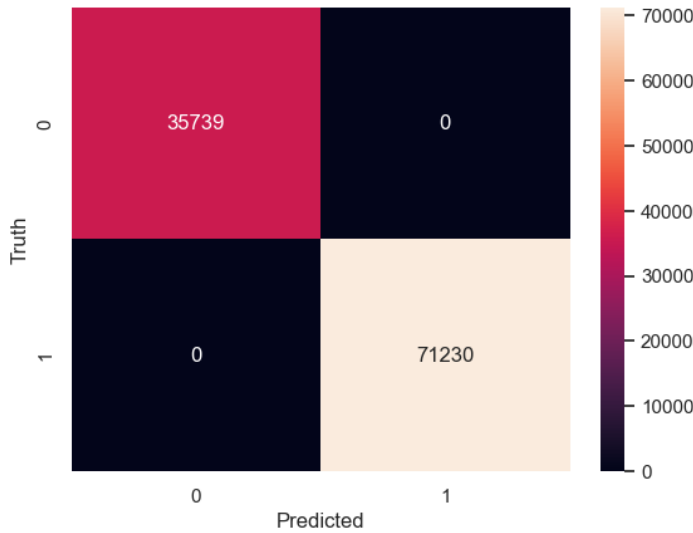


Figure 3: Confusion Matrix Showing the Machine Learning Model Predicting with Perfect Accuracy

Table 1: Machine Learning Model Performance Metrics

Labels	Precision	Recall	F1 Score	Support
Multireculated	1.00	1.00	1.00	35739
Unipapillary	1.00	1.00	1.00	71230
Accuracy			1.00	106969
Macro Average	1.00	1.00	1.00	106969
Weighted Average	1.00	1.00	1.00	106969

Cross-validation scores were also calculated to ensure that the accuracy of the model was generalizable to data that wasn’t specific to training the machine learning model itself. When done on a 5-fold cross validation model, all five scores contained 100% accuracy and and the standard deviation from these scores was 0%.

#### 3.1.2. PFAS Accumulation Model

This machine learning model’s accuracy was calculated using a few different methods since the model is a regression algorithm rather than a classifier. The first method used to calculate the accuracy of the model was to statistically evaluate the created regression line, which is done utilizing a  $R^2$  value and the Mean Absolute Error(MAE). The  $R^2$  value was 1.00, and the MAE was 0.00. These values indicate that the model was able to model the data perfectly. These values can be seen in Figure 4, where you can see how the graph depicts the alignment between the predicted and actual predictions. Additionally, these values were further justified through a cross-validation algorithm, which can prove the robustness of the model by verifying that the data split is not done by chance; it shows if the machine learning model is genuinely learning the data. The cross-validation statistics also validated that the machine learning model perfectly predicted the accumulation of PFAS, as it gave the same 1.00  $R^2$  value and 0.00 MAE. Additionally, these values can be further depicted in Figure 5, which shows the distribution of the residuals, which are calculated by subtracting the actual value from the predicted value, and the figure shows how the values are very close to 0, indicating that the model is predicting to almost perfection.

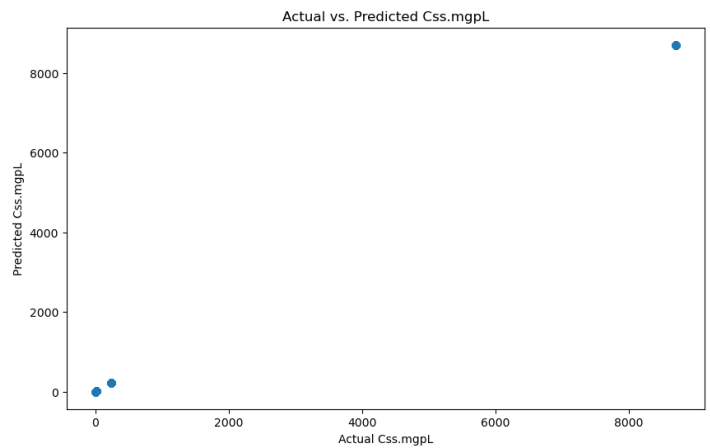


Figure 4: Comparison of the Predicted and Actual PFAS Accumulation Values

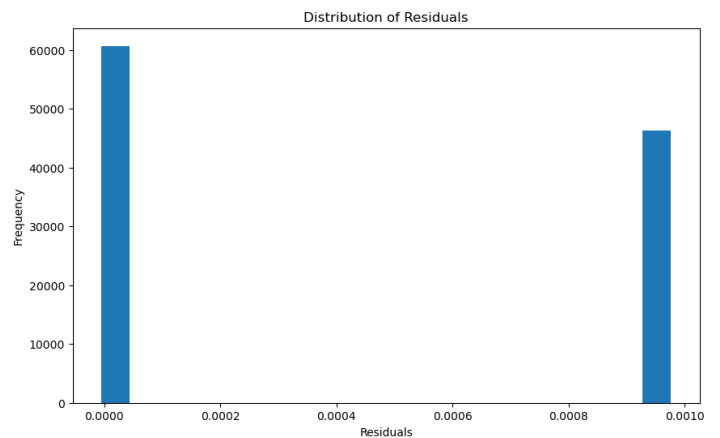


Figure 5: Residual Distribution Plot based on the Real PFAS Accumulation Values and the Predicted PFAS Accumulation Values



The cross validation results that were achieved for this model was a 0.00 MAE and a 1.00  $R^2$  on a 5-fold cross validation model.

### 3.1.3. Glomerular Total Surface Area vs Proximal Tubule Predictor

Similarly to the PFAS Accumulation Model, the accuracy of the machine learning model was calculated by finding the statistical values, which were verified through cross-validation, and the residuals were calculated. The initial statistical values calculated for the machine learning model were a 0.00 MAE and a 1.00  $R^2$  statistic. Furthermore, using cross-validation, the model still achieved an accuracy of 1.00. These values can be depicted through Figure 6, where it can be seen how the model is predicting the values with an incredibly high accuracy. Additionally, in Figure 7, you can see the plot of the distribution of the residuals, which shows an apparent discrepancy that spans a lot of data points. However, most of the data points are relatively close to the actual value.

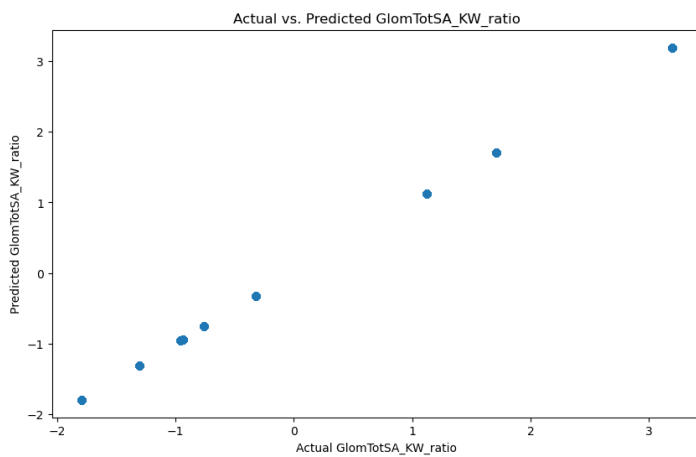


Figure 6: Comparisons of the Predicted and Actual Glomerular Total Surface Area and Proximal Tubule Ratio Values

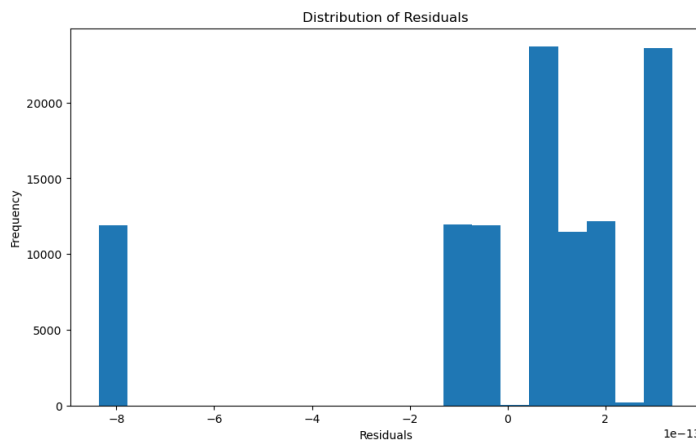


Figure 7: Residual Distribution Plot based on the Real Glomerular Total Surface Area and Proximal Tubule Ratio Values and the Predicted Glomerular Total Surface Area and Proximal Tubule Ratio Values

On a 5-fold cross validation the model is achieves a  $R^2$  value

of 1.00 and a MAE value of 0.00, showing the model is not just bound to the training data that was fed into it and it has based its information off of.

## 3.2. Feature Importance

After understanding the performance of the models, it is also essential to understand the key features indicating these specific outputs. Additionally, due to the specific models chosen as discussed in the methodology, it can be seen what features were critical to the specific models training rather than just looking at correlations within the dataset.

### 3.2.1. Kidney Type Prediction

The feature importances for the first model can be seen in Figure 8. The first model had some features with very high indications, but most did not correlate much with the type of kidney. The key features that can be seen in Figure X are the species of the anime and the proximal tubule diameter, which both seem to be understandable indicators. They both are reasonable indicators because the species directly affects the type and function of the kidney, which is critical to understanding which type it will be. Additionally, the proximal tubule diameter is a physical characteristic of kidneys, which indicates the type of kidney because it is a very clear descriptor of the kidney being affected.

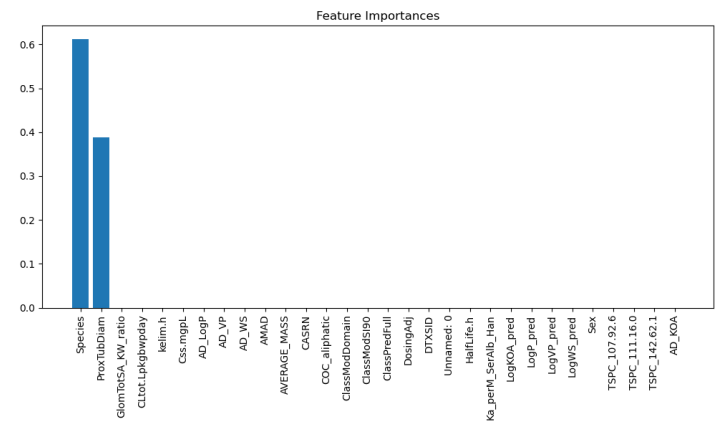


Figure 8: Feature Importance Plot for the Kidney Type Prediction Model

### 3.2.2. PFAS Accumulation Model

For this machine learning model, one feature that the model indicates has a direct correlation with the output of the model. The feature that the model is showing is the ClassPredFull, which is a metric inside of the dataset that tells us the level of exposure the body is to PFAS, which indicates that using this level, the machine learning algorithm is learning how to accurately predict how much PFAS is going to accumulate inside of the body and damage the kidney. This can be seen in Figure 9

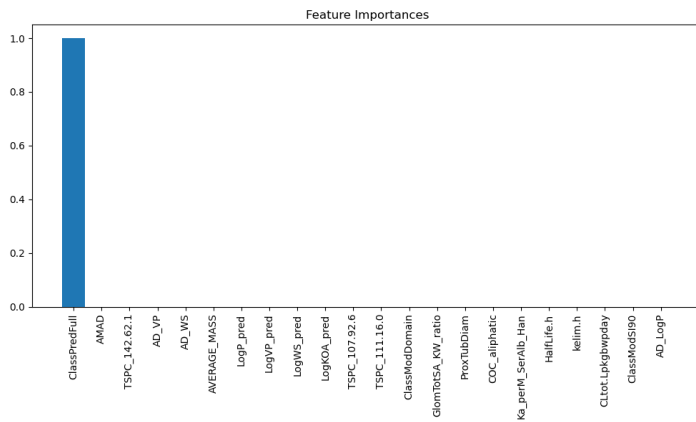


Figure 9: Feature Importance Plot for the PFAS Accumulation Model

### 3.2.3. Glomerular Total Surface Area vs Proximal Tubule Predictor

As seen in Figure 10, the third machine learning model indicates that two key features allow it to achieve a high accuracy. The two features that can indicate this are GlomTotSA\_KW\_ratio, which shows the ratio between the glomerular total surface area to the kidney weight, and the other feature, which can show a correlation to the output of this model is the diameter of the proximal tubule. Both of these features are understandable because they are descriptors of the physiological characteristics of the kidney or the specific proximal tubule, allowing for them to be correlated to the measurements that the machine learning model is concerned with predicting.

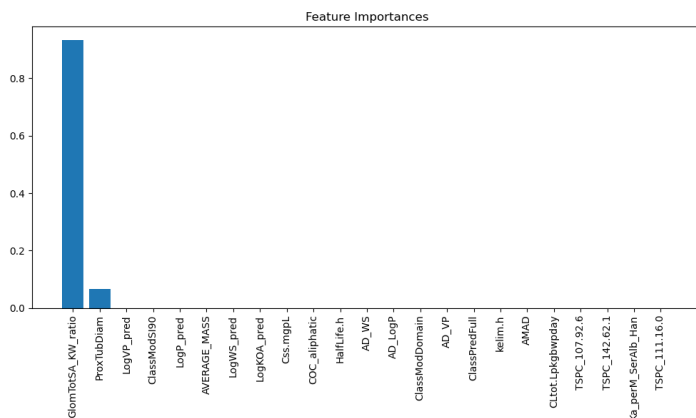


Figure 10: Feature Importance Plot for the Glomerular Total Surface Area vs Proximal Tubule Model

## 4. Biological Significance

As seen by the accuracy of the Kidney Type Prediction Model, we can safely verify previous studies and conclude that there is indeed a correlation between PFAS descriptors and kidneys to base the rest of our analysis. The ability of PFAS descriptors to predict kidney type with perfect accuracy suggests that PFAS compounds are not passive bystanders in the body but actively influence kidney morphology. As mentioned, PFAS, once absorbed into the bloodstream,

can infiltrate renal tissues, where they interact with cellular components and trigger molecular responses [36]. This interaction may lead to structural changes in the kidneys, potentially altering the distribution of kidney types [37]. The fact that PFAS descriptors alone can reliably distinguish between different kidney types underscores the pronounced and biologically meaningful impact of PFAS exposure on renal structures.

Furthermore, the model provides significance in early diagnoses and organ assessment by predicting the amount of PFAS accumulation in the kidneys utilizing an XGBoost Regressor. Understanding how PFAS chemicals accumulate in vital organs like the kidneys is crucial for assessing the long-term health impacts of PFAS exposure. By employing XGBoost, researchers gain a powerful tool for modeling and predicting these accumulations, allowing for more accurate risk assessments and informed policy decisions. This predictive capability can guide regulatory bodies and healthcare providers in designing effective strategies to mitigate PFAS exposure and protect public health. By forecasting PFAS accumulation, we move beyond reacting to environmental contamination and instead proactively manage and mitigate its impacts.

The estimation of the ratio of Glomerular Total Surface Area (GlomTotSA) to the Volume of the Proximal Tubule (ProxTubTotVol) also offers significant scientific insight into the health and performance of our kidneys. The GlomTotSA-to-ProxTubTotVol ratio holds the key to understanding the structural efficiency of the kidneys. The glomerular surface area represents the vast expanse of tiny filtering units within the kidney, where blood is meticulously sieved to remove waste and excess substances [38]. The volume of the proximal tubule reflects the space available for the reabsorption of essential substances into the bloodstream, a crucial process in maintaining bodily homeostasis [39]. When this ratio is higher, it suggests that the kidneys are adept at filtration and reabsorption, indicative of a well-functioning organ [40]. This signifies that the kidneys efficiently clear waste and retain vital compounds, reflecting good renal health. Conversely, a lower GlomTotSA-to-ProxTubTotVol ratio can signify potential kidney morphology and function issues. This may point to structural alterations within the kidneys that affect their ability to filter and reabsorb substances optimally [41]. On a chemical level, this ratio can also be linked to the renal clearance of various substances, including drugs and toxins [42]. Understanding these chemical processes is pivotal for predicting how the kidneys will handle different compounds, informing medication dosages and toxicity assessments.

This ratio further provides insight into the complex interplay of filtration, reabsorption, and structural integrity within the kidneys [43]. It allows researchers to investigate the effects of PFAS exposure on these processes, potentially uncovering early indicators of kidney health issues. Moreover, it has applications beyond PFAS research, as it can be used as a valuable biomarker for assessing kidney function and diagnosing renal diseases. The ability to estimate this ratio with precision, informed by predictive PFAS accumulation models, represents a powerful tool for advancing our understanding of kidney health and clinical interventions and treatments.

However, the GlomTotSA-to-ProxTubTotVol ratio also offers a fruitful perspective on the intricate relationship between kidney structure and its functional role, particularly in the context of estimating glomerular filtration rate (eGFR) [44]. The eGFR is a critical

indicator of kidney function, primarily based on factors like creatinine levels, age, sex, and race, ultimately reflecting how efficiently the kidneys filter waste products from the bloodstream [45]. The GlomTotSA-to-ProxTubTotVol ratio and eGFR (estimated glomerular filtration rate) in kidneys are interconnected in a complex and critical manner, biologically and chemically. This relationship is pivotal for understanding renal function, particularly how well the kidneys filter waste and maintain overall homeostasis in the body. The GlomTotSA-to-ProxTubTotVol ratio is essentially a representation of the glomerular surface area relative to the proximal tubular volume, and eGFR is an estimate of the rate at which the glomeruli in the kidneys filter blood [46].

Biologically, this relationship hinges on the intricate anatomy and physiology of the renal system. The glomeruli, small tuft-like structures within the kidney, are the primary filtration units. They filter blood and allow water and solutes to enter the tubular system while retaining larger molecules like proteins. The proximal tubules, on the other hand, are involved in reabsorbing valuable substances such as glucose and electrolytes [42]. The ratio of the glomerular surface area to the proximal tubular volume reflects the balance between filtration and reabsorption in the kidney. A high GlomTotSA-to-ProxTubTotVol ratio suggests efficient filtration relative to reabsorption, typically associated with better renal function [47]. When the kidneys function optimally, the eGFR is higher because the glomeruli are efficiently filtering a greater blood volume, and waste products are being excreted. Conversely, if the ratio is skewed or compromised due to kidney damage or disease, the eGFR decreases, indicating impaired filtration capacity and potential renal dysfunction [48].

