National College of Ireland

Drift Phenomenon based Email Spam Prediction with LSTM and GRU Approach

MSc Research Project
MSc in Data Analytics

Monika Rana
X21204497

School of Computing
National College of Ireland

Supervisor: Abubakr Siddig

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Monika Rana

Student ID: x21204497

Programme: MSc in Data Analytics **Year:** 2022-2023

Module: Research Project

Lecturer: Abubakr Siddig

Submission Due Date: 14th August 2023

Project Title: Drift Phenomenon based Email Spam Prediction with LSTM and GRU Approach

Word Count: 7486 **Page Count:** 22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Monika Rana

Date: 14th August 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	

Penalty Applied (if applicable):	

Drift Phenomenon based Email Spam Prediction with LSTM and GRU Approach

Monika Rana
X21204497

Abstract

This paper explores the complexities of drift in email spam and addresses the use of deep learning models to predict and filter spam emails that use the smart tactics. It provides an overview of the effects of these tactics on the efficiency of traditional spam filters and the need for advanced methods to identify and mitigate evolving spam threats. In this research two deep learning models Long Short-Term Memory networks (LSTMs) and Gated recurrent Unit (GRU) is used. The suggested models are made to capture temporal patterns, dynamic content structures, and contextual data that traditional filters find challenging to recognise. The deep learning models learn to recognize underlying patterns and adjust to the changing character of spam by being trained on labelled datasets including both traditional and drifting spam emails. The effectiveness of the suggested deep learning models in predicting and categorising spam emails with drift is analysed by experimental findings.

1 Introduction

1.1 Background

Email is an integral part of modern society's communication system, enabling the rapid and efficient global sharing of information. However, with the increasing reliance on it, email spam has become a significant issue. The need for effective email spam filtration mechanisms has never been more crucial. Traditional rule-based filtering techniques have had a difficult time keeping up with spammers' ever-evolving strategies. As a result, the phenomenon known as "drift" in email spam has developed, when spammers constantly adapt and change their methods to avoid detection. Existing spam filters are significantly challenged by this evolution, which reduces their effectiveness in correctly recognizing and filtering spam emails. To create cutting-edge strategies that can keep up with the spam environment's constant change, new research is required.

The drift phenomena in email spam refers to how spammers continuously adapt and develop their spamming strategies to get beyond conventional spam filters and detection techniques. Spammers adapt their strategies to get around spam filters and transmit their unwanted and frequently malicious messages straight to users' inboxes as spam filters become more sophisticated and effective at identifying and blocking spam emails.

1.2 Research Question and Objectives

The goal of the paper is to study

- what extent can LSTM and GRU deep learning algorithms successfully predict the presence of drift phenomena in email spam. The primary contribution of this research is to determine which of these architectures is better at handling dynamic and developing data patterns by contrasting the effectiveness of LSTM and GRU models in predicting the drift phenomenon?

Spammers' methods are growing more complex as a result of technological improvements. New tools and approaches are made accessible as technology develops. The use of these advancements to build stronger, more effective spam filtering systems can be explored through research. Continuous efforts are required to develop advanced, adaptive, and efficient spam filtration techniques to ensure a secure, and productive email. In order to maintain the efficiency of spam filters against new and evolving spamming strategies, mechanisms for dynamically updating spam filters must be developed, ensuring their continued effectiveness against new and emerging spamming techniques.
Research on the drift phenomena helps in gathering knowledge of the constantly changing tactics used by spammers, enabling the creation of preventative measures to reduce this malicious activity. Spammers frequently modify the language, subject lines, and images in their spam emails, making it challenging for rule-based filters to reliably identify patterns. They may employ genuine email services and hacked accounts to transmit spam or may use machine learning and artificial intelligence (AI) techniques to generate increasingly convincing spam content that closely resembles legal messages, making it more difficult for filters to tell them apart.

1.3 Document Structure.

This study report is divided into 7 sections, each of which demonstrates a different area of analysis. The second section summarises the work that other researchers have done in this area, highlighting their main findings and experimental methodology. The third section provides an outline of the methodology utilised to perform the research and the examination. Section 4 of the paper outlined the LSTM and GRU models' architectures which is the primary deep learning models applied for the analysis. Section 5 contains a description of the suggested procedure, how it was used, and the outcome that was discovered. The comprehensive analysis of the results and an explanation of the tools utilised are included in Section 6. Finally, Section 7 discusses Potential Future Studies, the Effectiveness of the Research Done, and the Main Findings of the Project.

2 Related Work

Numerous strategies have been put out over the past few decades to deal with the threat posed by spam emails. Machine learning techniques have become more popular among these due to

their ability to automatically identify patterns and features from enormous amounts of data. More specifically, deep learning algorithms, a subset of machine learning techniques inspired by the architecture and functioning of neural networks, have shown great success in a number of applications, including recognizing images, natural language processing, and speech recognition. With the ability to handle the inherent complexity and variety of spamming strategies, the application of deep learning in the field of spam email classification has resulted in positive outcomes.

