Neural Computing and Applications (2024) 36:4867–4883
https://doi.org/10.1007/s00521-023-09327-w

ORIGINAL ARTICLE

Predict customer churn using combination deep learning networks model

Van-Hieu Vu[1]

Received: 2 October 2022 / Accepted: 26 November 2023 / Published online: 21 December 2023

Abstract
Customers churn is an important issue that is always concerned by banks, and is put at the forefront of the bank's policies. The fact that banks can identify customers who are intending to leave the service can help banks promptly make policies to retain customers. In this paper, we propose a combined deep learning network models to predict customers leaving or staying at the bank. The proposed model consists of two levels, Level 0 consists of three basic models using three Deep Learning Neural Networks, and Level 1 is a logistic regression model. The proposed model has obtained evaluation results with accuracy metrics of 96.60%, precision metrics of 90.26%, recall metrics of 91.91% and F1 score of 91.07% on the dataset "Bank Customer Churn Prediction".

Keywords Customer churn · Banks · Machine learning

1 Introduction

In today's highly competitive business landscape, retaining customers has become a critical objective for commercial banks. Each bank employs its own strategies and policies, but accurately understanding customers' intentions to discontinue using their services poses a unique challenge. According to data, an increase in customer retention by just 5% can lead to a profit increase of at least 25% [1].

The main motivation of this research is to address several challenges:

- In today's highly competitive business context, customer retention has become a strategy for the banking industry. There is a need to develop effective methods and models to accurately predict customer churn, thereby allowing banks to take proactive measures to retain customers.
- Customer retention affects a bank's profits, customer churn has a negative impact on current customers and the opportunities it creates for competitors.
- Amid the rapid development of data science and the availability of powerful tools such as deep learning, research is driven by the opportunity to leverage these cutting-edge techniques. Deep learning models are known for their ability to automatically extract complex patterns from raw data, which makes them well-suited for improving predictions related to customer churn.
- To reduce false predictions and avoid overfitting, the study combined several deep learning models to create greater efficiency.

When using deep learning methods, data privacy is really a top concern. Banks can still apply deep learning methods to predict churn while respecting privacy regulations. By appropriate data processing such as removing unimportant attributes, identifying attributes and personal information are presented in the data preprocessing content.

The research's main contribution: Uses several deep learning models consisting of two stages that aim to leverage the strengths of different deep learning architectures and then combine predictions from individual models to Improve overall customer churn prediction accuracy. These models learn from the same data and combining the output, the ensemble model can capture patterns that

✉ Van-Hieu Vu
vvhieu@ioit.ac.vn
https://www.kaggle.com/code/kmalit/bank-customer-churn-prediction/data

[1] Institute of information technology, Vietnam Academy of Science and Technology, 18 Hoang Quoc Viet Street, Cau Giay, Hanoi 123080, Vietnam

individual models may have missed. This two-stage approach is an effective technique to improve prediction accuracy and reliability.

In recent years with the rapid development of data science, organizations have got powerful tools to solve various real world problems. This is an opportunity to solve customer retention problems in banking [2–4] and other fields through the application of machine learning and deep learning techniques such as E-commerce business [5–8], telecommunication industry [9–15], consumers' purchasing behavior [16], financial [3, 17]

In light of these challenges, this paper aims to develop an efficient machine learning model specifically designed to predict customer service churn [6–8, 17–22]. Researchers have also explored the combination of multiple machine learning algorithms to tackle the customer churn prediction problems such as in [4, 10, 14, 23, 24].

Starting from the idea of using combined models, the proposal is made to combine several deep learning models. Deep learning models have demonstrated the ability to automatically extract complex patterns and representations from raw data, which improves predictions. By combining multiple deep learning models, the proposed method aims to take advantage of the strengths of each model and improve the overall accuracy of the churn prediction.

In the next section, some related work are presented in Sect. 2. The in-depth explanation of combination learning methods, which form the foundation of the proposed approach is presented in Sect. 3. The describe of architecture and workflow of the proposed model, including the selection of deep learning models and their respective configurations is presented in Sect. 4. In Sect. 5, the experimental part of the study will be discussed, evaluating the performance of the proposed model based on various metrics and comparing it with existing approaches. Finally, the conclusions of the study, summarizing the key findings and discussing their implications for the banking industry is presented in Sect. 6.

2 Related work

Currently, there are many works have proposed methods and models to solve the problem customer churn with many different methods and models. In Ref [25] have built a Stacking and Voting model system with four algorithms. The combination model consists of two levels with XGBoost (XGB) at level 1, and logistic regression (LR), decision tree (DT) and naive bayes classifier (NBC) at level 2. The results of the proposed system show a high degree of accuracy the best accuracy is 96.12% and 98.09% for old and new data sets, respectively, compared with other prediction systems. In another study [26], the authors have built 6 classification models, in turn, the classification models obtained the following: Logistic regression: 77.30%, decision tree: 78.93%, random forest: 87.22%, k-neighbors neighbors: 83.52%, AdaBoost: 83.55%, gradient boosting: 84.99% and XGBoost: 86.72%. In [27] used on the customer data set of a US bank from Kaggle to predict customer churn and the grid search method used to find the best hyper-parameters. To predict customer churn, the XGBoost algorithm achieved an accuracy of 84%, an accuracy of 83%, a recall of 84%, and an F1 score on the test set of 84%. In Ref [28] develop a customer service churn prediction approach with three smart models Random Forest (RF), AdaBoost and Support Vector Machine (SVM). This approach achieves the best results when applying the SMOTE Technique to overcome the unbalanced data set. The SMOTED data-driven method produced results with an F1 score of 91.90 and an overall accuracy of 88.7%.

In Ref [29] using supervised machine learning, Using ML, the gap will be solved between customers leaving and staying. An approach using the K-nearest neighbor (KNN) algorithm in which the dataset is appropriately grouped for the training and testing model depending on the weight ratios together with the XGBooster algorithm for improved accuracy.

In Ref [18] an empirical analysis of the impact of various hyper-parameters is presented when DNN is used to predict exit in the banking sector. The results from three experiments show that the deep neural network (DNN) model performs better when an activation function correction is used in the hidden layers and a sigmoid function is used in the output layer. The performance of DNN is better when the batch size is smaller than the test set data size; while, the RemsProp training algorithm has better accuracy when compared to the stochastic gradient descent (SGD) algorithm, Adam, AdaGrad, Adadelta and AdaMax.

Stacking involves using a machine learning model to learn how to best combine predictions from contributing team members. In voting, the members of the group are often a diverse set of model types, such as decision trees, naive Bayes, and support vector machines. Prediction is made by averaging the predictions, such as choosing the class with the most votes (statistics mode) or the greatest composite probability [30]. In [31] provided an efficient early detection solution using modern techniques by stacking data mining algorithms. Research results show that the integration of support vector machine (SVM) with Chi-squared automatic interaction detection decision tree (CHAID) brings good results. The results show the appropriate accuracy of the proposed service displacement prediction solution. In addition, the use of the superposition technique contributes to improving the results of customer churn detection.

3 Combination learning models

Combination model consisting of the average set of a combination of predictions from multiple trained models. One limitation of this approach is that each model contributes an equal amount to the overall predict, regardless of how well the model performs. A variation of this approach, known as weighted average aggregation, weighs the contribution of each member of the population by the model's confidence or expected performance on the data set. This allows good-performing models to contribute more and poor-performing models to contribute less. The weighted mean set provides an improvement over the model's mean set.

Combined model used in Ref [19], a machine learning model was used to build a predictive model by receiving input from a two-step cluster. In Ref [14] proposed ensemble models include combinations of other ensembles by combined learners as in composition of other learners. The ensemble models in Ref [32] were evaluated on prediction of the cryptocurrency price on the following hour. The experimental analysis indicates that ensemble learning and deep learning can be efficiently beneficial to each other. In Ref [33] propose a new combined classifier based on three pre-trained CNN models (VGG19, DenseNet169 and NASNetLarge) to process the ImageNet database and achieve high classification accuracy. In this proposed model, a transfer learning model based on each pre-trained model is built as a candidate classifier and the optimal output of three candidate classifiers is selected as the final classification result.