Chemically, the interaction is also governed by the intricate molecular exchange processes within the kidney. The glomeruli filter blood by using a combination of pressure-driven physical forces and the selective permeability of their basement membranes and podocyte cells [49]. These structures allow small molecules and water to pass into the renal tubules while preventing larger molecules like proteins from crossing over. The proximal tubules then selectively reabsorb essential molecules and regulate the concentration of electrolytes and waste products in the urine [50]. The ratio signifies the efficiency of these filtration and reabsorption processes. When the balance between filtration and reabsorption is disrupted, such as in kidney damage, inflammation, or glomerular dysfunction, the GlomTotSA-to-ProxTubTotVol ratio can change [51]. This imbalance decreases eGFR, as less blood is effectively filtered and more waste products may accumulate in the bloodstream [52]. In this chemical interplay, disruptions in the GlomTotSA-to-ProxTubTotVol ratio can serve as a valuable indicator of kidney function and provide insights into renal health or dysfunction. This ratio may also signify how effectively the kidneys handle the excretion of various substances. Efficient filtration and reabsorption are vital for maintaining electrolyte balance and eliminating waste, pharmaceuticals, and toxins from the body [53]. Therefore, if the ratio tilts towards a more significant glomerular surface area relative to the proximal tubule volume, it may suggest that the kidneys can process substances more effectively, potentially contributing to higher eGFR values.

## 5. Discussion

### 5.1. Limitations

The models were based off of a single dataset, so a dataset with more diversity and different populations is necessary to truly assess the causality and relationships that the models are indicating are true from the data provided. Additionally, investigations into the biological mechanisms which are highlighted through the research study and experimental validation are necessary for understanding the true biological processes and pathways that are associated with PFAS and its effect on the body. Also, there are some confounding variables which could have potential effects on the results of the finding, such as environmental exposures which have the ability to effects the observed relationships.

### 5.2. Conclusion

This study demonstrates machine learning's potential to elucidate the intricate impacts of PFAS on kidney health. The models presented confirm discernible links between PFAS and renal function. The kidney type classifier verifies that PFAS alters morphology. The PFAS accumulation regressor enables clinical monitoring to protect at-risk groups. Estimating the GlomTotSA/ProxTubTotVol ratio provides significant insights into PFAS's effects on filtration and reabsorption efficiency.

The models showcase PFAS's multifaceted effects on kidney structure and function. The techniques pave the way for enhanced risk assessment, improved clinical surveillance, and targeted therapeutics. This study underscores the power of machine learning to unravel PFAS toxicity mechanisms. It makes significant contributions to the current understanding of PFAS nephrotoxicity.

The high predictive accuracy, feature importance analyses, and model interpretability reveal concrete biological impacts of PFAS on kidneys. PFAS descriptors actively influence morphology and accumulation. The ratio estimation sheds light on filtration and reabsorption dynamics. These findings confirm PFAS's pronounced effects on renal physiology.

This research demonstrates machine learning's immense potential for elucidating PFAS nephrotoxicity. The models provide actionable clinical insights and expand mechanistic knowledge of PFAS-kidney interplay. This paves the way for more informed risk analysis, earlier diagnosis of PFAS-associated kidney damage, and targeted therapies. The study sets a strong foundation for future machine-learning investigations into PFAS toxicity.

**Conflict of Interest** The authors declare no conflict of interest.

**Acknowledgment** We would like to thank the University of North Texas for providing us with the resources and support to conduct this research. The invaluable guidance and encouragement from our professors and mentors have been instrumental in shaping the direction and scope of this study. We would also like to acknowledge the National Kidney Foundation for their support and inspiration in conducting this project. Finally, we would also like to thank our families for supporting us throughout our research.

## References

- [1] J. Glüge, M. Scheringer, I. T. Cousins, J. C. DeWitt, G. Goldenman, D. Herzke, R. Lohmann, C. A. Ng, X. Trier, Z. Wang, "An overview of the uses of per- and polyfluoroalkyl substances (PFAS)," *Environmental Science: Processes & Impacts*, **22**, 2345–2373, 2020, doi:10.1039/DOEM00291G.
- [2] P. Dewapriya, L. Chadwick, S. G. Gorji, B. Schulze, S. Valsecchi, S. Samanipour, K. V. Thomas, S. L. Kaserzon, "Per- and polyfluoroalkyl substances (PFAS) in consumer products: Current knowledge and research gaps," *Journal of Hazardous Materials Letters*, **4**, 100086, 2023, doi:10.1016/j.hazl.2023.100086.
- [3] B. E. Blake, S. E. Fenton, "Early life exposure to per- and polyfluoroalkyl substances (PFAS) and latent health outcomes: A review including the placenta as a target tissue and possible driver of peri- and postnatal effects," *Toxicology*, **443**, 152565, 2020, doi:10.1016/j.tox.2020.152565.
- [4] D. Renfrew, T. W. Pearson, "The Social Life of the "Forever Chemical"," *Environment and Society*, **12**, 146–163, 2021, doi:10.3167/ares.2021.120109.
- [5] M. Kotthoff, J. Müller, H. Jüriling, M. Schlummer, D. Fiedler, "Perfluoroalkyl and polyfluoroalkyl substances in consumer products," *Environmental Science and Pollution Research*, **22**, 14546–14559, 2015, doi:10.1007/s11356-015-4202-7.
- [6] S. A. Bălan, V. C. Mathrani, D. F. Guo, A. M. Algazi, "Regulating PFAS as a Chemical Class under the California Safer Consumer Products Program," *Environmental Health Perspectives*, **129**, 2021, doi:10.1289/EHP7431.
- [7] M. N. Ehsan, M. Riza, M. N. Pervez, M. M. O. Khyum, Y. Liang, V. Naddeo, "Environmental and health impacts of PFAS: Sources, distribution and sustainable management in North Carolina (USA)," *Science of The Total Environment*, **878**, 163123, 2023, doi:10.1016/j.scitotenv.2023.163123.
- [8] J. Fabregat-Palau, M. Vidal, A. Rigol, "Examining sorption of perfluoroalkyl substances (PFAS) in biochars and other carbon-rich materials," *Chemosphere*, **302**, 134733, 2022, doi:10.1016/j.chemosphere.2022.134733.
- [9] G. Jha, V. Kankarla, E. McLennon, S. Pal, D. Sihi, B. Dari, D. Diaz, M. Nocco, "Per- and Polyfluoroalkyl Substances (PFAS) in Integrated Crop–Livestock Systems: Environmental Exposure and Human Health Risks," *International Journal of Environmental Research and Public Health*, **18**, 12550, 2021, doi:10.3390/ijerph182312550.
- [10] B. E. Blake, S. M. Pinney, E. P. Hines, S. E. Fenton, K. K. Ferguson, "Associations between longitudinal serum perfluoroalkyl substance (PFAS) levels and measures of thyroid hormone, kidney function, and body mass index in the Fernald Community Cohort," *Environmental Pollution*, **242**, 894–904, 2018, doi:10.1016/j.envpol.2018.07.042.
- [11] M. L. Ljubicic, A. Madsen, A. Juul, K. Almstrup, T. H. Johannsen, "The Application of Principal Component Analysis on Clinical and Biochemical Parameters Exemplified in Children With Congenital Adrenal Hyperplasia," *Frontiers in Endocrinology*, **12**, 2021, doi:10.3389/fendo.2021.652888.
- [12] M. Bonato, F. Corrà, M. Bellio, L. Guidolin, L. Tallandini, P. Irato, G. Santovito, "PFAS Environmental Pollution and Antioxidant Responses: An Overview of the Impact on Human Field," *International Journal of Environmental Research and Public Health*, **17**, 8020, 2020, doi:10.3390/ijerph17218020.
- [13] K. M. Fraley, H. N. Fraley, D. Arthur, E. J. Walther, "Per- and Polyfluoroalkyl Substances (PFAS): Anglers May Be Exposed to Harmful Chemicals in Their Catch," *Fisheries*, **45**, 138–144, 2020, doi:10.1002/fsh.10389.
- [14] L. Anderko, E. Pennea, "Exposures to per- and polyfluoroalkyl substances (PFAS): Potential risks to reproductive and children's health," *Current Problems in Pediatric and Adolescent Health Care*, **50**, 100760, 2020, doi:10.1016/j.cppeds.2020.100760.
- [15] S. H. Baker, A. Kinde, "The Pathway to a Green New Deal: Synthesizing Transdisciplinary Literatures and Activist Frameworks to Achieve a Just Energy Transition," *Environs: Environmental Law and Policy Journal*, **44**, 1–40.
- [16] P.-I. D. Lin, A. Cardenas, R. Hauser, D. R. Gold, K. P. Kleinman, M.-F. Hivert, A. M. Calafat, T. F. Webster, E. S. Horton, E. Oken, "Per- and polyfluoroalkyl substances and kidney function: Follow-up results from the Diabetes Prevention Program trial," *Environment International*, **148**, 106375, 2021, doi:10.1016/j.envint.2020.106375.
- [17] W. PFALLER, M. RITTINGER, *QUANTITATIVE MORPHOLOGY OF THE RAT KIDNEY*, 17–22, Elsevier, 1980, doi:10.1016/B978-0-08-025517-0.50011-4.
- [18] J. Yun, E.-C. Jang, S.-C. Kwon, Y.-S. Min, Y.-J. Lee, "The association of perfluoroalkyl substances (PFAS) exposure and kidney function in Korean adolescents using data from Korean National Environmental Health Survey (KoNEHS) cycle 4 (2018–2020): a cross-sectional study," *Annals of Occupational and Environmental Medicine*, **35**, 2023, doi:10.35371/aoem.2023.35.e5.
- [19] J. W. Stanifer, H. M. Stapleton, T. Souma, A. Wittmer, X. Zhao, L. E. Boulware, "Perfluorinated Chemicals as Emerging Environmental Threats to Kidney Health," *Clinical Journal of the American Society of Nephrology*, **13**, 1479–1492, 2018, doi:10.22215/CJN.04670418.
- [20] B. Conway, A. Badders, T. Costacou, J. Arthur, K. Innes, "Perfluoroalkyl substances and kidney function in chronic kidney disease, anemia, and diabetes," *Diabetes, Metabolic Syndrome and Obesity: Targets and Therapy*, **Volume 11**, 707–716, 2018, doi:10.2147/DMSO.S173809.
- [21] F. Coperchini, L. Croce, G. Ricci, F. Magri, M. Rotondi, M. Imbriani, L. Chiovato, "Thyroid Disrupting Effects of Old and New Generation PFAS," *Frontiers in Endocrinology*, **11**, 2021, doi:10.3389/fendo.2020.612320.
- [22] A. Raza, S. Bardhan, L. Xu, S. S. R. K. C. Yamijala, C. Lian, H. Kwon, B. M. Wong, "A Machine Learning Approach for Predicting Defluorination of Per- and Polyfluoroalkyl Substances (PFAS) for Their Efficient Treatment and Removal," *Environmental Science & Technology Letters*, **6**, 624–629, 2019, doi:10.1021/acs.estlett.9b00476.
- [23] C.-W. Huang, R. Lu, U. Iqbal, S.-H. Lin, P. A. Nguyen, H.-C. Yang, C.-F. Wang, J. Li, K.-L. Ma, Y.-C. Li, W.-S. Jian, "A richly interactive exploratory data analysis and visualization tool using electronic medical records," *BMC Medical Informatics and Decision Making*, **15**, 92, 2015, doi:10.1186/s12911-015-0218-7.
- [24] D. E. Dawson, C. Lau, P. Pradeep, R. R. Sayre, R. S. Judson, R. Tomero-Velez, J. F. Wambaugh, "A Machine Learning Model to Estimate Toxicokinetic Half-Lives of Per- and Polyfluoro-Alkyl Substances (PFAS) in Multiple Species," *Toxics*, **11**, 2023, doi:10.3390/toxics11020098.
- [25] R. Paul, A. S. M. L. Hoque, "Clustering medical data to predict the likelihood of diseases," in *2010 Fifth International Conference on Digital Information Management (ICDIM)*, 44–49, 2010, doi:10.1109/ICDIM.2010.5664638.
- [26] X. Gong, L. Liu, S. Fong, Q. Xu, T. Wen, Z. Liu, "Comparative Research of Swarm Intelligence Clustering Algorithms for Analyzing Medical Data," *IEEE Access*, **7**, 137560–137569, 2019, doi:10.1109/ACCESS.2018.2881020.
- [27] J. Ma, Z. Yu, Y. Qu, J. Xu, Y. Cao, "Application of the XGBoost Machine Learning Method in PM2.5 Prediction: A Case Study of Shanghai," *Aerosol and Air Quality Research*, **20**, 128–138, 2020, doi:10.4209/aaqr.2019.08.0408.
- [28] H. Liang, K. Jiang, T.-A. Yan, G.-H. Chen, "XGBoost: An Optimal Machine Learning Model with Just Structural Features to Discover MOF Adsorbents of Xe/Kr," *ACS Omega*, **6**, 9066–9076, 2021, doi:10.1021/acsomega.1c00100.
- [29] J. Li, X. An, Q. Li, C. Wang, H. Yu, X. Zhou, Y. ao Geng, "Application of XG-Boost algorithm in the optimization of pollutant concentration," *Atmospheric Research*, **276**, 106238, 2022, doi:10.1016/j.atmosres.2022.106238.
- [30] A. Ogunleye, Q.-G. Wang, "XGBoost Model for Chronic Kidney Disease Diagnosis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **17**(6), 2131–2140, 2020, doi:10.1109/TCBB.2019.2911071.
- [31] X. Zhang, C. Yan, C. Gao, B. A. Malin, Y. Chen, "Predicting Missing Values in Medical Data Via XGBoost Regression," *Journal of Healthcare Informatics Research*, **4**, 383–394, 2020, doi:10.1007/s41666-020-00077-1.
- [32] R. Sapir-Pichhadze, B. Kaplan, "Seeing the Forest for the Trees: Random Forest Models for Predicting Survival in Kidney Transplant Recipients," *Transplantation*, **104**, 905–906, 2020, doi:10.1097/TP.0000000000002923.

- [33] A. Subasi, E. Alickovic, J. Kevric, "Diagnosis of Chronic Kidney Disease by Using Random Forest, 589–594, 2017, doi:10.1007/978-981-10-4166-2\_89.
- [34] M. Z. Alam, M. S. Rahman, M. S. Rahman, "A Random Forest based predictor for medical data classification using feature ranking," *Informatics in Medicine Unlocked*, **15**, 100180, 2019, doi:10.1016/j.imu.2019.100180.
- [35] P. G. Polishchuk, E. N. Muratov, A. G. Artemenko, O. G. Kolumbin, N. N. Muratov, V. E. Kuz'min, "Application of Random Forest Approach to QSAR Prediction of Aquatic Toxicity," *Journal of Chemical Information and Modeling*, **49**, 2481–2488, 2009, doi:10.1021/ci900203n.
- [36] E. Gorrochategui, S. Lacorte, R. Tauler, F. L. Martin, "Perfluoroalkylated Substance Effects in *Xenopus laevis* A6 Kidney Epithelial Cells Determined by ATR-FTIR Spectroscopy and Chemometric Analysis," *Chemical Research in Toxicology*, **29**, 924–932, 2016, doi:10.1021/acs.chemrestox.6b00076.
- [37] J. Zhao, P. Hinton, J. Chen, J. Jiang, "Causal inference for the effect of environmental chemicals on chronic kidney disease," *Computational and Structural Biotechnology Journal*, **18**, 93–99, 2020, doi:10.1016/j.csbj.2019.12.001.
- [38] R. B. Jain, A. Ducatman, "Associations between the concentrations of  $\alpha$ -klotho and selected perfluoroalkyl substances in the presence of eGFR based kidney function and albuminuria: Data for US adults aged 40–79 years," *Science of The Total Environment*, **838**, 155994, 2022, doi:10.1016/j.scitotenv.2022.155994.
- [39] B. M. Brenner, J. L. Troy, T. M. Daugharty, I. F. Ueki, D. P. Nicholas, C. F. Wong, "On the Mechanism of Inhibition in Fluid Reabsorption by the Renal Proximal Tubule of the Volume-Expanded Rat," *Journal of Clinical Investigation*, **50**, 1596–1602, 1971, doi:10.1172/JCI106647.
- [40] R. Quigley, "Androgens stimulate proximal tubule transport," *Gender Medicine*, **5**, S114–S120, 2008, doi:10.1016/j.genm.2008.03.011.
- [41] A. Quan, S. Chakravarty, J.-K. Chen, J.-C. Chen, S. Loleh, N. Saini, R. C. Harris, J. Capdevila, R. Quigley, "Androgens augment proximal tubule transport," *American Journal of Physiology-Renal Physiology*, **287**, F452–F459, 2004, doi:10.1152/ajprenal.00188.2003.
- [42] A. A. McDonough, "Mechanisms of proximal tubule sodium transport regulation that link extracellular fluid volume and blood pressure," *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, **298**, R851–R861, 2010, doi:10.1152/ajpregu.00002.2010.
- [43] B. Kaissling, I. Hegyi, J. Loffing, M. Hir, "Morphology of interstitial cells in the healthy kidney," *Anatomy and Embryology*, **193**, 1996, doi:10.1007/BF00186688.
- [44] P. Delanaye, R. P. Radermecker, M. Rorive, G. Depas, J. M. Krzesinski, "Indexing glomerular filtration rate for body surface area in obese patients is misleading: concept and example," *Nephrology Dialysis Transplantation*, **20**, 2024–2028, 2005, doi:10.1093/ndt/gfh983.
- [45] J. R. Nyengaard, T. F. Bendtsen, "Glomerular number and size in relation to age, kidney weight, and body surface in normal man," *The Anatomical Record*, **232**, 194–201, 1992, doi:10.1002/ar.1092320205.
- [46] C. C. Geddes, Y. M. Woo, S. Brady, "Glomerular filtration rate what is the rationale and justification of normalizing GFR for body surface area?" *Nephrology Dialysis Transplantation*, **23**, 4–6, 2007, doi:10.1093/ndt/gfm662.
- [47] G. Vervoort, B. Veldman, J. H. M. Berden, P. Smits, J. F. M. Wetzels, "Glomerular hyperfiltration in type 1 diabetes mellitus results from primary changes in proximal tubular sodium handling without changes in volume expansion," *European Journal of Clinical Investigation*, **35**, 330–336, 2005, doi:10.1111/j.1365-2362.2005.01497.x.
- [48] W.-A. S. Mula-Abed, K. A. Rasadi, D. A. Riyami, "Estimated Glomerular Filtration Rate (eGFR): A Serum Creatinine-Based Test for the Detection of Chronic Kidney Disease and its Impact on Clinical Practice," *Oman Medical Journal*, **27**, 108–113, 2012, doi:10.5001/omj.2012.23.
- [49] S. Rayego-Mateos, R. Rodrigues-Diez, J. L. Morgado-Pascual, F. Valentijn, J. M. Valdivielso, R. Goldschmeding, M. Ruiz-Ortega, "Role of Epidermal Growth Factor Receptor (EGFR) and Its Ligands in Kidney Inflammation and Damage," *Mediators of Inflammation*, **2018**, 8739473, 2018, doi:10.1155/2018/8739473.
- [50] J. Tang, N. Liu, E. Tolbert, M. Ponnusamy, L. Ma, R. Gong, G. Bayliss, H. Yan, S. Zhuang, "Sustained Activation of EGFR Triggers Renal Fibrogenesis after Acute Kidney Injury," *The American Journal of Pathology*, **183**, 160–172, 2013, doi:https://doi.org/10.1016/j.ajpath.2013.04.005.
- [51] C. A. O'Callaghan, B. Shine, D. S. Lasserson, "Chronic kidney disease: a large-scale population-based study of the effects of introducing the  $\text{eGFR}_{\text{CKD-EPI}}/\text{eGFR}_{\text{MDRD}}$  formula for eGFR reporting," *BMJ Open*, **1**, e000308, 2011, doi:10.1136/bmjopen-2011-000308.
- [52] N. Richards, K. Harris, M. Whitfield, D. O'Donoghue, R. Lewis, M. Mansell, S. Thomas, J. Townend, M. Eames, D. Marcelli, "The impact of population-based identification of chronic kidney disease using estimated glomerular filtration rate (eGFR) reporting," *Nephrology Dialysis Transplantation*, **23**, 556–561, 2008, doi:10.1093/ndt/gfm839.
- [53] J. A. Hirst, M. D. L. A. V. Montes, C. J. Taylor, J. M. Ordóñez-Mena, E. Ogburn, V. Sharma, B. Shine, T. James, F. D. R. Hobbs, "Impact of a single eGFR and eGFR-estimating equation on chronic kidney disease reclassification: a cohort study in primary care," *British Journal of General Practice*, **68**, e524, 2018, doi:10.3399/bjgp18X697937.

