Phishing URL Detection: A novel hybrid Approach using Long Short-Term Memory and Gated Recurrent Units

B.A.S. Dilhara

ABSTRACT
Phishing is one of the oldest types of cyber-attack, which mostly comes in the form of camouflaged URLs to delude the users to disclose their personal information for malevolent purposes of the attacker. It is one of the easiest ways of inducing people into revealing their personal credentials including credit card details. Usually, most phishing attacks come up as fake websites pretending to mimic a trustworthy website, and the attackers use these malicious website URLs for successful data breaches. Therefore, it is a necessity to filter up which URLs are benign, and which are malicious. This study proposes three non-hybrid deep learning models, namely CNN (1D), LSTM, GRU, and four hybrid deep learning models, namely GRU-LSTM, LSTM-LSTM, BI (GRU)-LSTM, and BI (LSTM)-LSTM. Based on the results obtained, it was found that BI (GRU)-LSTM model was the best performing model with an accuracy of 93.91%, precision of 93.94 %, recall of 93.38 %, and F1-Score of 93.66 %. Thus, the primary objective of this paper is to provide an insight into the hybrid deep learning approaches in phishing URL detection by evaluating their accuracy, precision, recall, and f1 score.

Keywords
Deep Learning, URL Classification, Hybrid Approach, Phishing URL detection

1. INTRODUCTION
Cyber-crimes including financial scams, identity thefts, and installation of malwares are basically executed via malicious URLs which systematically gather the personal information of users while causing severe financial psychological problems to the victim. Therefore, it can be considered a timely need to create robust techniques to detect malicious URLs in order to minimize the aforementioned scams where the victims are rigorously blackmailed by the perpetrators. Typically, the initiation of a phishing attack starts with an email that seemed to be send to the victim by an authentic organization and then it requests an update of user information accessing through the link given in the email. The malicious URLs are the ones that are being compromised and use for cyber attacking. To launch an attack these malicious websites or the URLs host, use various unrequested content in form of spam phishing or driven by download to launch an attack.

Among the techniques use for phishing, buying a domain that is similar to the domain, that is to be attacked is one common approach. Since human brain is not accustomed for detecting minor character changes, it will not detect the difference betweenhttps://www.facebok.com and https://www.facebook.com. Since many users neglect what is there at the end of link, the attackers get the maximum use of URLs with a longer path. Ultimately the users ended up accessing to unsolicited websites while giving authority over their sensitive information to unknown third parties. Organizations use tremendous effort in term of resources and money to prevent possible data breaches. However, employees being the greatest asset for a company, sometimes become the greatest liability, due to their vulnerability to cyber threats. Along with social engineering, phishing sites has become one of the easiest ways of accessing sensitive and credential information.

Among the piles of email that people receive daily, there can be emails that contain malicious URLs; a click on such malicious URL can lead to a data breach. Even though the employees are being trained to identify these sites, attackers use software like HTTrack [1] for duplicating entire websites. With these kinds of approaches, even the trained users can be tricked into exposing their credentials. These attacks can cause billions of worth damages and it is a necessity to implement an effective method to efficiently diagnose these malicious URLs and maintain public internet security. The phishing attack on the Royal Bank of Canada in June 2004 [2], is one famous phishing attack where attackers sent fraudulent emails pretending to be the bank and requesting the customers to verify their account details through the link they had provided. It also stated the fact that if the account holders did not provide the necessary details their account will be blocked. For instance, around 33,000 cases of phishing attacks which totalled to a loss of 687 million dollars were reported in the year 2012[2]. According to a survey conducted by Anti Phishing Alliance of China, there had been 435193 number of phishing websites by the end of the year 2018[3].

The user awareness is the easiest and simplest method to minimize any potential threat. Through the awareness programs and frequently reminding the employees in an organization the risk of being exposed can be minimized to a certain extent. Since, humans are the weakest spot in any organizations, attackers find their way to trick them, sometimes even the security experts Though these measures have controlled the scenario to some extent, the users' knowledge, and behaviour in utilizing these plays a key factor.

When considering the legal solutions, it was the United States of America (USA) that first enacted laws against phishing activities [4]. Followed by USA, many other countries too have enacted laws. Based on these laws the phishing attackers can be arrested and sued. In 2004, by the Federal Trade Commission (FTC), which is a government agency for consumers protection, had include phishing in the computer crime list [4]. In 2005 and 2006 both United Kingdom's and Australia's governments have strengthened the punishments against phishing attacks and creation of such sites [4]. However, the attackers have proved the world that a legal frame is not enough to stop them. So, it is necessary to detect these malicious URLs to prevent phishing-based attacks

2. LITERATURE REVIEW

2.1 Traditional Approach
Blacklist and whitelist methods, heuristic methods, visually similarity methods and machine learning methods are four

categories of automatic phishing detection methods that are being used currently [3] as the technical approaches. However, machine learning methods have a more effective, novelty approach when compared to the other methods. For the Blacklist and whitelist methods, based on previous detections of phishing URLs Black and whitelists are constructed [3]. A database search performed whenever a URL is visited, and a warning is generated if the URL is present in the blacklist [5].

Since, this method relies on a list of previous detections it may find difficulties in diagnosing newly emerged phishing sites. Due to this, attackers can bypass the currently prevailing blacklists to launch phishing attacks. However, despite the cons, the simplicity and efficiency make it a commonly used technique in antivirus systems [4].

Heuristic method is an extension of blacklist and whitelist methods where the idea is to create a blacklist signature [4] to the identified phishing sites. By scanning the websites for these assigned signatures, a warning is raised if a malicious website is found. This method can detect newly emergedattacks which makes it superior to the black and whitelist method. However, this method is complicated, time consuming and has a higher false positive rate [3]. Visual similarity method is based on capturing images of website's appearances to sort them by comparing with suspicious websites for visual similarity and it can also achieve a similar accuracy when compared with the black and whitelist method [3].

Decades ago, it was the blacklisting and heuristic approaches that was prominent phishing URL detection. Even today, there exist a few instances that uses centralized phishing and URL blacklisting solutions like PhishTank. Even though, these methods are still being used, they seem to be inefficient, since it takes a very longer time to detect, report and confirm and publish a malicious URL in a blacklisting database [17]. In order to overcome the issues, the traditional methods had researchers started working on machine learning methodologies for effective phishing URL detection.

2.2 Machine Learning Approach

Due to the ability to learn from massive datasets and effectively predict the outcome, the machine learning methods are widely used over the traditional trends. Black and whitelist methods depend on keyword matching and matching of URL syntax [9]. However, the machine learning methods usually use feature selection to extract sensitive information. When the URL based features are converted to a feature vector those features can be directly be applied on a machine learning algorithm. In most research papers [6][7][8], malicious URL detection is addressed as a binary classification problem, where the aim is to classify whether the URL is benign or malicious. Blacklisting and Heuristics methods, machine learning models can be used to identify unseen Phishing URLs. Machine learning approach can be attained through either supervised or unsupervised techniques [4].

In [18], it uses six machine learning classifiers on 2 different datasets and the highest True Positive Rate (TPR) was obtained for the Random Forest Classifier at both instances. Authors in this paper [19] propose a multilayer filtering model where the model contains 4-layer classifier. Models first filter is a black and whitelist filter, the second layer is Naïve Bayesian filter, the third is a Classification and Regression decision tree filter and the final layer is a Support Vector Machinery (SVM) filter [19]. It was the model with 4 layers of classifiers, that had the highest accuracy rate of 79.55% and a precise rate of 87.64%. When the models were tested alone, their values lie below the multi-layer filter model. This suggests the fact that combination of classifiers as filters can improve the accuracy and the precision rates.

2.3 Deep Learning Approach

Deep Learning is model that conceptualize the behavioral patterns of the human brain and Deep Learning Algorithms are a subset of machine learning algorithms [20] and also capable Neural Networking. Since deep learning and ANN are so deeply connected it is impossible to define them separately. Same as in machine learning algorithms Deep Learning algorithms too have different flavours which serves different purposes. Multi- Layer Perceptron Neural Network (MLPNN) consists of 2 hidden layers [12]. This can solve complex problems with several parameters and can handle datasets with large number of features. Convolutional Neural Network (CNN) consists with one or more convolutional layers with fully connected layers on top and it has pooling layers, and weights [20]. Being able to train easily than feed forward neural networks and their ability to train with standard back propagation makes it a highly attractive architecture to be used [20].

Chen et.al in their study [13], have proposed a phishing detection system using Long Short-Term Memory (LSTM) recurrent neural networks, where LSTM can be used to analyze complex high-dimensional massive data. The proposed model had reached an accuracy of 99.1%. In the study by Bahnsen et.al [14], they have compared a feature engineering approach followed by random forest classifier against the Recurrent Neural Networks (RNN). The proposed model was tested using URLs collected from common crawl and an accuracy of 98.7%.