A more general way of this approach is to replace the linear weighted sum model (e.g., linear regression) used to combine the predictions of the submodels with any learning algorithm. This approach is called stacking generalization, or stacking for short.

On combination, an algorithm takes the output of the submodels as input and tries to learn how to best combine the input predictions to make a better output prediction. The stacking process can be visualized with two levels: Level 0 and Level 1 as shown in Fig. 1.

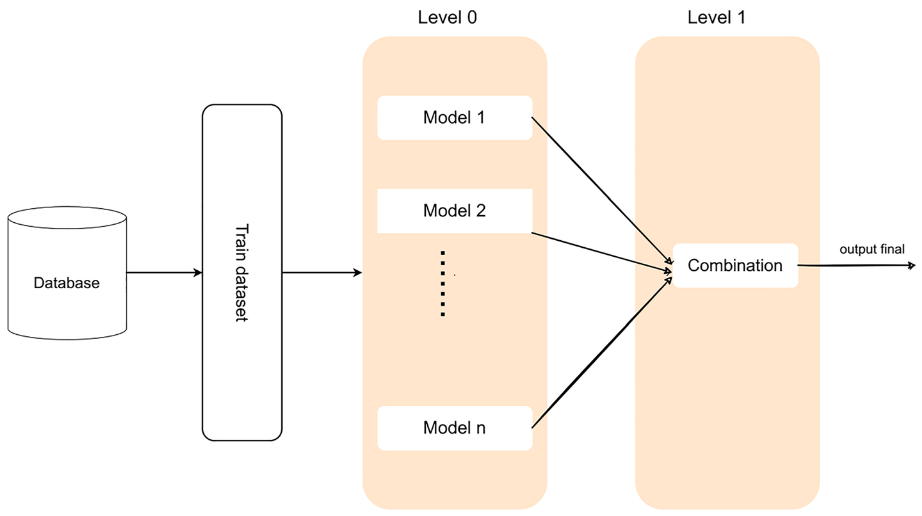
Interestingly, although stacking is described as a combinatorial learning method with two or more level 0 models, it can be used in cases where there is only a single level 0 model. In this case, the level 1 model, or meta-learner, learns to correct the predictions from the level 0 model. It is important that the meta-learner is trained on a separate dataset from the samples used to train the zero-level models to avoid over-fitting. A simple way to achieve this is to break down the training dataset into a training and test set. Level 0 models are then trained on the training set. The level 1 model is then trained using a test set, where the raw inputs are first passed through the level 0 models to get the predictions used as inputs to the level 0 model first. One limitation of the waiting test set approach for training stacked models is that Level 0 and Level 1 models are not trained on the full data set.

A stacked generalization combination can be developed for regression and classification problems. In the case of classification problems, better results have been seen when using the predictive probability of the class as input for meta-learning instead of the class label.

4 Proposed model

In this study, a combine model is proposed to predict bank customer churn. We use a method that combines multiple models (short as Level 0) into one final model (short as Level 1) as in [34] to achieve higher prediction accuracy. In the Level 0 models will be trained on the training data

Fig. 1 Combination model



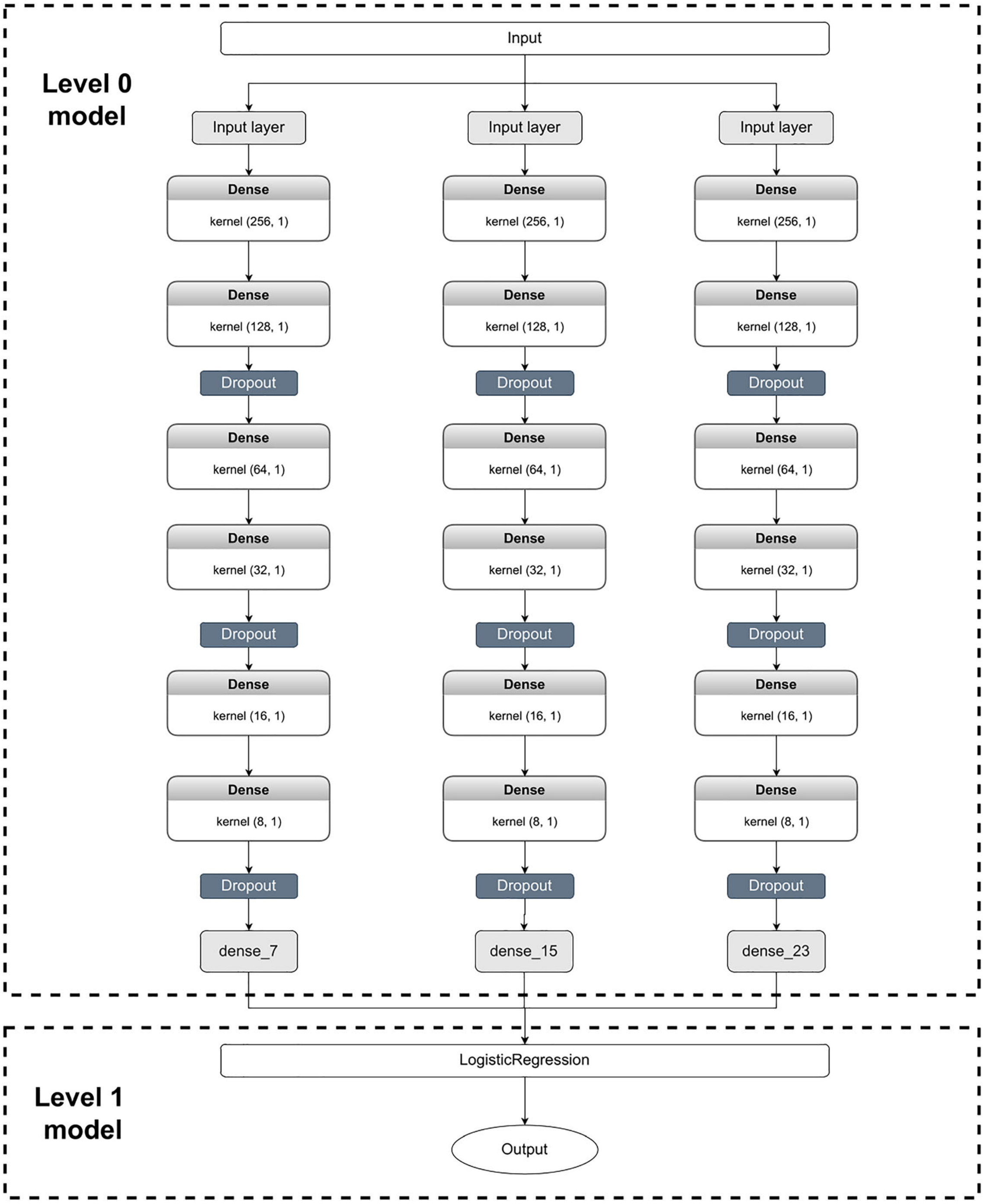


Fig. 2 Proposed system **ComDNN-LG**

set. Finally, the Level 1 model will be trained based on the predictions from the Level 0 models as input data.

