Journal of Neonatal Surgery
ISSN(Online): 2226-0439
Vol. 14, Issue 29s (2025)
https://www.jneonatalsurg.com

A Novel Deep Learning Based Customer Churn Prediction For The Banking Sectors With Efficient Feature Selection Strategy

B.Thenmozhi[1], Dr. S.Vimala[2], Dr. C.Jeyabharathi[3]

[1]Ph.D Research Scholar, Mother Teresa Women's University,Kodaikanal, Tamilnadu, India.
Email ID: phdcs19p614bt@gmail.com
[2]Associate Professor, Dept. of Computer Science, Mother Teresa Women's University, Kodaikanal,Tamilnadu, India.
Email ID: vimalaharini@gmail.com
[3]Computer Instructor Grade -I (PG), Govt. Higher Secondary School,Thalaiyuthu, Palani,Tamilnadu, India.
Email ID: bharathi_guhan@yahoo.com

.Cite this paper as: B.Thenmozhi, Dr. S.Vimala, Dr. C.Jeyabharathi, (2025) A Novel Deep Learning Based Customer Churn Prediction For The Banking Sectors With Efficient Feature Selection Strategy. *Journal of Neonatal Surgery*, 14 (29s), 545-556.

ABSTRACT
Every sector is seeing a significant growth in the number of service providers. When deciding where to invest their money, customer in the banking sector have an abundance of options these days. Customer engagement and customer churn have thus emerged as major concerns for the majority of banks. This research proposes a unique Deep Learning (DL) with efficient Feature Selection (FS) strategy for Customer Churn (CC) Prediction (CCP) in a bank. The suggested system follows for steps to predict the CC. In step 1, the data is preprocessed, where; the missing values (MV) are filled and the dataset is normalized by Z-Score Normalization (ZSN). In step 2, the preprocessed data is passed to the next step in which K-Nearest Neighbor (KNN) technique is implemented to handle the imbalance dataset. In step 3, the optimal features are computed from the balanced dataset using Brownian Movement centered Dragonfly Optimization Algorithm (BMDOA). Finally, in step 4, the churn customers or non-churn customer classification is done by using Modified Weight and Activation centered Bidirectional Gated Recurrent Unit (MWABGRU), in which the weight is optimally tuned by BMDOA. The findings show that the suggested one outperformed state-of-the-art (SOTA) methods across all the evaluation metrics.

Keywords: *Bank customers churn prediction (BCCP), Deep Learning (DL), Data Preprocessing, Dataset Balancing, Feature Selection (FS), Classification, and Bank Customer Churn Dataset*

1. INTRODUCTION

In an industry full of exciting and demanding companies, customers are important entities for any business. Customers may swiftly switch services or even suppliers when they have a choice of several service providers in a competitive industry [1]. Unhappiness, growing expenses, subpar quality, a lack of functionality, or privacy concerns could all be contributing factors to this switch (CC). CC has a direct impact on a number of businesses in several industries, such as banking, airline services, and telecommunications. As a result, the banking sector has seen this tendency the most [2]. Poor customer service, unnecessary bank fees, and other situations are likely to cause customers to leave any banking system. Because it expenses more to obtain a novel customer or subscriber than it ensures to keep an current one, customer retention (CR) is frequently difficult for businesses. However, CR methods will be focused exclusively on customers who are likely to depart or unsubscribe from an organization's service if it can forecast these customers with ease [3, 4]. However, in the financial sector, where massive amounts of data are analysed in order to extract information for productive and profitable operations, CC management is crucial [5]. Positive outcomes were obtained by several researchers that applied Machine Learning (ML) algorithms for CCP on customer data from electronic banking [6, 7]. CC has access to certain calculable machine learning estimations on the market, such as Decision Trees (DT), Naive Bayes (NB), XG boost, and others [8, 9]. One hundred million or more training samples may be included in the bank training dataset. If each training sample has a moderate amount of data, it may not be feasible to learn from such a large volume of data using conventional ML techniques. Recently, DL has been employed extensively for Churn Prediction (CP) in an effort to address these shortcomings. However, the FS on DL models' potentialfor forecasting the rate of churn in the banking sector has only been examined in a small number of research.

The influence of manual FS on the DL framework's effectiveness in predicting the churn rate appears to have some empirical support as a result. Therefore, heuristic information acquired from empirical research is currently insufficient to support FS [10]. This study suggests a novel DL technique for predicting the CC, solves an efficient FS, and resolves a data imbalancing problem. The following are the primary goals of the suggested work:

• To deal with the imbalance dataset and keep the model from becoming biased towards one class, the suggested solution makes use of KNN.

• The BMDOA algorithm is used by the suggested system to choose the features in the best possible way. By choosing the best features and removing the unnecessary and redundant ones, it improves process accuracy and boosts algorithm prediction ability.

Employing MEAGRU to classify the churn or non-churn customer, in which the weight values of the GRU is optimally chosen by BMDOA and the traditional Activation Function (AF)of RNN is replaces with Hard swish and ReLU to increases the classification accuracy and reduce the computational cost

The following is the remainder of the paper: The most current surveys related to the suggested method are shown in Section 2. The suggested methodology is offered in Section 3. The results and discussion are covered in Section 4. Conclusions for future study are summarized in Section 5.

