University of Sindh Journal of Information and Communication Technology (USJICT)
Volume 9, Issue 1, 2025
ISSN-E: 2523-1235, ISSN-P: 2521-5582
Website: http://sujo.usindh.edu.pk/index.php/USJICT/

Evaluating the Robustness of RNN Models on Diverse Sequential Datasets

Imra Shoukat, Mujeeb Ur Rehman

[1,2]Department of Computer Science, University of Management and Technology, Sialkot, 51040, Pakistan
imrashoukat7@gmail.com, mujeeb.rehman@skt.umt.edu.pk

Abstract: Recurrent Neural Networks are deep neutral networks that handle sequence data due to their ability to capture and replacement sequential dependencies in the data. This characteristic makes them ideal for applications such as natural language processing, speech recognition, and time series prediction. In this research paper RNN methods are applied and analyzed using the UCI Student Performance dataset and the UCI Human Activity Recognition (HAR) dataset. It studies the effect of hyper parameter optimization and data set balancing on critical performance system of measurement such as model accuracy, memory usage, and time taken for prediction. By altering critical features, the number of RNNs, the dropout rate, and the batch size, the effect of these parameters on model performance is analyzed. The results show that changing the number of RNN units and batch size optimization improves model performance, as well as the effect of computational efficiency. The RNN test produces 99.8% accuracy in the UCI Student Performance data set and 95.11% in the HAR data set. In adding, balanced datasets help to improve generalization by reducing bias and over fitting. A train-test split of 70%-30% is testified to provide the best overall precision compared to computational resource costs. This research tells us how hyper parameter optimization and data handling modify and improve the performance of RNN models. Class balance was achieved using synthetic oversampling techniques based on SMOTE. This approach encouraged uniform distribution through all classes.

Keywords: RNN; Sequence Modeling; Hyper parameter tuning; Dataset Balancing; Test-Train Split; Performance Evaluation.

I. INTRODUCTION

Machine Learning are specially design to solve complex problems automatically in various industries. Among numerous neural networks architectures, Recurrent Neural Networks (RNNs) are the fastest at sequential data processing because of their information retain over time. Because of this memory feature, RNNs are considered to be one of the most effective models in speech recognition, time-series prediction, and natural language processing. Unfortunately, standard RNNs have problems with capturing long-term dependencies because of the vanishing gradient problem. To address this issue, advanced versions with higher memory capacity and better performance include Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU) [5].

Although RNNs usually have a good performance in many applications, one of the more general research problems is the optimization of their performance for real-application datasets. Particularly, there is a need for understanding how to tune hyper parameters to optimize model accuracy and computational resources. In this Research Paper, RNN models are applied to two varied datasets: the UCI Student Performance dataset and the UCI Human Activity Recognition (HAR) dataset. These datasets capture two different real-world situations: predicting academic performance and classifying human activities. The analysis conducted upon these datasets aims to propose hyper parameter configurations that give the best performance and to study how data spreading actually affects the model performance.

Recurrent Neural Networks (RNNs) are used in various fields due to their ability to process sequential data effectively

A. Natural Language Processing (NLP)

RNNs are play central role to language modeling, machine translation, and sentiment analysis by learning the related relationship between words [11].

B. Speech Recognition

RNN models, especially LSTM networks, have demonstrated improved speech-to-text conversion accuracy by better managing sequential audio data [12].

C. Time-Series Forecasting

RNNs find widespread application to prediction future patterns from past data, hence being well-suited for applications like prediction of forecasting stock price variations and weather conditions [13].

D. Image Captioning

Through the use of the combination of RNNs and convolutional neural networks (CNNs), the models are able to automatically produce descriptive

captions for images by utilizing the advantages of the two architectures [14].

This Research Paper provides a comprehensive analysis of how hyper parameter optimization and data supply on RNN performance. Thought systematic experimentation, we analyze how important parameters affect model accuracy, memory consumption, and prediction time; these parameters are the number of RNN units, batch size, and dropout rate. Moreover, we use balanced and unbalanced data to predict how it impacts model development and the effect variance of training-testing splits has on computational complexity.

The contributions of this paper are then summarized are as follows: (1) We train and compare RNN models on two very different datasets: the UCI Student Performance dataset and the UCI Human Activity Recognition (HAR) dataset, (2) We exhaustively tune important hyper parameters (RNN units, batch size, dropout rate) and analyze their impact on accuracy, memory usage, and prediction latency, (3) We explore the effect of data on distribution by conducting an experiment comparing the balanced dataset vs. unbalanced dataset (4) We detect a 70%-30% training-testing split as the best set-up to maximize predictive accuracy while minimizing computational resources , and (5) We provide a complete performance analysis, reporting system of measurement such as accuracy, precision, recall, F1-score, and ROC-AUC under different experimental conditions.

This study has several major contributions. It widely evaluates the performance of different RNN models on two different sequential datasets and analyses the impact that data spreading–balanced against unbalanced–has on model accuracy, as well as how different train-test split methods affect generalization performance. Also examined are the critical hyper parameters such as dropout rate, batch size, and number of units in RNNs which, when enhanced, significantly improve prediction accuracy and reduce computational demands.

The structure of this paper is outlined as follows

In Section II we review related work on RNNs, hyper parameter tuning and the effect of data distribution. Section III describes the datasets and experimental different models, as well as model architecture and experimental process. Section IV provides the results and a thorough analysis of the model performance. Section V presents the main findings, and Section VI concludes the paper while suggesting directions for future research.

II. Related Work

This section explains existing research techniques on RNN models have shown as ideal solutions for processing sequential input and are widely used across various models. The following section deep dives into three areas of interest:

- RNN architecture and its systems hierarchy.
- How hyper parameter tuning affects the model performance.
- The implications of data balancing in predictive accuracy results.