4.1 Combined deep learning models

To solve the bank's customer churn prediction problem, some previous studies proposed a machine learning (ML)-based computer technique to distinguish between customers who are likely to leave and stay. Limitations, ML operates in a manual feature-based approach and lacks representative features in customer data at the bank. To overcome those limitations, deep learning (DL) is proposed in the pre-train phase and fine-tuned to generate an automatic feature representation for multi-class classification of service disengagement situations. In the pre-training phase, stacked learning sets are used for feature learning; in the

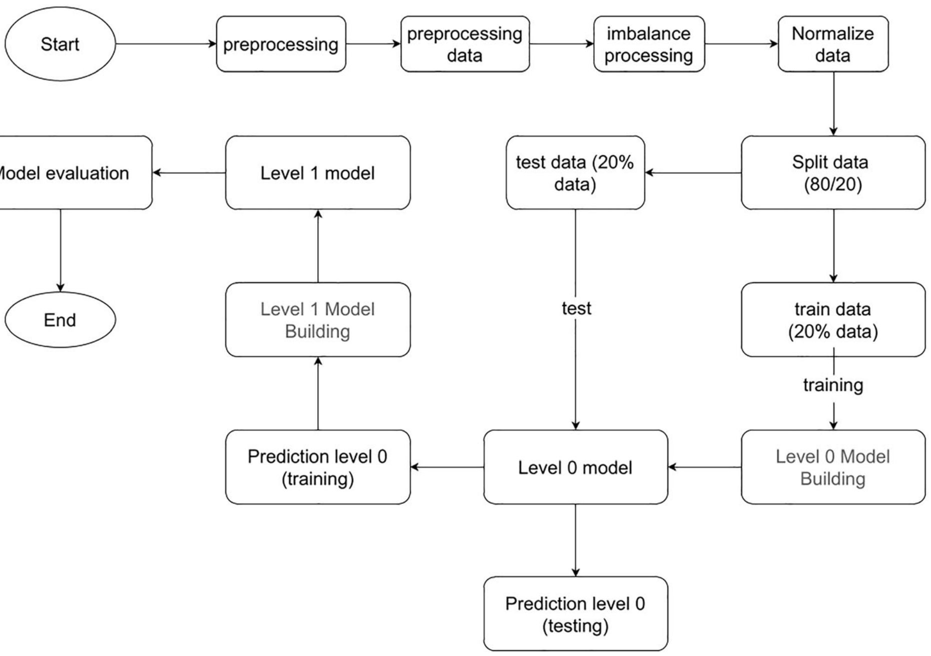


Fig. 3 Block diagram of ComDNN-LG the proposed system

fine-tuning phase, a deep neural network (DNN) is implemented as a classifier.

The two-stages process of combining several DNN models is as follows:

- Stage 1 (Training each deep learning model separately): Several separate deep learning models with different architectures are trained separately on the same data set. These models learn and make predictions about customer churn on each data sample.
- Stage 2 (Model Combination): Predictions from several individual deep learning models created in stage 1 are combined to produce the final prediction. An ensemble is created by taking as input the predictions made by each of the individual models on the data. The ensemble model learns to combine these individual predictions to produce a final prediction. The resulting ensemble model is usually a simple model (in this case a K-Nearest Neighbors or Logistic Regression classifier), which takes the predictions made by the individual models as input. It then uses this set to make predictions on the test data, giving a final prediction of customer churn prediction.

This two-stages process is implemented in a two-level system architecture as shown in Fig. 2.

The proposed model is short to as **ComDNN-LG** with Level 0 consisting of many so-called base models. In Level 0, the basic models are responsible for making predictions that characterize the model at Level 1. In Level 1, there will be a model called the metamodel, which is responsible for making the final predictions from the features that are the predictions of the base models. We use Level 0 as the base models consisting of three Deep Learning Network (DNN) classification models called **DNN 1**, **DNN 2** and **DNN 3**. In Level 1 (Combination level) is a Logistic Regression model. The structure of the implementation steps of the proposed model is as Fig. 3.

An artificial neural network is architecturally a stack of many interconnected layers, consisting of three layers: Input layers, hidden layers and output layers). When the structure of the neural network in the hidden layer is more than or equal to two, the neural network is called a deep learning neural network (DNN) [35]. The training and prediction process is performed according to the Algorithm 1.

Algorithm 1 Deep learning network combinatorial algorithm

```
Input:
    Train dataset $\mathbb{D} \Leftarrow \{\mathbb{X}_i, y_i\}_{i=1}^{m}$,
    $\mathbb{X}_i \in \mathbb{R}^n$, $y_i \in \mathbf{Y}$
Output: combinatorial classifier $\mathbf{H}$
 1: Initialization: deep learning models $\{modelDNN\}_{t=1}^{\mathbf{T}}$
 2: split training and test data: $\{\mathbb{D}_{train}, \mathbb{D}_{test}\} \Leftarrow \mathbb{D}$
 3: for $t = 1$ to $\mathbf{T}$ do
 4:     $modelDNN_t.fit(\mathbb{D}_{train}, \mathbb{D}_{test})$                ▷ train model
 5:     save( $modelDNN_t$)                                     ▷ save model
 6: end for
 7: $combineX \Leftarrow \emptyset$
 8: for $t = 1$ to $\mathbf{T}$ do
 9:     $modelDNN_t \Leftarrow$ load($modelDNN_t$)                   ▷ load models
10:     $\hat{\mathbf{y}}_t \Leftarrow modelDNN_t.predict(\mathbb{D}_{test})$            ▷ predict by model
11:     if $combineX == \emptyset$ then
12:         $\{combineX_t, y_t\} \Leftarrow \{\mathbb{D}_{test}, \hat{\mathbf{y}}_t\}$
13:     else
14:         $\{combineX_t, \mathbf{y}_t\} \Leftarrow dstack((combineX_t, \hat{\mathbf{y}}_t))$    ▷ Stacked library by
    numpy.dstack
15:     end if
16: end for
17: Initialization: $model \Leftarrow LogisticRegression()$
18: Training data $\{combineX\}$ in LogisticRegression model
19: for $t = 1$ to $\mathbf{T}$ do
20:     $model.fit(combineX_t, \mathbf{y}_t)$
21:     $\hat{\mathbf{y}}_t \Leftarrow model.predict(combineX_t)$
22: end for
23: Return: $H = \{\{combineX\}, \{\hat{\mathbf{y}}\}\}$
```

Table 1 Database description

Feature name	Feature description
Row number	Number row from 1 to 10000
Customer Id	Id unique to identify bank customers
Surname	Customer’s surname
Credit score	Customer’s credit score
Geography	Customer’s Country
Gender	Male or Female
Age	Age of customer
Tenure	Number of years customers have been with the bank
Balance	Customer’s bank balance
Num of products	The number of banking products the customer is using (savings accounts, mobile banking, online banking, etc.)
Has Cr card	The binary flag indicates whether the customer has a credit card with the bank
Is Active Member	Binary flags indicate whether the customer is an active member of the bank
Estimated salary	Estimated Client Salary in Dollars
Exited	Binary flag 1 if customer has closed account with bank and 0 if the customer has not closed the account

Table 2 Data statistics

	Count	Mean	Std	Min	Max
Credit score	10000.0	650.5288	96.653299	350	850
Age	10000.0	38.921800	10.487806	18.00	92
Tenure	10000.0	5.012800	2.892174	0	10
Balance	10000.0	76485.889288	62397.405202	0.00	250898.09
Num of products	10000.0	1.5302	0.581654	1.0	4.0
Has Cr card	10000.0	0.7055	0.455840	0.00	1.00
Is active member	10000.0	0.5151	0.499797	0.0	1.0
Estimated salary	10000.0	100090.239881	57510.492818	11.58	199992.48
Exited	10000.0	0.2037	0.402769	0.0	1.0

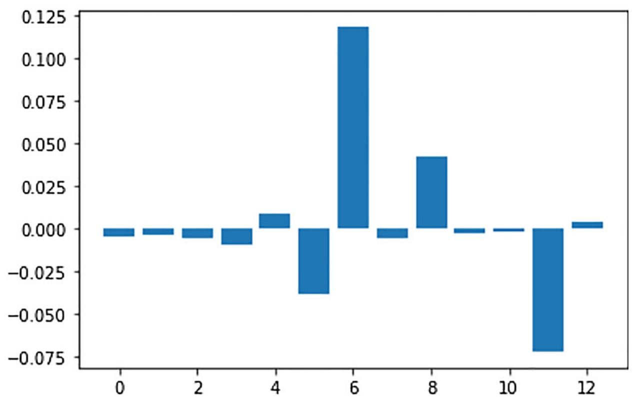


Fig. 4 Representing the importance of features

4.2 Characteristics of the combined deep learning model

In the pre-training phase, stacked learning sets are used for feature learning, a deep neural network is implemented as a classifier. At Level 0 comprises three DNNs classification models which make prediction features for the Level 1 model. In Level 1, a Logistic Regression model is used to make the final predictions based on the predictions of the base models.