2.1 Classification Techniques for predicting Email Spam

A recent study (Vinitha, Renuka and Kumar, 2023) highlights the difficulties with spam detection and how LSTM neural networks are used to deal with email spam. They show how effective this method is compared to other deep learning techniques and how LSTMs can maintain valuable past information, which has been essential to achieving high accuracy. They emphasise on the variety of approaches and technology being used to deal with the ongoing issue of spam in the world of technology and detail the LSTM's architecture starting with the Forget Gate, describing how it chooses which information is relevant and what should be passed ahead, then the Input Gate, highlighting its role in determining which new information should be included in the cell state, followed by the Cell State module, describing how it selects and stores data from the new state, and finally the Output Gate, which controls the relevance of the next concealed state. Various methods, including Artificial Neural Networks, Multi-Layer Perceptions, and Recurrent Neural Networks, are being compared to the Long Short-Term Memory (LSTM) methodology that has been proposed. The comparison is based on metrics like precision, recall, and accuracy to assess performance, and they found that LSTM had an accuracy of 97.4%. They concluded that when compared to other deep learning techniques, LSTM's capacity to retain the memory of previously collected information can in fact lead to superior performance in the detection of spam emails.

Authors (AbdulNabi and Yaseen, 2021) discuss about the ongoing difficulty in identifying unwanted emails like spam and phishing emails. Although many models and methods for automatically identifying spam emails have been developed, none of them have yet demonstrated 100% prediction accuracy. Traditional machine learning techniques as well as new deep learning methods both have proved successful among the proposed model by them. They compare the models using a publicly available corpus of spam and ham (non-spam) emails as part of their study on a BERT-based model for the automatic identification of spam. They used BERT, which makes use of attention layers, to comprehend the text's context, which influences its classification decision. The generated findings are then evaluated in comparison to a deep neural network (DNN) model that contains stacked dense layers and a bidirectional Long Short-Term Memory (BiLSTM) layer as a baseline and also with the classifiers like Naive Bayes (NB) and k-Nearest Neighbours (k-NN). The authors achieved an accuracy of 98.07% using the Naive Bayes classifier and accuracy of 83.33% using the SVM classifier. Additionally, the proposed deep learning models for Email Spam Detection

achieved the following accuracies: LSTM model: 97.15% accuracy, BiLSTM model: 98.34% accuracy and BERT model: 99.14% accuracy. The deep learning models consistently outperformed the previous studies in terms of accuracy, with the BERT model achieving the highest accuracy of 99.14%. This shows that when compared to conventional machine learning techniques, the suggested deep learning approaches for email spam detection are more effective.

Prior Studies (Rahman and Ullah, 2020) suggested a new method for identifying spam communications based on the use of sentiment analysis on the textual information in the email body. To produce effective and precise results, they built model that integrates methods such as Word Embeddings, Bidirectional LSTM networks, and Convolutional Neural Networks. The study covers spam's negative aspects, such as how it wastes users' time and poses security issues. The author further discusses about the Machine learning algorithms, which frequently uses supervised techniques like Naive Bayes, Decision Trees, and SVM for spam detection. They provided a comprehensive description of the proposed architecture which included Word-Embedding that turns words into dense vectors in order to capture context and semantic significance, Convolutional Network which was employed before the Bidirectional LSTM Network to extract more complex features and shorten training times.• and Bidirectional LSTM Network which combines LSTM with the capability of taking into account both the previous and subsequent sequences for sentiment and sequential characteristics analysis. Based on sentiment and textual sequentially, the architecture seeks to categories email communications. They showed this approach accelerates training time and facilitates the extraction of higher-level features for the Bidirectional LSTM network boasting accuracy rates in the range of 98% to 99%.

2.2 Drift Detection by Machine Learning Technique

Researchers (Mujtaba et al., 2017) provide light on the various contexts in which email classification has been employed as well as highlighting the existing concepts and methods. The study explores various aspects of email classification, including its application areas, classification techniques employed, and performance measures. The author discusses the existing research is based on static datasets, but real-time environments introduce dynamic factors that impact classifier performance and evaluating classifiers in real-time conditions, where incoming emails vary, and poses a challenge. The study identified five primary email classification techniques: supervised machine learning, semi-supervised machine learning, unsupervised machine learning, content-based learning, and statistical learning. Among these, supervised machine learning was most prevalent, with Support Vector Machines (SVM) showing superior performance, followed by decision trees and the Naive Bayes technique. Regarding the phenomenon of concept drift, where data distribution changes over time, authors argues that filters require adaptive or incremental learning to maintain accurate classification and by handling noise, duplicate can improve classifier accuracy and performance. They propose that by designing methods to incrementally add or remove features without rebuilding

the entire model can keep classifiers up-to-date with evolving trends in spam or phishing techniques.

Existing Research (Hayat et al., 2010) show the strategy that involves analyzing the variance in the distribution of email contents in order to identify any changes that might be signs of spam tactics emerging. Naive Bayes was the classifier utilized in this investigation. The suggested solution uses the language model technique to show how the content is distributed within a block of emails. For each word in a document, a language model assigns a probability. Considering this, any change in the way data is distributed can be seen as a content change. They discuss in the paper that concept drift is a change that occurs when the distribution between two blocks that follow considerably differs. The Kullback-Leibler divergence (KL-Divergence), a non-symmetric and non-negative measure, is used to express this difference in numerical terms. According to this method, each time a new block of emails is received, its language model is computed and the deviation between it and the previous block's language model is calculated. Concept drift is recognized, if this variation goes above a certain threshold. Once concept drift is detected, the approach effectively adapts the model changes to ensure correct email classification. For this, they demonstrate in the paper that the both Naïve Bayes classifier and the language model used for concept drift detection is outdated and need updating and three steps approach is followed Updating Term Probabilities, Constructing New Language Model and Updating Classification Model. They experiment with a block length of 300, where the training data length is 500. Their results confirm the method's effectiveness in concept drift detection and model adaptation. In terms of accuracy, they achieve 9% improvement for block length 500 and 8% for block length 300.