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

## Deploying Trusted and Immutable Predictive Models on a Public Blockchain Network

Brandon Wetzel, Haiping Xu

Computer and Information Science Department, University of Massachusetts Dartmouth, Dartmouth, MA 02747, USA

### ARTICLE INFO

Article history:

Received: 25 March, 2024

Revised: 12 May, 2024

Accepted: 29 May, 2024

Online: 16 June, 2024

Keywords:

Machine learning

Predictive model

Public blockchain

Smart contract

On-chain storage

### ABSTRACT

Machine learning-based predictive models often face challenges, particularly biases and a lack of trust in their predictions when deployed by individual agents. Establishing a robust deployment methodology that supports validating the accuracy and fairness of these models is a critical endeavor. In this paper, we introduce a novel approach to deploying predictive models, such as pre-trained neural network models, in a public blockchain network using smart contracts. Smart contracts are encoded in our approach as self-executing protocols for storing various parameters of the predictive models. We develop efficient algorithms for uploading and retrieving model parameters from smart contracts on a public blockchain, thereby ensuring the trustworthiness and immutability of the stored models, making them available for testing and validation by all peers within the network. In addition, users can rate and comment on the models, which are permanently recorded in the blockchain. To demonstrate the effectiveness of our approach, we present a case study focusing on storing vehicle price prediction models and review comments. Our experimental results show that deploying predictive models on a public blockchain network provides a proficient and reliable way to ensure model security, immutability, and transparency.

### 1. Introduction

Tasks that once required intensive manual labor can now be automated with the aid of artificial intelligence and machine learning (AI/ML) models, guaranteeing efficient completion in just minutes [1]. However, as AI/ML technologies revolutionize every aspect of our lives, ensuring the validity and fairness of these models becomes crucial. For example, in the field of recruitment, many organizations are incorporating machine learning methods into their hiring processes when dealing with high volumes of job applications. To address bias concerns, New York City implemented a groundbreaking law in July 2023 that requires all automated employment decision tools (AEDTs) to undergo bias audits prior to deployment, and the results of these audits must be made available to the public [2]. While the new regulations can help to ensure the trustworthiness of AI/ML models, it is widely recognized that such an approach shifts the responsibility for trust to a centralized regulatory body, which itself cannot guarantee its trustworthiness. An improved solution should allow users to validate the machine learning-based tools on their own and expose every facet of the tools to the public. This transparency would enable users to validate the functionality and efficiency of AI/ML tools and the absence of bias, thus fostering trust in these tools. This need for trust is not limited to AI/ML

tools but pervades various domains where intricate algorithms may pose a challenge to human comprehension. Hence, there is a vital necessity to develop a practical and trustworthy mechanism for deploying immutable and publicly accessible AI/ML-based tools, such as predictive models, that can be easily tested and validated by users.

A blockchain network is a peer-to-peer, decentralized ledger that eliminates the need to trust a centralized audit mechanism. Blockchain was initially recognized for its key role in cryptocurrency systems such as Bitcoin and Ethereum, securing and decentralizing transaction ledgers [3], [4]. However, the use of blockchain technology extends far beyond cryptocurrency applications. Blockchain technology can be employed to ensure the integrity and permanence of data in diverse industries, signifying data that is verifiable and untamperable [5]-[7]. A blockchain securely organizes data into linked blocks using cryptographic techniques, ensuring a tamper-proof and immutable record of transactions and smart contracts. The decentralized design of the blockchain network, with its features of immutability, security, and transparency, naturally addresses the issue of public accountability for data stored in the blockchain network. Therefore, if AI/ML-based predictive tools are deployed on a public blockchain, users of the AI/ML systems no longer need to rely on internal auditing for assurance. This paper uses ML-based neural network predictive models as an example to demonstrate how to deploy such models on a public blockchain

\*Corresponding Author: Haiping Xu, University of Massachusetts Dartmouth, Dartmouth, MA 02747, Email: [hxu@umassd.edu](mailto:hxu@umassd.edu)

for predicting vehicle prices, an area that is susceptible to biases driven by company or dealer interests. Machine learning models might inadvertently set prices too high or too low for specific vehicle types in order to optimize profit. The main goal of this paper is to provide a secure way of deploying machine learning tools that support user validation of predictive models to ensure that the models operate in the best interest of the user, rather than serving corporate goals. To this end, the model should receive vehicle parameters and generate price predictions, allowing users to validate the predicted prices even if they do not have specialized knowledge. In addition, users should be able to obtain validation from other users, which could be supported by the establishment of a comprehensive rating system.

In this paper, we present a novel approach to deploying predictive models using smart contracts by storing the parameters of the predictive models on a public blockchain. This method allows a user to recreate the predictive models on a local computer using the stored model parameters. The user can then validate a version of the model by running certain test cases to ensure the required accuracy and fairness of the model predictions. In addition, we define a mutable meta-block (*MB*) attached to the beginning of the blockchain to record indexing information of all models deployed on the blockchain network. The *MB* can be used to efficiently retrieve the predictive models and their review comments, thereby significantly improving the performance of the blockchain network system.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 presents the public blockchain framework that supports the use of smart contracts to store predictive models. It also describes the structural design of the regular blocks for storing predictive models and review comments, and the meta-block for efficiently retrieving models and the review comments. Section 4 describes in detail the procedures for deploying and retrieving predictive models and reviews. Section 5 presents a case study and the analysis results. Section 6 concludes the paper and mentions future work.

## 2. Related Work

Blockchain is a secure means of storing immutable and protected transactional data through a peer-to-peer decentralized ledger system. Recent advances have extended the traditional application of blockchain in cryptocurrencies to the storage of medical records, including the use of InterPlanetary File System (IPFS) in blockchain based healthcare secure storage solutions [8]. In [9], the authors proposed a blockchain model for creating a secure system for storing and sharing Electronic Health Records (EHRs). Their strategy is to store large amounts of medical data in the cloud, while the blockchain is dedicated to storing metadata related to EHRs. In [10], the authors introduced a storage framework integrating blockchain and IPFS for efficient transaction storage within the blockchain. In their architecture, the primary patient reports are stored in a decentralized off-chain storage system using IPFS, and the blockchain exclusively stores hash values of these reports. This approach effectively reduces the overall block size within the blockchain. In [11], the authors overcame the security concerns of pure cloud storage and provided a robust and scalable solution that utilizes blockchain technology to handle big data storage of healthcare multimedia

files. The approach introduces a hierarchical cloud-based blockchain framework for storing large amounts of healthcare data, resulting in significant efficiencies. Similarly, in [12] and [13], blockchain is again used as a step towards digital healthcare by storing medical images and EHRs directly in the blockchain instead of storing their metadata. Their goal in storing healthcare data via blockchain is to apply the security of blockchain to the healthcare system, mitigating the threat of tampering and increasing security. Blockchain technology has also been used in other areas to store data that may be public but must be trusted to be accurate, which is more similar to the intended application of this paper. For example, in [14], the authors proposed a public auditing mechanism for cloud storage systems using blockchain. They demonstrated that the proposed scheme is secure against type I/II/III/IV adversaries. In [15], the authors described a system that utilizes blockchain technology for video surveillance storage and sharing of videos that are encrypted but publicly accessible with a decryption key. Their approach consists of encrypting and storing camera-acquired video off-chain using distributed IPFS and storing the associated metadata in the blockchain. In addition, some other researchers proposed a secure method for sharing sensitive financial data using blockchain [16]. Their methodology involves recording access control rules, hash values, and storage addresses of financial data in the blockchain, while storing the actual financial data in a distributed database external to the blockchain. While blockchain technology has been effective in ensuring the immutability of stored data across various domains, our approach differs from existing methods in that, in addition to storing regular transactional data, we deploy ML-based predictive models in the blockchain network that are publicly available for validation and adoption by all users.

The challenge of bias and fairness in ML models has been a prevalent issue that currently lacks a universally accepted solution. A comprehensive survey on bias and fairness across various ML domains identified many areas of possible bias that can exist in AI/ML models [17]. ML models investigated in a number of real-world commercial use cases have shown that data integrity, learning parameters, and the lack of safeguards against bias can have a significant impact on the fairness of a model. The survey provides a taxonomy for determining the fairness of models and suggests that this is a pervasive problem for which researchers must find practical solutions. Furthermore, as noted in [18], mitigating bias in datasets for supervised ML is a pressing need for the emerging field of ML. Expanding on this idea, some researchers used statistical methods to assess the significant amount of bias in ML algorithms. For example, in [19], gender bias in facial recognition algorithms was measured and found to be prevalent, whereas in [20], the bias in the training data and its effect on the predictive results were detected and analyzed. The above approaches introduce methods for analyzing and assessing the amount of bias in ML models, and generally describe an approximation of the amount of bias in certain areas of ML. However, each of these proposed solutions for fairness and bias assessment requires trust in a centralized organization. In the absence of trust, e.g., when the model organizer has a strong incentive to manipulate the price of an object, a satisfactory solution remains elusive. In contrast, our approach utilizes the inherent security offered by blockchain technology for the storage of ML models. By employing decentralized hosting of ML models

within a blockchain network, each user can independently verify the accuracy of the models deployed by different organizations. This approach eliminates the need for users to worry about the fairness and potential bias of the models, as they can perform validation checks on each model individually.

The prevalence of substantial bias in ML models necessitates the need for dedicated research to mitigate bias. While bias mitigation relies on suitable bias assessment methods tailored to each unique problem, distinct solutions are needed. In [21], the authors introduced a novel approach to mitigating bias by visually displaying the attributes of an ML model to indicate bias in an intuitive manner. Subsequently, manual adjustments are required to rectify these attributes by modifying the dataset. In [22], the authors proposed a methodology for assessing fairness specific to dynamic pricing methods, where the fairness constraints on price prediction are assessed by a centralized entity. This provides a valuable mechanism for measuring and enforcing the fairness of model-based price prediction, even when the demand for the item is unpredictable. In [23], the authors described how to mitigate bias in neural network-based ML models trained on image data. The authors showed three main categories of bias mitigation methods, namely data-level techniques to balance the training data, model-level approaches to modify the learning algorithm, and adversarial approaches to identify biases in the data. Unlike the above methods, our approach reduces bias by employing a review system to ensure that users can easily find better-reviewed ML models, which tend to be less biased. Our bias mitigation strategy is centered on minimizing the bias directly encountered by users. Furthermore, our approach is versatile and applicable to a wide range of ML models, bypassing the reliance on statistical rigor and instead emphasizing user satisfaction.

The fusion of blockchain and machine learning is an emerging frontier, with ongoing research underscoring its promising potential. In [24], the authors presented various techniques concerning the preservation and processing of big data necessary for ML models. While this decentralized machine learning approach has advantages in terms of mitigating bias and improving fairness, the proposed solution falls short in describing the potentials for evaluating and validating the models, focusing mainly on the data stored in the blockchain. In [25], the authors described an ML-based approach where training gradients are validated by a decentralized blockchain network. They proposed a decentralized learning framework called LearningChain, where organizations can collaborate on training models by computing gradients together to update specific models. In [26], a similar approach to [25] was used to compute the parameters of an on-chain ML in a decentralized multi-threaded environment. Using association rule mining as an example, the authors showed how blockchain can be used to enable trusted machine learning. In [27], the authors further advanced the idea of using blockchain in the learning process by introducing explainable and traceable algorithms that make the learning process meaningful and trackable. Their study showed that smart contract-based training and prediction techniques produced mean square errors similar to scikit-learn-based prediction models. In [28], the authors analyzed the design space regarding the integration of machine learning into blockchain applications, a concept termed ML on chain. They presented a taxonomy for ML on chain, categorizing existing and prospective approaches based on design attributes and their

characteristics. While the above methods inherently offer the fundamental security advantages associated with blockchain technology, our approach is different because we only store the model parameters on the blockchain, excluding the data used in training, as all training is done off-chain. Note that storing training data on a large public blockchain would be costly and impractical. Thus, by focusing on storing only the essential components needed to reconstruct ML models, our approach optimizes deployment efficiency on a blockchain platform. Furthermore, we allow users to review a model to verify its fairness. This enables even non-technical users to make informed decisions about the degree of bias present in the model. Table 1 summarizes the main contributions and novelties of our approach by comparing it with existing approaches in terms of five key features.

Table 1: Comparison with Existing Approaches

Approach	Trustless	Fairness	Scalability	Security	Performance
Bias Mitigation Methods [21]-[23]	No	Yes	N/A	No	N/A
Conventional Blockchain [25]-[28]	Yes	No	No	Yes	No
Our Approach	Yes	Yes	Yes	Yes	Yes

As shown in Table 1, a feature can be supported (Yes) or not supported (No) by an approach, or not applicable (N/A) to an approach. The features we consider include whether the method is trustless, whether the fairness of the model can be verified, whether the method is scalable and secure, and whether the method supports improved performance in terms of efficiently searching for stored predictive models. Methods that focus on bias mitigation are typically centralized [21]-[23], so they do not support trustless computing. Since the fairness constraints of stored models are assessed by centralized entities, no security mechanisms are provided to users. On the other hand, while conventional blockchain approaches support trustless and secure computing [25]-[28], they typically do not support bias mitigation for stored models and are also not scalable due to on-chain training of predictive models. Furthermore, since conventional blockchains do not contain a meta-block, they do not support efficient search for stored predictive models.

### 3. Deploying Predictive Models on a Public Blockchain

#### 3.1. A Framework for a Public Blockchain with Smart Contracts

Each peer in a blockchain network may have a complete copy of the blockchain with a number of blocks. Figure 1 shows a framework for a public blockchain with smart contracts, as well as the list of transactions and the list of smart contracts stored in block  $B_2$ . As shown in the figure, each block in the blockchain consists of three main parts: a block header, a list of transactions, and a list of smart contracts. The block header is defined as a 5-tuple ( $BID$ ,  $HPB$ ,  $BTS$ ,  $NTR$ ,  $NSC$ ), where  $BID$  is the block ID,  $HPB$  is the hash value of the previous block,  $BTS$  is the timestamp when the block was created, and  $NTR$  and  $NSC$  are the number of transactions and the number of smart contracts contained within the block, respectively. Transactions are used to record new transactional information added to the blockchain, such as associated data related to a predictive model and review data of the model provided by users. Each transaction is defined as a 5-tuple ( $TID$ ,  $UID$ ,  $TTS$ ,  $DES$ ,  $DAT$ ), where  $TID$  is the ID of the transaction,  $UID$  is the ID of the user who initiated the transaction,



TTS is the timestamp when the transaction was created, DES is the detailed description of the transaction, and DAT is any associated data stored with the transaction.