2. LITERATURE SURVEY

The current papers that are relevant to the suggested method are surveyed in this section. Using a special customer-level dataset from a major Brazilian bank, **Renato Alexandre de Lima Lemos *et al.* [11]**investigated CCP in the banking sector. First, preprocessing of the data was done, including normalising all numerical variables and removing features with near-zero variance. In order to predict which features could be most helpful in efficiently separating the classes, the system then conducted an exploratory and visual analysis of the data. The technology might identify a possible good predictor of evasion for all types of customers of the institution based on the visual analysis. The findings showed that the system achieved 80.2% accuracy than the existing methods.

An adaptive ensemble algorithm extreme gradient boosting (XGBOOST) hyper-tuned meta classifier was suggested by **B. Srikanth *et al.* [12]**for CCP. The system first filled in the values with the mean and mode and identified the missing data through preprocessing. Next, the numerical data conversion was performed based on the label encoding (E). After that, the FS was done based on hybrid DragonFly algorithm and Firefly algorithms (FA). Finally, the prediction of CC was done based on XGBOOST. The results showed that the system achieved 97.85% accuracy (Acc) than the existing methods.

An experimental examination of Hyperparameters (HP) for DL-based CP in the banking sector has been suggested by **Edvaldo Domingos *et al.* [13]**. Initially, the data preprocessing was performed to remove the missing values on the collected Bank churn prediction dataset. After that, the Deep (NN) Neural Network (DNN) and MultiLayer Perceptron (MLP) was applied on the collected dataset forCCP. The consumer was labelled a churner if the classification was higher than the threshold (T) (0.5). The customer was categorised as a non-churner elsewhere. According to the experimental findings, the system's maximum Acc was between 83.85% and 85.45%.

Credit card CCP was produced using ML by **Dana AL-Najjar *et al.* [14]**. To start, the gathered credit card churn dataset was preprocessed in order to impute MV. Random forest (RF), NN, CR-Tree, C5 tree, Bayesian network, chi-square automatic interaction detection (CHAID) tree, support vector machine (SVM), quest tree, multinomial (LR) logistic regression (MLR), and an LR model were then used as the basis for the CCP. Here, KNN and two-step clustering were used to validate the prediction models' capabilities. The system outperformed the SOTA approaches with a maximum Acc of 97.5%.

A CC early warning model was established by **Zizhen Qin *et al.* [15]**using the RF technique. This model focusses on bank credit card CC and predicts the probabilities of future CC. First, the acquired bank churn dataset was preprocessed for minimizing the data count and noise and to turn the information into a format that could be processed by a computer. Next, the data imbalance problem was handled by SMOTE technique. Finally, the RF method was applied on the balanced dataset to predict the churn customer. The framework maintained a high Acc rate of 91.42%~92.2%, according to the findings.

The above-mentioned surveys the recent works related to Customer Churn Prediction in banking sector. All the above shows the better prediction results, but it has some deficiencies to predict the Customer Churn. FS is important to build efficient model. Numerous customer features were included in the CC data set, and not every element enhance the efficiency of churn prediction. The predictive performance of the model may be hampered by the dataset's excessive redundancy and irrelevant factors. But the above-mentioned works directly uses the features from the dataset for customer churn prediction. So it decreases the prediction performance and increases the overfitting problem. Also, the most of the above-mentioned works used DL algorithm to classify the churn customer with higher efficiency. But the random hyperparameter (HP) (weights, bias, etc.) of the network is a critical issue, because the random HP increases the processing time along with the computational complexity. This paper uses efficient Feature Selecion with novel optimization method and DL to classify the customer churn with optimal HP tuned approach. These are explained in the following section.

3. SUGGESTED METHODOLOGY

The phases of the bank Customer Churn Prediction system's implementation are shown in this section. As illustrated in Figure 1, the process of bank Customer Churn Prediction consists of four stages, including data preprocessing, dataset balancing, Feature Selection, and classification. Initially, the collected data from the bank CC dataset is preprocessing by removing the MV and z-score based normalize the dataset. Following that, dataset balancing is performed by KNN approach and the optimal features from the balanced dataset are selected by using BMDOA. Finally, the MWAGRU is applied to predict the bank customer as churn customer or non-churn customer, in which the weight is optimally chosen by BMDOA.

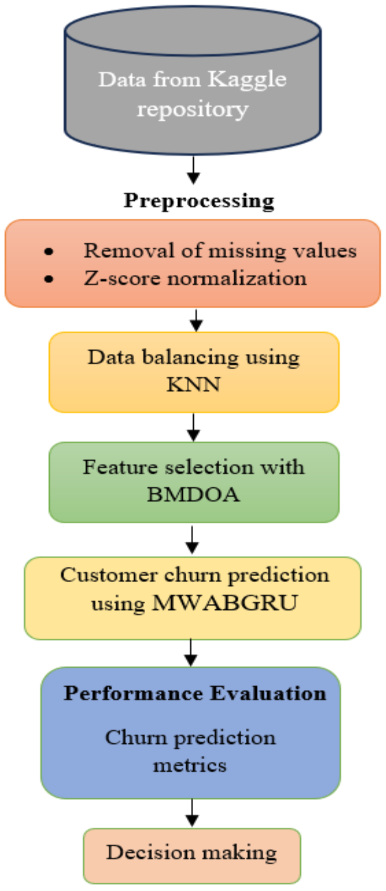
Data from Kaggle repository
Preprocessing
Removal of missing values
Z-score normalization
Data balancing using KNN
Feature selection with BMDOA
Customer churn prediction using MWABGRU
Performance Evaluation
Churn prediction metrics
Decision making