Researchers (Delany et al., 2005) have shown the developments in methodologies and techniques for addressing drift concept. They aim of the paper is to contribute to the advancement of drift concept research and its practical application in real-world scenarios. They define the Concept drift as a situation where the statistical properties of the target variable being predicted by a model changeover time in unexpected ways. This can lead to poor predictions and decisions, if not appropriately addressed. They also talk over about different types of drift, the framework for detecting drift concept, and some challenges associated with the process. The report mentioned four different drift concepts, Sudden Drift where a new concept occurs abruptly, Gradual Drift: A new concept gradually replaces an old one over time, Incremental Drift: An old concept incrementally changes to a new concept over time, Reoccurring Concepts where an old concept may reoccur after some time and intermediate concept as transformations between concepts, where concepts may not change at an exact timestamp but can transition over a period of time. Multivariate two-sample tests are discussed as a tool for concept drift identification in some circumstances. As the distribution drift may not always be included in the target characteristics, paper emphasises how crucial it is to choose the right target features.

2.3 Selection of an Algorithm

The sequential association between words in emails, which is significant, can be captured by LSTM and GRU architecture, making it suitable for email spam prediction. Due to spammers' evolving strategies, email spam trends can change over time. As they can modify their internal states in response to new data, LSTM and GRU architectures are highly suited for capturing such dynamic patterns. For modelling the drift phenomenon, this adaptability is extremely helpful. They can learn patterns in email content because they inherently capture semantic and contextual information from text. This proves highly helpful for identifying small variations in spam content that might signify drift. Also, Email lengths can vary greatly, and LSTM and GRU can analyse inputs of various durations without the need for fixed-size inputs in contrast to other deep learning algorithms. Due to LSTMs gated structure, it is able to efficiently record relationships in longer sequences, which tends to be common with email content. Thus, these advantages of LSTM and GRU align effectively with the dynamic nature of email spam and the drift challenge.

3 Research Methodology

The project uses a dataset that was obtained from online platform. It contains 5525 emails with two columns Text and Target. Text column constitute of text message of the email and Target column consist of categorical value 0 and 1 where 0 corresponds to Ham and 1 corresponds to Spam. The coding language used is python and software used for coding is google collab.

Process Involves in creating the Model

Data Preprocessing: Text preprocessing is an essential steps of model building because it enhances the model's performance enabling the model to discover relevant trends easily and generating accurate predictions.

- Data Cleaning - Noise in the form of special characters, symbols, HTML tags, and unnecessary information often appears in raw text data. By eliminating this noise, text preprocessing helps to clean the data by making the text more comprehensible and enhancing the quality of the incoming data.

    - Removal unlabelled Rows – It involves removal of rows where there is no label 0 or 1. By removing this, model can efficiently consider the relevant aspect of the data and is able to comprehend the connections and patterns required for precise prediction.
    - Converting all float values of target variable in to 0 or 1 – In order to make data format compatible for machine algorithm to understand and accurately predict it is essential to represent the values in to target classes.

- Removal of Punctuation - As punctuation symbols like periods, commas, and question marks, and others, often don't add much to the meaning of the text, their removal is preferable.

- Removal of Stop Words - Stop words include expressions like “and,” “the,” “is,” “in,” “of,” and others that are commonly used in sentences but rarely add much to the context of the text.

- Removal of Special Character - Special characters, like symbols and other non-alphanumeric characters, which frequently don't contribute to the text's fundamental meaning are removed.

- Removal of UML - Email Text normally have UML and weblinks which acts as noise for the dataset.

- Removal of Html - Removing HTML tags from text data are essential. These tags can contain formatting and structural information that is not relevant for text analysis and its removal is crucial so that models focus solely on the textual content.

- Converting text to lower case - This normalization ensures that similar words are treated the same way. This is essential for accurately counting and identifying words. It also improves computational efficiency and reduce memory usage.

- Stemming – Stemming is used in email spam prediction to handle text data variations, increase generalization, minimize word variations, and improve feature extraction. It helps to provide a text representation that is more reliable and efficient, which is advantageous for machine learning techniques used in spam identification.

- Tokenization - It is the process of breaking down a text string of emails into smaller components called tokens, such as a sentence or a document. Words, sub words, characters, or any other meaningful units can be used as tokens to represent text data in a way that machine learning models can comprehend. Both LSTM and GRU require fixed-length input sequences and tokenization ensures that each email, regardless of its length, is converted into a sequence of tokens with a consistent length. Any word that is not included in the vocabulary can be considered an out-of-vocabulary (OOV) word during tokenization. In order for the model to be generalised to new and unexplored data, it is essential that it be able to handle terms it hasn't encountered during training.

- Padding - padding is done to ensure that all sequences of email text have the same length. It involves adding special tokens (such as <PAD>) to the sequences to make

them uniform in length. It's common to process data in batches when training machine learning models. They must all be the same length in order to process numerous sequences simultaneously. Padding makes ensuring that each sequence in a batch has the same number of tokens. Padding makes certain that every sequence has an equal number of steps (time steps or tokens). This helps the model converge more smoothly during training since it can focus on learning patterns and relationships rather than being affected by varying sequence lengths

- Encoding – It refers to the process of converting text data into numerical representations that machine learning algorithms can understand and process. To enable LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) models to understand patterns, relationships, and semantics present in the text data, text must be converted into numerical values. To some extent, semantic meanings of words are captured by encoding. Although particular digits might not accurately represent word meanings, the patterns and correlations that the model learns from these numerical representations might help it.

  .

Model Building:
The Embedding layer transforms the integer-encoded words into dense vectors with fixed sizes in both LSTM and GRU models. The final Dense layer with a sigmoid activation function outputs the probability of the email being spam. Once the model is trained on the training set, it is evaluated on test data.