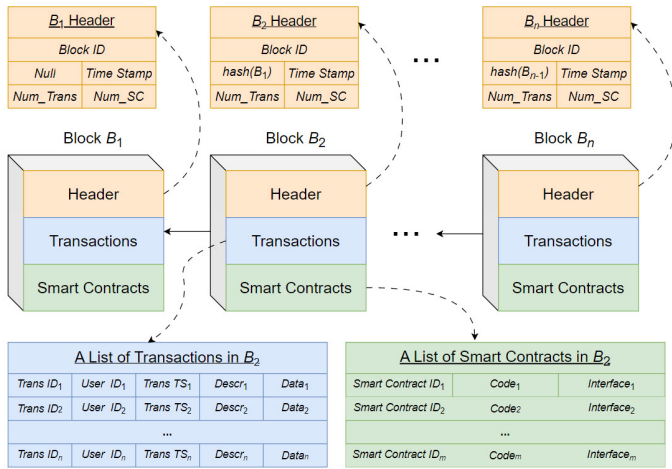


Figure 1: A Framework for a Public Blockchain with Smart Contracts

Smart contracts are initially defined as self-executing code stored in the blockchain and are often seen as the terms of an agreement between two parties. These terms are encoded directly into the smart contract, which is publicly accessible, allowing both parties to fully agree to these terms. In this paper, we broaden the concept of smart contracts to include methods for storing and accessing immutable data on the blockchain. We define a smart contract as a 3-tuple  $(SID, COD, INF)$ , where  $SID$  is the identifier of the smart contract,  $COD$  is the code of the smart contract, which is usually stored as binary code, and  $INF$  is the interface of the smart contract, which defines the methods that can be invoked. The process of deploying a smart contract involves creating a transaction  $T$ , where  $T.DES$  describes the smart contract to be deployed, including the smart contract's location in the block, and  $T.DAT$  may include any associated data, e.g., a test dataset of a predictive model that is encoded in the smart contract.

Running unverified code, such as smart contracts, on a user's local machine can pose potential security risks. To address this issue, we can assign the responsibility of verifying the trustworthiness of smart contracts on a blockchain to full-node peers. Additionally, executing smart contracts on a virtual machine, such as the Ethereum Virtual Machine (EVM), can help minimize security vulnerabilities by restricting their interactions within the virtual environment. This approach ensures a deterministic execution of smart contracts, as the virtual machine maintains full control over the execution environment.

### 3.2. Storing Predictive Models Using Smart Contracts

Smart contracts being used as storage for predictive models provide a layer of abstraction that increases the efficiency for users to access and verify predictive models stored on a blockchain network. For example, a predictive model stored with a smart contract can be efficiently extracted by invoking a corresponding method defined in the smart contract. We call a smart contract that stores a predictive model a Model Smart Contract (MSC), which contains the serialized parameters of the model and a set of methods that can be used to access and verify the model. Every version of a model must have both a model ID

(MID) and a version ID (VID), where MID and VID refer to the type of model and an instance of that model, respectively. For example, when MID refers to a set of predictive models used to forecast a company's stock price, each model version refers to a specific prediction model. Models sharing the same MID can be published by the same publisher at various times or by different publishers, but it is essential for each model to have a unique VID.

In our approach, predictive models are trained off-chain and only trained models are stored on-chain. Each peer can extract and recreate an exact copy of the trained model on a local machine by invoking a smart contract method defined in the corresponding MSC, which then allows the user to validate and execute the model without having to trust a centralized party. A smart contract containing a predictive model requires an associated transaction to be stored in the same block, documenting its deployment, location, and related data. The review comments of a model are also recorded in transactions but do not require an associated smart contract to access them. Instead, they are stored as plain text in a review transaction, directly accessible to peers. If transaction  $T$  tracks the deployment of an MSC,  $T.DES$  and  $T.DAT$  describe the MSC and its associated data, respectively. In contrast, if transaction  $T$  records the review data for a model version,  $T.DES$  and  $T.DAT$  contain the VID of the model and the corresponding review of the model, respectively. Figure 2 shows  $m$  versions of a predictive model that are stored on a blockchain network.

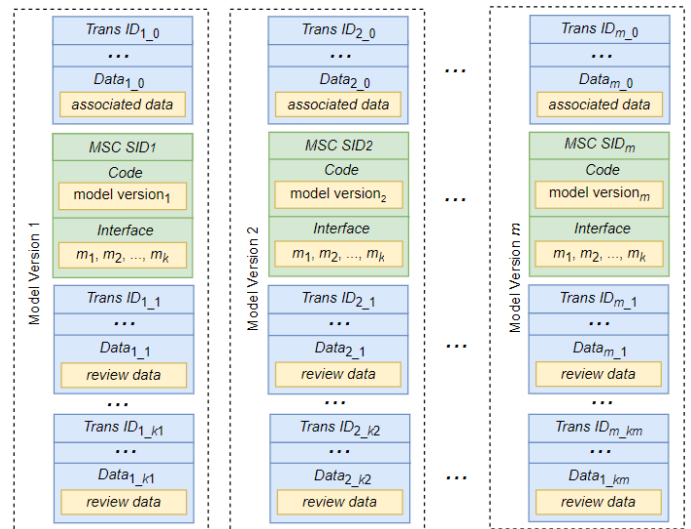


Figure 2: A Predictive Model with Multiple Versions and Review Data

As shown in Figure 2, each model version VID is stored in an MSC that defines its smart contract ID, the code of the model version, and an interface listing the methods that can be invoked by users. Each MSC model version  $i$ , where  $1 \leq i \leq m$ , stored in block  $B$  is accompanied by a transaction with ID  $i_0$  that describes the model version and is stored in the same block  $B$ . Suppose model version  $i$  has  $k_i$  reviews. These reviews are stored in transactions with IDs from  $i_1$  to  $i_{k_i}$ . Note that since the reviews for a model version are uploaded at different times, the review transactions can be stored in different blocks other than block  $B$ .

### 3.3. The Structure of a Meta-Block

In order to store indexing information for efficiently finding specific versions of a predictive model and all reviews about the

model versions, we introduce the *MB*, which is a mutable meta-block attached to the blockchain. When a new block is approved and added to the blockchain, each peer needs to update its *MB* so that it contains the latest indexing information about the models, versions, and review comments contained in the new block. Figure 3 shows the structure of the *MB* that consists of three parts, namely header, model index, and review index.

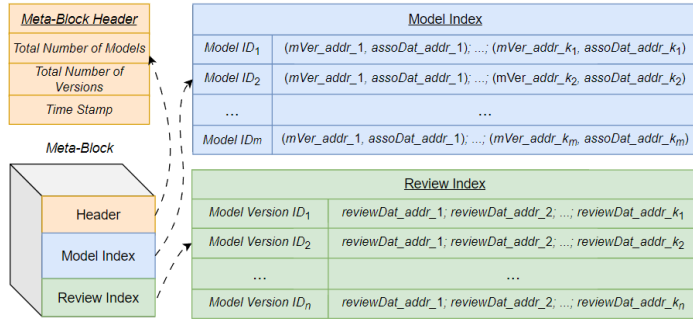


Figure 3: The Structure of a Meta-Block

The *MB* header contains the general information of the *MB*, which is defined as a 3-tuple  $(NM, NV, TS)$ , where *NM* and *NV* are the total number of models and the total number of versions, respectively, and *TS* is the timestamp when *MB* was last updated. The model index and the review index modules contain indexing information about model versions, associated data and review data in terms of their addresses in the blockchain. The address of a model version is defined as a pair  $(BID, ISC)$ , where *BID* is the ID of the block that stores the smart contract containing the model version, and *ISC* is the index of the smart contract within the list of smart contracts in the block. Similarly, the address of the model associated data or review data is defined as a pair  $(BID, ITR)$ , where *BID* is the ID of the block that stores the associated data of the model or its review data, respectively, and *ITR* is the index of the transaction that records the information about the model version and review data within the list of transactions in the block. As shown in the figure, the model index module can be implemented in the form of a HashMap, where the key is an *MID* and the output value is a list of pairs  $(mVer\_addr, assoDat\_addr)$ , containing the model version address and the model associated data address, i.e., the address of the smart contract with the predictive model and the address of the transaction that records information about the smart contract, respectively. Likewise, the review index module can also be implemented in the form of a HashMap, where the key is a *VID* of a model version, and the output value is a list of addresses that can be used to locate the transactions that record the reviews about the model version.

#### 4. Deployment and Retrieval of Predictive Models

##### 4.1. Process Overview

To support efficient peer usage of a blockchain network, we require a distinction between full-node peers and regular peers, where a full-node peer contains a full copy of the entire blockchain, while a regular peer is not required to maintain a full copy but can make a request to a full-node peer for information retrieval and uploading. In this paper, we allow regular peers to retrieve a predictive model from a full-node peer and also to send reviews to a full-node peer for publication. Full-node peers have the privilege of deploying models by adding an *MSC* and its

accompanying transaction to a new block. When a new block is added, it is broadcast to all full-node peers for updating. Figure 4 shows the process of model deployment and extraction on a blockchain with the *MB* and *m* blocks. As shown in the figure, the blockchain contains models deployed by *n* full-node peers. There are also *r* regular peers, each of which is connected to a full-node peer and can utilize this connection to retrieve a model or post a review for the model. A full-node peer connected to the regular peer can add these retrieval transactions and reviews to a new block. Once the new block contains enough transactions, the full-node peer can broadcast the new block to all full-node peers on the network for approval and adoption.

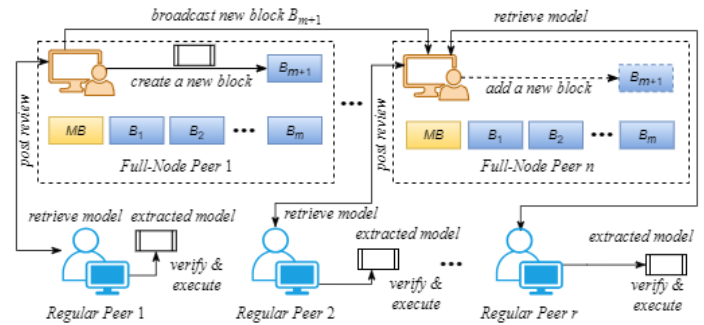


Figure 4: An Overview of the Model Deployment and Extraction Process

##### 4.2. Generating a Meta-Block

The process of generating an *MB* involves searching each block in a blockchain. While searching a block, the addresses of models and reviews are found and added to the *MB*. Each model transaction *tr*, where *tr.DES* describes an *MSC* with an *MID*, must have the accompanying smart contract found in the same block. The address of *tr* and the address of the smart contract described in *tr.DES* are paired and added to the *MB*'s *Model Index* HashMap with *MID* as a key. If the pair represents the first instance of a model, an empty list of addresses is created before the pair of addresses can be added. On the other hand, if *tr* is a review transaction that contains a review on a particular model *VID*, its address is added to the *Review Index* HashMap with *VID* as a key. The complete process to generate an *MB* for a public blockchain is described in Algorithm 1. According to the algorithm, we search for transactions in each block of the blockchain. For each transaction, if it describes the deployment of an *MSC* with *MID* and the model has not been recorded in *MB*, the pair  $\langle MID, [(mVer\_addr, assoDat\_addr)] \rangle$  is added to the *Model Index*, where *mVer\_addr*, *assoDat\_addr* are the address of the model version and the address of the associated data transaction, respectively. On the other hand, if the transaction is a review transaction for a model version *VID*, the pair  $\langle VID, [reviewDat\_addr] \rangle$  is added to the *Review Index*, where *reviewDat\_addr* is the address of review transaction for the model version *VID*. This process is repeated for all transactions in each block of the blockchain until all blocks have been processed. While generating an *MB* on an existing blockchain can be a time-consuming process, this process only needs to be performed once to be usable, thereby significantly reducing the time it takes to search and extract data from the blockchain. Additionally, a peer can download an already existing *MB* from another peer that has the latest *MB*, thus eliminating the need to search through the blockchain to create a new *MB*. Note that when a new block is

added to the blockchain, the *MB* can be updated in a similar manner as defined in Algorithm 1.

---

**Algorithm 1: Generating a Meta-Block for a Public Blockchain**


---

**Input:** A public blockchain *PB* with *n* blocks

**Output:** A generated *MB*

---

1. Create an empty *MB* with *Model Index* and *Review Index* HashMap  $\Pi_M$  and  $\Pi_R$ , respectively
  2. Initialize the total number of models *NM* to 0
  3. Initialize the total number of model versions *NV* to 0
  4. Set the time stamp *TS* to the current time
  5. **for** each block *B* in *PB*
  6.   **for** each transaction *tr* in block *B*
  7.     **if** *tr.DES* describes a model with *MID* stored in *MSC*
  8.       **if**  $\Pi_M$  does not contain any versions with key *MID*
  9.         Add  $\langle MID, [(mVer\_addr, assoDat\_addr)] \rangle$  to  $\Pi_M$
  10.        Increase *NM* by 1
  11.     **else** Update the list of model versions in  $\Pi_M$  with key *MID*
  12.    **else if** *tr.DES* describes a review of a model version *VID*
  13.     **if**  $\Pi_R$  does not contain any reviews of *VID*
  14.        Add  $\langle VID, [reviewDat\_addr] \rangle$  to  $\Pi_R$
  15.        Increase *NV* by 1
  16.     **else** Update the list of reviews in  $\Pi_R$  with key *VID*
  17. **return** *MB*
- 

#### 4.3. Deploying Models and Posting Reviews

The process of deploying a classifier can only be done by a full-node peer. This process must first check whether the corresponding model version has been deployed. The *MB* is used to ensure that a particular version of a model has not yet been deployed to the blockchain by checking whether the version being deployed has been indexed by the *MB*. Formally, we define a model as a 4-tuple (*MDA*, *MID*, *VID*, *ASD*), where *MDA* is the data required to recreate the model, stored in a standard way of storing model data such as HDF5 [29]; *MID* and *VID* are the model ID and version ID, respectively; and *ASD* is the associated data of model version *VID*. Deploying a model  $\alpha$  requires the creation of a transaction  $T_\alpha$  and a model smart contract  $MSC_\alpha$ , where  $MSC_\alpha$  contains a method for extracting model data for the model version. The transaction  $T_\alpha$  contains a description of  $MSC_\alpha$  including the use cases of the model and the address of  $MSC_\alpha$ .  $T_\alpha$  must also contain any associated data,  $ASD_\alpha$ .  $T_\alpha$  and  $MSC_\alpha$  are then added to a new block *B* that is being deployed. Deploying review  $\beta$  requires the creation of a review transaction,  $T_\beta$ , where  $T_\beta$  contains the review being posted. A new block *B* may contain multiple models or reviews, as defined by the full-node peer running the algorithm. Once block *B* is ready, it is broadcast to all full-node peers for approval before it can be published and added to the blockchain. If block *B* is successfully published, the *MB* must be updated to index the new data in block *B*. Algorithm 2 describes the process of generating and deploying a new block with multiple new models on a public blockchain. The efficiency of this process is predominantly influenced by the duration needed to broadcast block *B* and the quantity of transactions and smart contracts contained within the new block. Note that the process of publishing a review is similar to the process of deploying a model, but without the need to create an *MSC*. Therefore, in Algorithm 2, we do not show the process of adding a review transaction to the new block *B*. Once the new block *B* is approved and deployed, the *MB* needs to be updated to

add the model information and the review data addresses to the HashMaps *Model Index* and *Review Index*, respectively.

---

**Algorithm 2: Generate and Deploy a New Block**


---

**Input:** A public blockchain *PB* with an *MB*; a list of new models  $L_{NM}$

**Output:** Boolean value indicating successful or failed deployment

---

1. Let *B* be a new block to be generated and deployed
  2. **for** each predictive model  $\alpha$  in  $L_{NM}$
  3.   **if** *MB* contains key  $\alpha.MID$  in the *Model Index* HashMap
  4.     Let  $\Gamma_M$  be a list of deployed models with key  $\alpha.MID$
  5.     **if**  $\Gamma_M$  contains a model with version ID  $\alpha.VID$
  6.        **continue** // the model version has already been deployed
  7.     Create a new model smart contract  $MSC_\alpha$  for model  $\alpha$
  8.     Add  $MSC_\alpha$  to the list of smart contracts in *B*
  9.     Create a new transaction  $T_\alpha$  with model data  $\alpha.ASD$  and a description of  $MSC_\alpha$
  10.    Add  $T_\alpha$  to the list of transactions in *B*
  11. Broadcast *B* and await consensus approval determination
  12. **if** new block *B* is not approved
  13.    **return** *false* // failed deployment
  14. **else** Update *MB* to include new information from *B*
  15. **return** *true* // successful deployment
- 

#### 4.4. Extracting Models and Reviews

For a regular peer, the process of extracting a model of *MID* from the blockchain requires a connection to a full-node peer in order to make the request. When a full-node peer receives such a request, it retrieves all its model versions of *MID* and their reviews from the blockchain, and then returns the result as a list of model instances to the regular peer. Algorithm 3 describes the process of extracting all predictive models of *MID* and their reviews from the public blockchain *PB* with an *MB*.

---

**Algorithm 3: Extracting Predictive Models and their Reviews**


---

**Input:** A public blockchain *PB* with an *MB*; a model *MID* to retrieve all its model versions and their reviews

**Output:** A list of model instances of *MID* along with their reviews

---

1. Initialize  $L_M$  to an empty list of model instances of *MID*
  2. **if** *MB* contains the key *MID* in the *Model Index* HashMap
  3.   Let  $\Gamma_M$  be a list of  $\langle mVer\_addr, assoDat\_addr \rangle$  with key *MID*
  4.   **else return**  $L_M$  //  $L_M$  is an empty list, i.e., no model is found
  5.   **for** each pair  $\langle mVer\_addr, assoDat\_addr \rangle$  in  $\Gamma_M$
  6.     Let the model to which the pair refers be model  $\alpha$  with  $VID_\alpha$
  7.     Create a new empty model instance *A*
  8.     Extract model  $\alpha$  stored at  $mVer\_addr$
  9.     Extract associated data  $assoDat$  stored at  $assoDat\_addr$
  10.    Add model  $\alpha$  and associated data  $assoDat$  to *A*
  11.    **if** *MB* contains the key  $VID_\alpha$  in the *Review Index* HashMap
  12.     Let  $\Gamma_V$  be a list of  $reviewDat\_addr$  with key  $VID_\alpha$
  13.     **for** each  $reviewDat\_addr$  in  $\Gamma_V$
  14.       Extract the review  $reviewDat$  stored at  $reviewDat\_addr$
  15.       Attach  $reviewDat$  to *A*
  16.     Add *A* to the list of model instances  $L_M$
  17. **return**  $L_M$
- 

As shown in Algorithm 3, the full-node peer first searches for *MID* in its *Model Index* HashMap. If the model with *MID* does not exist, an empty list of model instances is returned. Otherwise, each model version must be extracted from the blockchain and added to a list of model instances  $L_M$ . For each pair  $\langle mVer\_addr, assoDat\_addr \rangle$ , both the model  $\alpha$  stored at  $MSC\_addr$  and the

model data *assoDat* stored at *assoDat\_addr* are extracted from the corresponding blocks in the blockchain and added to a model instance *A*. Then, the full-node peer searches for *VID\_α* in its *Review Index* HashMap. For each review stored at address *reviewDat\_addr*, it is extracted from the corresponding block in the blockchain and attached to the model instance *A*. Once all review comments have been retrieved, *A* is added to the list of model instances *L<sub>M</sub>*. Finally, when all model instances have been retrieved, *L<sub>M</sub>* is returned to the regular peer with the retrieval request. Note that without the *MB*, the extraction process requires traversing the entire blockchain. With the *MB*, it is possible to quickly find the addresses of predictive models and their review comments, thus making the extraction process very efficient.