The proposed model's architecture is depicted in Fig. 2, and the implementation steps are outlined in Fig. 3. The DNNs consists of input layers, hidden layers, and output layers. When the hidden layer structure contains two or more layers, it is referred to as a DNN.

The training and prediction process of the proposed model follows the algorithm described in Algorithm 1. The algorithm initializes the deep learning models and splits the training and test data. It then iterates over the base models to train them and save the trained models. Next, the algorithm combines the predictions from the base models into a stacked representation. Finally, a Logistic Regression model is initialized, trained on the combined data, and used to make predictions.

Overall, the proposed model aims to leverage the strengths of both ML and DL techniques to improve the accuracy of bank customer churn prediction. By combining multiple base models and incorporating feature representation through pre-training aims to improves accuracy predictions compared to traditional ML approaches.

4.3 Description of the proposed model algorithm

The proposed method is a superimposed model designed to enhance prediction accuracy by leveraging a stacking technique that combines multiple Level 0 models into a Level 1 model. The entire process involves several steps, including data preparation, Level 0 model training, Level 1 model training, and prediction.

4.3.1 Data preparation

After the data set is divided into training and test sets, the training dataset is used for training Level 0 DNN models. The test dataset is used to evaluate the performance of the final model (Level 1).

Table 3 Deep neural network training parameters

Epoch	Batch_size	Activation function	Optimizer
{50, 100, 150, 200, 220, 250, 300, 500}	{7,16, 32, 64, 128}	{sigmoid, tanh, ReLU}	{adam, RMSProp}

Table 4 Level 0 model parameters

Level 0 model	Layer		Dropout	Activation function	Optimal	Batch_size
DNN 1	Conv1	256 128	0.1	Relu	RMSProp	7
	Conv2	64 32	0.1	Relu		
	Conv3	16 8	0.1	Relu		
DNN 2	Conv1	256 128	0.1	Tanh	RMSProp	7
	Conv2	64 32	0.1	Tanh		
	Conv3	16 8	0.1	Tanh		
DNN 3	Conv1	256 128	0.1	Tanh	Adam	7
	Conv2	64 32	0.1	Tanh		
	Conv3	16 8	0.1	Tanh		

4.3.2 Level 0 model training

The Level 0 models play a crucial role in the proposed method as they are responsible for making predictions that contribute to the final prediction at Level 1. In this approach, the Level 0 models are designed as deep learning network classification models, specifically referred to as DNN 1, DNN 2, and DNN 3.

Each Level 0 model is trained using the training data set. The training process involves feeding the customer data into the DNN models and adjusting the model parameters to minimize the prediction error. After training, the trained Level 0 models are saved for later use.

4.3.3 Level 1 model training

The Level 1 model makes the final predictions based on the predictions of the Level 0 models. In this proposal, the Level 1 model is the Logistic Regression model. The prediction results from the Level 0 models are combined to create a new set of features. Then, the Logistic Regression model is trained using this new set of features. The training process involves adjusting the parameters of the Logistic Regression model to optimize its performance in prediction customer churn.

4.3.4 Prediction

Once the Level 1 model is trained, it can be used to make predictions on new, unseen data. To make a prediction, the customer data is first passed through each of the Level 0 models to generate predictions at the Level 0 model. These predictions are then combined and fed into the Level 1 model, which generates the final prediction of whether a customer is likely to churn or not.

By combining the predictions of multiple Level 0 models and using a Level 1 model to make the final prediction, the proposed method aims to leverage the strengths of both ML and DL techniques. The Level 0 models capture different aspects of the customer churn problem, and the Level 1 model learns to weigh and combine these predictions to make a more accurate overall prediction. The use of deep learning in the Level 0 models allows for learning feature representation, which helps to overcome the limitations of manual feature engineering in traditional ML approaches.

The effectiveness of the proposed method can be evaluated by measuring its prediction accuracy on the test data set. Additionally, other performance metrics such as precision, recall, and F1 score can be used to assess the model's ability to correctly identify churned customers.

5 The experiment

5.1 Dataset description

Because of the law on the protection of personal information, commercial banks will not have the right to disclose information about customers' accounts and transactions. Therefore, in this study, we will experiment on the dataset "Churn_Modelling.csv" downloaded from

Table 5 Evaluate performance DNN 1 model

DNN 1								
Epochs	Precision (%)	Recall (%)	F1-score (%)	Acc train (%)	Acc test (%)	Loss train (%)	Loss test (%)	Time (s)
50	83.78	83.78	83.78	90.90	83.80	22.50	43.80	98.00
100	86.34	86.01	86.06	94.50	86.10	13.90	86.10	180.73
150	86.72	86.65	86.59	95.20	86.60	12.30	63.50	280.03
200	88.13	88.12	88.12	97.60	88.10	7.20	58.00	372.37
220	86.16	86.10	86.10	97.20	86.10	7.80	68.40	503.16
250	87.03	86.65	86.62	98.50	86.70	4.50	67.80	509.70
300	86.42	86.14	86.14	96.90	86.20	7.90	68.80	665.79
500	88.09	87.82	87.72	99.30	87.70	2.20	144.10	878.35

Table 6 Evaluate performance DNN 2 model

DNN 2								
Epochs	Precision (%)	Recall (%)	F1- score (%)	Acc train (%)	Acc test (%)	Loss train (%)	Loss test (%)	Time (s)
50	81.81	81.77	81.77	84.40	81.80	37.10	41.40	101.05
100	86.49	86.38	86.40	93.70	86.40	18.50	35.40	182.19
150	85.27	85.24	85.19	94.10	85.20	17.60	38.50	299.37
200	88.36	88.34	88.34	97.10	88.30	9.20	36.10	377.89
220	88.41	88.11	88.17	96.70	88.20	10.40	38.90	487.08
250	88.29	88.05	88.03	98.00	88.10	6.80	40.40	530.00
300	87.26	86.90	86.90	96.80	86.90	9.70	40.10	531.91
500	88.23	88.03	87.95	99.10	88.00	4.00	38.90	901.70

Table 7 Evaluate performance DNN 3 model

DNN 3								
Epochs	Precision (%)	Recall (%)	F1-score (%)	Acc train (%)	Acc test (%)	Loss train (%)	Loss test (%)	Time (s)
50	82.90	82.76	82.74	86.50	82.80	30.20	38.90	94.93
100	83.92	83.79	83.81	91.00	83.80	23.00	37.60	194.60
150	83.57	83.45	83.36	94.10	83.40	22.80	38.50	283.29
200	81.25	81.21	81.20	81.70	81.20	40.70	41.80	372.13
220	85.86	85.72	85.76	93.60	85.80	17.40	36.40	412.00
250	79.98	79.90	79.88	82.40	79.90	40.70	44.30	484.83
300	81.72	81.72	81.71	82.30	81.70	40.20	41.50	790.65
500	80.43	80.43	80.41	83.70	80.40	37.00	43.30	911.78

Kaggle.com [36] on July 27, 2022. The generated data set does not belong to any bank at all, only describes the data that may exist in the bank. The data set is for research and experimental purposes, in the dataset there are 14 data fields and 10,000 observations. The data field "Exited" represents the customer leaving, of which there are 2037 leaving observations labeled as '1' accounting for 20.37% and there are 7963 non-disengaging observations labeled as '0' accounts for 79.63%. Details of these features are given in Table 1.