Figure 1: Workflow of the suggested methodology

3.1 Data Preprocessing

The Bank Customer Churn dataset is the primary source of the data. To find any potential problems or errors in the data, the initial pre-processing step is carried out after data collection (DC). Herein, the suggested system performs removal of MV and ZSN. These are explained as follows:

- **Removal of missing values**

Missing data may be the result of redundant data or the removal of crucial information from the dataset. Therefore, it is crucial to remove MV from the dataset in order to identify accurate results. The mean-based imputation of MV is applied here. The mean imputation (MI) technique is used to impute the MV for every variable. For every variable, MI determines the mean of the observed values. In particular, for models that are sensitive to feature magnitude, ZSN was chosen to guarantee that all features are on the same scale. Then MI is selected because it is a computationally efficient way to handle missing data when the missingness is random and the proportion of MV is small. It ensures that no valuable data points (DP) are lost and helps maintain the distribution of the feature.

- **Z-Score Normalization**

This method scales a feature's values to have a standard deviation (SD) of one and a mean of zero. This is accomplished by deducting each dataset value's feature mean and dividing the result by the SD. The following is its mathematical definition:

$$\underline{\underline{NV}} = \frac{\left(\widehat{O}_v - \mu\right)}{\sigma} \tag{1}$$

Where, $\widehat{O}_v$ refers the original value in the dataset, μ indicates the mean of data, and σ denotes the standard deviation.

3.2 Dataset Balancing

Research on customer data sets has revealed an unbalanced class distribution for CC analysis. The following situation could arise since the sample size of CC is considerably smaller than that of non-churn customers: the classification Acc is good, but the CCP Acc is low. Therefore, for correct classification and minimize the error, the dataset balancing is very important. In this work, the suggested system uses KNN to balance the dataset from the preprocessed dataset. The KNN algorithm is the most popular algorithms to handle the Class Imbalance (CI) problem. Only the feature vectors (FV) and class labels of the training information are stored by the KNN algorithm, which essentially doesn't need any training. In the feature space (FS), a new sample is given for testing, and its class is predicted as the majority class label from its KNN instance to the class of its nearest neighbour. It may function in the following areas:

- Initially, it determines k as the number of nearest neighbors. For each observation in the dataset, it calculates the Euclidean distance (ED) between each observation, and then adds the distance and the observation to an ordered set.
- Next, it sorts the ordered set of distances and observations in ascending order based on the distances.
- After that, it picks the first k entries from the sorted ordered set. Finally, it returns the majority class from the selected k entries.

3.3 Feature Selection

The balanced dataset is used to perform the Feature Selection in this phase. Following data balancing, Feature Selection significantly improves classification performance. The BMDOA is used in this work to reduce the size, computational complexity, and over-fitting while also selecting the best features from the dataset. Using the idea that dragonflies search their prey and avoid natural opponents, the DOA is new swarm intelligence (SI) algorithm. Its long convergence time and issues with local optima are among of its drawbacks, despite its ease of use and robust search capabilities.

To mitigate these limitations, this paper uses a Brownian Movement (BM) for final position updating of dragonflies (DF) and solves the above-mentioned limitations. Thus this BM is incorporated in conventional DOA is termed as BMDOA. The working process of BMDOA is explained as follows:

To begin, initialize the dragonfly population depend on the dataset as $\widehat{Y}_d = \left\{\widehat{Y}_1, \widehat{Y}_2, \widehat{Y}_3 ....\widehat{Y}_n\right\}$. The fitness value (FV) is then determined by computing the individual's fitness with the highest accuracy possible. It is statistically defined as follows:

$$fitness = Max\left(Accuracy\right) \tag{2}$$

$$Accuarcy = \frac{P_{ve}^{+} + P_{ve}^{-}}{TV_L} \tag{3}$$

Where, P_{ve}^{+} denotes the true positive (TP), P_{ve}^{-} indicates the true negative (TN), and TV_L refers to the overall sample count in the balanced dataset. Next, the new position of dragonflies can be obtained with the following step function $\Delta\widehat{Y}_{\tau+1}$ which is modeled as:

$$\Delta\widehat{Y}_{\tau+1} = \left(\hat{s}\,\overline{\overline{SE}}_u + \hat{a}\,\overline{\overline{AL}}_u + \hat{c}\,\overline{\overline{CH}}_u + \hat{f}\,\overline{\overline{FD}}_u + \hat{e}\,\overline{\overline{EY}}_u\right) + \varpi\,\Delta\widehat{Y}_\tau \tag{4}$$

Here, the separation, alignment, cohesion, attraction to food, and distraction from the enemy of the u^{-th} individual were represented as $\overline{\overline{SE}}_u, \overline{\overline{AL}}_u, \overline{\overline{CH}}_u, \overline{\overline{FD}}_u, and\ \overline{\overline{EY}}_u$ .Separation, alignment, cohesion, food, and enemy factors are denoted by $\hat{s}, \hat{a}, \hat{c}, \hat{f}, and\ \hat{e}$. The inertia weight is denoted by ϖ. The iteration counter is shown by τ.

$$\widehat{Y}_{\tau+1} = \widehat{Y}_{\tau} + \Delta\widehat{Y}_{\tau+1} * BM_w \tag{5}$$

$$BM_w = \sqrt{\frac{\alpha}{\chi}} * \vec{R}_N * PN_m \tag{6}$$

Where, BM_w indicates the BM that is used to improve the randomization stage of the DOA.