## 5. Case Study and Simulation Results

The case study presented in this section demonstrates the efficacy of storing predictive models on a public blockchain. The stored models are multilayer perceptron (MLP) neural network classifiers that predict vehicle prices based on given parameters such as age and mileage. MLP classifiers were chosen due to their relatively large storage requirements compared to other ML methods such as linear regression. By showcasing the successful storage of MLP classifiers on a public blockchain network, we enhance our confidence in the adaptability of our approach to other types of predictive models and domains. This provides a springboard for more ambitious future research efforts. The MLP classifiers used in this case study are deployed by vehicle dealerships with an *MID* for each model, and a *VID* for each version of these models. For simplicity, in this case study, the *MID* and *VID* contain the vehicle model and vehicle year as substrings, respectively. For example, the classifier that predicts the price of a 2011 Toyota Camry, deployed by car dealer *D* on March 15, 2024, may have an *MID* of “Toyota Camry” and a *VID* of “2011\_D\_03152024”.

### 5.1. Environment Settings

We have developed two different modules: one using the Ethereum blockchain [4] and the other providing users with a web interface to interact with the blockchain. To experiment on the Ethereum blockchain, we employed Ganache, a tool commonly used for testing and development [30]. As with many smart contracts on Ethereum, the smart contracts developed in this case study were also coded in Solidity. While this approach is experimentally tested on the Ethereum network, it also involves simulating processes such as connecting to a full-node peer to retrieve data from the blockchain. In addition, since the approval of new blocks takes place through a consensus mechanism [31], we simulated the approval times of new blocks, assuming that the times are normally distributed with appropriate means and standard deviations (*STDs*). In this sense, we collect data points partly from measurements of real-world implementations and partly from simulated values that best fit real-world scenarios.

The web interface was created to allay the concerns of regular users about the complexity of blockchain usage and to establish a secure environment for conducting the case study. Regular users can interact with the user-friendly interface to search for classifiers and their review comments, select suitable classifiers, extract classifiers and execute them on their local machines. It also allows regular users to create, publish, and read reviews

comments through the interface. The website interacts with Ethereum through the *Web3.js* tool, facilitating user engagement with the Ethereum nodes. A regular peer can use the website by connecting to a full-node peer and assembling a classifier locally with the support of the *TensorFlow.js* tool. Review comments on a classifier are uploaded and viewed through the website by regular users. Various types of users such as companies or dealers can create and train classifiers with TensorFlow before deploying them to the blockchain through the interface.

### 5.2. Space Efficiency for Storing Classifiers

Space efficiency is crucial in a public blockchain because the blockchain needs to be stored on every full-node peer. This experiment provides a basis for the expected spatial storage of our proposed solution. In an MLP classifier, parameters are numerical values that quantify the complexity of the classifier. TensorFlow contains efficient methods for storing a classifier that allow us to isolate these parameters. The number of parameters *nP* in an MLP classifier can be calculated as in (1).

$$nP = \sum_{i=1}^{n-1} ((Layer_i + 1) * Layer_{i+1}) \quad (1)$$

where *Layer<sub>i</sub>* refers to the number of neurons in layer *i* of the classifier. The rationale for this equation is that for each neuron in a layer, the number of connections to that node must be the same as the number of neurons in the previous layer plus a connection from a bias neuron. For example, a model with 5 input neurons, 1 output neuron, and 3 hidden layers, each with 128 neurons, would have a total of 33,536 parameters. This quantifies the complexity of the classifier, which allows for comparisons between the different scenarios in this case study.

In our approach, when determining the storage size of a classifier, we consider the size of the classifier itself as well as the overhead of storage, including the storage size of the methods defined in a smart contract and the size of the transactions that track the deployment of the classifier. The storage overhead is usually stable, while the size of the stored classifier can be efficiently measured by the number of bytes stored for its parameters. In addition, we must also account for the number of reviews for each classifier. We define a random number of reviews for each classifier by skewing the normal distribution [32], [33], and define its *pdf* (probability density function) and *cdf* (cumulative density function) using the *pdf* and *cdf* of the normal distribution as shown in (2-5).

$$pdf_{norm}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad (2)$$

$$cdf_{norm}(x) = \int_{-\infty}^{\frac{x-\mu}{\sigma}} pdf_{norm}(t) dt \quad (3)$$

$$pdf_{skew}(x) = 2 * pdf_{norm}\left(\frac{x-\xi}{\omega}\right) * cdf_{norm}\left(\alpha \frac{x-\xi}{\omega}\right) \quad (4)$$

$$cdf_{skew}(x) = \int_{-\infty}^{\frac{x-\xi}{\omega}} pdf_{skew}(t) dt \quad (5)$$

In (2-5), the parameters  $\mu$  and  $\sigma$  represent the mean and *STD* of a normal distribution and are set to 0 and 1, respectively. Parameters  $\alpha$ ,  $\xi$ , and  $\omega$  represent the *shape*, *scale* and *location* of a skew normal distribution and their values are chosen to be 10, 0, and 200, respectively, to better match real-world scenarios. The rationale for using this distribution and the chosen parameter values is based on the observations on websites such as Amazon

and others with reviews of items, where only a few items are highly popular and reviewed but most items are unpopular and have much fewer reviews. With this understanding, we can adjust the parameter values to create a right-skewed distribution we would find in the real world. The chosen values can produce a highly skewed distribution, similar to the distribution of product reviews on Amazon. This distribution is visualized as in Figure 5, where the distribution has a high variance and a significant skew to the right. The mode can be visually shown to be approximately 70 reviews, although the mean is expected to be larger due to the significant skew.

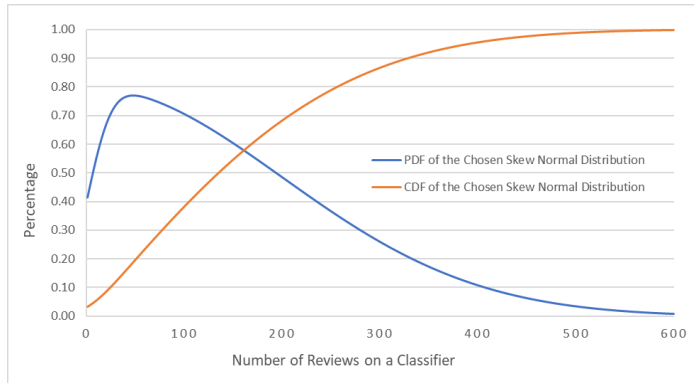


Figure 5: An Example of Skew Normal Distribution of Reviews

The mean, *STD* and skewness of a skew normal distribution can be calculated as in (6-8):

$$mean = \xi + \omega\delta\sqrt{\frac{2}{\pi}}, \text{ where } \delta = \frac{\alpha}{\sqrt{1+\alpha^2}} \quad (6)$$

$$STD = \omega\sqrt{1 - \frac{2\delta^2}{\pi}} \quad (7)$$

$$skewness = \frac{4-\pi}{2} \frac{(\delta\sqrt{2/\pi})^3}{(1-2\delta^2/\pi)^{3/2}} \quad (8)$$

Table 2 shows the resulting mean, *STD*, and skewness of the number of reviews, as well as the *MAX* and *MIN* chosen for the skew normal distribution.

Table 2: Chosen Parameters for the Size of Review Storage

	Number of Reviews <sup>1</sup>	Review Transaction Size (KB) <sup>2</sup>
<b>Mean</b>	159	0.35
<b>STD</b>	121	0.03
<b>Max</b>	600	0.6
<b>Min</b>	0	0.25
<b>Skewness</b>	0.96	0

<sup>1</sup> Number of Reviews follows a skew-normal distribution with skewness of 0.96

<sup>2</sup> Review Transaction Size follows a normal distribution (skewness = 0)

As shown in Table 2, we assume the review transaction size follows a normal distribution with a mean of 0.35KB, a standard deviation of 0.03, and a skewness of 0. Note that review transactions stored on Ethereum require a minimum transaction size, while the maximum size is set to prevent malicious attacks on the blockchain network, such as posting unreasonably large reviews. With a generated random number of reviews for a classifier, we can calculate the total space  $S_{cla}$  to store the classifier and its reviews on Ethereum as in (9).

$$S_{cla} = MSCSize(claComp) + RevSizes(numRev) \quad (9)$$

In (9),  $claComp$  is the classifier complexity (i.e., the number of model parameters),  $MSCSize$  is a function that computes the amount of storage space required to store the model's *MSC* in a block,  $numRev$  is a random variable representing the number of reviews, and  $RevSizes(numRev)$  is the size of  $numRev$  randomized review transactions. The function  $MSCSize$  was determined from experimental test data on the Ethereum network, where a linear relationship was found between the complexity of a classifier and the size of the associated *MSC* containing the method of extracting the model parameters of a classifier.

Figure 6 shows the 500 data points generated based on the chosen parameters listed in Table 2 and the storage space required on Ethereum. Each data point is generated by randomizing the model complexity (i.e., a random number of parameters in the model) and yields the storage space required to store the model and the associated transaction describing the model. We further generate the random number of reviews and their review transaction sizes based on the distributions provided in Table 2. The storage sizes of the classifier and all its reviews are summed to yield the total size required to store the model on the Ethereum blockchain network.

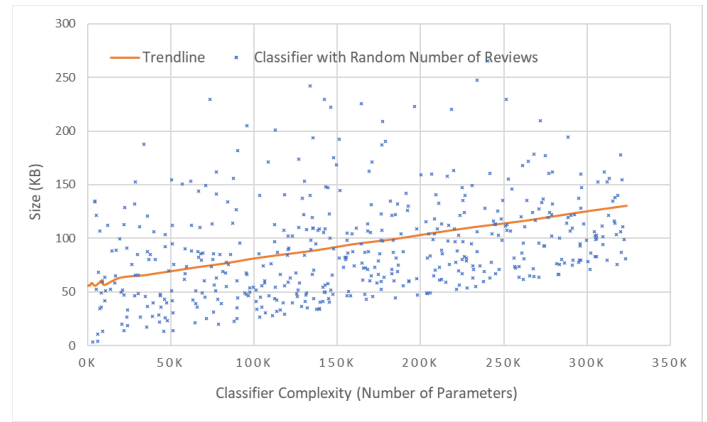


Figure 6: Storage Size of Uploaded Classifier and Their Reviews

As shown in Figure 6, the trendline clearly demonstrates that the average size of storing a model on Ethereum is reasonably small and increases with classifier complexity. While the presence of highly reviewed models puts the trendline slightly above the large clustering of models, the size required to store such classifiers is typically less than 250KB, even at the far end of the range where high-complexity classifiers are accompanied by many reviews. This result is promising and demonstrates both the spatial efficiency and scalability of this approach. For example, if there are 100 car dealerships, each deploying 50 models per year, with approximately 200K parameters in each model, for 10 years, the blockchain storage requirement would be merely 10GB. It is worth noting that while we used Ethereum for our experiments in this paper, our vision for the future is to develop dedicated public blockchain networks for storing specific types of predictive models. Thus, such dedicated public blockchain networks could be scalable and available for a long period of time (e.g., 10+ years). In addition, we could also consider mitigating the scalability issue further by using historical blockchains, such as those discussed in [34].

### 5.3. Time Efficiency for Deploying a New Block

The amount of time to deploy a new block containing multiple classifiers includes the time to broadcast a new block to other full-node peers and the time to approve the new block using a consensus mechanism. The broadcast time can be estimated based on the broadcast rate and sizes of the classifiers included in the new block. As shown in Table 3, we assume that the size of the classifier is normally distributed with a mean value of 200KB for storing the classifier parameters with different model complexity and the overhead required to store the model on Ethereum. Note that the classifier size distribution requires a maximum value to avoid unrestricted file upload attacks on the blockchain network and a minimum classifier size to avoid deployment of trivial models on the blockchain. Considering the uncertainty of network traffic and the possible bandwidth constraints that may be enforced by the full-node peers, we assume that the broadcast rate and the approval time (i.e., the consensus time) are also normally distributed with the parameters listed in Table 3.

Table 3: Parameters for Deploying a New Block with Multiple Classifiers

	Broadcast Rate (KB/s)	Classifier Size (KB)	Consensus Time (s)
Mean	500	200	12
STD	300	20	2
Max	1,000	1,000	100
Min	200	10	8

Our estimate of the approval time is based on the average time it takes to add a new block to the Ethereum blockchain, which is about 12 seconds [35]. Similarly, the consensus time must be able to time out (e.g., 100 seconds) when there are not enough full-node peers to approve a new block. Since classifiers must be deployed through blocks, the number of transactions or classifiers in each block directly affects the time required for deployment. The time required to broadcast a new block to other full-node peers increases with the number of transactions and classifiers, as well as the sizes of the transactions and classifiers. The time to deploy a new block  $T_{deploy}$  can be generated using (10), where  $Clasizes(numCla)$  returns the size of  $numCla$  classifiers in the new block,  $BCR$  is a random broadcast rate, and  $ConsTime$  is a random consensus time.

$$T_{deploy} = \frac{Clasizes(numCla)}{BCR} + ConsTime \quad (10)$$

Figure 7 shows the time required to deploy a new block with 1 to 10 classifiers. There are 30 data points for each number of classifiers, each representing a new block, and the variables are chosen randomly according to the distributions defined in Table 3. The trendline in the figure shows how much deployment time is expected to be needed as the number of classifiers increases. With 10 classifiers, the expected deployment time is about 18 seconds, which is a reasonable waiting time for the users. We note that among the above data points, the time used for the approval process accounts for the largest percentage of the total deployment time, while the percentage of the time used for broadcasting new blocks increases slightly with the number of classifiers. Based on the experimental results, it is desirable to include more classifiers in the same block when there is a high demand for uploading classifiers in a short period of time.

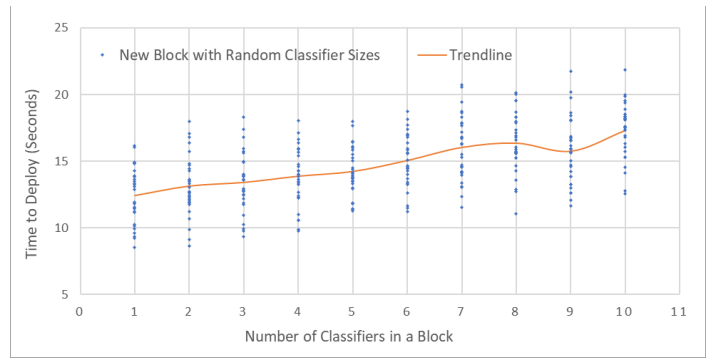


Figure 7: Time to Deploy a New Block

### 5.4. Generation of a Meta-Block

Another key use case for this approach is the generation of an *MB* by a full-node peer. This process is largely affected by the number of transactions in each block and the total number of blocks in the blockchain. Since there is usually an upper limit to the number of transactions that can be reasonably placed in a block, the time to generate an *MB* would be approximately linear with the number of blocks in the blockchain. In a real-world setting, a full-node peer needs to connect to other full-node peers to verify the consistency of its blockchain before generating an *MB*. The time to connect to other full-node peers for such verification is assumed to be normally distributed with a mean time of 10 seconds and a *STD* of 2 seconds. The time  $T_{genMB}$  to generate a meta-block can be calculated using (11), where  $GenMB$  is the time to generate an *MB* based on experimental tests, and  $ConnPeer$  is the randomly simulated time to connect to a full-node peer node.

$$T_{genMB} = GenMB + ConnPeer \quad (11)$$

Figure 8 shows the simulation time to generate an *MB* based on the number of blocks in the blockchain.

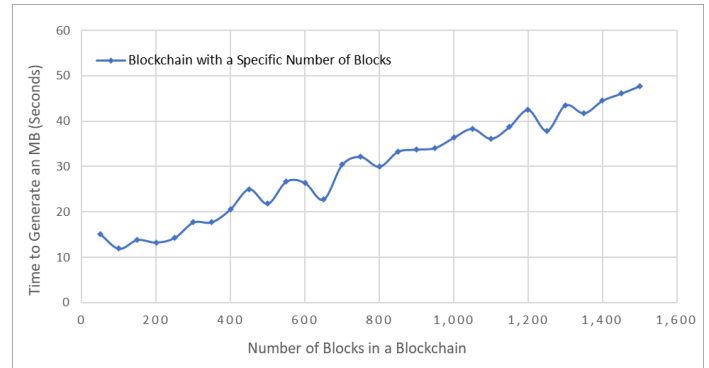


Figure 8: Time to Generate a Meta-Block

Note that Figure 8 only shows up to 1,500 blocks; however, based on the degree of linearity shown in the figure, we can infer how long it takes to generate an *MB* for a blockchain with a larger number of blocks. Additionally, the stochastic impact of the time to connect to a full-node peer is reduced when the number of blocks increases. This is because the connection time is centered on a 10-second average, so as the number of blocks increases, the impact of the connection time becomes smaller compared to the total generation time. For a blockchain with 1,500 blocks, the time required to generate an *MB* in less than a minute is reasonable,

and the fact that it only needs to be executed once demonstrates the effectiveness of this approach.