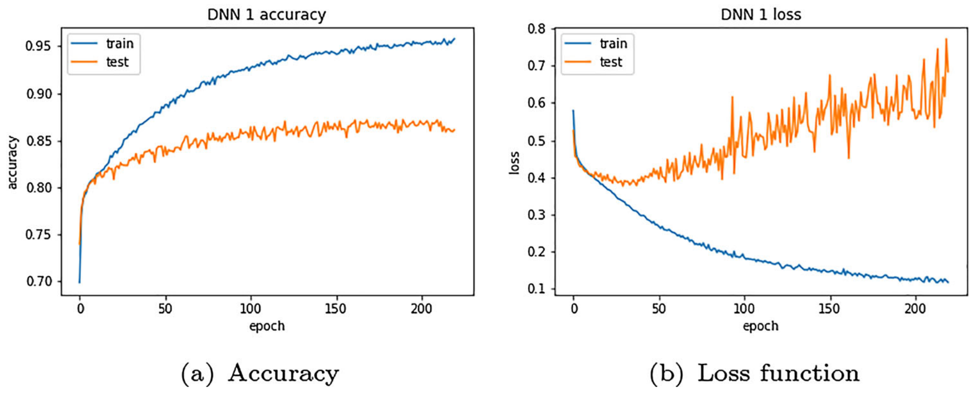


Fig. 5 Evaluate Accuracy and Loss for **DNN 1** model

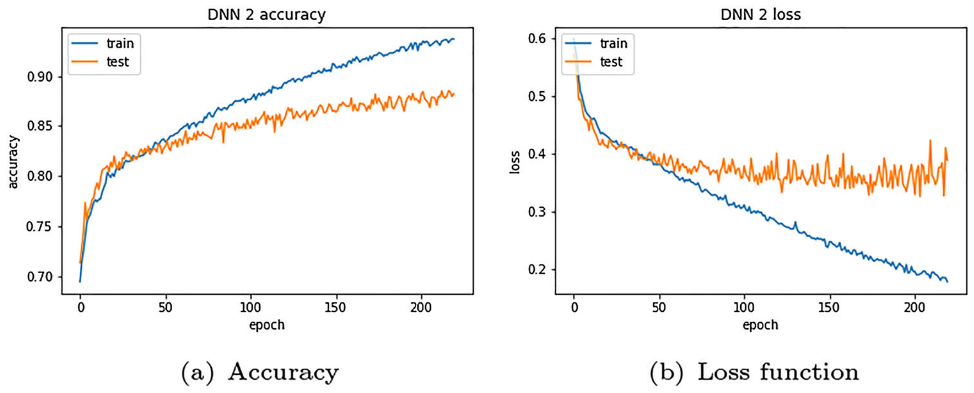


Fig. 6 Evaluate Accuracy and Loss for **DNN 2** model

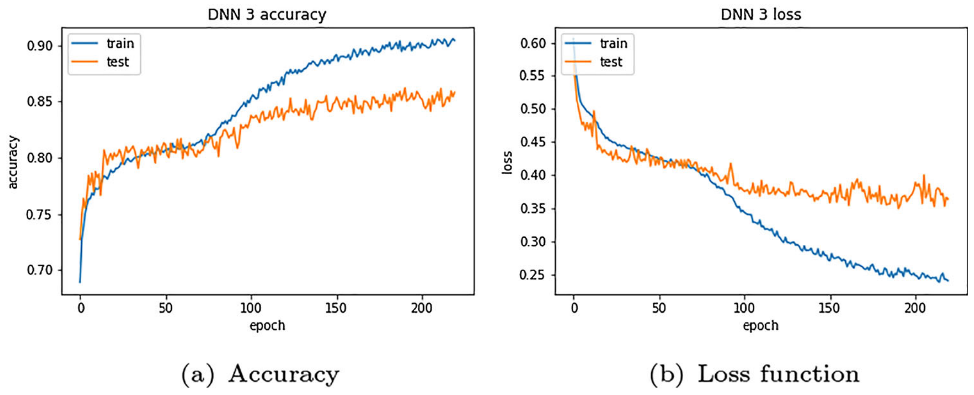


Fig. 7 Evaluate Accuracy and Loss for **DNN 3** model

Table 8 Evaluate performance when epoch = 50

Epochs = 50				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	59.66	54.75	65.09	78.13
Recall	88.72	75.39	74.10	70.51
F1-score	71.34	63.43	69.31	74.12
Accuracy	86.10	83.10	87.20	90.40

5.2 Data preprocessing and cleaning

First of all, the data needs to be preprocessed and cleaned. This is an important stage in the data mining process because it directly affects the success rate of the problem. Therefore, it is necessary to look for irrelevant, noisy and unreliable data, which can be transformed when necessary. Descriptions of the data fields used for prediction after

Table 9 Evaluate performance when epoch = 100

Epochs = 100				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	69.87	71.51	61.85	85.75
Recall	93.45	91.69	89.42	87.91
F1-score	79.96	80.35	73.12	86.82
Accuracy	90.70	91.10	86.90	94.70

Table 10 Evaluate performance when epoch = 150

Epochs = 150				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	74.09	71.87	64.31	85.43
Recall	92.89	88.83	90.10	87.82
F1-score	82.43	79.46	75.05	86.61
Accuracy	92.20	91.00	88.20	94.70

Table 11 Evaluate performance when epoch = 200

Epochs = 200				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	82.82	77.75	45.58	89.09
Recall	92.39	92.65	79.79	90.03
F1-score	87.35	84.55	58.02	89.56
Accuracy	94.90	93.60	78.00	96.00

Table 12 Evaluate performance when epoch = 220

Epochs = 220				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	77.23	75.68	69.94	90.26
Recall	94.78	94.26	91.12	91.91
F1-score	85.11	83.95	79.14	91.07
Accuracy	93.70	93.10	90.80	96.60

Table 13 Evaluate performance when epoch = 250

Epochs = 250				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	80.30	79.05	55.68	90.49
Recall	92.67	93.15	75.55	90.71
F1-score	86.04	85.52	64.11	90.60
Accuracy	93.80	93.60	82.70	96.20

Table 14 Evaluate performance with epoch = 300

Epochs = 300				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	78.37	78.41	55.81	88.70
Recall	93.02	95.72	76.80	91.89
F1-score	85.07	86.21	64.65	90.27
Accuracy	92.80	93.20	81.30	95.60

Table 15 Evaluate performance with epoch= 500

Epochs = 500				
	DNN 1 (%)	DNN 2 (%)	DNN 3 (%)	ComDNN-LG (%)
Precision	85.15	83.92	55.11	92.03
Recall	95.09	95.09	79.44	94.39
F1-score	89.85	89.16	65.07	93.20
Accuracy	95.40	95.10	81.70	97.10

preprocessing are listed in Table 2. These are the features used to decide the prediction in this study.

5.2.1 Remove unimportant features and data transformation

Feature importance refers to the scoring of all input features for a given model. Scoring is simply a representation of the “importance” of each feature. A higher score means that the particular feature will have a greater influence on the model being used to predict a particular variable. The removal of data or features that have no effect on the customer’s departure or stay in the bank will be considered as the removal of irrelevant data. Keeping such attributes can sometimes affect the performance of the classifier. While considering the data set, the features Row number, Customer Id, Surname, and Geography have nothing to do with the prediction. Therefore, these features will be removed. In Fig. 4 show that the feature 0, feature 1, feature 2, and feature 3 can be removed.

Data transformation is the transformation from one form to another. The data is properly checked and structured to improve data quality and protect in the application from occurrences such as nulls, duplicates, incorrect indexes, and incompatible formats. For example convert gender Female to 0 and Male to 1.

5.3 Deep neural network parameters and models

During training of deep learning models, different epochs are used to tune model parameters, avoid overfitting, and stop training when necessary. Using multiple epochs allows the model to gradually approach a more optimal point and adjust parameters to optimize performance. Besides, combining some activation functions and optimizers. This combination creates performance optimization, this fine-tuning can find the optimal combination to achieve the highest performance for the network architecture. Choosing the best batch size for a model often depends on the model architecture, data set size, and resources. The use of multiple batch sizes in experiments is to determine the most suitable batch size for the model. The table 3 below shows the parameters used.