Defining Model

- Number Of Layers - The term "layers" refers to the total number of connected neuronal layers layered one on top of the other. The importance of the number of layers resides in its capacity to extract hierarchical features and representations from data. Deeper layers allow the model to learn complex patterns by building a hierarchy of features. Each layer can learn to represent more abstract features based on the representations learned by the previous layers.

- Type of Activation Function - Different types of activation functions have specific characteristics that affect how the network learns and behaves. Activation functions impact how gradients flow during backpropagation, which affects how well the model can be trained. Common activation functions include ReLU, sigmoid, tanh, and their variants.

- Number of Units in Each Layer - The number of units in each layer of a neural network is a key design decision that significantly affect the model's performance and behaviour.

- Number of epochs- The number of epochs in training a neural network is another important hyperparameter that is needed to decide upon. An epoch represents one

complete pass through the entire training dataset during the training process. For this research 10 is considered.

- Dense Layer - Dense layers provide flexibility for neural networks to learn complex relationships in data by connecting each neuron to every neuron in the previous layer.

Evaluation - In order to understand the performance of model and make informed decisions about their deployment, evaluation of the model prediction is critical. The following metric is used for classification

•Accuracy: It shows the proportion of correct predictions. Out of all the occurrences in the dataset, it calculates the percentage of accurately predicted instances

Accuracy = Number of Correct Predictions / Total Number of Predictions

Number of Correct Predictions: The sum of instances where the model correctly predicts whether an email is spam or not spam

Total Number of Predictions: The total number of instances in the dataset.

•Precision: The ratio of true positive predictions to the total predicted positives. It focuses on the quality of positive predictions. Precision is an important statistic for LSTM and GRU models, particularly when assessing how well they perform in predicting email spam. Optimizing for high precision can be significant if the cost of false positives is significant (e.g., misclassifying a non-spam email as spam and causing critical communications to be lost).

Precision = (True Positives + False Positives) / True Positives

True Positives (TP): The number of instances that were correctly predicted as positive (spam emails)

False Positives (FP): The number of instances that were incorrectly predicted as positive (instances that were predicted as spam but are not spam).

•Recall: It is also known as Sensitivity or True Positive Rate. It is defined as ratio of true positive predictions to the total actual positives. It measures the model's ability to identify all positive instances and focuses on the ability of the model to capture all positive instances, which is important in applications where missing positive instances (false negatives) is a concern.

Recall = (True Positives + False Negatives) / True Positives

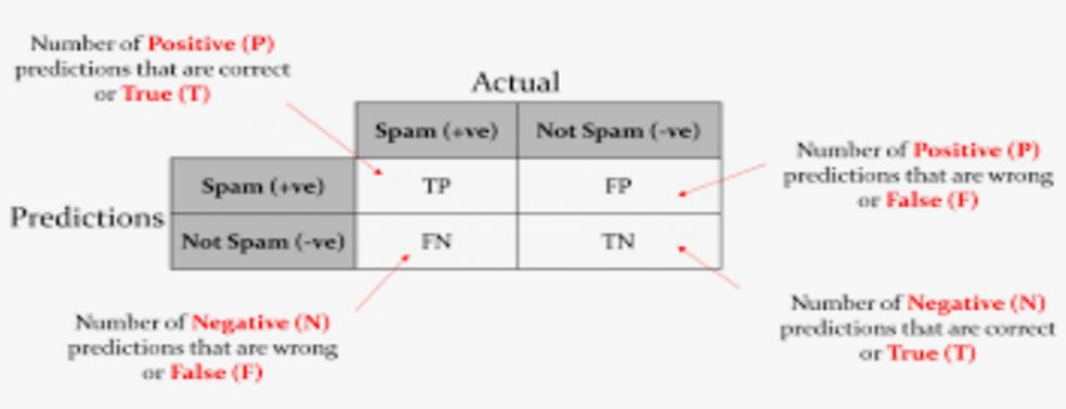
Positives (TP): The number of instances that were correctly predicted as positive (spam emails in this case).

False Negatives (FN): The number of instances that were actually positive (spam emails) but were incorrectly predicted as negative (non-spam).

•F1 Score: It is a mean of precision and recall. It provides a balanced measure between precision and recall and combines both precision and recall into a single value. The F1 score ranges between 0 and 1, where a higher value indicates better model performance

$$F1Score = 2 \times Precision \times Recall / (Precision + Recall)$$

•Confusion Matrix: It is a table that shows the counts of true positives, true negatives, false positives, and false negatives. It helps in comparing the distribution of TP, FP, FN, and TN between models and help to identify specific patterns of misclassification.



True Positive (TP): The number of instances that were correctly predicted as positive (spam) by the model.

False Positive (FP): The number of instances that were incorrectly predicted as positive (spam) by the model when they were actually negative (not spam).

False Negative (FN): The number of instances that were incorrectly predicted as negative (not spam) by the model when they were actually positive (spam).

True Negative (TN): The number of instances that were correctly predicted as negative (not spam) by the model.

4 Design Specification

After preprocessing, the entire sentence should be taken into account when deciding whether to classify an email as spam or not. It is crucial for accurate classification since LSTM and GRU needs to remember the previous data in order to send the data to the network.