2.4 Hybrid Deep Learning Approach

Peng et.al [15], in their study have proposed a hybrid approach which uses Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) based on the attention mechanism for phishing URL detection. Here URL features like texture information, lexical information and host information are extracted, filtered, and pre-processed. The feature matrix more relevant to the output are chose according to the weight of the attention mechanism and input to the constructed parallel processing model called CNN LSTM, combining CNN and LSTM to get local features. Furthermore, the extracted local features are merged to calculate the global features of the URLs detected and finally the URL classification is done using SoftMax classifier. The accuracy of the proposed model is 98.26% which was evaluated based on URL dataset collected from PhishTank.

Furthermore, Yazhmozhi et al. [9] have proposed an anti-phishing system to protect the users from phishing attacks. Similar to [15],[8], they use a model that uses both LSTM and CNN with a large dataset consisting of 200,000 URLs. The phishing URLs were collected from Phish Tank and virus total, and the legitimate URLs were collected from various sources using an API called Yandex SearchAPI. The model was trained for 200 epochs and found that the model performed with an accuracy of 96% and a precision of 97%.

Authors Vaitkevicius et.al [16] proposes a URL detection mechanism based on Recurrent Neural Networks and other types of deep neural networks. The results are additionally compared with the performance of classical classification algorithms on the same dataset with 48 features extracted. Here, the features which are extracted in this study shows a

significant increase in classification accuracy. The study has also focused on comparing CNN, RNN, Gated Recurrent Units (GRU), LSTM, GRU-LSTM and LSTM-RNN. Moreover, this study clearly shows RNNs with explicit memory implementation, like LSTM and GRU, outperform classic RNNs without memory by a significant difference of 0.06 which is statistically very significant. So, it becomes obvious that using a specialized RNN like LSTM has a clear advantage over classic RNNs. Furthermore, out of the studies reviewed this is the only study which uses GRU, and the

3. METHODOLOGY

Based on the literature review, it was obvious that using a hybrid approach by using deep learning algorithms has shown prominent results in the deep learning domain. The primary objective of this methodology section is to develop a hybrid prediction model using LSTM and GRU to detect phishing URLs.

3.1 Dataset Selection

The selection of data sets is a critical task in this research. In order to complete the research in a reasonable amount of time, an existing dataset was used rather than manually collecting the URL data. The quality of the dataset has a significant impact on the result. Here, the dataset [22] is taken from Mendeley Data and used in this study. To build the dataset, researchers used publicly available lists of phishing and legitimate websites. Multiple users have verified the PhishTank registry, but it was included in the phishing website list along with legitimate websites from publicly available, community-labelled, and organized lists and Alexa top ranking websites. The data is comprised of the features extracted from the lists of website addresses. Total 111 features are included in the data set, with 96 of them coming directly from the website URL and the other 15 being extracted via custom Python code [22].

The dataset contains 111 attributes, excluding the target phishing attribute, which indicates whether the specific instance is legitimate (value 0) or phishing (value 1). It also provided two versions of the dataset, where the first provided with a total of 58,645 instances which almost had a 1:1 ratio between the target class [22]. It consisted of 30,647 instances of phishing websites and 27,998 instances of legitimate websites. The dataset's second variant has 88,647 instances, of which 30,647 are classified as phishing and 58,000 as legitimate, with the aim of simulating a real-world situation where more legitimate websites are present. Fig 1 shows the distribution of classes in both datasets.

The first dataset, which is also shown in the fig 1 as dataset small, was chosen out of the two. This is due to an imbalance in the target class instance sizes in the second dataset, which is also labelled dataset full in the fig. On imbalanced datasets, machine learning models produce either general or highly specific rules. When using an imbalanced dataset, the classifier favors majority instances while ignoring minority ones [21]. In addition, classifiers tend to overfit the training data, resulting in poor classification accuracy on unseen data

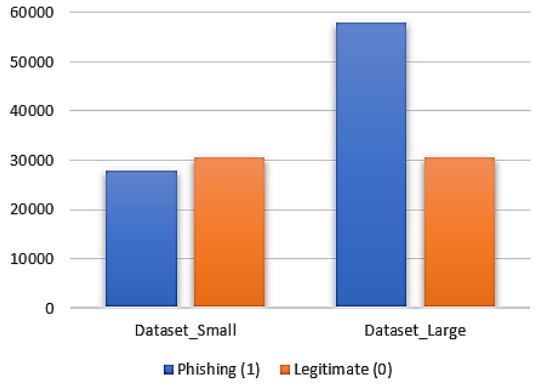

Fig 1: Dataset Distribution

3.2 Proposed Approach

In this study, we are proposing a hybrid approach in detecting phishing URLs using deep learning techniques. So, in this study, we are proposing 7 deep learning models. Some of these models are independent models while some of them are hybrid models. Here, the proposed LSTM model, GRU model and 1D CNN model falls under the category of independent models while LSTM-LSTM, Bi (GRU)-LSTM, GRU-LSTM and Bi (LSTM) - LSTM models fall under the category of hybrid models.

Initially in this study models were tested individually and then these models were combined to build up the hybrid models. However, the CNN (1D) model was not used in the hybrid approach. The main reason for this is that, even though CNN (1D) can be used in text classification problems, it was not found to be very promising since CNN are mainly used for image processing problems. Therefore, only the LSTM and GRU models along with the bidirectional approach was used here. Fig 2 and Fig 3 depicts how the proposed approach looks like.

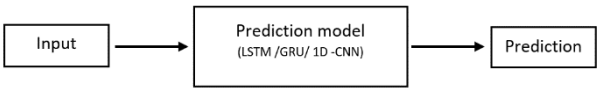

Fig 2: Proposed model for non-hybrid models

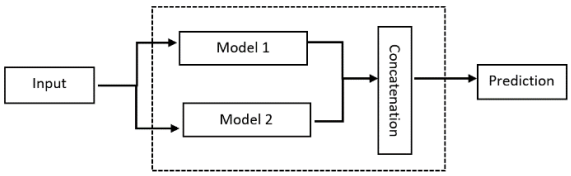

Fig 3: Proposed model for hybrid models

4. DESIGN AND IMPLEMENTATION

As previously stated, 7 models will be developed namely CNN (1D), LSTM, GRU, LSTM-LSTM, Bi (GRU)-LSTM, GRU-LSTM and Bi (LSTM) – LSTM can be used, and their detailed structure will be discussed here.Initially, before the preparing the models the selected data set should be pre-processed so that it will be suitable as an input for the model. The dataset was vectorized based on the URL input and it was later converted to the desired format based on the model requirement. Pickle library was used insaving pre-processed data and target files. Fig 4 depicts the model flow of best performed model.

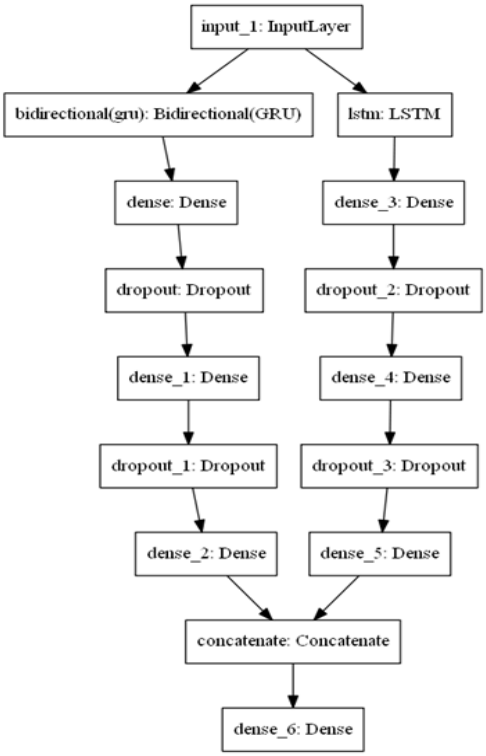

Fig 4: Bi (GRU) – LSTM Model Flow

4.1 Dataset Pre-processing

In this study, the small_dataset as mentioned in the dataset selection section was used. The main reason for this was that small_dataset had an almost equal data distribution for the 2 classifying classes. As previously discussed, imbalance datasets can cause several issues and as a result full_dataset. So only the small dataset was used in the training process and if it is necessary full_dataset can be used in the testing process.

In order to train the dataset properly, dataset should be preprocessed initially. The preprocessing happens through several stages. As the initial step, the dataset was checked out for null values. The reason for this is that, in any real-world dataset there can be few null values and having these null values obstruct the process of proper model training. So, it will not matter whether the problem is classification, regression, or any other kind, null (NaN) values should be removed from the dataset in order to achieve better results. Apart from that, the dataset was shuffled. The dataset shuffling is necessary to minimize the overfitting. In this dataset the phishing and benign classes were not randomly distributed. So, this shuffling process minimizes the variance while making sure that model trains well while keeping it general. In addition to this, shuffling helps the training converge faster, learning the order of training is prevented and also the bias during the training process also gets prevented.