In this proposal, we use a hybrid architecture of three deep neural network models. The selection of the number of models and network configurations is analyzed based on experiments and experimental validation for the problem and analyzed data set. This combination aims to determine the optimal combination for the problem based on the size of the data set, computational resources and model performance. In training, the combined use of many models requires large computational resources. Sometimes combining too many training models on a small data set can cause overfitting to occur. Choosing a sufficient number of combined models brings efficiency to the problem, ensures architectural diversity, simple maintenance process, and achieved computing capacity are requirements of the system.

5.4 Experimental results

According to the proposed model architecture, the Level 0 deep learning networks are first set up with the parameters as Table 4.

Three different Deep Neural Network models, namely DNN 1, DNN 2, and DNN 3 at Level 0 were trained with distinct configurations and optimization algorithms. The architecture in each model uses Conv1, Conv2 and Conv3 layers. These are commonly used convolutional layers for feature extraction in deep learning models. The activation functions and optimization algorithms used in each model are as follows:

- DNN 1: ReLU activation function with the Optimal optimizer.
- DNN 2: Tanh activation function with the RMSProp optimizer.
- DNN 3: Tanh activation function with the Adam optimizer.

5.4.1 Activation functions and optimization algorithms

DNN 1 employs the Rectified Linear Unit (ReLU) activation function and the Optimal optimizer. ReLU is known for its ability to handle nonlinear relationships effectively, enabling the model to capture complex patterns in the data. The Optimal optimizer is utilized to optimize the model’s

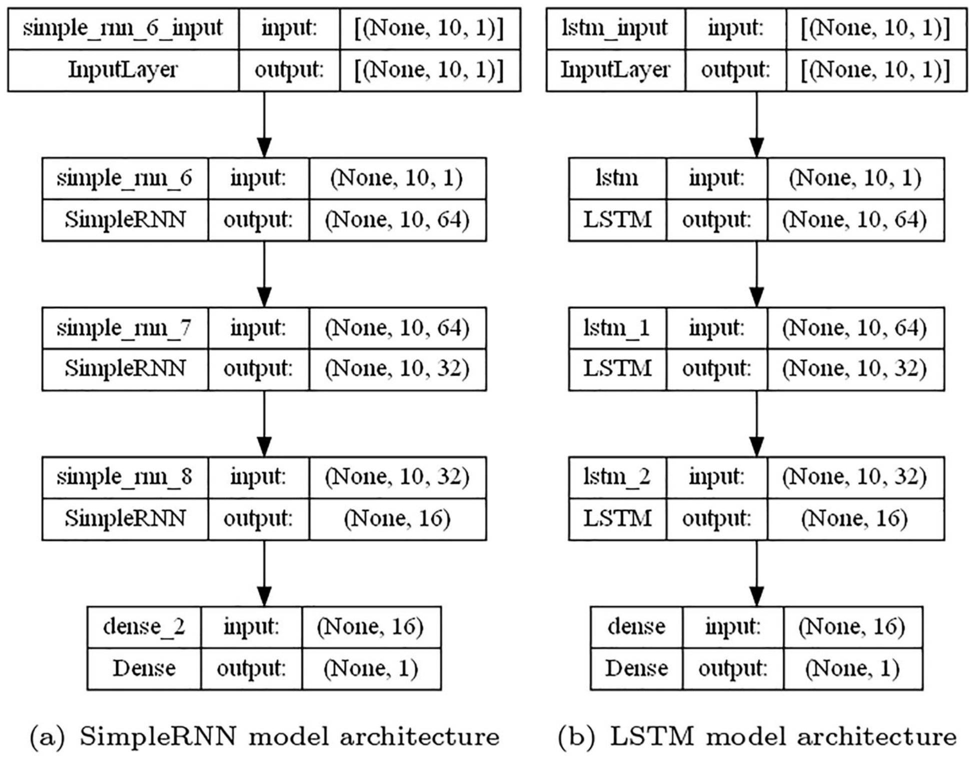

simple_rnn_6_input | input: | [(None, 10, 1)]
InputLayer | output: | [(None, 10, 1)]
simple_rnn_6 | input: | (None, 10, 1)
SimpleRNN | output: | (None, 10, 64)
simple_rnn_7 | input: | (None, 10, 64)
SimpleRNN | output: | (None, 10, 32)
simple_rnn_8 | input: | (None, 10, 32)
SimpleRNN | output: | (None, 16)
dense_2 | input: | (None, 16)
Dense | output: | (None, 1)
(a) SimpleRNN model architecture
lstm_input | input: | [(None, 10, 1)]
InputLayer | output: | [(None, 10, 1)]
lstm | input: | (None, 10, 1)
LSTM | output: | (None, 10, 64)
lstm_1 | input: | (None, 10, 64)
LSTM | output: | (None, 10, 32)
lstm_2 | input: | (None, 10, 32)
LSTM | output: | (None, 16)
dense | input: | (None, 16)
Dense | output: | (None, 1)
(b) LSTM model architecture

Fig. 8 Architecture of SimpleRNN and LSTM models

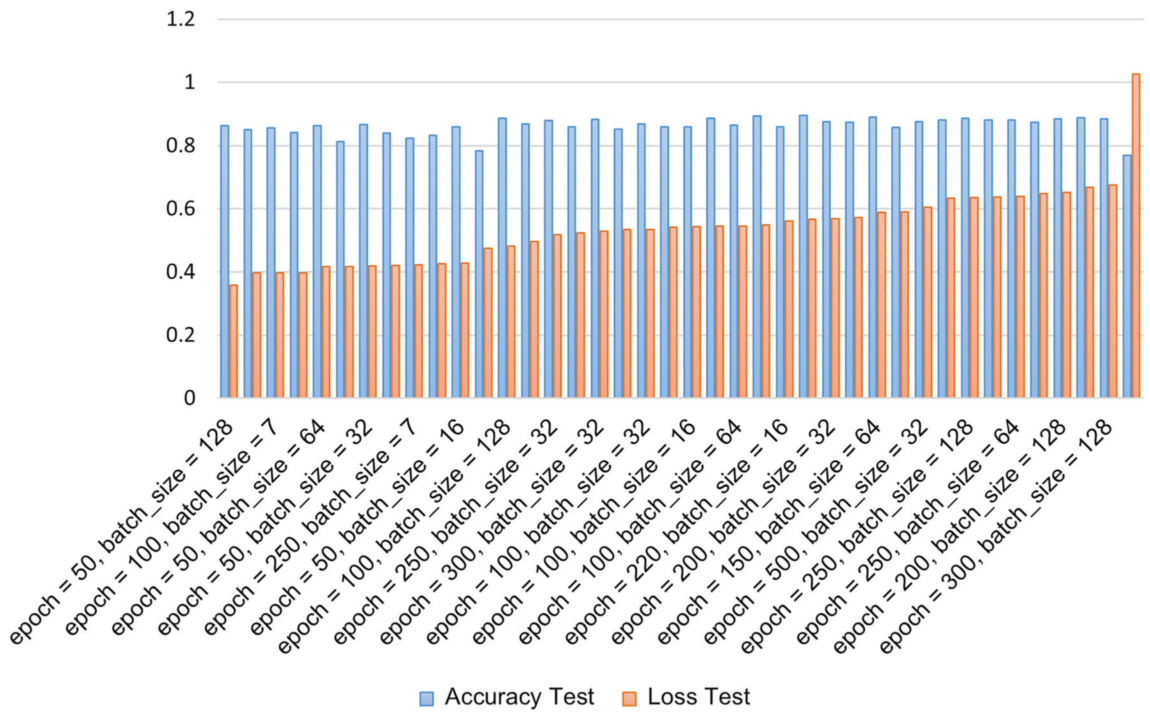


Fig. 9 Performance evaluation over different epochs and batch_sizes of the SimpleRNN architecture

weights and biases during training, aiming to minimize the loss function and improve overall performance.