The motion time period of an agent in seconds is thus denoted by α. The number of sudden motions for the same agent in relation to time is denoted by χ. The random number is shown by $\vec{R}_N$. PN_m indicates the periodic motion of the DF that was distributed throughout time. The procedure will be continued till the maximum number of iterations is reached. The position is also updated if the DF has neighbours. The MWABGRU is now used for predicting the CC using the optimised features as input.

BMDOA Feature Selection Pseudo code:

```
1. Initialize the parameters:
   - N: Number of dragonfly agents (population size)
   - D: Sum of dimensions (features in the dataset)
   - T: Maximum count of iterations
   - c1, c2: Coefficients controlling the search (exploration vs exploitation)
   - w: Brownian movement intensity (a random exploration factor)
   - F: Feature matrix (original dataset with all features)
   - Label vector (target churn labels)
2. Generate initial population of dragonflies (solutions):
   - For each dragonfly (i = 1 to N):
      - Randomly initialize a solution (a binary vector of length D, where every element represents the inclusion (1) or exclusion (0) of a feature).
      - Calculate the fitness of the dragonfly based on feature subset (use evaluation metric, e.g., classification accuracy using a model like logistic regression, decision tree, etc.).
3. Evaluate fitness of initial population:
   - For each dragonfly:
      - Evaluate the subset of features selected by the dragonfly.
      - Calculate its fitness value (e.g., classification accuracy, cross-validation score, or another suitable evaluation metric).
      - Store the best fitness value and the corresponding feature subset.
4. Main optimization loop (for t = 1 to T):
   - For each dragonfly (i = 1 to N):
      a) **Exploration Phase** (Brownian movement-based exploration):
         - Generate a random step size `r` using Brownian motion (e.g., normal distribution with mean 0 and variance w).
         - Update the dragonfly's position (feature subset) as:
            `Position[i] = Position[i] + r`
         - Apply boundary conditions (ensure binary encoding remains within [0, 1]).
       b) **Exploitation Phase** (Dragonfly movement based on the best solution found so far):
         - For each dragonfly, calculate the distance to the best solution found so far (global best).
```

```
            - Move towards the global best solution with a velocity proportional to the distance between the current and best solutions.
            - Update the feature subset accordingly (this is an exploitation of the promising solutions).
        c) **Combine Exploration and Exploitation**:
            - Adjust the position (feature subset) of each dragonfly as a weighted combination of exploration (random Brownian movement) and exploitation (movement towards the best solution).
        d) **Evaluate the fitness of each dragonfly**:
            - Calculate the fitness of each new solution (subset of features).
            - If the current fitness is superior than the previous fitness, update the dragonfly's location and best fitness.
5. After T iterations:
    - Select the feature subset corresponding to the best fitness score.
6. Return the optimal feature subset and the model performance based on these features.
```

Explanation of Key Steps:

Initialization: We initialize the population of dragonflies, where each dragonfly represents a potential solution in the form of a binary vector, indicating whether each feature is selected (1) or not selected (0).

Fitness Evaluation: For each dragonfly, its feature subset is evaluated by running a model (such as DT, LR, etc.) on the dataset using the selected features. A fitness function (FF), typically based on model performance (Acc, F1 score, etc.), are employed for assessing the feature subset.

Exploration and Exploitation:

1. **Exploration**: This is driven by Brownian motion, where a random step is applied to the dragonfly's position to explore the feature space.
2. **Exploitation**: The dragonfly also moves towards the best feature subset found so far, improving the search process by focusing on areas of the feature space that have shown promising results.

Optimization: The algorithm alternates between exploration and exploitation (E-E), refining the feature subset selection over multiple iterations to find the optimal subset that leads to the best model efficiency.

Final Output: After the defined number of iterations, the best feature subset is returned, which will likely reduce dimensionality, improve model performance, and help reduce overfitting.

3.4 Classification

At last, the Modified Weight and Activation centred Bidirectional Gated Recurrent Unit (MWABGRU) is used to do the CC classification from the feature set that was best chosen. One GRU takes the input in a forward manner, while the other takes it backwards. This type of sequence processing model is called a BGRU. Both short-term and long-term dependences can be learnt by BGRU. The three main parts of every GRU are a candidate hidden state, an update gate, and a reset gate.

These are associated with random weights to compute the output. This randomly chosen weight matrix increases the computational complexity and takes more time to train the parameter. So the proposed system uses BMDOA, which optimally chosen weights for the corresponding input and reduce the complexity of the system. At the same time it increases the classification accuracy with less processing time. In addition, usually, the BGRU used sigmoid and tanh AF, which hard to solve the gradient vanishing problem (GVP) and so it produces the overfitting issues. Hence, the suggested system replaces Hard swish instead of sigmoid and rectified linear unit (ReLU) instead of tanh activation to solve the VGP. Thus the optimal weights and modified activation incorporations in traditional BGRU is termed as MWABGRU. The configuration of BGRU is visualized in figure 2.