Another aspect that was considered in pre-processing this dataset is that what features are to be used in the training data. However, usually it is not necessary to use feature selection steps in deep learning-based algorithms, as they can learn representations from raw input data and can process accordingly. So, here a separate feature selection was not done.

After these steps, the dataset was separated into 2 parts namely data and target. The ‘data’ is consisted of 111 features as described in the Dataset selection section and the ‘target’ is consisted of the two classification classes in binary: ‘0’ if benign and ‘1’ if phishing. The separated ‘data’ and ‘target’ was then converted to two separate NumPy arrays and then saved into two physical files using the Pickle library. The saved file can be then used with the proposed deep learning models without further pre-processing.

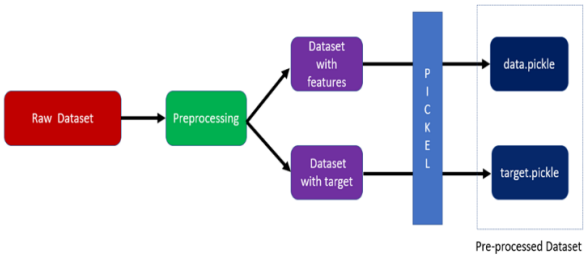

Fig 5: Dataset pre-processing and saving to a physical file as NumPy arrays

4.2 Training the Models

The pre-processed dataset was then used to train the desired deep learning model. Proposed deep learning models and their architecture were previously discussed under Model overview section. So, at the training stage, the pre-processed dataset is feed into the proposed 7 deep learning models.

The training dataset is then feed into the respective deep learning models and the model is allowed to train. Here, CNN (1D) model and all the other models were trained for 25 epochs. The main reason for this was that training of LSTM and GRU models consume a lot of time compared to the CNN (1D) models. Apart from that, when passing the input for these proposed models, the data should be passed in the required format by the model. The reshaping was not done initially at the pre-processing stage because of this reason as different models requesting different input formats. For an example, CNN (1D) requires the input in 3D format. So, the data shape should be changed accordingly.

Before providing the data as the input for the deep learning model, it was passed through a function called train_test_split. This function is imported from the Sklearn library, and it serves the purpose of splitting the data arrays into 2 separate subsets namely, training data and testing data and the other benefit of this function is, it is not necessary to manually divide the dataset because the Sklearn’strain_test_split function can divide the dataset with features into training features and testing features while, the target dataset can be divided into test target and train target. The training dataset is being used for training the deep learning model. Here, in this study training dataset with features, contains URL features to determine whether a URL is phishing or not while the train target dataset contains whether the URL is phishing or not base on the provided features. During the splitting of the dataset using the train_test_split, the test size argument can be provided. The test size is used to determine the split size of the dataset. The test size usually accepts a value between 0 to 1. In this study, a test size of 0.1 was used. This means 10% of the dataset is being used for testing while the remaining 90% is used for training the model. That means 10% out of 58645 data in this dataset, is for testing while rest is for training. Fig 6 below depicts this flow.

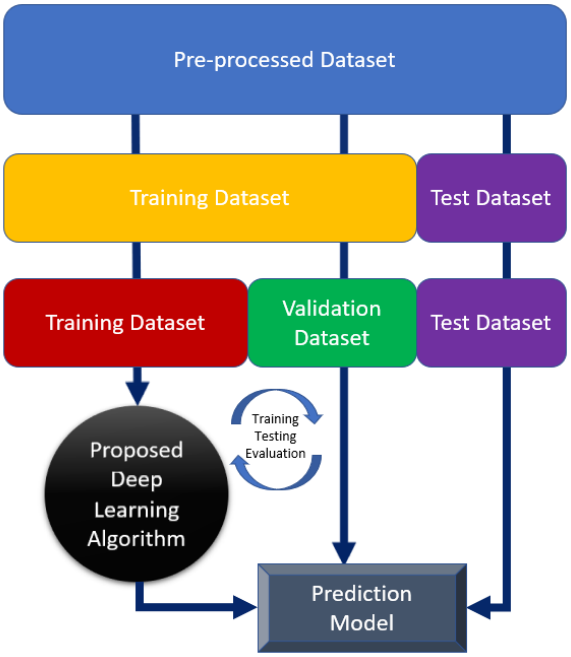

Fig 6: A Visualization of Splits and training flow

4.3 Testing the models

The trained model was then tested using test data and test data is used only after the model is being trained. That is by training and validation sets. As previously discussed, the test data size was 10% from the pre-processed dataset. Even though model testing and model evaluation seem to be similar, they are two different things. Model testing usually involves performing explicit checks for the expected behaviors of the models while metrics and plots that summarize performance on a validation or test dataset are included in model evaluation. The training dataset might be training the model perfectly. However, there should be a guarantee that the model will perform similar in real scenarios. Here, model will see the training dataset for the first time, which hasn't been used earlier. Basically, this is an unseen dataset for the trained model. So, based on this dataset, the performance of the model can be determined. As a best practice the same test dataset should not be overused. Based on the test data set we can determine whether the model is overfitting or not. If the model fits a training set better than the testing set, overfitting is likely to be present.

training set. Once the dataset size is defined, the training process can be started. Once the training is done, models are then evaluated with the validation set. Since, this process is iterative it can embrace changes needed for a model that can be re-evaluated based on the results which ensures that the test dataset remains unused to be used for testing an evaluated model. This process can be summarized using the fig 7 which is shown below.

According to the fig 7, the next most important task is to evaluate the trained and validated model. The evaluation is done using evaluation parameters. In this study evaluation is mainly done using confusion matrix and its other related evaluation parameters like accuracy, precision, F1 score and recall. Evaluation parameters which are being used in this study in order to gain a better understanding about the models, will be discussed below.

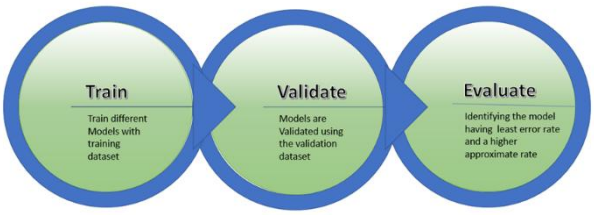

Fig 7: Iterative process of training, validating, and evaluating the deep learning model

4.4 Model Evaluation Parameters

The performance evaluation metrics are being used in two phases in a typical data classification problem: training (process of learning) and testing (process of testing). The classification algorithm was optimized using the evaluation metrics during the training stage. To put it in another way, the evaluation metric was used as a discriminator to identify and select the best solution for predicting the future classifier evaluations. In addition, the evaluation metrics is being used as an evaluator during the testing stage to determine the effectiveness of the created classifier when tested with unseen data.

Evaluating the performance of a machine learning model is a fundamental aspect of machine learning. This helps in refining the parameters and selecting the best and most appropriate model from which are being tested. Moreover, selecting a proper evaluation parameter helps in visualizing the model's success or failure in an efficient manner. So, it is a necessity to find out appropriate performance evaluation parameters to express the developed models.

In general, many generative classifiers use accuracy as a metric to determine the best solution during classification training. Nevertheless, the accuracy has several flaws, including a lack of uniqueness, discriminability, informativeness, and a bias toward the majority class data [23]. Most studies use confusion matrix for the evaluation of machine learning models. Apart from that metrics like accuracy, precision, recall and F1-score were used in this research.

4.4.1 Confusion Matrix

The model evaluation can be considered as one of the best solutions for binary classification problems, during the classification training and can be defined based on confusion matrix [23]. A typical confusion matrix usually consists of 2 rows and 2 columns that gives the count of True Positives (TP), count of True Negatives (TN), count of False Positives (FP) and count of False Negatives (FN). Each row in a confusion matrix represents an observed class, while each column represents a predicted class. A confusion matrix's entries are integer numbers, and the sum of TP, TN, FP, and FN equals the number of test data. Table 1 given below represents a confusion matrix.

Table 1. Confusion Matrix

Predicted Values \ Actual Values	Positive	Negative
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

In the confusion matrix the True Positive (TP) and True Negative (TN) depicts the correctly classified negative and positive instances while False Positive (FP) and False Negative (FN) depicts the misclassified positive and negative instances. For an example, when the actual value is 1 and predicted value is also one, it can be considered as a True Positive Value and when the actual value is 0 and the predicted value is 0, then it can be considered as a False Negative value. Moreover, if the actual value is 0 and the predicted value is 1 then it can be considered as a False Positive (FP), while the predicted value is0 and the actual value is 1, it is then called a False Negative. In addition, False Positive is also known as Type I error and False Negative is also known as Type 2 error.