RNNs are neutral networks that are especially used to handle sequential data. Nevertheless, traditional RNNs are not good at handling long-term dependencies due to the vanishing gradient problem [2]. This problem has been addressed by advanced architectures in the form of Long Short-Term Memory networks and Gated Recurrent Units networks, which improve to memory [4,6]. These advancements methods make LSTMs and GRUs extremely well-suited for techniques like time-series forecasting and natural language processing [5].

Hyper parameter tuning is extremely important to improve RNN performance. Parameters such as the number of elements in RNN, learning rate, dropout rate, and batch size are primary parameters, which effect the capability of the model to generalize to new data. Enhance of these variables leads to improved accuracy and reduce over fitting. For illustration, adding additional units will enhance pattern matching but will come at a computational cost [7], whereas adding dropout avoids over fitting as it standardizes the model performance [8].

Balanced datasets are also important for exact RNN prediction. Imbalanced datasets make model cause biased outputs and do not simplify well [3]. Oversampling, under sampling, and data augmentation are some of the methods that remove this problem by giving a more balanced opinion of all modules. Research validates that models trained on balanced datasets are more likely to generate more exact and more robust outcomes.

Instantly, RNN performance trusts on architectural enhancements as well as data training methods. This research improves on previous work by systematically examining the impact of hyper parameter optimization and data balancing across many application areas [1].

Recurrent Neural Networks have played a very important role in order to expand machine learning approaches to sequential data processing. By having the ability to hold previous values, they can keep track of temporal dependencies and are particularly significant for usage such as language modeling and time series analysis. Unfortunately, initial versions of RNNs were quite delayed by the vanishing and exploding gradient problems, warning their performance at learning long-term dependencies.

In deciding such issues, Long Short-Term Memory networks came into play as an advance solution. LSTMs employ gate-based mechanisms to control information passing and effectively model long-term memory dependencies. On the same note, Gated Recurrent Units (GRUs) came up as an efficient alternative to LSTMs with correspondingly high performance at lower computational costs [10].

The versatility of RNNs has received them their popularity in various fields. For natural language processing, they have been applied for machine translation as well as for sentimentality analysis. For speech recognition, the RNN supports in the modelling of sequential addictions of audio patterns. Moreover, in time series prediction, they analyze

historical information to forecast forthcoming trends, and thus they specify their ability in identifying temporal structures. Traditional developments have added consideration mechanisms to RNNs, giving them the strength of focusing on prominent parts of input sequences and hence enhancing performance in such tasks as machine translation. More, hybrid approaches based on combinations of RNNs and CNNs have come forward, copying the best attributes from both architectural models for application areas like image captioning and video analysis [9].

The DERNN-LNLT model enhances hyper spectral image reconstruction in CASSI systems by addressing real-world problems like sensing inaccuracies and noise which are generally neglected by DUNs. To address these challenges, the model adds a Degradation Estimation Network (DEN) that improves imaging and combines it with the Local and Non-Local Transformer (LNLT) to efficiently learn spatial and spectral dependencies at low computation cost. Furthermore, re-centering the DUN into an RNN structure where parameters are shared across discrete stages allows improvement on learning efficiency as well as hyper parameter economy while achieving state-of-the-art performance on both synthetic and real datasets [21].

While the effectiveness of ML algorithms has been a major emphasis for researchers, data preprocessing is often ignored as it is done separately from model training. This separation ignores the critical feedback loop between data quality and model performance within the framework of unequal or noisy patterns. To address this issue, RW-ML proposes a novel solution which refines the data within a learning algorithm outlining machine learning subdivision. In RW-LSTM, preprocessing is incorporated with recurrent structures and strengthens models' accuracy as opposed to RO-LSTM and Informer architectures. Building on this, Corrector LSTM or cLSTM adjusts cell states to alter complexities in temporal patterns improving the learning process [22]. Contemporary variants of RNNs—including GRU, Bidirectional RNNs, Attention-enhanced RNNs, and XLSTM—added due to the gradient stability and training effectiveness issues have also enriched the structure [23].

By using affordable EEG devices such as NeuroSky and Brainlink, this research integrates the two disciplines of neuroscience and artificial intelligence by creating models that predict certain cognitive states: attention and meditation. This work aims at forecasting these mental states for real-time brain-computer interface (BCI) applications, utilizing recurrent neural networks (RNNs) such as long short-term memory (LSTM) and gated recurrent units (GRU) models which are competent in sequential data processing. Results indicate that the LSTM model is the best performer for meditation prediction with an RMSE of 10.90 while GRU yielded greater accuracy for attention prediction with RMSE of 11.79 outperforming eSense proprietary algorithm in both cases. Moreover, both models have inference time's less than 50 milliseconds which makes them ideal for interactive systems including robotic control. Thus far, this research shows that inexpensive EEG hardware can be used to detect cognitive states in real time with high accuracy and lays groundwork towards RNN-based BCI technology advancements [24].

Energy forecasting is an important component of manage stable aquaporin systems that rely on photovoltaic (PV) energy sources. In this study we develop an integrated technique using Long Short-Term Memory Recurrent Neural Networks (LSTM-RNN) and Random Forest (RF) to forecast meaningful measures such as voltage, current, power, and solar irradiance. The LSTM development, can express sequential data and demonstrates sequences, while the RF method can select the relevant characteristics as well involve non-linear relationships. The integrated approach was able to provide a RMSE of 0.0363 on the irradiance and is more accurate than either LSTM or RF models alone. The development has demonstrated power and provides reliable energy and is well suited for greater application to forecast renewable energy [25].

Generally, while RNNs have taken an enormous leap in sequential data processing, work is still progressing to increase their abilities, moving outside existing shortcomings and growing their potential with emerging architectures and merged approaches. Second, RNN performance also depends on architectural innovations as well as data research methodology. This research investigates further into ongoing research by scientifically evaluating the impact of hyper parameter adjustment and data balancing across various areas of application.

III. METHODOLOGY

Recurrent Neural Networks implementation is a sequential process to enable accurate and efficient model performance with numerous sets of datasets. This research paper compares two broadly used datasets: the UCI Student Performance dataset and the UCI Human Activity Recognition dataset. They are chosen because they own typical features—one for academic performance activities and the other for physical activity classification—to provide a general comparison of how the RNN model can act on different types of data.