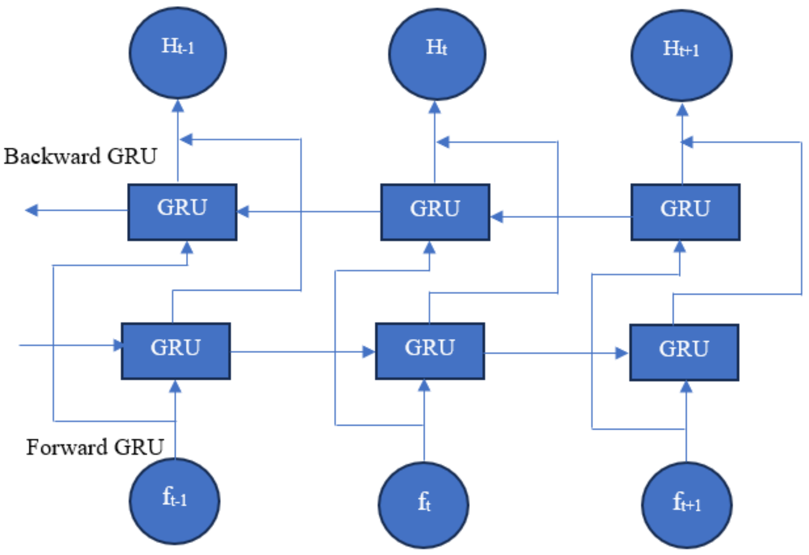

Figure 2: Structure diagram of BGRU

Initially, the forward GRU takes the optimal features set as input $\breve{F} = \{\breve{f}_1, \breve{f}_2, \breve{f}_3, \ldots \breve{f}_n\}$ generated by the BMDOA algorithm. In order to carry out a number of linear functions and AF, the forward (HL) hidden layer combines $\breve{F}_t$ and the forward hidden state $\overrightarrow{H_{t-1}}$ at time step $t-1$. There are two gates in each GRU, namely an update gate $(\breve{U}_t)$ and reset gate $(\breve{R}_t)$. To guarantee that crucial byte sequences are sent on to the following stage, the update gate regulates how much of the prior data is kept in the current state and the reset gate establishes the extent to which irrelevant byte sequences are ignored. The following formulation can be used to express each relationship in the forward GRU:

$$\breve{U}_t = \eta^* \left(\breve{W}_{\breve{U}} \left[\overrightarrow{H_{t-1}}, \breve{F}_t \right] \right) \tag{7}$$

$$\breve{R}_t = \eta^* \left(\breve{W}_{\breve{R}} \left[\overrightarrow{H_{t-1}}, \breve{F}_t \right] \right) \tag{8}$$

$$\tilde{h}_t = \kappa^* \left(\breve{W}_{\breve{F}_t} \left[\breve{R}_t, \overrightarrow{H_{t-1}}, \breve{F}_t \right] \right) \tag{9}$$

$$\overrightarrow{H_t} = (1 - \breve{U}_t)\overrightarrow{H_{t-1}} + \breve{U}_t \, \tilde{h}_t \tag{10}$$

Where, $\breve{W}$ indicates the weight matrix of forward GRU, which is optimally selected by BMDOA, $\tilde{h}_t$ indicates the candidate HL, $\overrightarrow{H_t}$ indicates the forward hidden state output, η^* indicates the hard swish AF, which is calculated using equation (11), and κ^* refers the ReLU AF, which is calculated using equation (12). A variant of AF called "hard swish" is based on Swish but substitutes a piecewise linear analogue for the computationally costly sigmoid. A ReLU is an AF that solves the VGP and adds the nonlinearity (NL) property to a DL model.

$$\eta^* = 2 * \breve{F} * \max\left(0, \min\left(1, \left(\varphi \, \breve{F} * 0.2 + 0.5\right)\right)\right) \tag{11}$$

$$\kappa^* = \max\left(0, \breve{F}\right) \tag{12}$$

Where, $\breve{F}$ refers the input features set and φ refers the trainable parameter. Next, the backward GRU takes the input (optimal features set) in reverse as an input.

Similar to the forward GRU, it carries out a sequence of computations at each t to produce a hidden state $\overleftarrow{H}_t$. The final output, which is either a churn customer or a non-churn customer, is then determined by combining this state with $\overrightarrow{H}_t$, as indicated in equation (13).

$$H_t = \left[\overleftarrow{H}_t, \overrightarrow{H}_t\right] \qquad (13)$$

Comparison with Traditional Activation Functions:

To adequately compare the modified activation functions with traditional ones, the following aspects should be considered:

a. Performance (Learning Speed & Convergence):

- **Traditional Activation Functions**: Functions like sigmoid and tanh can lead to **vanishing gradients** during backpropagation, which slows down training and can cause convergence issues. ReLU tends to perform better by mitigating this problem, but it suffers from the **dying ReLU** problem, which can slow down learning when neurons stop updating.
- **Modified Activation Functions in MWABGRU**: The modifications are designed to improve gradient propagation, possibly through better weight scaling or adaptive behavior that adjusts the activation response based on training dynamics. This could lead to faster convergence, especially in deep or complex recurrent architectures, where the vanishing gradient problem is more pronounced.