Based on the TP, FP, TN and FN values, several other performances enhancement metrics like accuracy, precision, recall (sensitivity) and F1 Score can be generated, and these parameters has been used in the evaluation of the proposed models.

4.4.2 Accuracy

As depicted previously, accuracy can be considered as the most used metric for performance evaluation of both binary and multi-class classification problems. Apart from that, based on the calculated accuracy, quality of the generated result is evaluated, relying on the correct prediction percentage over total number of available instances. There is another metric called error rate, which is the complement of accuracy, and this evaluates the generated solution by the percentage of its incorrect predictions [23]. However, in this study we will only be using accuracy as a performance evaluation metric.

The major advantage of accuracy is its easiness to calculate with less complexity and it is also suitable for multi class and multi label classification problems while making it easier to be understandable by humans. The equation 1 given depicts how accuracy can be calculated.

$$Accuracy(acc) = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

4.4.3 Precision

Precision can be defined as a measure of positive patterns that are correctly predicted from the total predictions of a positive class and the equation 2 can be used to determine the precision.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

4.4.4 Recall

Recall is usually used in measuring the fraction of positive patterns that are correctly being classified. This is also known as sensitivity. Equation 3 can be used to determine the recall.

$$Recall(r) = \frac{TP}{TP + TN} \quad (3)$$

4.4.5 F1-Score

F1 Score is a metric which displays the harmonic mean between recall and precision and equation 4 depicts the equation for F1 Score calculation.

$$F1\ Score = \frac{2 * p * r}{p + r} \quad (4)$$

Even though, there are numerous performance evaluation parameters available, in this study only the aforementioned parameters are being used to determine the effectiveness of the proposed models. Based on these evaluation parameters the performance of the model can be predicted. The results obtained through this evaluation are discussed in detail under the next section of Results and discussion.

5. RESULTS

Results obtained by evaluating the performance will be discussed in this section. Each of the proposed model in this study performance will be evaluated based on the evaluation parameters.

5.1 CNN (1D)

For the evaluation of proposed CNN (1D) model several evaluation parameters like accuracy, precision, F1-Score, and recall was used. Initially accuracy of the model was tested, and it was 87.32 %. Apart from that while training the model, based on the validation dataset, validation accuracy was tested, and it was 87.90 % for the final epoch. Furthermore, for each epoch loss along with the validation loss was calculated. The graph depicted in the fig8 shows the variation of accuracy and validation accuracy over the trained epochs.Here fig 8 clearly depicts that accuracy and validation accuracy are varying at a closer margin.

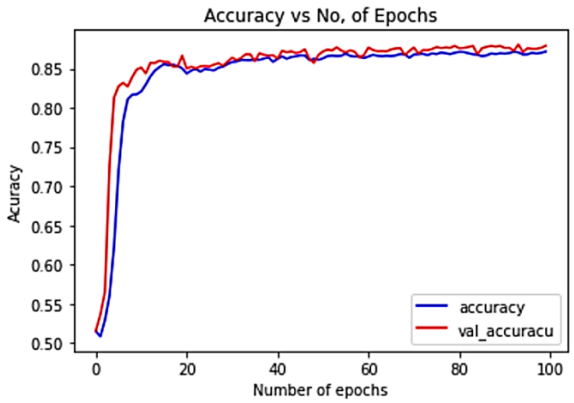

Fig 8: Variation of Accuracy and Validation Accuracy over the number of epochs in CNN (1D) model

For the CNN (1D) model the validation loss and loss are 9.02 % and 9.91 %. Given below in the fig 9 is a confusion matrixfor this model. This consist of values for TP, TN, FP, and FN. Based on the confusion matrix as shown in fig 9, the values of TP, FP, TN, and FN are as follows. The True

positive rate is 87.04 %, the false positive rate is 11.46 %, the true negative rate is 88.53 % and false negative rate is 12.95%. When considering about the precision, this CNN (1D) model achieved a precision of 88.20 %. Apart from that the recall of this proposed model is 87.04 % and F1-Score is 87.13

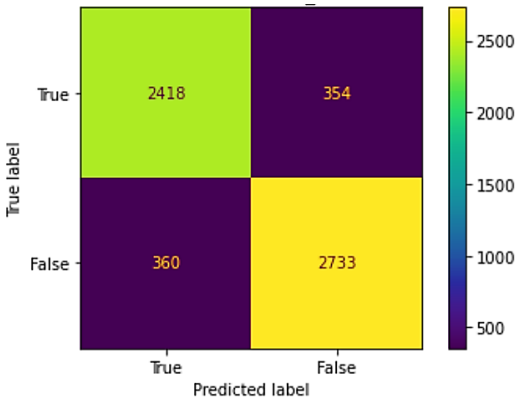

Fig 9: Confusion Matrix of CNN (1D) model

5.2 LSTM

At the testing of evaluation parameters of the proposed LSTM model an accuracy of 92.71 %, validation accuracy of 92.78%, loss of 18.57 %, and validation loss of 19.06%. The graph depicted in the fig 10 shows the variation of accuracy and validation accuracy over the trained epochs. According to the fig 10, it is visible that accuracy and validation accuracy are varying at a slightly different margin. However, it is visible that at the final epoch of training, both accuracy and validation accuracy is reducing the margin and validation loss and loss also displays a similar scenario. According to the confusion matrix, False positive rate is 6.60 %, False Negative rate is 8.27 %, True positive rate is 91.72 % and True Negative rate is 93.39%.Based on the TP, FP, FN, and TN a Precision of 92.93 %, a Recall of 91.72 %, and a F1 Score of 92.32 % was achieved.

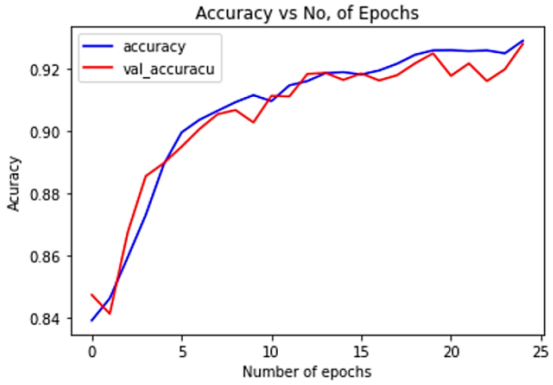

Fig 10: Variation of Accuracy and Validation Accuracy over the number of epochs in LSTM model

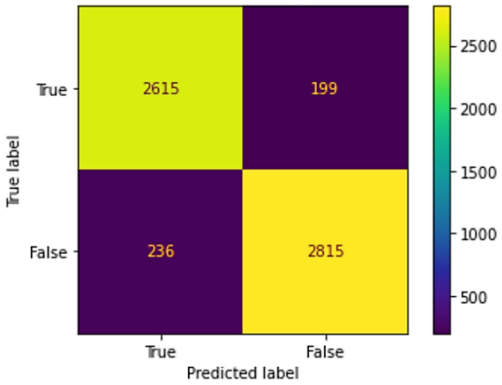

Fig 11: Confusion Matrix of LSTM model

5.3 GRU

GRU is the final non hybrid model that was tested and evaluated and at the testing of evaluation parameters of the proposed GRU model an accuracy of 93.41 %, validation accuracy of 93.10%, loss of 17.22 %, and validation loss of 18.00%. The graph depicted in the fig 12shows the variation of accuracy and validation accuracy over the trained epochs.In fig 12 it clearly shows there is a drastic variation of accuracy and also loss among different epochs but towards the end of training the variations has a subtle difference.

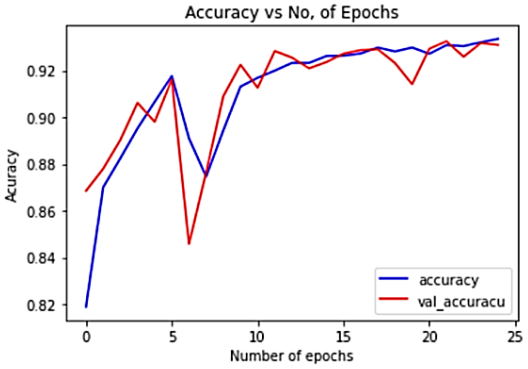

Fig 12: Variation of Accuracy and Validation Accuracy over the number of epochs in GRU model

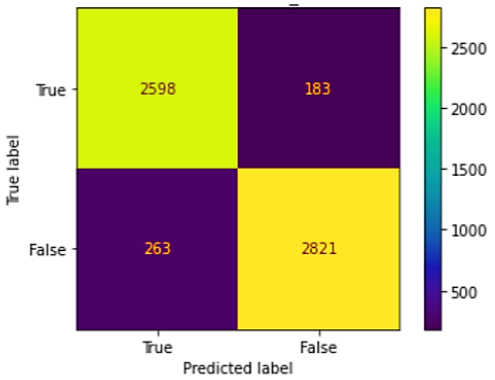

Fig 13: Confusion Matrix of GRU model

Confusion matrix as shown in the fig 13, was then used to determine the FP, TN, TP, and FN rates and these values can be depicted as follows. The False positive rate is 6.09 %, False Negative rate is 9.19 %, True positive rate is 90.81 %, and the true Negative rate is 93.90 %. In fig 13 is a comparison matrix of the proposed GRU model. Based on the

confusion matrix the determined precision, recall and F1 score as follows. The precision is 93.41 % while the recall is 90.80 %. Apart from that the F1 score recorded a value of 92.09 %. A detailed comparison of these values along with the other values recorded by the aforementioned models will be discussed in the next section.