The approach is divided into different phases: dataset selection, data preprocessing, model training, and performance testing. To evaluate the model impact of data distribution on model performance, each dataset is subdivided into balanced and unbalanced subsets with variable ratios. Also, that several training and testing splits are active to compare how dataset size influences model accuracy and generalization. The suggested framework employs RNN models custom-made to each dataset, and hyper parameters are tuned to achieve maximum accuracy as well as computational efficiency.

The following Figure 1 shows that complete RNN process, including both data preparation steps and both model training and performance testing steps

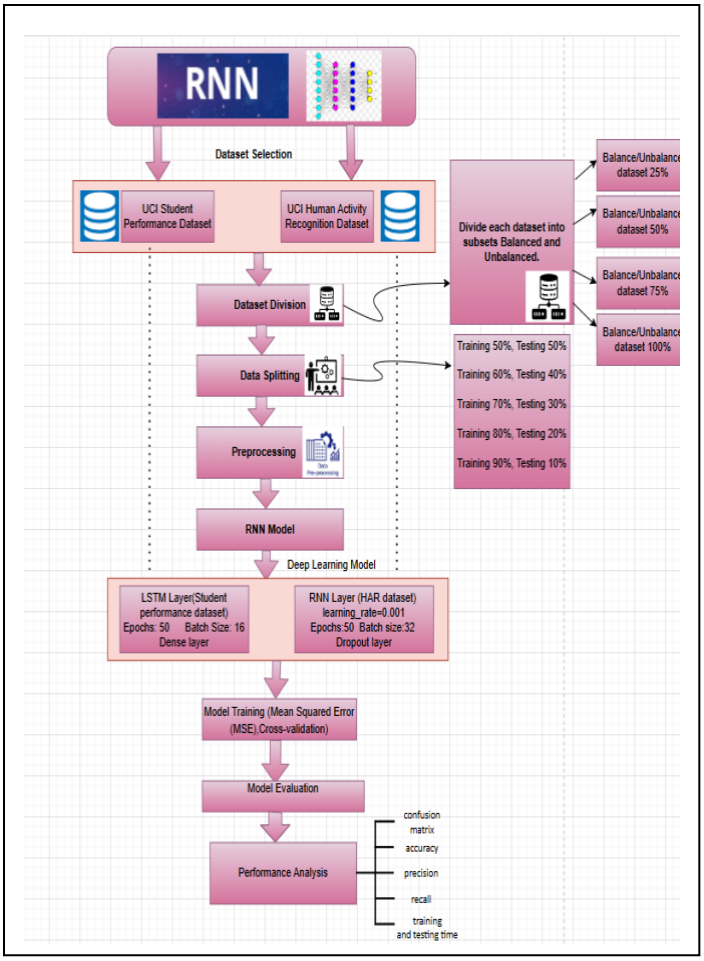

Figure 1. Proposed Methodology.

The Fig 1 demonstrates the RNN-based approach working in this research paper. The development begins with the utilization of two datasets: the UCI Student Performance and the UCI Human Activity Recognition dataset. For testing the model ability to handle numerous data distributions, each dataset is divided into balanced and unbalanced sets of four balance levels: 25%, 50%, 75%, and 100%. This allows enables a testing of the model's flexibility to data imbalance. During the data splitting phases, the datasets are split into training and testing sets in five different proportions (50-50%, 60-40%, 70-30%, 80-20%, and 90-10%). This variation ensures that there is a determination of how training size affects model accuracy and generalization.

In order to achieve balanced representation of the classes, we applied synthetic data generation methods based on the SMOTE concepts, which aided in having an equal number of samples for all categories present input in training sets.

Preprocessing of the data involves cleaning, normalization, and transformation so that the data can be used by the RNN model. This process ensures that there is better model convergence and less interference from noise. RNN architectures are trained separately for different datasets. For the UCI Student Performance dataset, a Long Short-Term Memory (LSTM) model is employed, trained for 50 epochs with a batch size of 16. The UCI HAR dataset is signified using a standard RNN layer with a dropout to prevent over fitting, a learning rate of 0.001, 50 epochs, and a batch size of 32. Both models apply the Mean Squared Error (MSE) loss function, with cross-validation being used to ensure unbiased performance evaluation.

For the UCI Student Performance dataset, as it is a regression problem, the model was trained with Mean Squared Error (MSE) as the loss function. For the HAR classification task, the model was trained with sparse categorical cross-entropy as it is designed to handle multi-class outputs well.

The performance of the model is measured in terms of significant parameters like accuracy, precision, recall, confusion matrix, and training and testing time. These parameters give a comprehensive idea about the predictive power and efficiency of the model. During the stage of performance analysis, the result is compared by dataset type, balance level, and train-test split. Through this analysis, the best model configuration is found while illustrating how data distribution difference and hyper parameters affect the RNN's accuracy in prediction. Through this step-by-step method, the research provides useful insight into improve the performance and overview of the RNN model across various datasets and data states.

The hyper parameters we used in our experimentation are shown in Table 1 and Table 2 along with their descriptions and values.

TABLE I. HYPERPARAMETER LIST UCI STUDENT PERFORMANCE

UCI Student Performance hyper parameter list		
Parameter	***Value***	***Description***
LSTM	64	Specifies the number of memory units in the LSTM layer. A higher value enables the model to capture more complex patterns but may increase both training time and memory consumption.
Epochs	50	Defines how many times the model will process the entire training dataset. More epochs may enhance learning but can also increase the risk of overfitting.
Batch Size	16	Indicates the number of samples processed before updating the model's weights. Smaller batch sizes may slow down training but often improve learning, particularly with LSTM models.
Loss Function	MSE	Ideal for regression tasks where the objective is to predict continuous values. For classification tasks, consider using alternatives like binary-crossentropy or categorical-crossentropy.
Optimizer	Adam	Adam is a widespread optimizer for long short term memory models because of its learning rate. Alternatives like RMSprop or SGD can also be used, depending on the dataset and model needs.