b. Accuracy and Generalization:

- **Traditional Activation Functions**: In many cases, traditional activation functions provide sufficient performance, but they may fail to generalize well in specific tasks, particularly when the model faces long-term dependencies in sequential data (common in tasks like time series (TS) prediction or natural language processing).
- **Modified Activation Functions in MWABGRU**: By addressing vanishing/exploding gradients and improving gradient flow, modified AF could allow the network to learn more effectively from long sequences, potentially leading to better Acc and generalization, particularly in **sequence-based tasks** like CCP (in the context of the Bank CC dataset).

c. Robustness to Vanishing and Exploding Gradients:

- **Traditional Activation Functions**: Functions like sigmoid and tanh are known to cause issues with the **vanishing gradient problem**, especially in deeper networks, leading to ineffective training.
- **Modified Activation Functions in MWABGRU**: These functions are likely designed to **prevent vanishing gradients** by allowing gradients to remain stable during backpropagation (BP). This would be especially important for recurrent networks, where long-term dependencies need to be captured without losing important gradient information.

d. Computational Efficiency:

- **Traditional Activation Functions**: Functions like ReLU are computationally efficient and widely used in deep learning models, making them well-suited for large-scale applications.
- **Modified Activation Functions in MWABGRU**: While these functions might introduce some complexity (e.g., through adaptive or weighted behavior), their computational efficiency should be evaluated based on the training time and accuracy improvements they bring.

4. RESULTS AND DISCUSSION

The authors now offer the findings from the suggested method, along with a discussion of their implications, in accordance with the work's stated aims. The suggested work is implemented using an NVIDIA Geforce 930M graphics card, an Intel Core i7 processor, 8 GB of RAM, and Python 3.7. The experiment's dataset is covered in Section 4.1, and the performance analysis is shown in Section 4.2.

4.1 Dataset Descriptions

The method assesses and validates the effectiveness of the suggested method using the Bank CC Dataset. The attributes of this dataset that relate to ABC Multistate Bank are customer_id, credit_score, country, gender, age, tenure, balance, products_number, credit_card, active_member, estimated_salary, and churn. In this instance, testing uses 30% of the data, whereas training uses 70%.

https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-datasetis the link to view it. To provide a comprehensive dataset description, it's vital to cover the following main features:

1. Churn vs Non-Churn Distribution

- **Churn**: The rate or count of customers who have left the service (churned).
- **Non-Churn**: The rate or count of customers who have stayed with the service.
- It's useful to highlight the **class imbalance** in this context, as imbalanced distributions can affect model performance, requiring specific techniques like oversampling or weighted loss functions.

2. Feature Types

- **Categorical**: Features like customer demographics, service type, region, etc.
- **Numerical**: Features like age, account balance, usage statistics, etc.
- **Datetime**: Any timestamp-based features, such as last transaction date or account creation date.
- **Text**: Any unstructured data, e.g., customer feedback.
- It's important to mention the **data types** for each feature and whether any encoding or transformation is required for categorical features (e.g., one-hot encoding, label encoding).

3. Size and Time Period

- **Number of Samples**: overall count of customers (rows) in the dataset.
- **Time Period**: The time frame over which the data was collected (e.g., monthly churn data, yearly customer retention).

4. Label Encoding

- If churn status (churn vs. non-churn) is encoded numerically, you may want to mention the mapping used (e.g., 0 for non-churn, 1 for churn).

By providing these details, you can ensure that the dataset description gives a clear picture of the data and any steps that might have been taken during preprocessing. Acc, precision (P), recall (R), f-measure, Area Under Curve (AUC), Brier, Kappa, Expected Maximum Profit of the Customer Churn (EMPC), False omissions rate (FOR), and classification time are the metrics used to compare the results of the proposed MWABGRU to the current Gated Recurrent Unit (GRU), Recurrent NN (RNN), MLP, and RF classifiers.These analyses could be shown in the following figure and table.

1. Accuracy

The framework's overall ratio of accurate predictions is measured by the simple metric known as acc. It is defined as the proportion of fully predicted instances (including churn and non-churn) to all instances in the dataset.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (14)$$

- True Positives (TP): Samples where the framework correctly predicted churn.
- True Negatives (TN): Samples where the framework accurately predicted non-churn.
- False Positives (FP): Samples where the framework accurately predicted churn when the true class is non-churn.
- False Negatives (FN): Samples where the framework inaccurately predicted non-churn when the true class is churn.

2. Precision (P)

P calculates the ratio of accurate positive predictions (churn) among all the cases that the model predicts as churn. "Of all the customers predicted to churn, how many actually churned?" is the question it answers.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (15)$$

3. Recall (R) (Sensitivity):

R is a measure of the ratio of real churn cases that the framework accurately identified. It responds to the query: "Of all the customers who actually churned, how many did the model successfully predict as churn?"

$$R = \frac{TP}{TP+FN} \quad (16)$$

4. F-measure (F1 Score):

The harmonic mean of P and R is the F1 score. It provides a equilibrium among the two measures and is particularly helpful in imbalanced datasets when you need to weigh the relative value of P and R.