5.5. Analysis of Search and Retrieval Time

The search and retrieval (S&R) time for classifiers and their reviews is the main waiting time for regular users; therefore, we must minimize this time as much as possible. For each retrieval request made by a regular peer, several versions and their review comments must be retrieved and sent back. While the search time through an *MB* is expected to be efficient, regular peers that do not maintain the blockchain need time to connect to a full-node peer, which retrieves the requested data and returns results to the regular peer. Thus, the S&R time  $T_{S\&R}$  consists of three components as shown in (12): the time required to connect to a full-node peer from the regular peer (*ConnPeer*), the time required for the full-node peer to search for each requested classifier and its review comments, and the time required to send this data back to the regular peer (i.e., the download time). Note that the download time is based on the total size of the classifiers *Clasizes*, the total size of all reviews *RevSizes*, and the download rate (*DLR*) for downloading the classifiers and their reviews.

$$T_{S\&R} = ConnPeer + Search(n) + \frac{Clasizes+RevSizes}{DLR} \quad (12)$$

The search time is measured on an Ethereum blockchain network. According to Algorithm 3, the expected search time *Search(n)* is approximately linear as the number of blocks *n* increases. The number of reviews for each model version follows the skew normal distribution defined in Table 2. The size of a classifier, the size of a review and the download rate are randomly generated following the normal distribution defined in Table 4.

Table 4: Parameters for Searching and Retrieving Classifiers

	Review Size (KB)	Classifier Size (KB)	Download Rate (KB/s)	Connection Time (Sec)
Mean	0.1	100	500	10
STD	0.03	50	300	2
Max	0.35	800	1,000	100
Min	0	10	200	1

As shown in Table 4, the size of each review and classifier is significantly different from those listed in Table 2 and Table 3, respectively. This is because in this experiment, we only need to download the reviews and classifiers themselves and not the transactions associated with them. It is important to note that transactions stored with Ethereum usually contain additional information that is not necessary for a regular peer to download. The time to download data is determined by the amount of data downloaded, defined as the last term in (12). In addition, we assume that *ConnPeer* is normally distributed, with a mean time of 10 seconds and a *STD* of 2.

Figure 9 shows the time required to search and retrieve three arbitrarily chosen classifiers and their reviews versus the number of blocks in a blockchain. To demonstrate the effectiveness of our approach, two different search methods are included in the figure: one is our approach using *MB*, and the other is a conventional blockchain method that does not use *MB*. The conventional approach searches for all transactions from the newest block to the oldest block and saves reviews and model data during the search. We conducted 250 experiments with each approach,

where a data point represents a request from a regular peer. The S&R time for each data point is calculated using (12), where the search times were measured by simulations on Ethereum.

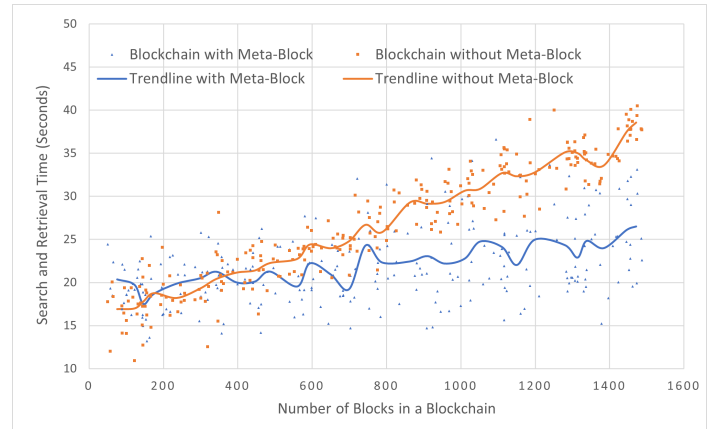


Figure 9: Search and Retrieval Time for Classifiers and Their Reviews

In order to clearly demonstrate the effectiveness of our approach, we show the trendlines for both methods. As can be seen from the trendlines, the average S&R time of the blockchains using *MB* remains relatively stable. The difference between a large number of blocks and a small number of blocks is about 5 seconds, while the conventional method without using *MB* takes significantly longer as the number of blocks increases. Thus, based on the experimental results, our approach greatly improves the ability to search a blockchain compared to a conventional approach without using *MB*.

It is worth noting that the experimental results are based on real-world data collected from the Ethereum blockchain network, as well as simulated data designed to demonstrate the practical implications of the case study. The inclusion of carefully selected distribution values enhances the credibility of the case study and increases confidence in the case study solution.

6. Conclusions and Future Work

As machine learning becomes an increasingly important part of many people’s lives, the need to address the issue of trust in these algorithms grows every day. The approach proposed in this paper aims to completely eliminate the need for trust by combining machine learning models with a decentralized public blockchain network. Machine learning models are stored on a blockchain through smart contracts, which contain methods for accessing and validating the stored models. Transactions are used to track the deployment of models and record their reviews. Thus, model users do not need to trust individual organizations, but only peer reviews and their own audits of machine learning algorithms. To further enhance the user experience, we add a meta-block at the beginning of the blockchain to record indexing information for all models and reviews stored on the blockchain. Our experimental results on Ethereum show that our approach is effective and efficient when deploying predictive models on a public blockchain network.

For future work, we plan to develop dedicated public blockchain networks for deploying predictive models. Storage and computation costs can be reduced as the scale of a dedicated public blockchain network is more manageable. The impact of

congestion on the blockchain is also a key issue that needs to be addressed [36]. In our current approach, the approval of new blocks and the retrieval of classifiers and their reviews are handled by full-node peers. In future work, we may consider developing efficient load balancing mechanisms for full-node peers [34] to mitigate the impact of blockchain network congestion on system performance. Furthermore, combined with the trustless nature of blockchain, we can bring more systematic changes to the way the machine learning models are created. This could include publishing metrics for training data or goals for the training process on the blockchain. Additionally, training could take place entirely on the blockchain, rather than off-chain as in our approach. While an on-chain approach may lead to scalability issues, it allows users to trust the process of creating the model rather than the model itself. This can be important in situations where the trustworthiness of a model cannot be easily validated by regular users. To elaborate on this approach, a new smart contract could be designed to handle on-chain training, where full-node peers have the ability to define new methods and execute them to train predictive models. Finally, we can consider implementing automated auditing of new AI/ML models [37]. This automated approach helps eliminate undesirable AI/ML models from being stored on the blockchain, thus saving model storage space. Based on previous work [38], new models can be initially deployed in a temporary block, transitioning to a permanent block upon successful completion of auditing. This auditing procedure can be automated, facilitated by classifiers trained on historical data to discern undesirable models.

### Conflict of Interest

The authors declare no conflict of interest.

### Acknowledgment

We thank the editors and all anonymous referees for spending their valuable time carefully reviewing this paper and making many suggestions for improvement. We also thank the University of Massachusetts Dartmouth for providing financial support to the first author to complete this work.

### References

- [1] Sarker, "AI-based modeling: techniques, applications and research issues towards automation, intelligent and smart systems," *SN Computer Science*, **3**(158), 1-20, 2022, doi: 10.1007/s42979-022-01043-x.
- [2] NYC 311, "Automated employment decision tools," The Official Website of the City of New York, July 2023. Retrieved on September 1, 2023 from <https://portal.311.nyc.gov/article/?kanumber=KA-03552>.
- [3] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," White Paper, Bitcoin Project, October 2008. Retrieved on January 15, 2023 from <https://bitcoin.org/bitcoin.pdf>.
- [4] V. Buterin, "Ethereum: a next-generation smart contract and decentralized application platform," Ethereum Whitepaper, 2014. Retrieved on May 15, 2023 from <https://ethereum.org/en/whitepaper/>.
- [5] O. Dib, K.-L. Brousmiche, A. Durand, E. Thea, E. B. Hamida, "Consortium blockchains: overview, applications and challenges," *International Journal on Advances in Telecommunications*, **11**(1&2), 51-64, 2018.
- [6] H. Guo, X. Yu, "A survey on blockchain technology and its security," *Blockchain: Research and Applications*, **3**(2), February 2022, doi: 10.1016/j.bcr.2022.100067.
- [7] Thamrin, H. Xu, "Cloud-based blockchains for secure and reliable big data storage service in healthcare systems," In Proceedings of the 15th IEEE International Conference on Service-Oriented System Engineering (IEEE SOSE 2021), 81-89, Oxford Brookes University, UK, August 2021, doi: 10.1109/SOSE52839.2021.00015.
- [8] S. Kumar, A. K. Bharti, R. Amin, "Decentralized secure storage of medical records using blockchain and IPFS: a comparative analysis with future directions," *Security and Privacy*, **4**(5), 1-16, April 2021. doi: 10.1002/spy2.162.
- [9] H. Wang, Y. Song, "Secure cloud-based EHR system using attribute-based cryptosystem and blockchain," *Journal of Medical Systems*, **42**(152), 1-9, July 2018, doi: 10.1007/s10916-018-0994-6.
- [10] S. Liu, H. Tang, "A consortium medical blockchain data storage and sharing model based on IPFS," In Proceedings of the 4th International Conference on Computers in Management and Business (ICCMB 2021), 147-153, Singapore, January 2021, doi: 10.1145/3450588.3450944.
- [11] Thamrin, H. Xu, "Hierarchical cloud-based consortium blockchains for healthcare data storage," 2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C), 644-651, Hainan, China, December 2021, doi: 10.1109/QRS-C55045.2021.00098.
- [12] S. D. Ashwini, A. P. Patil, S. K. Shetty, "Moving towards blockchain-based solution for ensuring secure storage of medical images," In Proceedings of the 2021 IEEE 18th India Council International Conference (INDICON), 1-5, Guwahati, India, December 19-21, 2021, doi: 10.1109/INDICON52576.2021.9691516.
- [13] S. Ballal, Y. Chandre, R. Pise, B. Sonare, S. Patil, "Blockchain-based decentralized platform for electronic health records management," In Proceedings of the 2023 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS), 1-5, New Raipur, India, October 06-08, 2023, doi: 10.1109/ICBDS58040.2023.10346392.
- [14] S. Ajarapu, S. K. Pasupuleti, "Blockchain based certificateless privacy preserving public auditing for cloud storage systems," In Proceedings of the 2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC), 286-291, Solan, Himachal Pradesh, India, November 25-27, 2022, doi: 10.1109/PDGC56933.2022.10053241.
- [15] Y. Jeong, D. Hwang, K. Kim, "Blockchain-based management of video surveillance systems," In Proceedings of the 2019 International Conference on Information Networking (ICOIN), 465-468, Kuala Lumpur, Malaysia, January 2019, doi: 10.1109/ICOIN.2019.8718126.
- [16] Z. Su, H. Wang, H. Wang, X. Shi, "A financial data security sharing solution based on blockchain technology and proxy re-encryption technology," In Proceedings of the IEEE 3rd International Conference of Safe Production and Informatization (IICSPI), 462-465, Chongqing City, China, 2020, doi: 10.1109/IICSPI51290.2020.9332363.
- [17] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, A. Galstyan, "A Survey on bias and fairness in machine learning," *ACM Computing Surveys*, **54**(6), Article No. 115, 1-35, July 2022, doi:10.1145/3457607.
- [18] V. N. Mandhala, D. Bhattacharyya, D. Midhunchakkaravarthy, "Need of mitigating bias in the datasets using machine learning algorithms," In Proceedings of the 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), 1-7, Chennai, India, 2022, doi: 10.1109/ACCAI53970.2022.9752643.
- [19] M. Atay, H. Gipson, T. Gwyn, K. Roy, "Evaluation of gender bias in facial recognition with traditional machine learning algorithms," In Proceedings of the 2021 IEEE Symposium Series on Computational Intelligence (SSCI), 1-7, Orlando, FL, USA, December 2021, doi: 10.1109/SSCI50451.2021.9660186.
- [20] S. Rohani, R. Baeza-Yates, "Measuring bias," In Proceedings of the 2023 IEEE International Conference on Big Data (BigData), 1289-1298, Sorrento, Italy, 2023, doi: 10.1109/BigData59044.2023.10386679.
- [21] H. Wang, S. Mukhopadhyay, Y. Xiao, S. Fang, "An interactive approach to bias mitigation in machine learning," In Proceedings of the 2021 IEEE 20th International Conference on Cognitive Informatics & Cognitive Computing (ICCI\*CC), 199-205, Banff, AB, Canada, October 2021, doi: 10.1109/ICCI\*CC53683.2021.9811333.



- [22] M. C. Cohen, S. Miao, Y. Wang, "Dynamic pricing with fairness constraints," SSRN, September 2021. Retrieved on October 1, 2023 from [https://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=3930622](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3930622).
- [23] Y. Wang, H. Liu, "De-biasing methods in neural networks: a survey," In Proceedings of the 2023 International Conference on Machine Learning and Cybernetics (ICMLC), 458-463, Adelaide, Australia, July 2023, doi: 10.1109/ICMLC58545.2023.10327985.
- [24] H. Maheshwari, U. Chandra, D. Yadav, A. Gupta, R. Kaur, "Machine learning and blockchain: a promising future," In Proceedings of the 4th International Conference on Intelligent Engineering and Management (ICIEM), 1-6, London, United Kingdom, May 09-11, 2023, doi: 10.1109/ICIEM59379.2023.10166343.
- [25] X. Chen, J. Ji, C. Luo, W. Liao, P. Li, "When machine learning meets blockchain: a decentralized, privacy-preserving and secure design," In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), 1178-1187, Seattle, WA, USA, December 10-13, 2018, doi:10.1109/BigData.2018.8622598.
- [26] T. Wang, "A unified analytical framework for trustable machine learning and automation running with blockchain," In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), 4974-4983, Seattle, WA, USA, 2018, doi: 10.1109/BigData.2018.8622262.
- [27] S. Badrudoja, R. Dantu, Y. He, A. Salau, K. Upadhyay, "Scalable smart contracts for linear regression algorithm," International Conference on Blockchain Technology and Emerging Applications (BlockTEA 2022), Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, **498**, 19-31, Springer, Cham, April 2023, doi: 10.1007/978-3-031-31420-9\_2.
- [28] B. Gu, A. Singh, Y. Zhou, J. Fang, F. Nawab, "ML on chain: the case and taxonomy of machine learning on blockchain," In Proceedings of the 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 1-18, Dubai, United Arab Emirates, May 1-5, 2023, doi: 10.1109/ICBC56567.2023.10174908.
- [29] M. Folk, G. Heber, Q. Koziol, E. Pourmal, D. Robinson, "An overview of the HDF5 technology suite and its applications," In Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases, 36-47, Uppsala Sweden, March 25, 2011, doi: 10.1145/1966895.1966900.
- [30] ConsenSys, "What is Ganache?" Overview - Truffle Suite, ConsenSys Software Inc., 2022. Retrieved on March 15, 2024 from <https://archive.trufflesuite.com/docs/ganache/>.
- [31] al-Qerem, A. Hammarsheh, A. M. Ali, Y. Alslman, M. Alauthman, "Using consensus algorithm for blockchain application of roaming services for mobile network," International Journal of Advances in Soft Computing & its Applications, **15**(1), 99-112, March 2023, doi: 10.15849/IJASCA.230320.07.
- [32] O'Hagan, T. Leonard, "Bayes estimation subject to uncertainty about parameter constraints," *Biometrika*, **63**(1), 201-203, 1976, doi: 10.1093/biomet/63.1.201.
- [33] S. K. Ashour, M. A. Abdel-hameed, "Approximate skew normal distribution," *Journal of Advanced Research*, **1**(4), 341-350, October 2010, doi: 10.1016/j.jare.2010.06.004
- [34] M. Felipe, H. Xu, "A scalable storage scheme for on-chain big data using historical blockchains," In 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security Companion (QRS-C), 54-61, IEEE BSC 2022, Guangzhou, China, December 5-9, 2022, doi: 10.1109/QRS-C57518.2022.00017.
- [35] Ethereum, "Blocks," Ethereum Documents, February 27, 2024. Retrieved on March 15, 2024 from <https://ethereum.org/developers/docs/blocks>.
- [36] S. Ahn, T. Kim, Y. Kwon, S. Cho, "Packet aggregation scheme to mitigate the network congestion in blockchain networks," In Proceedings of the 2020 International Conference on Electronics, Information, and Communication (ICEIC), 1-3, Barcelona, Spain, January 19-22, 2020, doi: 10.1109/ICEIC49074.2020.9051158.
- [37] J. R. Beckstrom, "Auditing machine learning algorithms: a white paper for public auditors," *International Journal of Government Auditing*, **48**(1), 40-41, Winter Edition, 2021.
- [38] R. Ming, H. Xu, "Timely publication of transaction records in a private blockchain," In 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), 116-123, IEEE BSC 2020, Macau, China, December 11-14, 2020, doi: 10.1109/QRS-C51114.2020.00030.

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

## Automated Performance analysis E-services by AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC

Rebwar Khalid Muhammed<sup>1</sup>, Kamaran Hama Ali Faraj<sup>2,3</sup>, Jaza Faiq Gul-Mohammed<sup>4</sup>, Tara Nawzad Ahmad Al Attar<sup>2</sup>, Shaida Jumaah Saydah<sup>5</sup>, Dlsoz Abdalkarim Rashid<sup>2</sup>

<sup>1</sup>Computer Science Institute, Sulaimani Polytechnic University, Sulaimani, Kurdistan Region, Iraq

<sup>2</sup>Department of Computer Science, University of Sulaimani, Sulaimani, Kurdistan Region, Iraq

<sup>3</sup>Department of Computer, Collage of Engineering and Computer Science, Lebanse Frence University, Erbil, Iraq

<sup>4</sup>Department of Computer Engineering University of Sulaimani, Sulaimani, Kurdistan Region, Iraq

<sup>5</sup>Ministry of education, Kirkuk Education Department of Kurdish Studies, Hawazen Preparatory School for Girls, Kirkuk, Iraq

### ARTICLE INFO

Article history:

Received: 24 February, 2024

Revised: 17 May, 2024

Accepted: 18 May, 2024

Online: 22 June, 2024

Keywords:

Hybrid AES-RSA

Hybrid AES-ECC

Hybrid AES- ElGamal

### ABSTRACT

Recently Network safety has become an important or hot topic in the security society (i.e., Encryption and Decryption) developed as a solution of problem that have an important role in the security of information systems (IS). So protected/secure the shared data and information by many methods that require in all internet faciality, data health and the cloud computing that suggestively increased our data every in milliseconds unit. This performance analysis by two factors namely Encryption, Decryption and throughput time of three Hybrid Encryption schemes namely; Hybrid AES-RSA, Hybrid AES-ECC, and Hybrid AES-ElGamal which are based on Encryption and Decryption times by milliseconds unit in the form of throughput. The results evaluation shows clear distinctions schemes capabilities such as; Encryption and Decryption as well as throughput time consume. Nevertheless, the Hybrid AES-RSA emerges as the fastest types. Both encryption and decryption outcome with superior throughput. Hybrid AES-ECC and Hybrid AES-ElGamal results are slower processing times and making them more suitable for scenarios where performance is not the primary concern. The choice between these schemes should consider not only performance but also security requirements and specific application required for testing and realize to select Hybrid AES-RSA due to better performance in milliseconds. The programing language for proposed system is JAVA, this mean that all testing is by JAVA and discover that the Hybrid AES-RSA is better in performance. The security proposed is Hybrid AES-RSA for automated recruitment system is best.