TABLE II. HYPERPARAMETER LIST HAR

UCI Student Performance hyper parameter list		
Parameter	***Value***	***Description***
RNN Layers	1	Specifies how many RNN layers are stacked. More layers can increase the model's ability to learn complex patterns.
Learning Rate	0.001	The rate at which the optimizer adjusts the model's weights during training. A lower learning rate may slow down the process, while a higher rate can lead to instability.
Batch Size	32	The number of training samples processed before updating the model's weights. Smaller batch sizes result in more frequent updates but can extend training time.
Epochs	50	The total number of times the entire training data is passed through the model. Increasing epochs may improve learning but could cause overfitting.
Dropout Rate	0.5	A technique to prevent overfitting by randomly setting a portion of input units to zero during training.
Activation Function	tanh ,relu	The functions used to introduce non-linearity in the model. Common choices like tanh for RNN layers and relu for dense layers help enhance the model's capacity to learn complex patterns.
Loss Function	sparse_ categor ical_cr ossentr opy	The function used to evaluate and optimize the model during training. For tasks involving multiple classes, sparse_categorical_crossentropy is often used.
Optimizer	Adam	The algorithm responsible for adjusting the model's parameters to minimize the loss function. Optimizers like Adam or SGD are widely used to boost model performance.

IV. EXPERIMENTATION

A. Configuration and Tools

The implementation is done on a DESKTOP-EPMB5JQ system with an Intel Core m7-6Y75 CPU (1.20 GHz, Turbo Boost up to 1.51 GHz), 8 GB RAM (7.88 GB available), and a 64-bit x64-based architecture with pen and touch support for up to 10 touch points. The Python environment is set up with Python 3.8 or higher, using major libraries like TensorFlow, Keras, NumPy, Pandas, Matplotlib, and Scikit-learn. Execution is mostly local with Google Colab being used as and when necessary, providing Tesla K80 GPU, Python 3.x runtime, 12 GB RAM, and 50 GB temporary storage. The research utilizes the UCI Student Performance dataset and the UCI Human Activity Recognition dataset, both of which are downloaded from the UCI Machine Learning Repository.

B. Dataset Description

This research paper uses two datasets from the UCI Machine Learning Repository: the UCI Student Performance dataset and the UCI Human Activity Recognition (HAR) dataset. The Student Performance dataset contains data on 649 students from two Portuguese secondary schools, with 33 attributes on demographics, social background, study habits, absenteeism, and academic history. The target variables—G1, G2, and G3—are student grades for three academic terms. Organized into a table structure, the data is divided between training and test subsets, with the grade distribution being mildly uneven, and in need of oversampling or under sampling to optimize the accuracy of models. Compared with this, the HAR dataset features 10,299 samples gathered from 30 volunteers completing six different activities: walking, walking upstairs, walking downstairs, sitting, standing, lying down. It is comprised of 561 sensor-based features collected using accelerometers and gyroscopes and is split into 7,352 training samples and 3,047 test samples. Although the dataset is fairly balanced, small variations in class distribution could need compensation during training.

Specifications of Balanced and Unbalanced datasets

- Unbalanced Dataset (100%)
  Contains all 10,296 samples as originally collected.
- Balanced Datasets (25%, 50%, 75%, 100%)
  Adjusted to ensure an equal number of samples across all six activity categories.

TABLE III. CLASS DISTRIBUTION BALANCE DATASET

Dataset Type	***Total Instances***	***Class 1***	***Class 2***	***Class 3***	***Class 4***	***Class 5***	***Class 6***
Unbalanced (100%)	10,296	1,572	1,716	1,716	1,716	1,639	1,639
Balanced (25%)	2,574	429	429	429	429	429	429
Balanced (50%)	5,148	858	858	858	858	858	858
Balanced (75%)	7,722	1,287	1,287	1,287	1,287	1,287	1,287
Balanced (100%)	10,296	1,716	1,716	1,716	1,716	1,716	1,716

Table III shows the distribution of activities over six activity classes in both unbalanced and balanced versions of the UCI Human Activity Recognition (HAR) dataset at various percentages (25%, 50%, 75%, and 100%). In the unbalanced dataset (100%), there are 10,296 instances with different class distributions. For example, Class 1 (WALKING) has 1,572 samples, whereas Class 6 (LAYING) has 1,639 samples.

The balanced data sets (25%, 50%, 75%, and 100%) provide equal numbers of instances in each activity class. The 25% dataset has 2,574 instances (429 instances per class), the 50% dataset contains 5,148 instances (858 instances per class), the 75% dataset has 7,722 instances (1,287 instances per class), and the 100% balanced dataset retains 10,296 instances with 1,716 samples per class. In contrast to the unbalanced dataset, its balanced counterparts maintain equal representation for every activity without any discrepancies in class distribution.

- Unbalanced Datasets (25%, 50%, 75%, 100%)
  Randomly, distribution across all six activity classes.

TABLE IV. CLASS DISTRIBUTION UNBALANCE DATASET

Dataset Type	*Total Instances*	*Class 1*	*Class 2*	*Class 3*	*Class 4*	*Class 5*	*Class 6*
Unbalanced (100%)	10,296	1,572	1,740	1,544	1,750	1,730	1,550
Unbalanced (75%)	7,722	1,179	1,305	1,158	1,312	1,297	1,470
Unbalanced (50%)	5,148	513	812	870	897	982	1,074
Unbalanced (25%)	2,574	326	390	420	436	470	532

Table IV shows the class distribution of the Unbalanced Datasets (25%, 50%, 75%, and 100%) that are obtained from the UCI Human Activity Recognition (HAR) dataset. In the Unbalanced 100% Dataset, the original distribution is preserved with different sample numbers for each activity (e.g., Class 1 has 1,572 samples, and Class 2 has 1,740). Unbalanced 75% Dataset has 7,722 samples, each class having the same proportional sample size reduction (e.g, Class 1 has 1,179 samples). Similarly, the Unbalanced 50% and 25% Datasets have the same distribution pattern but with fewer and fewer total samples. The datasets are still unbalanced since samples change in terms of classes.