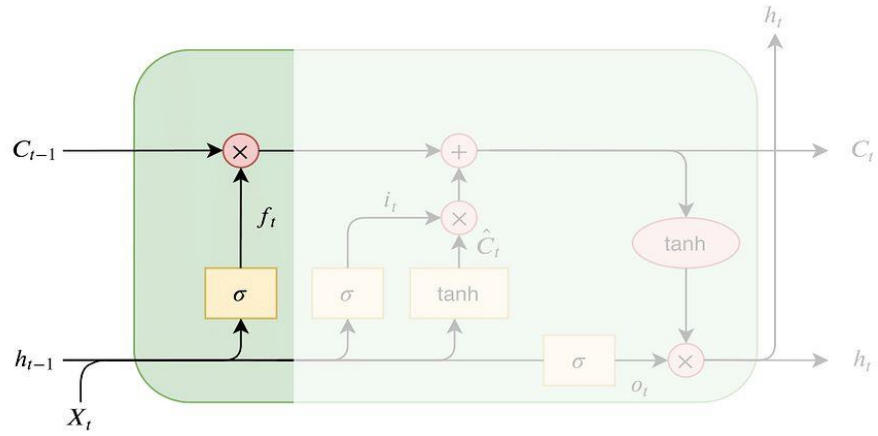
Long Short- Term Memory

The functioning of LSTM depends on the gates that regulate how the sequence of data enters, remains stored, and leaves the network. The information flow in the LSTM neural network is controlled by a set of gates: forget, input, and output. These gates behave as distinct neural networks, filters and control processes for bringing information into networks, storage, and then releasing it. Three elements affect the model's outcome: cell state, past hidden state, and the current input data.

Stages of LSTM architecture:

Forget Gate - In this stage, the neural network will decide which components of the cell state or long-term memory are crucial for spam classification depending on the weights (W_f) of previous hidden state and incoming data. The forget gate generates a forget gate output (f_t) by passing the current input (x_t), the prior hidden state (h_t-1), through a sigmoid activation (σ) function.

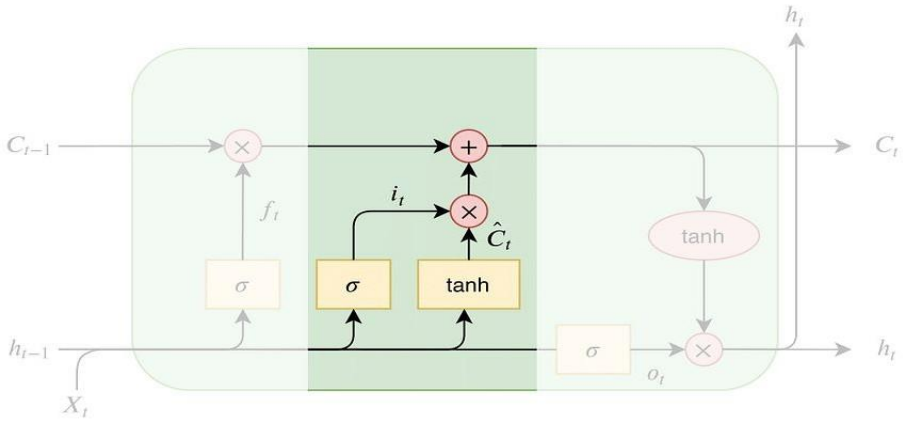
Function: f_t = σ (W_f * [h_{t-1}, x_t] + b_f)



This network is trained to provide an output value close to 0 for unnecessary component of email and a value close to 1 for relevant part. The output values are then multiplied element-wise with the prior cell state(ct-1). The unnecessary component of the cell state is then downweighed by a factor close to 0, which lowers impact on following stages.

Input Gate - It employs two parts: the input modulation gate (g_t), which modifies the cell state component, and the input gate (i_t), which selects which portions of the updated information to allow in. With the help of a tanh activation function, the input modulation gate creates a candidate cell state (g_t) using the current input (x_t) and the prior hidden state (h_t-1). A sigmoid activation function is applied to the current input (x_t) and the previous hidden state (h_t-1), which results in the input gate output (i_t).

Function: g_t = tanh(W_g * [h_{t-1}, x_t] + b_g)
i_t = sigmoid(W_i * [h_{t-1}, x_t] + b_i)



Overall, based on the output from the forget gate, the input gate uses the sigmoid function and the tanh function to decide what new information should be added to the cell state. As a result of activation similar to the forget gate, it also generates a vector of values in the range [0,1] through sigmoid function and [-1,1] through tanh function. A low output value from this gate indicates that the relevant cell state element should not be updated.

New Memory Network - It employs the tanh activation function, which can generate negative values and is crucial for minimising the impact of component on the cell state. Cell state update vector specifies how much each component of the longterm memory (cell state) should be adjusted based on the latest data.

Function: c_t = tanh(W_c * [h_{t-1}, x_t] + b_c)

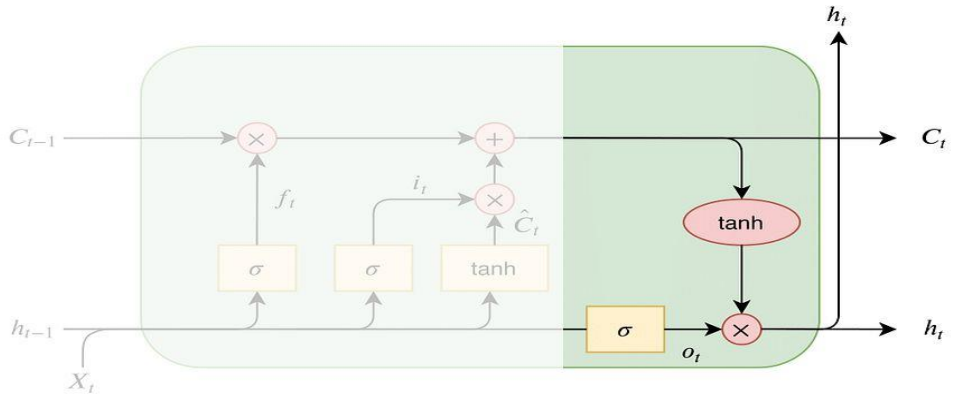
The cell state, which is the long-term memory of the LSTM network, is updated using the outcome of the combination of the new memory update and the input gate filter. The output of the cell state update is regulated by input gate filter which controls the new memory by point-wise multiplication so that only pertinent elements are added to the cell state.