## 1. Introduction

The rapidly growth of information networking technology (ICT) is main principals for common culture interchanging of the data very considerably. Since the huge amount of data transferred over the communication facility, data security modified the issue. The security requires in order to guard such data which communicates on unsecure channel [1]. The modification that occurred from past until now is create a generation in the security and changed to automated security [2] or online security.

Attributable to the development of ICT to necessity reputation of data has directed to the mentioned secure data in several methods. The Insecure information as considered one of the difficulties at the present time with the development of ICT and the use of Internet networks through data correspondence between various sides such as organizations and users. The term security of information refers to a methodology that employed to satisfy the security requirements [3]. The Encryption method is one of the targets that make users to ensure the secrecy and access of the data to the receiving from side without affecting it from any third party. Also, the algorithms of Cryptography prevent third party or public reading of private messages [4]. The working of Encryption and

\* Corresponding Author: Rebwar khalid, Sulaimani Polytechnic University, rebwar.khalid@spu.edu.iq

decryption is shown in Figure 1 [5]. The Figure 1 states that users Encrypts the message using secret key and send it through the communication channels. The Decrypt by Receiver the message using the secret key but Cryptography offers a number of security target to ensure the privacy of data with non-modification of data and so on. the great security advantages of Cryptography that widely in nowadays security [6].

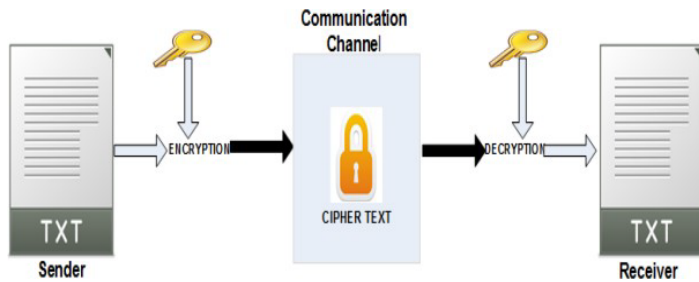


Figure 1: Working of Encryption and Decryption [1]

Cryptography algorithms can be intitled two important categories: Symmetric key Cryptography and Asymmetric key cryptography [7]. Figure 2 shows three types of Cryptography.

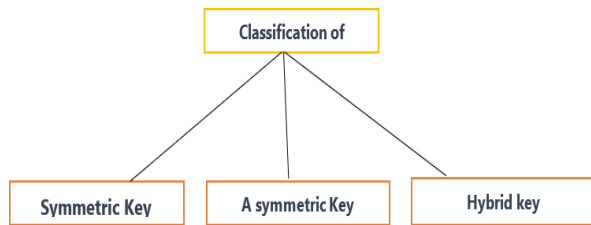


Figure 2: Shows three types of Cryptography [8]

## 2. Classification of Cryptography

### 2.1. Symmetric Key Cryptography

Symmetric algorithm is also called shared key Cryptography. During data transmission, the sender and the receiver share the same key for Encryption and Decryption [5]. The different types of Symmetric algorithms as same as to Data Encryption Standard (DES), Triple Data Encryption Standard (3DES), Advanced Encryption Standard (AES) and Blowfish [8]. Hence, AES is a type of symmetric key cryptography and very powerful and more reliable than the other.

### 2.2. Asymmetric Key Cryptography

Public key Cryptography is another name for the Asymmetric Algorithm with two keys element that namely 'Private key' and 'Public key'. While the data transmission the users encrypts the clear text with the help of public key known as the cipher text and the receiver Decrypts this cipher text with the help of its private key [8]. The different types of asymmetric algorithms are Rivest Shamir Adlemen (RSA), Diffie Hellman (DH) and Digital Signature Algorithm (DSA), ECC [8].

### 2.3. Hybrid Cryptography

The concept which combines the both shared key cryptography and public key Cryptographic techniques create new version of algorithms and mentioned as Hybrid Cryptography. A Hybrid Cryptosystem is a protocol using Symmetric and Asymmetric Cryptographic technics together, each to its best advantage. Hybrid Encryption is considered a highly secure type of encryption as long as the public and private keys are fully secure. Hybrid Cryptography is achieved through data transfer using unique session keys along with Symmetrical Encryption. public key Encryption is implemented for random Symmetric key Encryption. The recipient then uses the public key Encryption method to Decrypt the Symmetric key. Once the Symmetric key is recovered, it is then used to Decrypt the message [9]. The blended OR combined in between Symmetric and Asymmetric Cryptographic techniques and create Hybrid Cryptography with three key. However, the three key Hybrid must be more secure than the two key Asymmetric and one key Symmetric. The Table 1 below show the comparison between three different algorithms in respect of key type namely; Symmetric Cryptography, Asymmetric Cryptography and Hybrid Cryptography.

Table1: Shows the comparison between three different algorithms

Aspect	Symmetric Cryptography	Asymmetric Cryptography	Hybrid Cryptography
Key Type	Uses a single shared key	Uses a key pair (public and private keys)	Uses both shared and key pairs
Key Distribution	Securely sharing a single key can be challenging	Public keys can be freely distributed, private keys must be kept secret	Securely share a symmetric key, then use asymmetric encryption for key distribution
Speed	Generally, faster for encryption/decryption	Slower compared to symmetric encryption	Slower than symmetric but faster than pure asymmetric
Security	Vulnerable if the key is compromised	More secure due to the key pair, but still vulnerable if private key is compromised	Combines the strengths of both symmetric and asymmetric encryption for improved security

## 3. Related Work

In [7], a comparative theory of techniques Encryption in terms of (Symmetric and Asymmetric) keys to algorithms analyzed. In the Symmetric key Encryption (AES) algorithm is found to be better in terms of cost, security and implementation.

Nevertheless, the Asymmetric key Encryption (RSA) algorithm is much better in terms of speed and security. Nevertheless, in paper [7] show and discover that the two algorithms namely AES and RSA are good individually in term of the security and cost but didn't mention Hybrid techniques OR combined technique simultaneously. The AES with RSE are combined in our proposed system and came out with better results than before Paper [10] by Verma, P. Guha, and S. Mishra that comparative study of different key algorithms namely: AES, DES, 3DES, Blowfish and RSA are analyzed and compared with the results that found among the symmetric encryption algorithm, AES and Blowfish are the most secure and efficient algorithms. The performance and energy consumption of these algorithms are better compared to the others. In case of Asymmetric Encryption algorithm, RSA is secure and can be used for application in wireless network because of its good speed and security. However, in paper [10] show and compare between: AES, DES, 3DES, Blowfish and RSA algorithms individually but didn't mention Hybrid techniques OR combined technique simultaneously working between mentioned techniques. The AES, DES, 3DES, Blowfish and RSA algorithms are NOT combined and results came out individually.

In paper [11], ECC is explained in detail. Elliptic curve Cryptography (ECC) is a relatively newer form of public key Cryptography that provides more security per bit than other forms of Cryptography still being used today .

In paper [12], Encryption and Decryption of text using ECC is explained with mapping technique. It is concluded that ECC has low power consumption, less memory requirement, small key size and high security.

In paper [13], Hybrid Cryptography technique using AES and ECC is proposed. The system is intended to provide security to a variety of multimedia data ranging from text documents, images, audio, video. Proposed Hybrid system capable of Encrypting and Decrypting the sensitive data to protect it from unauthorized access and attacks.

In paper [14], Hybrid Cryptography approach implemented using AES and ECC. This system provides encryption to the multimedia data such as text, image, audio, video which resulted in an output with 100 percent accuracy without any loss of information.

In paper [15], Different text files are taken as input and encrypted using AES-ECC Hybrid approach. Analysis of AES Encryption with ECC is done on the basis of different parameters like storage requirement, Encryption time, Decryption time.

In paper [16], Hybrid approach for Encryption technique is implemented over a binary image, which provides more accuracy to the encryption process. The ECC and AES are combined in such a way that differentiates them from the usual manner of

Encryption. These days with the increasing trend of security it becomes essential to protect the data and information in a better way.

#### 4. Problem Definition

Cryptographic techniques provides the secure data transmission in automated performance analysis of E-services by AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC which be a target for our proposed system. The complications existing systems such as a Cryptographic technique with time consume processes. Nevertheless, exactly techniques that isn't includes the reliability checks on transmitted data and another issue in key changing the presence of security. Thus, the implement an operational Cryptographic algorithm in all phases must be well thought-out to sort a strong method (i.e. AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC) with complications in to the proposed method. The proposed Hybrid Cryptographic technique which uses the best features of Symmetric AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC. The designed technique helps to reduce the time complexity in Encryption-time, Decryption-time and Throughput-time. This paper solves the performance in time consume also select fewer time response by throughput-time. Finally, our proposed algorithms AES-Based Hybrid Cryptosystems with RSA.

#### 5. Description of Algorithms

Advanced Encryption Standard (AES), Rivest Shamir Adleman (RSA), Elliptic Curve Cryptography (ECC), and ElGamal are indeed powerful cryptographic procedures and use for different purposes and have different features. The AES is a Symmetric-key algorithm widely used for secure facts encryption while RSA operates as an Asymmetric-key algorithm for secure communication and digital signatures. ECC, another asymmetric-key algorithm, relies on elliptic curves for enhanced security with shorter key lengths. ElGamal, also asymmetric are utilized for public-key Encryption and digital signatures based on the discrete logarithm problem. In below explain in detail about all types of Algorithms.

##### 5.1. Advanced Encryption Standard (AES)

The AES algorithm are operating as a Symmetric block cipher system that employs a replace or exchange network. The data block length and key length in AES can be adjusted based on specific requirements, with key lengths available in 128, 192, and 256 bits. The iteration cycle numbers for these round keys are 10, 12, and 14 rounds, respectively. The AES algorithm primarily comprises three components: round change, turns, and key expansion. Each round transformation consists of a non-linear layer, a linear mixture layer, and an add round key layer. The

Encryption process of AES is illustrated in Figure 3. Also, there are three different key lengths with round key iteration.

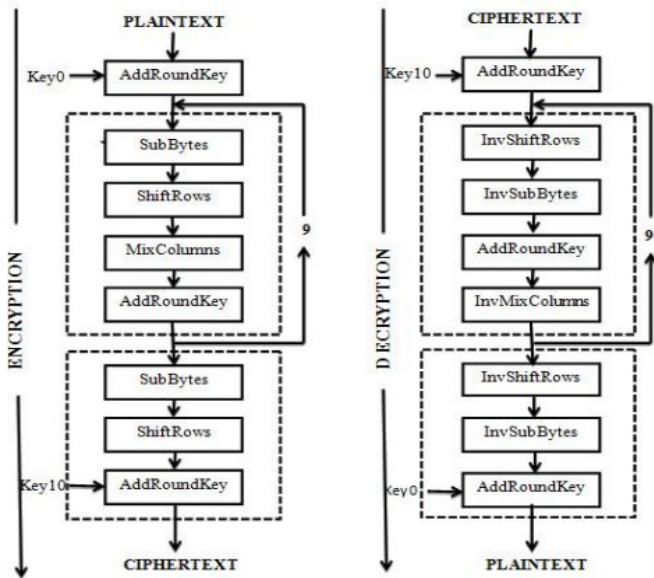


Figure 3: AES Encryption & Decryption process [17]

### 5.2. Elliptic Curve Cryptography (ECC)

In order to compare the public key cryptography that is comparatively more established by an elliptic curve cryptography (ECC) because offers more security per bit than other Cryptographic methods that are still in use nowadays technologies. Nevertheless the Mathematically an elliptic curves are cubic curves that are equivalent to tori topologically. Despite their name which is not closely related to the ellipse. The name of elliptic is integral. By Weierstrass normal equation form. The basic general elliptic curve used for cryptography is of the equation form

$$y^2 = x^3 + ax + b \tag{1}$$

which showed in Figure 4. The Curves of this form are defined by different values for  $a$  and  $b$  and modifying these values to visualize of the curve and expand, contract, or pinch off to be two separate pieces. Practically the Curves used for Cryptography to defined very large integer values for  $aa$  and  $bb$  respectively. The modification of  $a$  and  $b$  in the mentioned equation is modify the elliptic curve visualization. This mean that is a direct relation between  $a$  and  $b$  values and elliptic curve.

### 5.3. RSA Algorithm

The invented of the most widely Asymmetric key cryptosystem known as RSA algorithm is very powerful and user-friendly algorithms that Encryption and authentication Cryptosystem used since that time in many Cryptographic applications for instance an e-mail security, banking, e-commerce and digital signature over web operating systems over the internet facility.

The main security algorithm depends on the difficulty of finding prime numbers factor of large integers. RSA operation consists of three main stages key generation, Encryption and Decryption processes which are explained briefly in following [18].

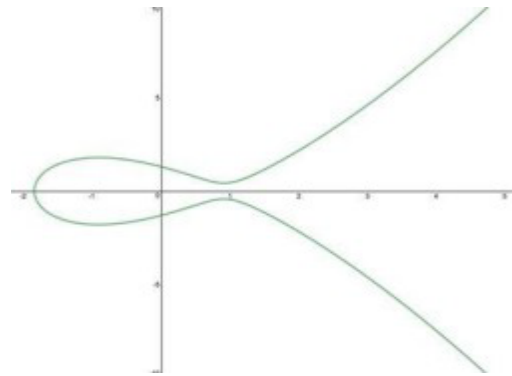


Figure 4: Simple elliptic curve visualization [11]

#### 5.3.1 Key Generation

- Select two large prime integer numbers  $p$  and  $q$  to calculate the modulus  $n$  using the formula  $n = p \times q$ .
- Calculate  $\phi$  using the formula  $\phi(n) = (p - 1) \times (q - 1)$ .
- Select an integer  $e$  which is the public exponent, such that  $\text{GCD}(e, \phi(n)) = 1$ ,
- where  $\text{GCD}$  refers to greatest common divisor function between two numbers.
- Calculate  $d$  which is the private exponent, such that  $e \times d = 1 \text{ mod } \phi(n)$ , where  $\text{mod}$  symbolizes to the modulus operation or the remainder after division.

Hence,  $(n, e)$  represents the public Encryption key, while  $(n, d)$  represents the private decryption key [18]. The two prime numbers are the target of the algorithms.

#### 5.3.2 Encryption/Decryption Processes

Let  $m$  be a message that wanted to be encrypted, then the encrypted message  $c$  is calculated via the public key  $(n, e)$  using the equation:  $c = m^e \text{ mod } n$ . To extract the original message  $m$ , the received encrypted message  $c$  is decrypted via the private key  $(n, d)$  using the equation:  $m = c^d \text{ mod } n$  [18].

### 5.4. ElGamal algorithm

Security mentioned algorithm depends over hard to calculating discrete logarithms of large prime numbers. If the same plaintext is encrypted using this cryptosystem, then a different cipher text is obtained in each time of Encryption. El-Gamal operation can be described in 3 main steps:

- 1) key generation, 2) Encryption and 3) Decryption processes which are explained briefly as follows.

### 5.4.1 Key Generation

- First, select a random prime number  $p$  and two other random numbers  $x$  and  $g$ , such that both of them are less than  $p$ .
- Calculate  $y$  using the formula:  $y = gx \text{ mod } p$ .
- Thus,  $(p, g, y)$  represents the public key which can be shared between a group of users, while  $x$  represents the private key which should be kept secret [19].

### 5.4.2 Encryption/Decryption Processes

In order to Encrypt a message  $m$ , firstly, a random integer number  $k$  is selected, such that  $k$  is relatively prime with  $(p - 1)$ . Secondly, the cipher text pairs  $(c1, c2)$  is calculated using the equations:  $c1 = g^k \text{ mod } p$  and  $c2 = (y^k \times m) \text{ mod } p$ . Finally, the cipher text  $(c1, c2)$  is transmitted to the recipient. To Decrypt the cipher text, pair  $(c1, c2)$ , the private key  $x$  is employed to recover the original message  $m$  using the equation:  $m = \frac{c2}{c1^x} \text{ mod } p$  [19].

The following pseud codes for ElGamal Algorithms namely Key Generation, Encryption and Decryption. Figure 5: Pseudo-code key generation stage: encryption and decryption.