$$F - \text{measure} = 2 \times \frac{P \times R}{P+R} \quad (17)$$

5. AUC - ROC

The model's capacity to differentiate between churn and non-churn customers is measured by the AUC. Plotting the TP rate (recall) compared to the false positive rate at different T levels is known as the ROC curve.

$$AUC = \int_0^1 \text{TP Rate (TPR)} \times \text{FP Rate (FPR)} \quad (18)$$

6. Brier Score

A tool for assessing the precision of predicted probabilities is the Brier score. It calculates the average squared difference among the actual binary outcomes and the predicted probability.

$$\text{Brier Score} = 1/N \sum_{i=1}^{N} (p_i - o_i)^2 \quad (19)$$

- The predicted probability for instance is denoted as p_i.
- The observed outcome (1 for churn, 0 for non-churn) is denoted as o_i

7. Kappa (Cohen's Kappa):

Cohen's Kappa is a metric that measures the agreement between two raters or predictions, adjusting for the possibility of chance agreement.

$$\kappa = \frac{p_0 - p_e}{1 - p_e} \quad (20)$$

- The Observed agreement (proportion of correct predictions) is denoted as p_0.
- The Expected agreement by chance is denoted as p_e

8. EMPC:

EMPC measures the potential profit gain from the CCP model by focusing on maximizing profits from correctly identified CP.

$$EMPC = \text{Profit} \times (\text{True Positives} - \text{False Positives}) \quad (21)$$

9. FOR:

The FOR quantifies the proportion of churn customers that are predicted as non-churn by the model.

$$FOR = \frac{FN}{FN+TN} \quad (22)$$

10. Classification Time:

Classification time measures the computational cost (time) taken by the frameworkin creating predictions for a given number of instances.

$$\text{Classification Time} = \frac{\text{Total Time Taken for Prediction}}{\text{Number of Instances Predicted}} \quad (23)$$

4.2 Performance Analysis

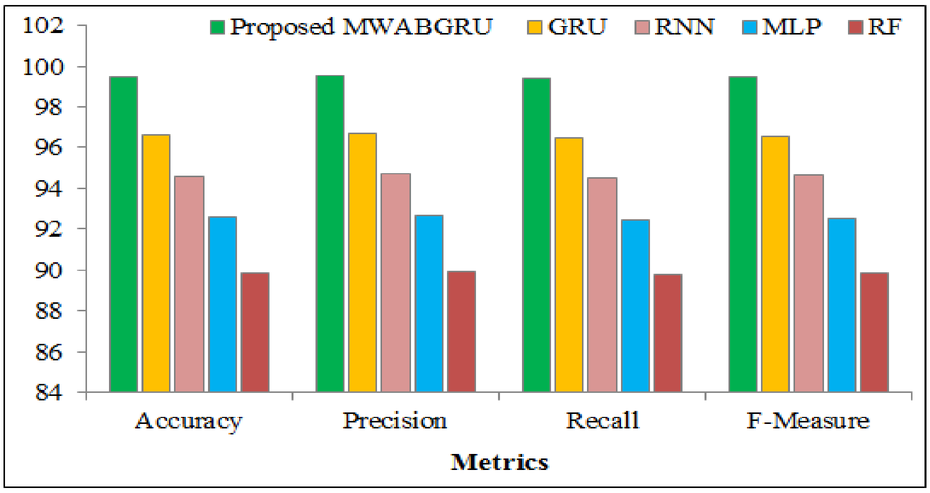

Figure 3: Evaluation results analysis

The results of comparing the suggested MWABGRU with the current GRU, RNN, MLP, and RF are shown in Figure 3. Acc, P, R, and f-measure are considered essential measures for assessing the model's efficacy. The proposed effort achieves a better percentage of results compared to the current methods. For instance, the suggested RF has a higher prediction rate of 99.43% Acc, 99.52% P, 99.36% R, and 99.44% f-measure, whereas the current RF has the lowest results among the techniques, with 89.82% Acc, 89.92%P, 89.74% R, and 89.91% f-measure. The performance of the suggested system is therefore found to be better than that of the traditional methods. Whether a customer is a churner or not can be easily determined due to its exact churn management. The effectiveness of the suggested work with the current GRU, RNN, MLP, and RF classifiers is then displayed in table 1 along with metrics for AUC, Brier, Kappa, EMPC, FOR, and classification time.

Table 1: results analysis of the suggested framework

Metrics	Suggested	GRU	RNN	MLP	RF
AUC (%)	98.51	95.63	93.41	91.25	89.52
Brier (%)	0.1056	0.1198	0.1741	0.1964	0.2132
Kappa (%)	0.8254	0.6965	0.5689	0.4975	0.3964
EMPC (%)	8.21	7.42	6.53	5.85	5.41
FOR (%)	98.56	96.35	93.21	91.62	90.41
Classification time (min)	0.84	1.07	1.93	2.32	3.06

All models were able to estimate the churn customers with varying performance, according to the table data. For example, AUC obtained by the suggested approach was improved by 2.88%, 5.1%, 7.26%, and 8.99% when compared to GRU, RNN, MLP, and RF, respectively. Likewise, for the remaining metrics, the suggested one archives superior results than the current approaches. For example, the suggested method achieves maximum outcomes of 0.1056% brier score, 0.8254% kappa, 8.21% EMPC, 98.56% FOR, respectively. Also, the classification time metrics is an important metric to show the efficiency of the system. If any system takes less time to classify the churn or non-churn customer, then it is regarded as a good system. In this work, the current GRU, RNN, MLP, and RF takes classification time of 1.07m, 1.93m, 2.32m, and 3.06m, but the suggested one takes just 0.84m time to classify the churn customer. Thus the outcomes from the experiments showed that the different configurations such as data balancing using KNN, FS using BMDOA, and the optimal weight and modified activation-based GRU boost the outstanding performance of the suggested work.