5.4 LSTM – LSTM

This is the firstly proposed hybrid model. As previously mentioned, this consist of two LSTM models concatenated together. In this model also accuracy, loss, validation accuracy and validation loss were calculated. At the evaluation, an accuracy of 92.63 % was achieved and loss was recorded as 20.53 %. Apart from this for the last epoch of this model training it achieved a validation accuracy of 92.71 % and validation loss of 19.85 %. When taking a look at the graph in the fig 14 of accuracy and validation accuracy variation with number of epochs, it is visible that at the initial epochs, there is a zigzag variation in between accuracy and validation accuracy. However, at the latter training epochs the training, variations get subtle and at the final epoch of training there is only a very slight marginal difference in between the accuracy and validation accuracy.

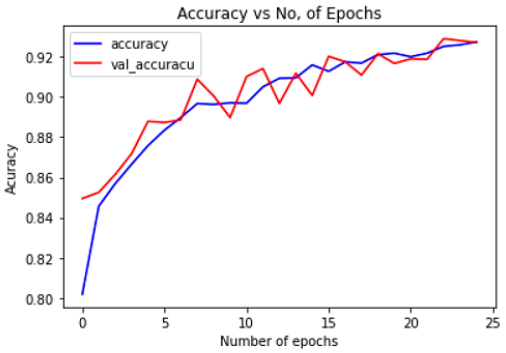

Fig 14: Variation of Accuracy and Validation Accuracy over the number of epochs in LSTM-LSTM model

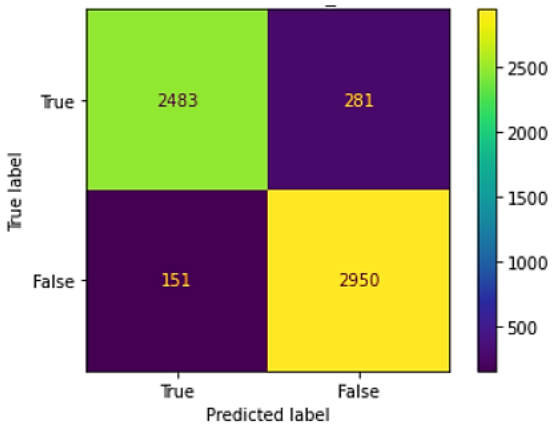

Fig 15: Confusion Matrix of LSTM-LSTM model

Confusion matrix depicted in fig 15 is used to determine the TP, TN, FP and FN values of the LSTM -LSTM model. A False positive rate of 8.69 %, False Negative rate of 5.73 %, True positive rate of 94.26 %, and a True Negative rate of 91.30 % was recorded and based on these values precision, recall and F1 score was calculated to evaluate the model. A rate of 89.83 % was recorded as the precision while recall was found to be 94.25 %. Moreover, the F1 score of this model was found to be 92.00 %.

5.5 GRU-LSTM

GRU – LSTM is the second hybrid model proposed in this study. As previously discussed, this model comprised of a GRU model and a LSTM model which provides the final output through a concatenation layer. This model achieved an accuracy of 92.07 % and a loss of 19.82 %. The validation loss of the final epoch is 20.55 % and the validation accuracy is 91.88 %. However, in the epoch, which was before the final one, a validation accuracy of was found to be 93.03 % and the validation loss is 18.58 %. Even in here, the graph initially depicts a zig zag variation but when the training epochs keeps increasing, the variation difference reduces gradually. Fig 16 depicts the variation of accuracy and validation accuracy with the training epochs.

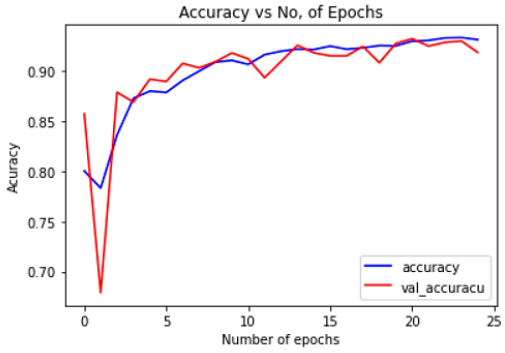

Fig 16: Variation of Accuracy and Validation Accuracy over the number of epochs in GRU-LSTM model

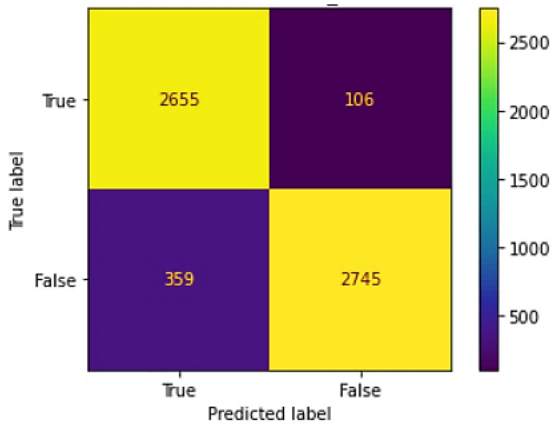

Fig 17: Confusion Matrix of GRU-LSTM model

Fig 17 depicts a confusion matrix, and it was used to determine the FP, TP, TN, and FN values for the GRU – LSTM model. According to the confusion matrix it was found that False Positive rate of 3.72 %, False Negative rate of 11.91 %, True Positive rate of 88.09 %, and True Negative rate of 96.28 % was recorded for this model. Similar with the other models, based on the TP, TN, FP, and FN values the precision, recall and the F1-Score was calculated. The recorded precision for this model was 96.16% and recall was 88.09 %. In addition to this F1 – Score were recorded as 91.95 %. Out of all the models discussed here, this model is having the highest precision. However, the precision itself cannot define the quality of a model.

5.6 Bi (GRU) - LSTM

This is also a hybrid model and as previously stated, this model uses a bidirectional GRU and a LSTM model. Similar to the previous hybrid models, the output is determined by concatenating the two models through a concatenation layer. This model achieved an accuracy 93.91 % and a validation accuracy of 93.53 %. Apart from that a loss of 18.01 % and a validation loss of 17.61 % was also achieved by this model. According to the fig 18 showing the variation of accuracy and validation accuracy, it shows a zigzag variation initially, however in the final 4 epochs the variation between the accuracy and validation accuracy is very little.

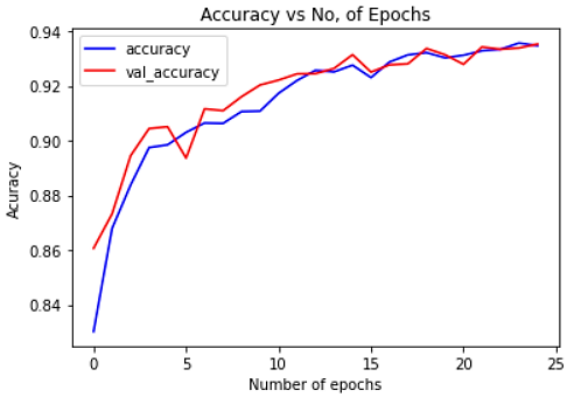

Fig 18: Variation of Accuracy and Validation Accuracy over the number of epochs in Bi (GRU)-LSTM model

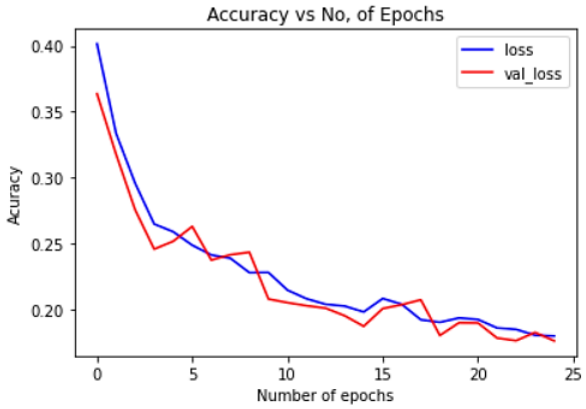

Fig 19: Variation of loss and Validation loss over the number of epochs in Bi (GRU)-LSTM model

Fig 19 depicts the variation of loss and validation loss. Similar to the fig 18 it shows a zig zag variation and then smooths out at the end, when the epochs progress.