V. RESULTS AND ANALYSIS

A. *Experiment with balanced and unbalanced datasets across different divisions based on performance metrics*

The performance of the RNN on 25%, 50%, 75%, and 100% balanced and unbalanced datasets are shown in Table V and VI below

TABLE V. PERFORMANCE METRICS OF BALANCED DATASETS.

Performance Metrics	*Model 1 (25%)*	*Model 2 (50%)*	*Model 3 (75%)*	*Model 4 (100%)*
Dataset	Balanced	Balanced	Balanced	Balanced
Accuracy	0.8219	0.8559	0.82	0.51
Precision	0.8401	0.8671	0.82	0.48
Recall	0.8219	0.8559	0.82	0.51
F1 Score	0.8197	0.8547	0.82	0.47
ROC-AUC	0.9791	0.9856	0.98	0.82

In the Table V and VI, Model 2 (50%) performs well overall on both the balanced and unbalanced datasets with excellent accuracy, precision, recall, and F1 score, and a good trade-off between training time and memory usage. Model 3 performs very well on the unbalanced dataset with the best accuracy, precision, recall, and ROC-AUC score. On the other hand, Model 4 (100%) seems to be overfitting in both cases, with decreased accuracy, precision, recall, and F1 score, particularly when dealing with the unbalanced dataset. Hence, choosing the best model should be based on the properties of the dataset and a tradeoff between accuracy and computing power.

TABLE VI. PERFORMANCE METRICS OF UNBALANCED DATASETS.

Performance Metrics	*Model 1 (25%)*	*Model 2 (50%)*	*Model 3 (75%)*	*Model 4 (100%)*
Dataset	Unbalanced	Unbalanced	Unbalanced	Unbalanced
Accuracy	0.7711	0.8503	0.8977	0.1377
Precision	0.7801	0.8547	0.9013	0.1285
Recall	0.7711	0.8503	0.8977	0.1377
F1 Score	0.7581	0.8359	0.8969	0.1304
ROC-AUC	0.9640	0.9860	0.9918	0.9406

In general, Model 2 (50%) is recommended for balanced datasets because of its optimal balance between speed and accuracy, whereas Model 3 (75%) is to be used for unbalanced datasets, even though it has higher resource requirements, for its improved class imbalance handling.

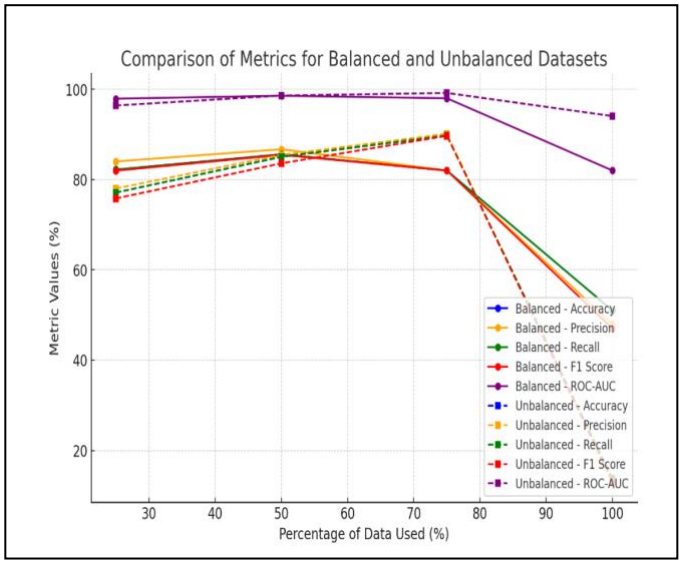

Figure 2. Comparison metrics.

This line graph in Fig 2 shows the performance of models trained on balanced and unbalanced datasets at different levels of data usage (25%, 50%, 75%, and 100%). The most significant evaluation metrics are Accuracy, Precision, Recall, F1 Score, and ROC-AUC. For the balanced dataset, performance is consistent, with improvement up to 50% data usage but a significant drop at 100%. In difference, models learned from the imbalanced dataset suffer a significant decline in performance at complete data usage, demonstrating issues in learning class imbalances. While ROC-AUC performance is constantly high for both datasets,

it is slightly lower with increased data usage. These findings establish that model stability and prediction consistency are improved with the use of a balanced dataset, thus being a better method for stable performance.

B. *Experiment with balanced and unbalanced datasets across different divisions based on resource usage*

The Table VI and VII shows some resource usage metrics for evaluating the performance of the model on balanced and unbalanced datasets across different metrics

TABLE VII. PERFORMANCE METRICS OF BALANCED DATASETS RESOURCES.

Performance Metric	***Model 1 (25%) Balanced***	***Model 2 (50%) Balanced***	***Model 3 (75%) Balanced***	***Model 4 (100%) Balanced***
Training Time (sec)	293.51	192.01	221.98	427.43
Testing Time (sec)	58.82	10.50	24.92	25.96
Memory Usage (MB)	831.17	66.31	895.77	833.30

TABLE VIII. PERFORMANCE METRICS OF UNBALANCED DATASETS RESOURCES.

Performance Metric	***Model 1 (25%) Unbalanced***	***Model 2 (50%) Unbalanced***	***Model 3 (75%) Unbalanced***	***Model 4 (100%) Unbalanced***
Training Time (sec)	437.19	554.36	756.03	556.03
Testing Time (sec)	21.50	37.35	13.99	11.12
Memory Usage (MB)	1314.88	1098.94	1042.46	821.30

The table VII and VIII show that the training time, testing time, and memory usage of balanced and unbalanced dataset models at different levels of data usage (25%, 50%, 75%, and 100%). Balanced dataset training time differs with the least at 50% data usage (192.01 sec) and maximum at full data usage (427.43 sec). However, training time for unbalanced datasets is always high, reaching a maximum of 756.03 sec at 75% usage before reducing slightly at 100%. Testing time is not uniform across models, with the lowest being that of the unbalanced dataset at 100% usage (11.12 sec) and the highest for the balanced dataset at 25% (58.82 sec). Memory usage is significantly greater in unbalanced models, peaking at 1314.88 MB when used at 25%, whereas balanced datasets have more reliable memory usage. These findings propose that unbalanced datasets require greater computational resources without necessarily enhancing performance, demonstrating the benefits of balanced datasets in terms of greater efficiency and stability.