DNN 2 and DNN 3 both utilize the hyperbolic tangent (Tanh) activation function, which can also capture nonlinear relationships in the data. However, they employ different optimization algorithms: DNN 2 uses the RMSProp optimizer; while, DNN 3 uses the Adam optimizer. RMSProp is an adaptive learning rate optimization

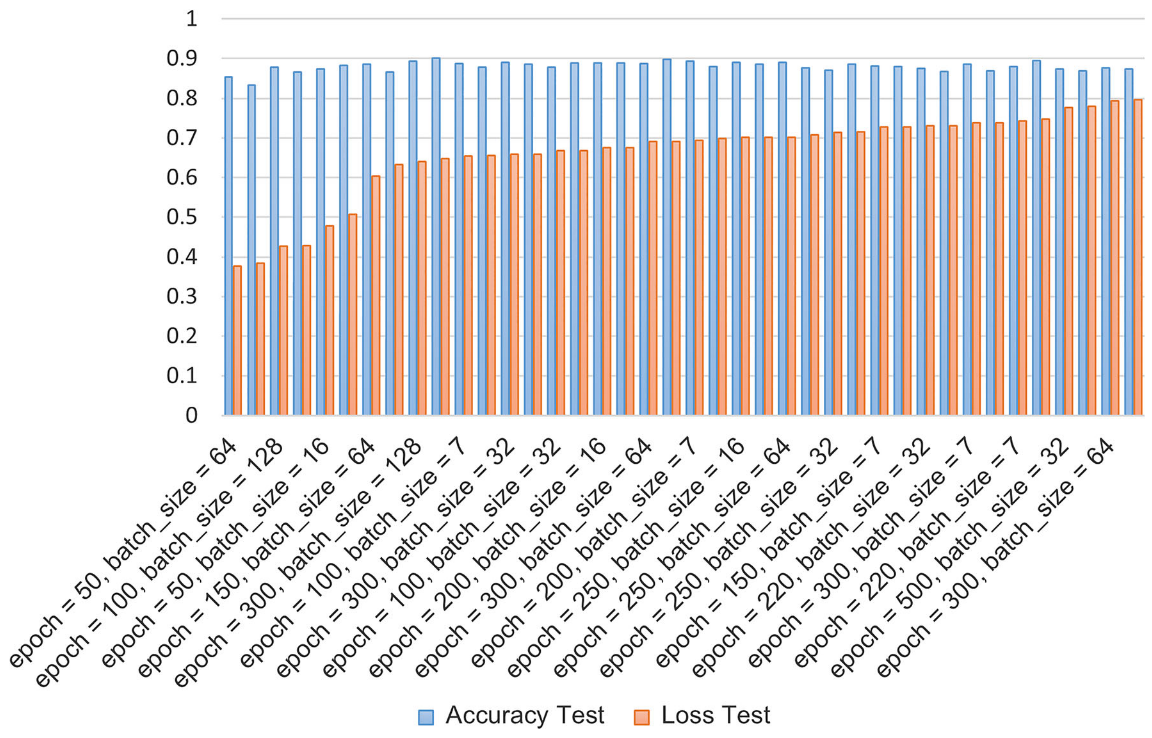


Fig. 10 Evaluate performance under different epochs and batch_size of LSTM architecture

Table 16 Compare the proposed with other methods

	ComDNN -LG (%)	DNN (%)	XGB* [29] (%)	KNN [29] (%)	Simple- RNN (%)	LSTM (%)
Precision	90.26	85.90	86.85	83.85	86.38	85.47
Recall	91.91	88.27	87.95	86.03	86.26	85.38
F1-score	91.07	94.92	96.73	95.16	86.32	85.42
Accuracy	96.60	91.47	92.13	90.36	86.30	85.34

algorithm that helps the model converge faster by adjusting the learning rate based on the recent gradients. Adam, on the other hand, combines the advantages of both RMSProp and AdaGrad algorithms and is known for its efficient handling of large datasets.

5.4.2 Adjusting the number of epochs during the training

In the experiment, the epochs in each DNN model are set to multiple values to evaluate and select the appropriate value when having the best performance results. If the number of epochs is too low, the model may not have enough iterations to learn complex patterns in the data. On the other hand, choosing large epochs can lead to overfitting because it is very good on the training data but not very well on the unseen data.

Finding the Optimal Epoch: By training the model with different epoch values, that can observe how the performance metrics evolve. Initially, as the number of epochs increases, the model's performance improves as it learns more from the data.

For the best performance during Level 0 training, epoch-based learning is used to find the best performance and avoid overfitting. Selected epochs include 50, 100, 150, 200, 220, 250, 300 and 500 on DNN 1, DNN 2 and DNN 3 models that have been evaluated for precision, recall, f1 score, accuracy train (acc train), accuracy test (acc test). Tables 5, 6, and 7 evaluate these metrics for DNN models at Level 0. According to this evaluation, we see that the highest performance is achieved when the epoch is 220.

The results are shown in Figs. 5, 6 and 7 with the best choice of epoch being 220. Accuracy and function evaluation charts loss (loss) on the training set (train) and the test set (test). The results on the graph show that the selection model effectively avoids over-fitting in training.

This suggests that after 220 epochs, the model's performance on the test data starts to decline, indicating the onset of overfitting. Thus, 220 epochs strike the optimal balance between capturing relevant patterns and avoiding overfitting for the Level 0 models.

Adjusting the epoch value is an essential hyperparameter tuning step in deep learning. It helps to ensure that the model learns the necessary information from the data without overfitting. By finding the optimal epoch value, we can improve the model's generalization ability and achieve better performance on unseen data.

5.4.3 Compare the combined model with the individual models

In this section, we demonstrate the efficiency of epoch selection in deep learning models. The deep learning models will also be compared with the proposed **ComDNN-LG** model in the following metrics: precision, recall, f1-score and accuracy. Evaluation results according to the metrics in Tables 8, 9, 10, 11, 12, 13, 14 and 15 show that the proposed model is more efficient.

Table 8 shows that when choosing epochs = 50, the proposed **ComDNN-LG** model shows a clear effect when outperforming the DNN 1, DNN 2 and DNN 3 models in precision (highest: 23.38% and lowest: 13.04%) and accuracy (highest: 7.3% and lowest: 3.2%).

Table 9 shows that when choosing epochs = 100, the proposed **ComDNN-LG** model shows more efficiency than the DNN 1, DNN 2 and DNN 3 models in precision metrics (highest: 23.9% and lowest: 14.24%) and accuracy (highest: 7.8% and lowest: 3.6%).

Table 10 shows that when choosing epochs = 150, the proposed model **ComDNN-LG** is more efficient than the DNN 1, DNN 2 and DNN 3 models in precision metrics (the highest: 21.12% and lowest: 11.34%) and accuracy (highest: 6.5% and lowest: 2.5%).

Table 11 shows that when choosing epochs = 200, the proposed model **ComDNN-LG** is more efficient than the DNN 1, DNN 2 and DNN 3 models in precision metrics (the highest: 43.51% and lowest: 6.27%) and accuracy (highest: 18% and lowest: 1.1%).

Table 12 shows that when epochs = 220, the proposed model **ComDNN-LG** is more efficient than the DNN 1, DNN 2 and DNN 3 models in precision metrics (highest: 20.32% and lowest: 13.02%) and accuracy (highest: 5.8% and lowest: 2.9%).

Table 13 shows that when epochs = 250, the proposed model **ComDNN-LG** is more efficient than the DNN 1, DNN 2 and DNN 3 models in precision metrics (highest: 34.81% and lowest: 10.19%) and accuracy (highest: 13.5% and lowest: 2.4%).

Table 14 shows that when epochs = 300, the proposed model **ComDNN-LG** is more efficient than the DNN 1, DNN 2 and DNN 3 models in precision metrics (highest: 32.89% and lowest: 10.28%) and accuracy (highest: 14.3% and lowest: 2.4%).

Table 15 shows that when epochs = 500, the proposed model **ComDNN-LG** is more efficient than the DNN 1, DNN 2 and DNN 3 models at precision metrics (highest: 36.92% and lowest: 6.88%) and accuracy (highest: 15.4% and lowest: 1.7%).