Function: c_t = f_t * c_{t-1} + i_t * g_t

Output Gate- This gate enables finding the new hidden state using the recently updated cell state (c_t) , the previous hidden state (h_{t-1}), and the latest input data (x_t). The output gate is a sigmoid activated network that serves as a filter, determining which components of the

updated cell state are significant and output as the new hidden state with the value ranges in [0,1].

Function: o_t = sigmoid(W_o * [h_{t-1}, x_t] + b_o)

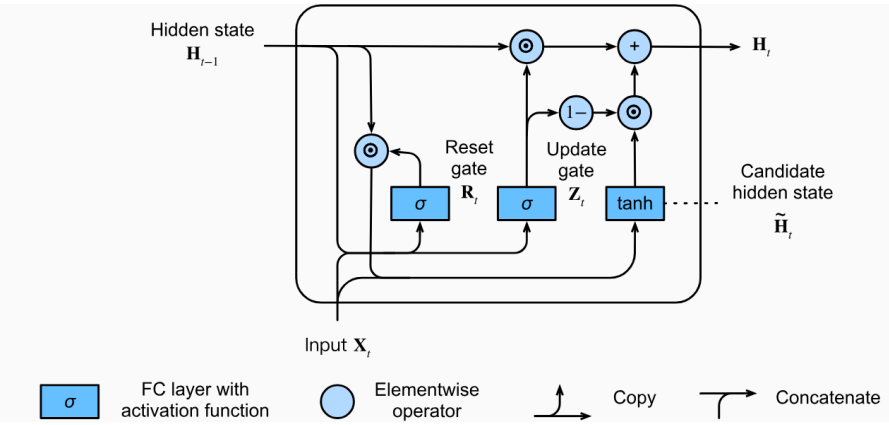


Hidden State Update - The updated hidden state (h_t) is computed by passing the updated cell state (c_t) through a tanh activation function to limit its values to [-1,1] and then element-wise multiplying it with the output gate output (o_t).

Function: h_t = o_t * tanh(c_t)

Gated Recurrent Unit

GRUs are LSTM versions that have been designed to have fewer parameters and shorter training times. The LSTMs' memory cell and hidden state are combined into a single "hidden state." The reset gate and the update gate are two more gate mechanisms found in GRUs. The reset gate determines how much of the previous hidden state should be forgotten, while the update gate controls how much of the new hidden state should be taken from the current input and from the previous hidden state.



Stages of GRU architecture:

GRU is intended to mimic sequential data by allowing information to be selectively remembered or lost over time, LSTM has a more complex architecture than GRU, which can make it harder to train and less computationally efficient.
LSTM with three gates—the input gate, output gate, and forget gate are used to update the memory cell state, which is kept separate from the hidden state.

Input Layer - The input layer provides the GRU with sequential data, such as a list of phrases or a graph of values.

Reset Gate: Recurrent computing takes place in the hidden layer. The prior hidden state and the current input are used to update the hidden state at each time step. A vector of numbers known as the hidden state serves as the network's "memory" of prior inputs.

Update Gate: The update gate decides how much of the potential activation vector should be incorporated into the brand-new hidden state. It generates a vector of integers between 0 and 1 that regulates the extent to which the candidate activation vector is absorbed into the new hidden state from the inputs of the previous hidden state and the current input.Activation Function:

Candidate activation vector: Combined with the current input and "reset" by the reset gate, the candidate activation vector is a modified version of the prior hidden state. It is calculated using a tanh activation function that confines the output to the range of -1 and 1.

Output layer: The output layer generates the network's output using the final hidden state as an input. This could be a single number, a series of numbers, or a probability distribution across classes.

5 Implementation

In order to implement the LSTM and GRU model on the dataset, python programming language is used and all the essential libraries are being installed and imported like nltk, keras, seaborn, contractions, pickle-mixin, numpy, pandas, string in collab notebook. The email spam dataset with 5525 rows and 2 column is then imported in csv format in to data frame.

After this the data is cleaned with all the preprocessing steps where all the null values, stop words, and missing values are handled, url and html tags are removed, text messages of the email are converted in to lower case and stemming is done and floating target values are converted in to target categorical values (0 or1). After preprocessing, label and Input encoding is done where text is tokenized in to token and then encoded. Padding is the next step where

the length of the largest sequence is estimated and according to it all the text sequence are made of seme length so that they can be fed in to the model.

After preprocessing, the dataset is split in to training set and test set in 4:1 ratio. With the training set the model is trained on ham and spam email messages and with test set the model predict the target value of the email text where 0 corresponds to ham and 1 corresponds to spam.

Following this LSTM and GRU model is created and is trained on training dataset. The below figure explains how the model is defined. After building and executing the model the accuracy rate of both model for the prediction is calculated and compared.