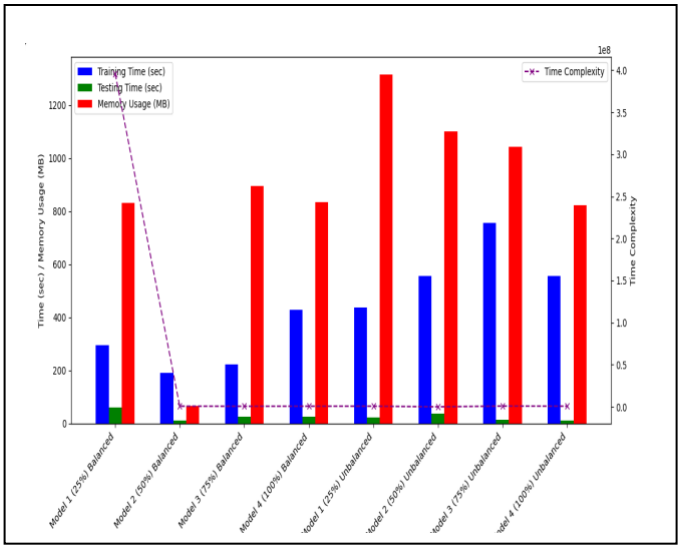

Figure 3. Performance Metrics.

The Fig 3 shows that comparison of performance between the models shows some interesting trends. With increasing dataset size, both training time and memory consumption have a tendency to increase. Out of the balanced models, Model 2 (50%) Balanced has the least training time of 192.01 seconds and testing time of 10.50 seconds. Model 4 (100%) Balanced has the maximum training time of 427.43 seconds. For the unbalanced models, there is typically an increased increase in training time and memory usage, with Model 1 (25%) Unbalanced using the most memory at 1314.88 MB. Test times for the unbalanced models vary, with Model 3 (75%) Unbalanced using the extended time at 37.35 seconds.

C. *Experiments with different train and test splits based on performance metrics*

In addition, we also perform experimentation across different training and testing splits including 50%-50%, 60%-40%, 70%-30%, 80%-20%, and 90%-10% ratios as shown in Table 7. The maximum accuracy of 0.8202 is obtained with a 70-30 split, which is the best configuration for this dataset. A 50-50 split is also good, resulting in an accuracy of 0.8138, which suggests well-balanced model performance. In contrast, the 90-10 split has the lowest performance with an accuracy of 0.7650, which is probably due to overfitting owing to the small test set. All precision and recall demonstrate similar behavior with optimal values at the 70-30 split. These results imply that the 70-30 split is the best choice to balance training and testing performance with excellent generalization.

TABLE IX. NUMBER OF SEQUENTIAL DATA ACROSS DIFFERENT TRAINING-TESTING RATIOS.

Training and Testing Percentage	*Accuracy*	*Precision*	*Recall*
50.0 and 50.0	0.8138	0.8153	0.8138
60.0 and 40.0	0.7619	0.7648	0.7619
70.0 and 30.0	0.8155	0.8202	0.8155
80.0 and 20.0	0.7956	0.7975	0.7956
90.0 and 10.0	0.765	0.7744	0.7650

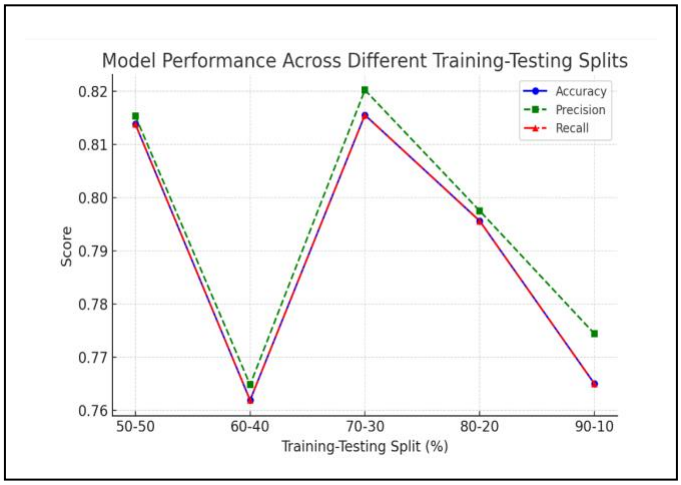

Figure 4. Performance Metrics across different training and testing splits.

The Figure 4 shows the performance of the RNN model across different training-testing splits. In this experimentation, the model reaches peak performance at 70%-30% training-testing ratio while maintaining the accuracy, precision, and recall for different training-testing splits. The 70-30 split shows the highest accuracy and precision, reinforcing its optimal performance for this dataset.

D. Experiment with training-testing splits datasets across different divisions based on resource usage

TABLE X. TRAINING-TESTING DATASET USAGE.

Training and Testing Percentage	*Training Time (s)*	*Testing Time (s)*	*Memory Usage (MB)*
50.0 and 50.0	197.36	4.86	1019.12
60.0 and 40.0	328.92	3.83	905.02
70.0 and 30.0	311.46	5.19	895.08
80.0 and 20.0	415.23	2.61	898.31
90.0 and 10.0	371.31	1.06	895.89

The comparison of these different training and testing splits is also shown graphically for clarity in Figure 5 as shown below.