5. CONCLUSION

In this study, the system suggests a novel DL based CCP for the banking sectors with a novel FS strategy. The suggested system mainly consists of four steps, namely, data preprocessing, dataset balancing, FS, and classification. To validate the efficiency of the suggested work, it uses Bank CC Dataset. The experimental analysis of the suggested MWABGRU is carried out with the existing GRU, RNN, MLP, and RF classifiers. The evaluation is executed with the support of Acc, P, R, f-measure, AUC, Brier, Kappa, EMPC, FOR, and classification time. The results highlighted that the suggested method achieves higher efficiency of 99.43% Acc, 99.52% P, 99.36% R, 99.44% f-measure, 98.51% AUC, 0.1056% Brier score, 0.8254% kappa, 8.21% EMPC, 98.56% FOR, and 0.84min classification time, respectively. The experimental analysis concluded that the suggested method achieves better outcomes when compared to the current approaches. Using sophisticated DL techniques, this study will be extended in order to improve the prediction rate and categorise the churn output as having a high, medium, or a low probability of churning.

REFERENCES

[1] Lalwani, P., Mishra, M. K., Chadha, J. S., & Sethi, P. (2022). Customer churn prediction system: a machine learning approach. *Computing*, 1-24.

[2] Usman-Hamza, F. E., Balogun, A. O., Capretz, L. F., Mojeed, H. A., Mahamad, S., Salihu, S. A., ... & Salahdeen, N. K. (2022). Intelligent decision forest models for customer churn prediction. *Applied Sciences*, *12*(16), 8270.

[3] Yahaya, R., Abisoye, O. A., & Bashir, S. A. (2021, February). An enhanced bank customers churn prediction model using a hybrid genetic algorithm and k-means filter and artificial neural network. In *2020 IEEE 2nd International Conference on Cyberspac (CYBER NIGERIA)* (pp. 52-58). IEEE.

[4] Muneer, A., Ali, R. F., Alghamdi, A., Taib, S. M., Almaghthawi, A., & Ghaleb, E. A. (2022). Predicting customers churning in banking industry: A machine learning approach. *Indones. J. Electr. Eng. Comput. Sci*, *26*(1), 539-549.

[5] Oluwatoyin, A. M., Misra, S., Wejin, J., Gautam, A., Behera, R. K., & Ahuja, R. (2022, July). Customer Churn Prediction in Banking Industry Using Power Bi. In *Proceedings of Third International Conference on Computing, Communications, and Cyber-Security: IC4S 2021* (pp. 767-774). Singapore: Springer Nature Singapore.

[6] Vo, N. N., Liu, S., Li, X., & Xu, G. (2021). Leveraging unstructured call log data for customer churn prediction. *Knowledge-Based Systems*, *212*, 106586.

[7] Agarwal, V., Taware, S., Yadav, S. A., Gangodkar, D., Rao, A. L. N., & Srivastav, V. K. (2022, October). Customer-Churn Prediction Using Machine Learning. In *2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS)* (pp. 893-899). IEEE.

[8] Prabadevi, B., Shalini, R., & Kavitha, B. R. (2023). Customer churning analysis using machine learning algorithms. *International Journal of Intelligent Networks*.

[9] Rahmaty, M., Daneshvar, A., Salahi, F., Ebrahimi, M., & Chobar, A. P. (2022). Customer churn modeling via the grey wolf optimizer and ensemble neural networks. *Discrete Dynamics in Nature and Society*, *2022*, 1-12.

[10] Dalli, A. (2022). Impact of hyperparameters on Deep Learning model for customer churn prediction in telecommunication sector. *Mathematical Problems in Engineering*, *2022*, 1-11.

[11] de Lima Lemos, R. A., Silva, T. C., & Tabak, B. M. (2022). Propension to customer churn in a financial institution: A machine learning approach. *Neural Computing and Applications*, *34*(14), 11751-11768.

[12] Srikanth, B., Papineni, S. L. V., Sridevi, G., Indira, D. N. V. S. L. S., Radhika, K. S. R., & Syed, K. (2022). Adaptive XGBOOST Hyper Tuned Meta Classifier for Prediction of Churn Customers. *Intelligent Automation & Soft Computing*, *33*(1).

[13] Domingos, E., Ojeme, B., & Daramola, O. (2021). Experimental analysis of hyperparameters for deep learning-based churn prediction in the banking sector. *Computation*, *9*(3), 34.

[14] AL-Najjar, D., Al-Rousan, N., & AL-Najjar, H. (2022). Machine learning to develop credit card customer churn prediction. *Journal of Theoretical and Applied Electronic Commerce Research*, *17*(4), 1529-1542.

[15] Qin, Z., Liu, Y., & Zhang, T. (2022). Research on Early Warning of Customer Churn Based on Random Forest. *Journal of Artificial Intelligence (2579-0021)*, *4*(3).