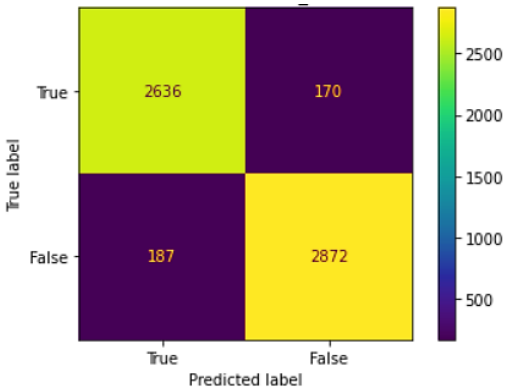

Fig 20: Confusion Matrix of Bi (GRU)-LSTM model

The confusion matrix for the bi (GRU) – LSTM model is depicted in the fig 20. According to the confusion matrix the TP, TN, FP, and FN values were determined. According to the confusion matrix it was found that False Positive rate of 5.58 %, False Negative rate of 6.62 %, True Positive rate of 93.37 %, and True Negative rate of 93.41 % was recorded for this model. For this model also, based on the TP, TN, FP, and FN values the precision, recall and the F1-Score were calculated. This model recorded a precision of 93.94 % and a recall value of 93.38 %. Apart from that, it also recorded a F1 – score of 93.66 %. Out of all the models discussed here, this model is having all precision, recall and F1 – score at a respectable state. For an example in the GRU – LSTM model had a very high precision, but its' recall was below 90%.So, overall, this can be considered as a good model out of the proposed ones. However, a detailed comparison will be done in a separate section.

5.7 Bi (LSTM) - LSTM

This is the proposed final hybrid model, and it comprised of a bidirectional LSTM model and LSTM model. As previously discussed, input values that passes through two different pathways gets concatenated at a concatenation layer and then the output is determined. This model achieved an accuracy 92.77 % and a validation accuracy of 92.85 %. Apart from that a loss of 18.70 % and a validation loss of 19.46 % was also achieved by this model. According to the fig 21 showing the variation of accuracy and validation accuracy, it shows a zigzag variation, however the variations are not smoother as the Bi (GRU) – LSTM model.

The TP, TN, FP, and FN values for the bi (LSTM) – LSTM model were determined using the confusion matrix and it is shown below in the fig 22. A False Positive rate of 8.32 %, False Negative rate of 6.00 %, a True Positive rate of 94.00 %, and a True Negative rate of 91.68 % was recorded. Based on the TP, TN, FP, and FN values Precision, Recall and F1 scores were calculated. This bi(LSTM)-LSTM model recorded a precision of 90.89 %, Recall of 94.00 % and a F1 – score of 92.41 %.

Since results of all the 7 proposed models are discussed here, it is then necessary to bring out a clear comparison between these results in determining which one is the best model and also to check whether there is an effectiveness in combining different models. So, in the next section results will be compared in determining the model having best outcome.

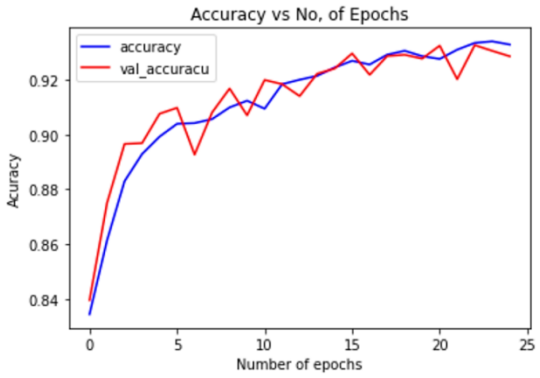

Fig 21: Variation of Accuracy and Validation Accuracy over the number of epochs inBi (LSTM)-LSTM model

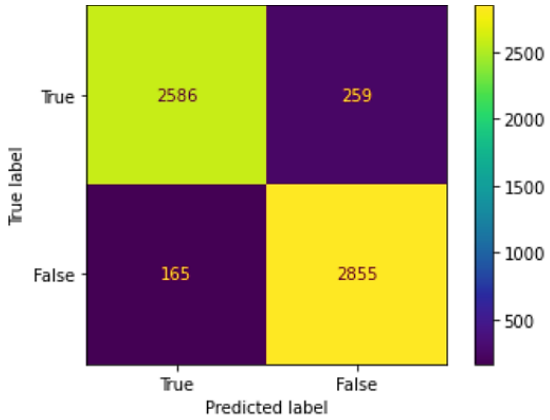

Fig 22: Confusion Matrix of Bi (LSTM)-LSTM model

6. RESULTS COMPARISON

Based on the results, it was obvious that different models had different outputs. So, based on these results it is necessary to determine the best performing model. The table 2 depicts all the results that were discussed in the previous section in a tabular format.When considering the FP values and FN values it is clear that for a model to be smart the True Positives and True Negatives should be at a higher level, while the False Positives and False Negatives should be at a lower level. Moreover, their False Negative Rate (FNR) and False Positive Rate (FPR) should be at a higher level while the True Positive Rate (TPR) and True Negative Rate should (TNR) be at a higher level. According to the table 2, when considering the FPR, the CNN (1D) model is having the highest FPR of 11.46% while the lowest FPR of 3.72 % was recorded by the GRU – LSTM model. When considering the other 3 proposed hybrid models, FPR of Bi (GRU) – LSTM model is below the FPR of Non - hybrid models. However, the FPR of Bi (LSTM) - LSTM and LSTM – LSTM models are higher than the LSTM and GRU non – hybrid models. According to the FNR column of the table 2, the highest FNR of 12.95 %, was recorded by the CNN (1D) model once again, while the lowest FNR of 5.73 % was recorded by the LSTM – LSTM model. Even though, the GRU – LSTM model recorded the lowest FPR, the FNR is at a higher level when compared with the other models. However, the other 2 hybrid models have a FNR below the values presented by non – hybrid models. So, when only considering the FNR and FPR, it can highlight the fact that, Bi (GRU) – LSTM is having a very good FPR and FNR, even though it is not belonging to the lowest category of FNR column or FPR column. The main reason for this selection is that the Bi (GRU) – LSTM model has the 2nd lowest FPR and the 3rd lowest FNR.

Even though, the lowest value is considered for the FPR and TPR, it is the highest value that is being considered for the TPR and TNR. When considering the column of TPR the highest value of 94.27 % was recorded for the LSTM – LSTM model and the lowest value of 88.53 % was recorded by the CNN (1D) model. Apart from that, both Bi (LSTM) – LSTM model and Bi (GRU) – LSTM model has recorded higher rates when compared to the non-hybrid models. However, the results displayed by the GRU-LSTM is not at a very healthy level when compared with the other hybrid models. According to the TNR column, the highest value of 96.28 % was recorded for the GRU – LSTM and the lowest value of 88.53 % was recorded for the CNN (1D) model. When considering the other hybrid models only the Bi (GRU) - LSTM model has a higher TNR compared to the non – hybrid models. However, the other 2 hybrid models namely Bi (LSTM)-LSTM and LSTM – LSTM is having a low TNR when compared to the non – hybrid models. Moreover, the LSTM – LSTM model which recorded the highest TPR has the second lowest TNR and the GRU – LSTM model which recorded the lowest TPR is having the highest TNR. This outsmarts the factor that the TPR or TNR itself cannot determine the quality of a model. Based on the TPR and TNR, it can be determined that, Bi (GRU) – LSTM model is once again a better model when compared to the other ones. Here, the Bi (GRU) - LSTM model is ranked 6 out of 7 in the TNR column while it is ranked 5 out of 7 in the TPR column making it a better model than the other proposed models.

Even though based on the TPR, TNR, FPR, and FNR, Bi (GRU) – LSTM was found to be well performing, these parameters itself cannot determine the quality of the model. So that it requires other evaluation parameters like accuracy, precision, recall and F1-Score as discussed in the previous section. In the 5th column of the table 2 it depicts the accuracy achieved by different models. One of the simplest performance indicators to employ is accuracy. However, the problem with accuracy is that it can create false illusions about the model, thus it's important to understand the dataset and algorithms before deciding whether or not to employ them.