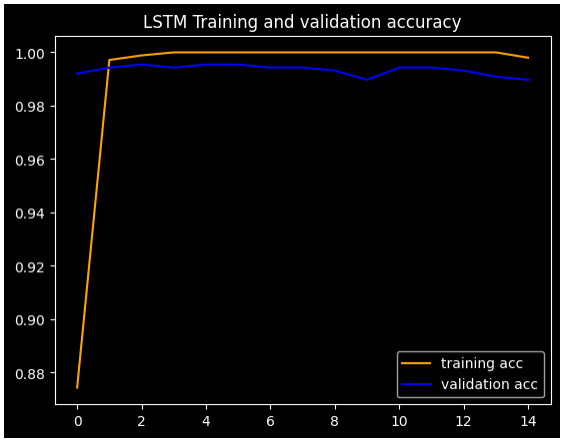
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 3111, 32)          5185888

 lstm (LSTM)                 (None, 50)                16600

 dropout (Dropout)           (None, 50)                0

 dense (Dense)               (None, 20)                1020

 dropout_1 (Dropout)         (None, 20)                0

 dense_1 (Dense)             (None, 1)                 21

=================================================================
Total params: 5,203,529
Trainable params: 5,203,529
Non-trainable params: 0
```

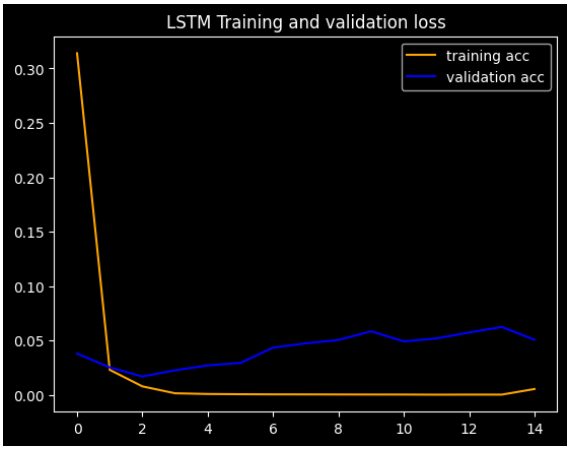
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 3111, 50)          8102950

 gru (GRU)                   (None, 50)                15300

 dropout_2 (Dropout)         (None, 50)                0

 dense_2 (Dense)             (None, 20)                1020

 dropout_3 (Dropout)         (None, 20)                0

 dense_3 (Dense)             (None, 1)                 21

=================================================================
Total params: 8,119,291
Trainable params: 8,119,291
Non-trainable params: 0
```

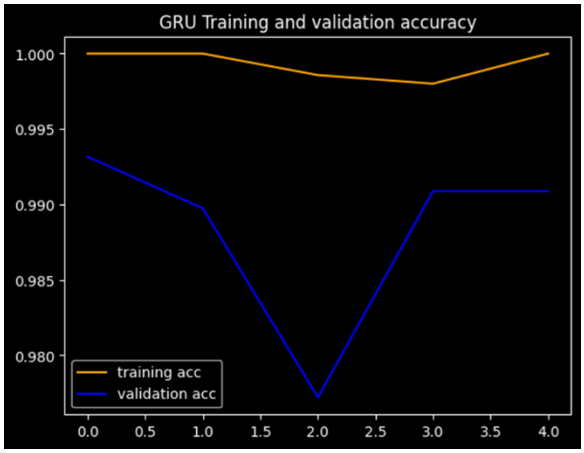
For the research purpose, and to determine which of these model LSTM and GRU are better in predicting the Email spam with drift, from the test dataset, 300 spam emails are separately taken out and is modified to form drift emails. Email consists of main three parts -headers, subject and body. Header is the section at the beginning of the email message that contains metadata and contain information about the email. It includes various fields that provide information about the sender, recipient, subject, date, and other details related to the email. Subject of an email is a concise and informative line of text that summarizes the main topic or purpose of the email. The body of an email refers to the main content of the email message. It's where we communicate our message, provide information, ask questions, share updates. For incorporating drift, the header of the spam emails are changed with the header of ham emails in order to make it resemble to legal emails and subject line, and body of spam emails are paraphrased to change its content.

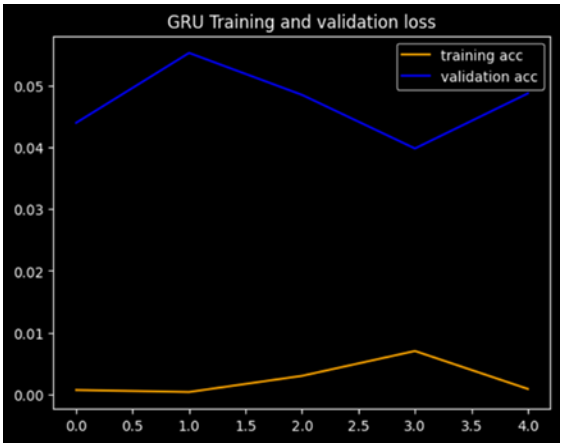
Further both the LSTM and GRU model is implemented on the drift-based spam emails and accuracy of spam prediction is evaluated and compared.

6 Evaluation









The above graph depicts training and validation accuracy and validation loss of both the model and it can be observed that with increasing epochs the accuracy of the models has increased for LSTM and was almost constant with small drop for GRU.

6.1 Case Study 1 – LSTM and GRU Model Evaluation on Spam Dataset

Confusion Matrix (LSTM Model):

```
print(confusion_matrix(pred_y, y_test))
```

```
35/35 [==============================] - 3s 55ms/step
[[759  13]
 [  0 326]]
```

True Positives (TP): In the matrix, the value 759 represents the number of emails that were actually spam and were correctly predicted as spam by the LSTM model.