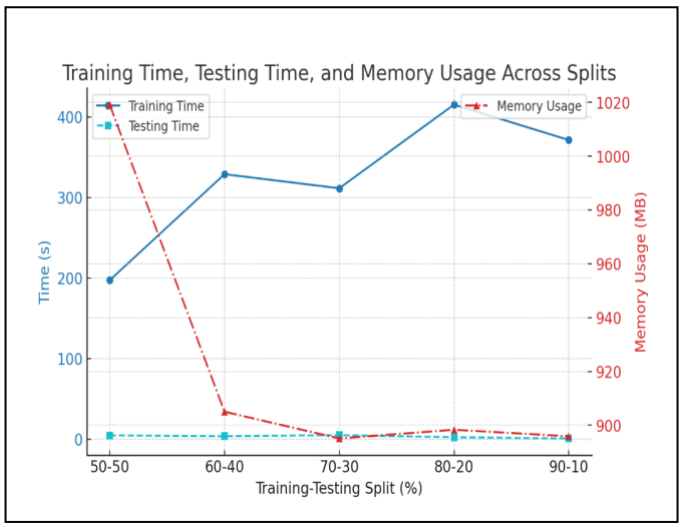

Figure 5. Resource Usage across different training and testing splits.

VI. RESULTS DISCUSSION

A. Pattern Analysis in Results of balanced and unbalanced datasets

The evaluation of the RNN model involves a comparison between balanced and unbalanced datasets for varying dataset sizes (25%, 50%, 75%, and 100%) and different training-testing splits. The findings tell that the RNN model performs best in all cases on balanced datasets, irrespective of size. Whether one trains with 25%, 50%, 75%, or the entire 100% dataset, balanced data delivers better performance than unbalanced data at identical proportions, as shown in Figure 6.

The enhanced performance with balanced datasets is due to the uniform representation of all classes, avoiding majority class bias. This helps ensure that the model learns patterns well from all classes, and this increases its ability to simplify.

On the other hand, unbalanced datasets make the model biased toward the main class, version it incapable of correctly classifying minority class instances, thus lowering generalization capacity. As a result, balanced datasets always perform better for any dataset size.

Additionally, the accuracy of the RNN model is enhanced when there is an increasing dataset, as proved in Figure 7. Machine learning algorithms usually favor greater datasets, which allow them to generalize and learn features better. The greatest possible accuracy, precision, recall, and F1 score occur if the complete 100% dataset is used. The performance tends to get higher with a progressive increase in dataset size from 25% up to

100%, representative the benefit of increased training set size.

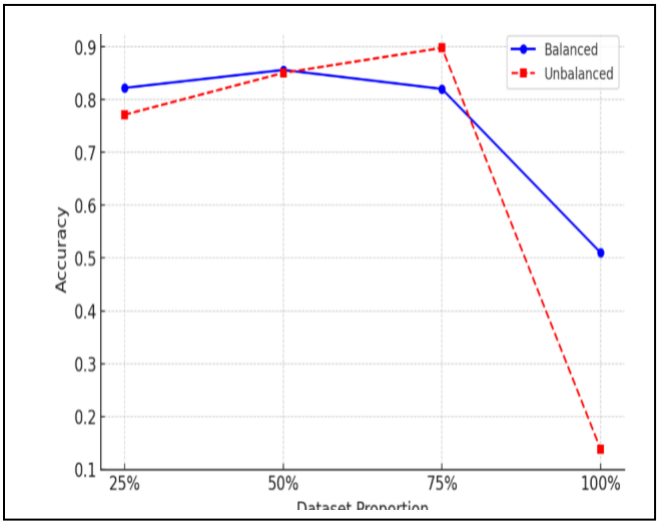

Figure 6. Accuracy-based comparison of balanced and unbalanced datasets.

In all of the studied train-test splits, the 70-30 split was the most reliable performer with the highest accuracy and F1 scores. This development likely happened because of a perfect combination of training and test data to learn the model and provide a test set large enough for an impartial evaluation. To duplicate these results, the dataset was shuffled randomly in a standard manner and the random stones was set in the same manner throughout all experimental runs.

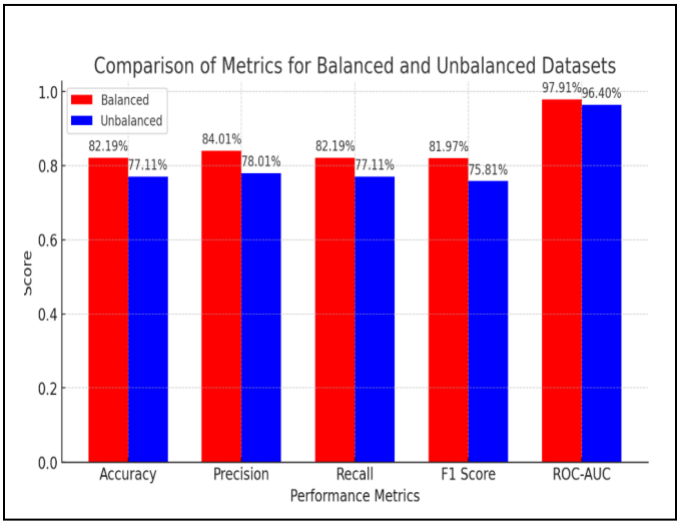

Figure 7. Performance-based comparison of different dataset divisions.

Moreover, the size of the dataset results in increased training time, testing time, and memory usage, as shown in Figure 8. A bigger dataset requires more samples for the model to process, calling for more computational power and memory. The entire 100% dataset takes the longest time to train and test among the 75%, 50%, and 25% datasets. This is a confirmation that a larger dataset improves the performance of the model but at the cost of higher computational resource usage.

The model leans towards to confuse activities with similar movement patterns particularly with the sitting and standing categories. This is possible due to sensor signals that were closely similar for both actions. In the future, a potential for improvement could be involved with some further feature engineering or task-specific preprocessing techniques.