#### ➤ Encryption Stage

```

ElGamal_Encryption (e1, e2, p, P) // P is the plaintext
{
    Select a random integer r in the group G = <Zp*, x>
    C1 ← e1^r mod p
    C2 ← (P × e2^r) mod p // C1 and C2 are the ciphertexts
    return C1 and C2
}
    
```

#### ➤ Decryption Stage

```

ElGamal_Decryption (d, p, C1, C2) // C1 and C2 are the ciphertexts
{
    P ← [C2 (C1^d)^-1] mod p // P is the plaintext
    return P
}
    
```

#### ➤ Key Generation Stage

```

ElGamal_Key_Generation
{
    Select a large prime p
    Select d to be a member of the group G = <Zp*, x> such that 1 ≤ d ≤ p - 2
    Select e1 to be a primitive root in the group G = <Zp*, x>
    e2 ← e1^d mod p
    Public_key ← (e1, e2, p) // To be announced publicly
    Private_key ← d // To be kept secret
    return Public_key and Private_key
}
    
```

Figure 5: Pseudo-code key generation stage: encryption and decryption

## 6. Methodology

Methodology of our proposed system is by Hybrid /blended

of two algorithms namely: (i.e. AES with RSA) and (i.e. AES with ECC), also (i.e. AES with ElGamal). The enhancement of security performance for developing and comparing an excellent result in milliseconds unit by Java programming language. Hence, there are an indirect relation between the performance-security and time- consume in milliseconds unit. The Figure 6 show indirect relation between performance-security and time-consume. The creation of web Ecommerce and secured by one of the blended algorithms as a methodology and protect the website. As mentioned, that before the protection of website should be by one of the algorithms.

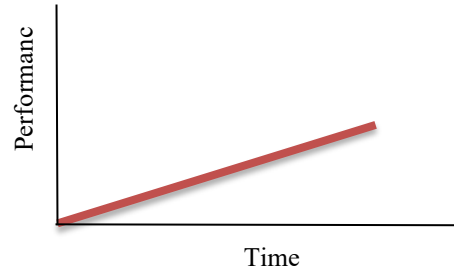


Figure 6: Time

Nevertheless, the performance analysis is an important part of our proposed system by Hybrid techniques between two algorithms and show that the best performance with less time unit and explained in Table below. The Table 2- show all file types with Hybrid algorithms of AES-RSA performance analysis. The main purpose of our proposed system is mixing two algorithms and test the six types of files such as: jpg, txt, pdf, pptx, mp3, mp4 with different file size over Hybrid AES-RSA. The Encryption time and throughput of all types that show it in Table 2 by blended AES and RSA

Table 2: Hybrid AES-RSA Encryption Performance Comparison: Encryption Time and Throughput

File Type	Text size (In Kb)	Encryption Time	Encryption Throughput
JPG	9328	120	71746.15
TXT	3765	55	68436.36
PDF	1136	25	45400.0
PPTX	1743	31	56193.54
MP3	57531	709	81142.45
MP4	4555	60	75900.0
Average Time		166.66	66469.75

The Hybrid techniques between two algorithms and show that the best performance with less time unit and explained in Table below. The Table 3 show all file types with Hybrid algorithms of AES-RSA performance analysis. The main purpose of our proposed system is mixing two algorithms and test the six types of files such as: jpg, txt, pdf, pptx, mp3, mp4 with different file size over Hybrid AES-RSA.

The decryption time and throughput of all types that show it in Table 3 by blended AES and RSA.

Table 3: Hybrid AES-RSA Decryption Performance Comparison: Decryption Time and Throughput

File Type	Text size (In Kb)	Decryption Time	Decryption Throughput
JPG	9328	179	52106.14
TXT	3765	75	50186.66
PDF	1136	50	22700.0
PPTX	1743	47	37063.82
MP3	57531	722	79681.44
MP4	4555	99	46000.0
Average Time		195.33	47956.34

The Hybrid techniques between two algorithms and show that the best performance with less time unit and explained in Table below. The Table 4 show all file types with Hybrid algorithms of AES- ECC performance analysis. The main purpose of our proposed system is mixing two algorithms and test the six types of files such as: jpg, txt, pdf, pptx, mp3, mp4 with different file size over Hybrid AES- ECC.

The Encryption time and throughput of all types that show it in Table 4 by blended AES and ECC.

Table 4: Hybrid AES- ECC Encryption Performance Comparison: Encryption Time and Throughput

File Type	Text size (In Kb)	Encryption Time	Encryption Throughput
JPG	9328	735	12689.79
TXT	3765	608	6190.78
PDF	1136	585	1940.17
PPTX	1743	595	2927.73
MP3	57531	1570	36643.31
MP4	4555	648	7027.77
Average Time		790.166	11236.59

The Hybrid techniques between two algorithms and show that the best performance with less time unit and explained in Table below. The Table 5 show all file types with Hybrid algorithms of AES- ECC performance analysis. The main purpose of our proposed system is mixing two algorithms and test the six types of files such as: jpg, txt, pdf, pptx, mp3, mp4 with different file size over Hybrid AES- ECC.

The Decryption time and throughput of all types that show it in Table 5 by blended AES and ECC

Table 5: Hybrid AES- ECC Decryption Performance Comparison: Decryption Time and Throughput

File Type	Text size (In Kb)	Encryption Time	Encryption Throughput
JPG	9328	186	50145.16
TXT	3765	100	37640.0
PDF	1136	65	17461.53
PPTX	1743	69	25246.37
MP3	57531	802	71733.16
MP4	4555	117	38923.07
Average Time		790.166	11236.59

The Hybrid techniques between two algorithms and show that the best performance with less time unit and explained in Table below. The Table 3 show all file types with Hybrid algorithms of AES- ElGamal performance analysis. The main purpose of our proposed system is mixing two algorithms and test the six types of files such as: jpg, txt, pdf, pptx, mp3, mp4 with different file size over Hybrid AES- ElGamal.

The Encryption time and throughput of all types that show it in Table 6 by blended AES and ElGamal.

Table 6: Hybrid AES- ElGamal Encryption Performance Comparison: Encryption Time and Throughput

File Type	Text size (In Kb)	Encryption Time	Encryption Throughput
JPG	9328	877	10635.11
TXT	3765	788	4776.64
PDF	1136	769	1475.942
PPTX	1743	763	2283.09
MP3	57531	1787	32193.62
MP4	4555	805	5657.14
Average Time		964.83	9503.59

The Hybrid techniques between two algorithms and show that the best performance with less time unit and explained in Table below. The Table 7 show all file types with Hybrid algorithms of AES- ElGamal performance analysis. The main purpose of our proposed system is mixing two algorithms and test the six types of files such as: jpg, txt, pdf, pptx, mp3, mp4 with different file size over Hybrid AES- ElGamal.

The Decryption time and throughput of all types that show it in Table 7 by blended AES and ElGamal.

Table 7: Hybrid AES- ElGamal Decryption Performance Comparison: Decryption Time and Throughput

File Type	Text size (In Kb)	Decryption Time	Decryption Throughput
JPG	9328	300	31090.0
TXT	3765	150	25093.33
PDF	1136	115	9869.56
PPTX	1743	127	13716.53
MP3	57531	1170	49170.94
MP4	4555	174	26172.41
Average Time		339.33	25852.12

In order to achieve more information from previous comparability between three performance which are: Hybrid algorithm of (AES-RSA, AES-ECC, AES- ElGamal). The Table below shows performance analysis for several Hybrid Encryption algorithm, in respect of the average time and throughput. encryption time and Encryption throughput are two parameters that complete and been blended together. In the context of the

presented data information. The three Hybrid /OR blended algorithms are explained below:

- Hybrid AES-RSA: The result blended AES-RSA Encryption algorithms by average time is 166.66 milliseconds. Nevertheless, an encryption time of throughput of blended AES-RSA is 66469.75 milliseconds. The encryption throughput for this Hybrid approach is much higher and more notable. this combination between two algorithms namely Hybrid AES-RSA, thus the Encryption Time and Encryption Throughput is much less than the other blend algorithms (Hybrid AES-ECC and Hybrid AES- ElGamal). However, the Hybrid AES-RSA is outcome is much better and fewer than the others blended algorithms, which are; Hybrid AES-ECC and Hybrid AES- ElGamal. All result is show in Table 8.

Table 8: Average Time Encryption and Throughput of AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC in milliseconds

Average Time	Encryption Time	Encryption Throughput
Hybrid AES-RSA	166.66	66469.75
Hybrid AES-ECC	790.166	11236.59
Hybrid AES-ElGamal	964.83	9503.59

In order to achieve more information from previous comparability between three performance which are: Hybrid algorithm of (AES-RSA, AES-ECC, AES- ElGamal). The Table below shows performance analysis for several Hybrid decryption algorithm, in respect of the average time and throughput. decryption time and decryption throughput are two parameters that complete and been blended together. In the context of the presented data information. The three Hybrid /OR blended algorithms are explained below:

- Hybrid AES-RSA: The result blended AES-RSA decryption algorithms by average time is 195.33 milliseconds. Nevertheless, an Encryption time of throughput of blended AES-RSA is 47956.34 milliseconds. The decryption throughput for this Hybrid approach is much higher and more notable. this combination between two algorithms namely Hybrid AES-RSA, thus the Decryption Time and Decryption Throughput is much less than the other blend algorithms (Hybrid AES-ECC and Hybrid AES- ElGamal). However, the Hybrid AES-RSA is outcome is much better and fewer than the others blended algorithms, which are; Hybrid AES-ECC and Hybrid AES- ElGamal. All result is show in Table 9.

Table 9: Average Time Decryption and Throughput of AES-Based Hybrid Cryptosystems with RSA, ElGamal, and ECC in milliseconds

Average Time	Decryption Time	Decryption Throughput
Hybrid AES-RSA	195.33	47956.34

Hybrid AES-ECC	223.16	40191.54
Hybrid AES-ElGamal	339.33	25852.12

However, the two pi-chart below give more information regarding our proposed system average Time Encryption and throughput of AES-Based Hybrid Cryptosystems with RSA, ElGamal, ECC. the Figure 7 show the best performance Hybrid AES-RSA over (Hybrid AES-ECC and Hybrid AES- ElGamal).

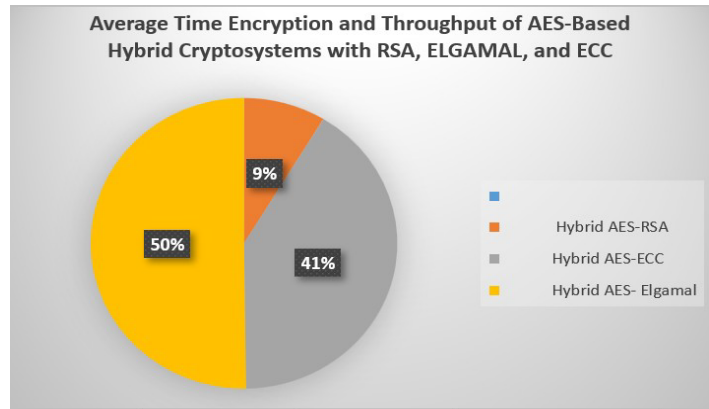


Figure 7: Average Time Encryption and Throughput (Hybrid AES-ECC and Hybrid AES- ElGamal)

The Hybrid AES-RSA Encryption method demonstrates an average Decryption time of 195.33 milliseconds, achieving a throughput of 47956.34 decrypts per second. In comparison, Hybrid AES-ECC exhibits a slightly longer decryption time of 223.16 milliseconds with a throughput of 40191.54 decrypts per second. Hybrid AES- ElGamal, on the other hand, has a higher Decryption time of 339.33 milliseconds and a throughput of 25852.12 Decrypts per second. which Mentioned and explained clearly in Table 9, by of information in Table 9 created a pie-chart Figure 8.

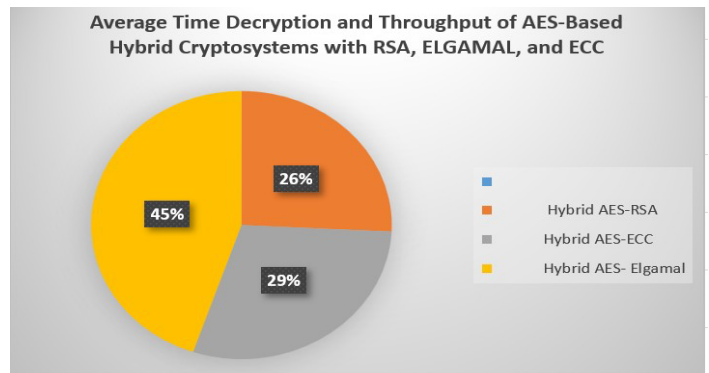


Figure 8: Average Time Decryption and Throughput (Hybrid AES-ECC and Hybrid AES- ElGamal)

## 7. Conclusion

In conclusion, the data analysis declared that the performance variations among the three Hybrid encryption methods namely



Hybrid AES-RSA, Hybrid AES-ECC, and Hybrid AES- ElGamal. The Hybrid AES-RSA proves superior speed in both schemes' encryption and decryption processes. This mean that making its an optimal choice for applications requiring swift data protection. Furthermore, it achieved the highest throughput for encryption and decryption and underlining its efficiency. Nevertheless, Hybrid AES-ECC and Hybrid AES-ElGamal exhibit slower encryption and decryption time-consume that is why making them less suitable for time-sensitive tasks. Although both are offer certain cryptographic advantages to reduced throughput may limit their applicability.

Eventually, the selection of an encryption scheme should be based on a careful consideration of security requirements and performance requirements. The speed is important for Hybrid AES-RSA stands out as the most favorable option, while Hybrid AES-ECC and Hybrid AES- ElGamal may be more appropriate in situations where performance is less critical, and specific security features are prioritized.

## References

- [1] P. Patil and R. Bansode, "Performance Evaluation of Hybrid Cryptography Algorithm for Secure Sharing of Text & Images," *International Research Journal of Engineering and Technology* 2020
- [2] K. H. A. Faraj, A. B. Kanbar, J. Gul-Mohammed, W. M. Hmeed, and S. F. Karim, "Cloud Computing Loading Time Over Different Operating Systems," *Science Journal of University of Zakho*, **8**(4):154–159, 2020, DOI: <https://doi.org/10.25271/sjuoz.2020.8.4.756>
- [3] Z. C. Oleiwi, W. A. Alawsi, W. C. Alisawi, A. S. Alfoudi, and L. H. Alfarhani, "Overview and Performance Analysis of Encryption Algorithms," *J. Phys. Conf. Ser.*, **1664**(1), 2020, DOI: <https://doi.org/10.1088/1742-6596/1664/1/012051>
- [4] P. Chinnasamy, S. Padmavathi, R. Swathy, and S. Rakesh, "Efficient Data Security Using Hybrid Cryptography on Cloud Computing," *Lect. Notes Networks Syst.*, **145**(September):537–547, 2021, DOI: [https://doi.org/10.1007/978-981-15-7345-3\\_46](https://doi.org/10.1007/978-981-15-7345-3_46)
- [5] F. Maqsood, M. Ahmed, M. M. Ali, and M. A. Shah, "Cryptography: A Comparative Analysis for Modern Techniques," *Int. J. Adv. Comput. Sci. Appl.*, **8**(6):442-448, 2017, DOI: <https://doi.org/10.14569/IJACSA.2017.080659>
- [6] P. Verma, J. Shekhar, P. Preety, and A. Asthana, "A Survey for Performance Analysis Various Cryptography Techniques Digital Contents," *International Journal of Computer Science and Mobile Computing*, **4**(1):522–531, 2015
- [7] N. Bisht and S. Singh, "A Comparative Study of Some Symmetric and Asymmetric Key Cryptography Algorithms," *International Journal of Innovative Research in Science, Engineering and Technology*, **4**(3):1028-1031, 2015, DOI: <https://doi.org/10.15680/IJRSET.2015.0403043>
- [8] S. Chandra and S. Paira, "A Comparative Survey of Symmetric and Asymmetric Key Cryptography," in 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE), 2014, 1-4, DOI: <https://doi.org/10.1109/ICECCE.2014.7086640>
- [9] P. Kuppaswamy and S. Q. Y. Al-Khalidi, "Hybrid Encryption/Decryption Technique Using New Public Key and Symmetric Key Algorithm," *Manage. Inf. Syst. Rev.*, **19**(2):1-13, 2014, DOI: <https://doi.org/10.6131/MISR.2014.1902.01>
- [10] P. Verma, P. Guha, and S. Mishra, "Comparative Study of Different Cryptographic Algorithms," *Int. J. Emerg. Trends Technol. Comput. Sci.*, **5**(2):58-63, 2016
- [11] R. Harkanson and Y. Kim, "Applications of Elliptic Curve Cryptography: A Light Introduction to Elliptic Curves and a Survey of Their Applications," in 12th Annual Cyber and Information Security Research Conference, USA, 2017, 1-7
- [12] K. Keerthi and B. Surendiran, "Elliptic Curve Cryptography for Secured Text Encryption," in International Conference on Circuits Power and Computing Technologies, India, 2017
- [13] S. C. Iyer, R. R. Sedamkar, and S. Gupta, "A Novel Idea on Multimedia Encryption Using Hybrid Crypto Approach," in 7th International Conference on Communication, Computing and Virtualization, 2016, 79:293-298
- [14] S. C. Iyer, R. R. Sedamkar, and S. Gupta, "An Efficient Multimedia Encryption Using Hybrid Crypto Approaches," *Int. J. Recent Trends Eng. Res.*, **2**:442-452, 2016
- [15] S. Sharma and V. Chopra, "Analysis of AES Encryption with ECC," in Proceedings of International Interdisciplinary Conference on Engineering Science & Management, 2016, 195
- [16] D. Ameta and S. Upadhyay, "A Hybrid Approach for Image Encryption Using Different Number Iterations in ECC and AES Techniques," *Int. J. Comput. Appl.*, **175**(3), 2017, DOI: <https://doi.org/10.5120/ijca2017915469>
- [17] N. Mathura and R. Bansode, "AES Based Text Encryption Using 12 Rounds with Dynamic Key Selection," in Proc. 7th Int. Conf. Commun., Comput., Virtualization, 2016, 131-135, DOI: <https://doi.org/10.1016/j.procs.2016.03.131>
- [18] F. Yousif, "Encryption and Decryption of Audio Signal Based on RSA Algorithm," *Int. J. Eng. Technol. Manage. Res.*, **5**(7):259-264, 2018, DOI: <https://doi.org/10.29121/ijetmr.v5.i7.2018.259>
- [19] O. A. Imran, S. F. Yousif, I. S. Hameed, W. N. Al-Din Abed, and A. T. Hammid, "Implementation of El-Gamal Algorithm for Speech Signals Encryption and Decryption," *Procedia Comput. Sci.*, **167**:1028–1037, 2020, DOI: <https://doi.org/10.1016/j.procs.2020.03.402>

**Copyright:** This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).