Table 2: Results obtained from the evaluation of the proposed Deep Learning models

Proposed Model / Metrics (in %)	FPR	FNR	TPR	TNR	Accuracy	Validation Accuracy	Precision	Recall	F1 -Score
CNN (1D)	11.46	12.95	87.04	88.53	87.32	87.90	88.20	87.04	87.13
LSTM	6.60	8.27	91.72	93.39	92.71	92.78	92.92	91.72	92.32
GRU	6.09	9.19	90.81	93.91	93.41	93.10	93.41	90.80	92.09
Bi(GRU)-LSTM	5.58	6.62	93.37	94.41	93.91	93.53	93.94	93.38	93.66
GRU-LSTM	3.72	11.91	88.09	96.28	92.07	91.88	96.16	88.09	91.95
Bi(LSTM)-LSTM	8.32	6.00	94.00	91.68	92.77	92.85	90.89	94.00	92.41
LSTM-LSTM	8.70	5.73	94.27	91.30	92.63	92.71	89.83	94.27	92.00

For an example, the accuracy should never be used as a measure when dealing with imbalanced datasets. According to the table 2, the highest accuracy of 93.91 % was achieved by the Bi (GRU) – LSTM model and the CNN (1D) model achieved the lowest accuracy of 87.32 %. So, this provides additional facts in proving Bi (GRU) – LSTM is a better model. When considering the validation accuracy also, it is obvious that Bi (GRU) – LSTM model is having the highest validation accuracy of 93.53 % as well. The accuracy has the advantage of being very easy to interpret, but it has the disadvantage of not being robust when the data is unevenly distributed or when there is a higher cost associated with a specific type of error.

The 7th column of the table 2 depicts the precision achieved by different proposed deep learning models. Even though, it seems there is a similarity between accuracy and precision, there lies a clear line in between them. Accuracy is how close a measurement is to the correct value for that measurement while the precision of a measurement system relates to how closely repeated measurements coincide (which are repeated under the same conditions). Out of the proposed models the highest precision of 96.16 % was achieved by the GRU – LSTM model while the CNN (1D) achieved the lowest precision of 88.20 %. The second highest precision of 93.94 % was achieved by Bi (GRU) – LSTM model. When considering the other 2 hybrid models, both Bi (LSTM) – LSTM and LSTM – LSTM models have a precision below the precision of GRU and LSTM models.

The 8th column of the table 2 depicts the recall achieved by the proposed models and differentiation between recall and precision is very important. Precision is as previously stated, is the ability to generate a similar result again and again. But recall actually determines how many Actual Positives our model captures by identifying them as Positive (True Positive). So, recall also play an important role in selecting the best model. The highest recall value of 94.27 % was achieved by the LSTM -LSTM model while the lowest value of 87.04 % was recorded by the CNN (1D) model. All hybrid models except GRU – LSTM is having a higher recall value when compared with the non-hybrid models.

The final or the 9th column of the table 2 depicts the F1-Score values achieved by the proposed models. The F1 – Score is basically a way of combining precision and recall together and also it is defined as the harmonic mean of a model's precision and recall. Using two values, it is frequently impossible to determine whether one algorithm is superior to another. For example, if one algorithm has higher precision but lower recall than another, determining which algorithm is superior is difficult. Regardless of the recall, there may be an impression that a model with higher precision is a better model. In most cases, however, the best model cannot be determined solely on the basis of precision. In such cases, a combined metric such as F1- Score is required. It will be easier to compare some precision and some recall by using it. In other words, with high precision but low recall, the classifier is extremely accurate, but it misses a significant number of instances that are difficult to classify making it not very useful. Based on the conducted study, it was found that the highest F1 – Score of 93.66 % was recorded by Bi (GRU) – LSTM model while the lowest F1 – Score was of 87.13 % was recorded by the CNN (1D) model. Even though the GRU – LSTM model had the highest precision, it recorded a very low recall. As a result of that, the F1 - Score achieved by GRU – LSTM model was found to be 91.95 %. In addition to this, it is an obvious fact that, the LSTM – LSTM model which recorded the highest recall has a low precision when compared to the other models. As a result, the F1 – score of this model is at a higher level as expected. So, this proves the fact of, having high precision is not enough to determine the quality of model.

Therefore, based on the table 2, it outsmarts the fact that out of the proposed models, the Bi (GRU) – LSTM model is the best model, and it has the better capabilities of detecting whether a phishing URL is malicious or not. Moreover, it also proved the fact that, even though CNN (1D) can be used for the text classification problems, it is not effective as the LSTM and CNN models. This fact is very apparent as it is depicted in the fig 23.

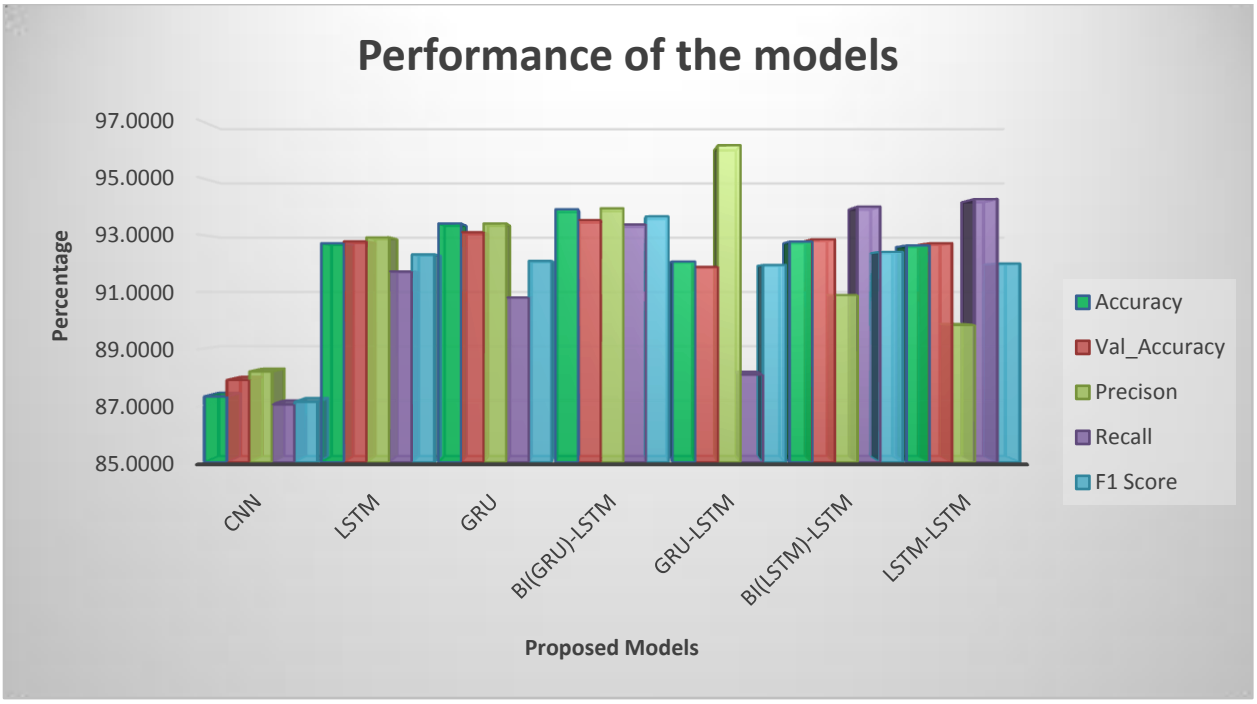

Figure 23: A Bar Graph showing the values of Accuracy, Validation Accuracy, Precision, Recall, and F1- Score of all the tested models

7. LIMITATIONS AND DRAWBACKS

The limitations in datasets create drawbacks in most machine learning and deep learning related studies. A small data set can generate incorrect predictions. Many machine learning techniques need massive volumes of data before they can provide usable results. A neural network is an excellent illustration of this. Neural networks are data-hungry devices that need a plethora of training data. The more data required to deliver valid findings, the bigger the architecture. Reusing data is never a good idea, and data augmentation might be beneficial in certain cases, but having more data is always the better option. Generation of proper datasets of a considerable size can greatly help researchers to generate models with better results. Apart from that absence of good features too result in poor performing models. When the models are trained using poor features, the overall quality of the model drops.

Another issue that has to be dealt with when dealing with deep learning models is the overfitting. Overfitting happens when the model fits well on the training data, but does not generalize well on fresh, unseen data. In other words, the model learns patterns that are unique to the training data and are irrelevant in other data. Validation measures like as loss or accuracy are used to detect overfitting. Typically, after a given number of epochs, the validation measure stops improving and starts to decline. Because the model strives to find the greatest match for the training data, the training metric is always improving. Even though finding more data for training is a difficult task, more training data can help in overfitting. Apart from that, to avoid the overfitting problems in classifiers a preprocessing stage can be included and in processing clustering can be used to sort out outliers. By preventing the inclusion of such noisy samples in training can also generate good results. In this study dropouts were used inside the deep learning models to reduce the overfitting by removing certain features randomly by making them zero. In addition to the dropouts, reducing the network's capacity by reducing the layers can also result in minimizing the effect of overfitting.