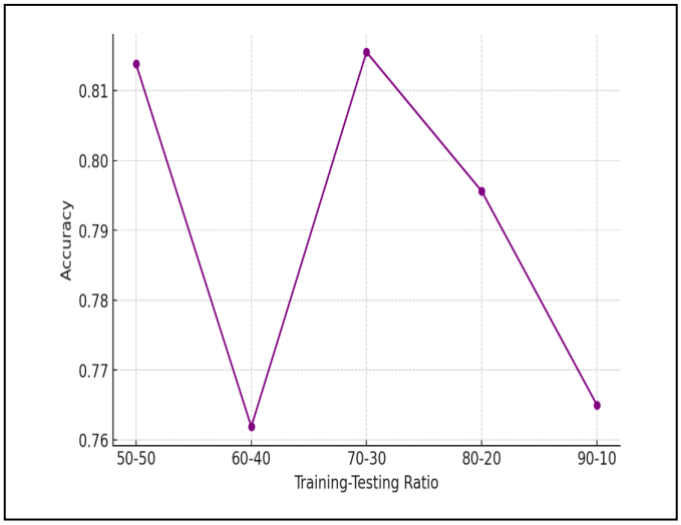

Figure 8. Training and testing time comparison.

VII. Conclusion and Future Directions

The experimental results show that Model 2 (50%) performs best for balanced datasets with high accuracy, precision, recall, and F1 score and with efficient use of computational resources. On the other hand, Model 3 (75%) has the best performance in unbalanced datasets with efficient handling of class imbalance at the cost of increased computational resources. Model 4 (100%) shows overfitting, mostly on unbalanced data, which reduces its overall performance. For training and testing splits, the 70%-30% setup is the best compromise between accuracy and computational complexity, while making the size of the training data have a significant impact on training time and memory usage.

Further, studies can explore ways to improve model performance on unbalanced datasets, such as data augmentation, class weighting, and synthetic data generation methods like SMOTE. Moreover, incorporating hybrid deep learning architectures combining LSTMs, CNNs, or transformer models could also improve classification outcomes. Research on lightweight models for real-time uses and minimalization of computational resource use for large-

sized datasets would also widen the scope for practical applications of these models in numerous fields.

ACKNOWLEDGMENT

We would like to express our gratitude to the developers at Kaggle for providing access to datasets and computational resources that were essential for our analysis.

REFERENCES

[1] Mienye, Ibomoiye Domor, Theo G. Swart, and George Obaido. "Recurrent neural networks: A comprehensive review of architectures, variants, and applications." *Information* 15.9 (2024): 517.

[2] Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult." IEEE transactions on neural networks 5.2 (1994): 157-166.

[3] Che, Zhengping, et al. "Recurrent neural networks for multivariate time series with missing values." *Scientific reports* 8.1 (2018): 6085.

[4] Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." *arXiv preprint arXiv:1412.3555* (2014).

[5] Greff, Klaus, et al. "LSTM: A search space odyssey." *IEEE transactions on neural networks and learning systems* 28.10 (2016): 2222-2232.

[6] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[7] Nitish, Srivastava. "Dropout: a simple way to prevent neural networks from overfitting." *J. Mach. Learn. Res.* 15 (2014): 1.

[8] Zaremba, Wojciech, Ilya Sutskever, and Oriol Vinyals. "Recurrent neural network regularization." *arXiv preprint arXiv:1409.2329* (2014).

[9] Mienye, Ibomoiye Domor, Theo G. Swart, and George Obaido. "Recurrent neural networks: A comprehensive review of architectures, variants, and applications." *Information* 15.9 (2024): 517.

[10] Yu, Yong, et al. "A review of recurrent neural networks: LSTM cells and network architectures." Neural computation 31.7 (2019): 1235-1270.

[11] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems 27 (2014).

[12] Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks." 2013 IEEE international conference on acoustics, speech and signal processing. Ieee, 2013.

[13] Liu, Zhenyu, et al. "Forecast methods for time series data: A survey." Ieee Access 9 (2021): 91896-91912.

[14] Vinyals, Oriol, et al. "Show and tell: A neural image caption generator." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[15] Sherstinsky, Alex. "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network." Physica D: Nonlinear Phenomena 404 (2020): 132306.

[16] Yin, Wenpeng, et al. "Comparative study of CNN and RNN for natural language processing." arXiv preprint arXiv:1702.01923 (2017).

[17] Li, Shuai, et al. "Independently recurrent neural network (indrnn): Building a longer and deeper rnn." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[18] Xiao, Jianqiong, and Zhiyong Zhou. "Research progress of RNN language model." 2020 ieee international conference on artificial intelligence and computer applications (icaica). IEEE, 2020.

[19] Zhao, Jingyu, et al. "Do RNN and LSTM have long memory?." International Conference on Machine Learning. PMLR, 2020.

[20] Al-Selwi, Safwan Mahmood, et al. "RNN-LSTM: From applications to modeling techniques and beyond—Systematic review." Journal of King Saud University-Computer and Information Sciences (2024): 102068.

[21] Dong, Y., Gao, D., Li, Y., Shi, G., & Liu, D. (2024). Degradation estimation recurrent neural network with local and non-local priors for compressive spectral imaging. *IEEE Transactions on Geoscience and Remote Sensing*.

[22] Baghoussi, Y. (2024). *Enhancing Forecasting using Read & Write Recurrent Neural Networks* (Doctoral dissertation, Universidade do Porto (Portugal)).

[23] Alonso, N. I. (2024). The Mathematics of Recurrent Neural Networks. *The Mathematics of Recurrent Neural Networks (October 27, 2024)*.

[24] Rivas, F., Sierra-Garcia, J. E., & Camara, J. M. (2025). Comparison of LSTM-and GRU-Type RNN Networks for Attention and Meditation Prediction on Raw EEG Data from Low-Cost Headsets. *Electronics*, *14*(4), 707.

[25] Dewi, T., Mardiyati, E. N., Risma, P., & Oktarina, Y. (2025). Hybrid Machine learning models for PV output prediction: Harnessing Random Forest and LSTM-RNN for sustainable energy management in aquaponic system. *Energy Conversion and Management*, *330*, 119663.