5.4.4 Compare the proposed with other methods

To evaluate the dataset with the Recurrent Neural Network (RNN) deep learning network model, we use the architecture (Simple RNN and LSTM). SimpleRNN and LSTM consist of a sequence of SimpleRNN and LSTM layers stacked on top of each other, then ending with a Dense layer. Figure 8 is the architecture of these two models.

Experiments on the SimpleRNN model show that the best performance is when epoch = 50 and batch_size = 128 (Fig. 9). The LSTM model shows the best performance when epoch = 50 and batch_size = 64 (Fig. 10). In addition, we can see that the number of errors increases when the epoch or batch_size value increases, then training overfitting occurs.

The proposed technique is compared with the techniques in [18] and in Ref [29] on the same data set [36]. Table 16 shows the test results of the proposed model from which it can be seen that using a combination of deep learning networks gave higher performance in precision and accuracy metrics of 92.03% and 97.1%.

6 Conclusion

The article proposes a deep learning neural network combined learning method to help commercial banks have a tool to predict customer abandonment. Based on the combination approach, which combines multiple machine learning models together, improves the performance of the model. The proposed model consists of two levels, Level 0 includes three deep learning neural network models, and Level 1 is a Logistics regression model. Combining many models into one significantly increases the ability to detect leaving customers, the proposed model obtains evaluation metrics including accuracy of 96.6%, precision of 90.26%, recall is 91.91% and F1-score of 91.07%, respectively, when using an epoch value of 220. The proposed model provides banks with a method to quickly and accurately identify customers who intend to leave the service since then offer appropriate policies for customer churn.

Data availability The data that support the findings of this study are "Bank Turnover Dataset" openly available in https://www.kaggle.com/datasets/barelydedicated/bank-customer-churn-modeling.

Declarations

Conflict of interest The author declare that they have no conflict of interests regarding the publication of this article.

References

1. Rozum JA (2001) Defining and understanding software measurement data. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, Tech. rep.
2. Alizadeh M, Zadeh DS, Moshiri B, Montazeri A (2023) Development of a customer churn model for banking industry based on hard and soft data fusion. IEEE Access 11:29759–29768
3. Rudd DH, Huo H, Xu G (2022) Improved churn causal analysis through restrained high-dimensional feature space effects in financial institutions. Human-Centric Intell Syst 2:70–80
4. Bharathi SV, Pramod D, Raman R (2022) An ensemble model for predicting retail banking churn in the youth segment of customers. Data 7:61
5. Wu X, Li P, Zhao M, Liu Y, Crespo RG, Herrera-Viedma EE (2022) Customer churn prediction for web browsers. Expert Syst Appl 209:118177
6. Matuszelański K, Kopczewska K (2022) Customer churn in retail e-commerce business: spatial and machine learning approach. J Theor Appl Electron Commer Res 17:165–198
7. Theodoridis G, Tsadiras AK (2022) Applying machine learning techniques to predict and explain subscriber churn of an online drug information platform. Neural Comput Appl 34:19501–19514
8. Xiahou X, Harada Y (2022) B2c e-commerce customer churn prediction based on k-means and svm. J Theor Appl Electron Commer Res 17:458–475
9. Vo NNY, Liu S, Li X, Xu G (2021) Leveraging unstructured call log data for customer churn prediction. Knowl Based Syst 212:106586
10. Saha L, Tripathy HK, Gaber T, El-Gohary H, El-Kenawy ESMT (2023) Deep churn prediction method for telecommunication industry. Sustainability 15:4543
11. Pustokhina I.V, Pustokhin D.A, Nguyen P.T, Elhoseny M, Shankar K (2021) Multi-objective rain optimization algorithm with welm model for customer churn prediction in telecommunication sector. Complex & Intelligent Systems
12. Yu R, An X, Jin B, Shi J, Move OA, Liu Y (2018) Particle classification optimization-based bp network for telecommunication customer churn prediction. Neural Comput Appl 29:707–720
13. Xu J, Li X, He Z, Zhou J (2022) Early warning of telecom customer churn based on multialgorithm model optimization. Front Energy Res 10:946933
14. Ahmed M, Afzal H, Siddiqi I, Amjad MF, Khurshid K (2018) Exploring nested ensemble learners using overproduction and choose approach for churn prediction in telecom industry. Neural Comput Appl 32:3237–3251
15. Zhao M, Zeng Q, Chang M, Tong Q, Su J (2021) A prediction model of customer churn considering customer value: an empirical research of telecom industry in china. Discrete Dyn Nat Soc 2021:1–2
16. Hasumoto K, Goto M (2022) Predicting customer churn for platform businesses: using latent variables of variational autoencoder as consumers' purchasing behavior. Neural Comput Appl 34:18525–18541
17. de Lima Lemos RA, Silva TC, Tabak BM (2022) Propension to customer churn in a financial institution: a machine learning approach. Neural Comput Appl 34:11751–11768
18. Domingos E, Ojeme B, Daramola O (2021) Experimental analysis of hyperparameters for deep learning-based churn prediction in the banking sector. Computation 9:34
19. Al-Najjar D, Al-Rousan N, Al-Najjar HM (2022) Machine learning to develop credit card customer churn prediction. J Theor Appl Electron Commer Res 17:1529–1542
20. Feng L (2022) Research on customer churn intelligent prediction model based on borderline-smote and random forest. In: 2022 IEEE 4th international conference on power, intelligent computing and systems (ICPICS) pp 803–807
21. Liu R, Ali S, Bilal SF, Sakhawat Z, Imran A, Almuhaimeed A, Alzahrani A, Sun G (2022) An intelligent hybrid scheme for customer churn prediction integrating clustering and classification algorithms. Appl Sci 12:9355
22. Lalwani P, Mishra MK, Chadha JS, Sethi P (2021) Customer churn prediction system: a machine learning approach. Computing 104:271–294
23. Awang M.K, Makhtar M, Udin N, Mansor N.F (2021) Improving customer churn classification with ensemble stacking method. Int J Adv Comput Sci Appl
24. De S, Prabu P (2022) A sampling-based stack framework for imbalanced learning in churn prediction. IEEE Access 10:68017–68028
25. Xu T, Ma Y, ryeol Kim K (2021) Telecom churn prediction system based on ensemble learning using feature grouping. Appl Sci
26. Kumar S.L (2021) Bank customer churn prediction using machine learning. International Journal for Research in Applied Science and Engineering Technology
27. Zhang T (2022) Prediction and clustering of bank customer churn based on xgboost and k-means. BCP Business & Management
28. Muneer A, Ali RF, Alghamdi A, Taib SM, Almaghthawi A, Ghaleb EAA (2022) Predicting customers churning in banking industry: a machine learning approach. Indones J Electr Eng Comput Sci 26:539–549
29. Dalmia H, Nikil CVSS, Kumar S (2020) Churning of bank customers using supervised learning. In: Innovations in Electronics and Communication Engineering: Proceedings of the 8th ICIECE 2019. Springer Singapore. pp 681–691
30. Witten IH, Frank E (2002) Data mining: practical machine learning tools and techniques with java implementations. ACM SIGMOD Rec 31:76–77
31. Shabankareh MJ, Shabankareh MA, Nazarian A, Ranjbaran A, Seyyedamiri N (2021) A stacking-based data mining solution to customer churn prediction. J Relationsh Market 21:124–147
32. Livieris IE, Pintelas EG, Stavroyiannis S, Pintelas P (2020) Ensemble deep learning models for forecasting cryptocurrency time-series. Algorithms 13:121
33. Huang GL, He J, Xu Z, Huang G (2020) A combination model based on transfer learning for waste classification. Concur Comput: Pract Exp 32:5751
34. Ting KM, Witten IH (2011) Issues in stacked generalization. J Artif Intell Res 10:271–289
35. Bayraci S, Susuz O (2019) A deep neural network (dnn) based classification model in application to loan default prediction. Theoretical and Applied Economics pp. 75–84
36. Bank turnover dataset. Retrieved from https://www.kaggle.com/datasets/barelydedicated/bank-customer-churn-modeling/metadata