Apart from that when the neural network architecture becomes heavy, it takes a longer time to train the models. Moreover, LSTMs and GRUs take longer time to train when compared with the CNN models. So, in order to train these LSTM and GRU models for higher number of epochs it is necessary to have more powerful computers. Unless it might cost even days for training if the number of epochs is around 100. For an example, in this research study average time for an LSTM epoch to get trained in the used computer having a 16 GB RAM, in this study is around 8 minutes. This means, if it is to be trained for 100 epochs it is going to cost 800 minutes or 13.3 hrs. So, this highlight the fact that it necessary to have a very powerful computer when training deep learning models; specially when it is related with LSTMs and GRUs.

Another drawback in phishing URL detection is that, since, deep neural network design is very complicated, understanding the underlying mechanics of a neural network model remains a mystery. Thus, it seems hard to reverse engineer and beat a deep learning-based detection system without getting the same collection of training data that was used to build the system.

8. FUTURE WORK

When compared to the other detection methods, several research areas in this domain are yet to be discovered. However, for the existing research work modifications can be done to obtain greater results. The analysis of the literature discussed above outsmarted the fact that the hybrid approach can significantly improve the detection of URLs at a better rate. So, one area that can enhance the detection rate is that, building the model architecture in a different method. Modifications of the deep learning models is a necessity because over time attackers find mechanisms to overrule the prevailing defense measures. The fine tuning of the deep learning algorithms can produce better results utilizing the

features to function at an optimum level. In this study, mainly use of shared layers was used in developing the hybrid models. As one future approach, more effort can be put into ensemble techniques like bagging, boosting and bagging and there by better prediction rates can be achieved.

9. CONCLUSION

In spite of the comprehensive studies and immense progress that the machine learning and deep learning approach on malicious URLs have gained in the recent years; yet this domain remains very challenging. At the initiation of this research the, the main objective was to implement 4 hybrid models based on LSTM and GRU to detect the phishing URLs efficiently. So, through this research it was possible to introduce new architectures for these proposed models and then compare them using evaluation parameters and there by determining the best model out of the proposed models. Based on this research it was identified that Bi (GRU) -LSTM is the best performing model and it has a clear advantage over the other proposed models in classifying whether a URL is phishing or not. The accuracy, precision, recall and F1-score of this model is 93.91 %, 93.94 %, 93.38 %, and 93.66 % respectively. Apart from that it was found that it was found that, the recurrent neural networks have a clear advantage over the CNN (1D) model, even though CNN (1D) is being used in text classification problems. In addition to this, this study provides a clear insight to increasing the quality of deep learning models when they are being combined in an effective way. It can be articulated that; this study provided a comprehensive intuition to the phishing URL detection in deep learning domain while portraying the use of hybrid approach for effective identification of malicious and benign URLs. Thus, in conclusion according to this research it was the bidirectional GRU model concatenated with the LSTM, succeeded in outsmarting all the other proposed models in achieving the best performing model.

10. ACKNOWLEDGMENT

The utmost sincere gratitude is conveyed to Associate professor Dr. DharshanaKasthurirathna,for his constant guidance and motivation to complete this research successfully.

11. REFERENCES

[1] A. Kulkarni and L. L. Brown, “Phishing websites detection using machine learning,” Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 7, pp. 8–13, 2019, doi: 10.14569/ijacsa.2019.0100702.

[2] J. James, L. Sandhya, and C. Thomas, “Detection of phishing URLs using machine learning techniques,” 2013 Int. Conf. Control Commun. Comput. ICCC 2013, no. December 2013, pp. 304–309, 2013, doi: 10.1109/ICCC.2013.6731669.

[3] E. Zhu, Y. Chen, C. Ye, X. Li, and F. Liu, “OFS-NN: An Effective Phishing Websites Detection Model Based on Optimal Feature Selection and Neural Network,” IEEE Access, vol. 7, pp. 73271–73284, 2019, doi: 10.1109/ACCESS.2019.2920655

[4] D. Sahoo, C. Liu, and S. C. H. Hoi, “Malicious URL Detection using Machine Learning: A Survey,” vol. 1, no. 1, pp. 1–37, 2017.

[5] M. Ferreira, “Malicious URL Detection using Machine Learning Algorithms,” pp. 114– 122, 2019.

[6] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, “URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection,” no. i, 2018.

[7] F. Vanhoenshoven, G. Napoles, R. Falcon, K. Vanhoof, and M. Koppen, “Detecting malicious URLs using machine learning techniques,” 2016 IEEE Symp. Ser. Comput. Intell. SSCI 2016, no. December 2017, doi: 10.1109/SSCI.2016.7850079.

[8] A. Blum, B. Wardman, T. Solorio, and G. Warner, “Lexical feature based phishing URL detection using online learning,” Proc. ACM Conf. Comput. Commun. Secur., pp. 54–60, 2010, doi: 10.1145/1866423.1866434.

[9] I. N. V. D. Naveen, K. Manamohana, and R. Verma, “Detection of malicious URLs using machine learning techniques,” Int. J. Innov. Technol. Explor. Eng., vol. 8, no. 4S2, pp. 389–393, 2019.

[10] M. A. Adebowale, K. T. Lwin, and M. A. Hossain, “Deep learning with convolutional neural network and long short-term memory for phishing detection,” 2019 13th Int. Conf. Software, Knowledge, Inf. Manag. Appl. Ski. 2019, no. March 2018, 2019, doi: 10.1109/SKIMA47702.2019.8982427.

[11] V. M. Yazhmozhi, B. Janet, and S. Reddy, “Anti-phishing System using LSTM and CNN,” 2020 IEEE Int. Conf. Innov. Technol. INOCON 2020, pp. 3–7, 2020, doi: 10.1109/INOCON50539.2020.9298298.

[12] M. Somesha, A. R. Pais, R. S. Rao, and V. S. Rathour, “Efficient deep learning techniques for the detection of phishing websites,” Sadhana - Acad. Proc. Eng. Sci., vol. 45, no. 1, 2020, doi: 10.1007/s12046-020-01392-4.

[13] W. Chen, W. Zhang and Y. Su, "Phishing Detection Research Based on LSTM Recurrent Neural Network", Communications in Computer and Information Science, pp. 638-645, 2018. Available: 10.1007/978-981-13-2203-7_52 [Accessed 15 March 2021].

[14] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, and F. A. Gonzalez, “Classifying phishing URLs using recurrent neural networks,” eCrime Res. Summit, eCrime, pp. 1–8, 2017, doi: 10.1109/ECRIME.2017.7945048.

[15] Y. Peng, S. Tian, L. Yu, Y. Lv, and R. Wang, “A Joint Approach to Detect Malicious URL Based on Attention Mechanism,” Int. J. Comput. Intell. Appl., vol. 18, no. 3, pp. 1– 14, 2019, doi: 10.1142/S1469026819500214.

[16] P. Vaitkevicius and V. Marcinkevicius, Composition of ensembles of recurrent neural networks for phishing websites detection, vol. 1243 CCIS. Springer International Publishing, 2020.

[17] R. Verma and A. Das, “What’s in a URL: Fast feature extraction and malicious URL detection,” IWSPA 2017 - Proc. 3rd ACM Int. Work. Secur. Priv. Anal. co-located with CODASPY 2017, pp. 55–63, 2017, doi: 10.1145/3041008.3041016.

[18] C. Liu, L. Wang, B. Lang, and Y. Zhou, “Finding effective classifier for malicious URL detection,” ACM Int. Conf. Proceeding Ser., pp. 240–244, 2018, doi: 10.1145/3180374.3181352.

[19] R. Kumar, X. Zhang, H. A. Tariq, and R. U. Khan, “Malicious URL detection using multi-layer filtering model,” 2016 13th Int. Comput. Conf. Wavelet Act.

Media Technol. Inf. Process. ICCWAMTIP 2017, vol. 2018- February, no. November, pp. 97–100, 2017, doi: 10.1109/ICCWAMTIP.2017.8301457.

[20] B. B. Benuwa, Y. Zhan, B. Ghansah, D. K. Wornyo, and F. B. Kataka, "A review of deep machine learning," Int. J. Eng. Res. Africa, vol. 24, no. February 2017, pp. 124–136, 2016, doi:10.4028/www.scientific.net/JERA.24.124.

[21] A. Mahani and A. Riad Baba Ali, "Classification Problem in Imbalanced Datasets," Recent Trends Comput. Intell., pp. 1–23, 2020, doi: 10.5772/intechopen.89603.

[22] G. Vrbančič, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," Data Br., vol. 33, p. 106438, 2020, doi: 10.1016/j.dib.2020. 106438..

[23] H. M and S. M.N, "A Review on Evaluation Metrics for Data Classification Evaluations," Int. J. Data Min. Knowl. Manag. Process, vol. 5, no. 2, pp. 01–11, 2015, doi: 10.5121/ijdkp.2015.5201.