False Positives (FP): The value 13 indicates the number of emails that were not actually spam but were incorrectly predicted as spam by the model.

True Negatives (TN): The value 326 represents the number of emails that were not spam and were correctly predicted as not spam by your model.

False Negatives (FN): The value 0 suggests that there were no emails that were actually spam but were predicted as not spam by the model.

```
LSTM Accuracy:  0.98816029143898
LSTM Precesion:  0.9616519174041298
LSTM Recall:  1.0
LSTM F1 Score:  0.9804511278195489
```

The overall accuracy of the model can be calculated as (TP + TN) / (TP + TN + FP + FN), which is (759 + 326) / (759 + 326 + 13 + 0) = 0.9881 or approximately 98.81%. This indicates that the model correctly classified 98.81% of the emails present in test set.

Precision: It measure of how many of the predicted spam emails were actually spam It is calculated as TP / (TP + FP), which is 759 / (759 + 13) = 0.9616 or approximately 96.61%. High precision means that when the model predicts an email as spam, it's likely to be correct.

Recall (Sensitivity or True Positive Rate): Recall indicates how many of the actual spam emails the model correctly predicted. It is calculated as TP / (TP + FN), which is 759 / (759 + 0) = 1 or 100%. High recall means the model is effectively capturing most of the actual spam emails.

F1-Score: It is calculated as 2 * (Precision * Recall) / (Precision + Recall). Since both precision and recall are high in this case, the F1-score should also be high 98.04%.

Confusion Matrix (GRU Model):

```
35/35 [==============================] - 2s 37ms/step
[[756   7]
 [  3 332]]
```

True Positives (TP): The value 756 indicates the number of emails that were actually spam and were correctly predicted as spam by the model.

False Positives (FP): The value 7 represents the number of emails that were not actually spam but were incorrectly predicted as spam by the model.

True Negatives (TN): The value 332 represents the number of emails that were not spam and were correctly predicted as not spam by the model.

False Negatives (FN): The value 3 suggests that there were a few emails that were actually spam but were predicted as not spam by the model.

Accuracy: The overall accuracy of the model can be calculated as (TP + TN) / (TP + TN + FP

```
GRU Accuracy:  0.98816029143898
GRU Precesion:  0.9616519174041298
GRU Recall:  1.0
GRU F1 Score:  0.9804511278195489
```

+ FN), which is (756 + 332) / (756 + 332 + 7 + 3) = 0.9881 or approximately 98.81%. This indicates that the model correctly classified 98.81% of the emails in the test set.

Precision: Precision is a measure of how many of the predicted spam emails were actually spam. It is calculated as TP / (TP + FP), which is 756 / (756 + 7) = 0.9616 or approximately 96.16%. This means that when the model predicts an email as spam, it's likely to be correct.

Recall (Sensitivity or True Positive Rate): Recall indicates how many of the actual spam emails the model correctly predicted. It is calculated as TP / (TP + FN), which is 756 / (756 + 3) 0.9990 or approximately 99.90%. This means the model is capturing a very high percentage of the actual spam emails.

F1-Score: it is high in this case due to the high values of both precision and recall.

The confusion matrix for the above two models suggests that the spam prediction by GRU model (98.81%) is same, with high accuracy, precision, and recall compared to LSTM model (98.81 %).

6.2 Case Study 2 – LSTM and GRU Model Evaluation on Spam Dataset with Drift

Confusion Matrix (LSTM Model):

```
10/10 [==============================] - 0s 19ms/step
[[300   0]
 [  1   0]]
LSTM - Drift 300 Accuracy:  0.9966777408637874
LSTM - Drift 300 Precesion:  nan
LSTM - Drift 300 Recall:  0.0
LSTM - Drift 300 F1 Score:  nan
```

True Positives (TP): The value 0 indicates that no emails that were actually spam and were correctly predicted as spam by the model.

False Positives (FP): The value 0 suggests that there were no emails that were not actually spam but were incorrectly predicted as spam by the model.

True Negatives (TN): The value 300 represents the number of emails that were not spam and were correctly predicted as not spam by the model.

False Negatives (FN): The value 1 indicates the number of emails that were actually spam but were predicted as not spam by the model.

After considering the drift:

Accuracy: The overall accuracy of the model can be calculated as (TP + TN) / (TP + TN + FP + FN), which is (0+300) / (0+300+0+1) = 0.9966 or approximately 99.66%. This indicates that the model correctly classified 99.66% of the emails in the test set with Drift.

Precision: Precision is a measure of how many of the predicted spam emails were actually spam. It is calculated as TP / (TP + FP), cannot be calculated using the standard formula (TP / (TP + FP)), as division by zero is not possible.

Recall (Sensitivity or True Positive Rate): Recall indicates how many of the actual spam emails the model correctly predicted. It is calculated as TP / (TP + TN), which is 0/ (300 + 0) = 0.

F1-Score: The F1-score, which balances precision and recall, cannot be calculated in this case due to the absence of true positives (TP = 0).

Confusion Matrix (GRU Model):

```
10/10 [==============================] - 0s 16ms/step
[[293   0]
 [  8   0]]
GRU - Drift 300 Accuracy:  0.973421926910299
GRU - Drift 300 Precesion:  nan
GRU - Drift 300 Recall:  0.0
GRU - Drift 300 F1 Score:  nan
```

True Positives (TP): The value 0 indicates that there were no emails that were actually spam and were correctly predicted as spam by the model.

False Positives (FP): The value 0 suggests that there were no emails that were not actually spam but were incorrectly predicted as spam by the model.

True Negatives (TN): The value 293 represents the number of emails that were not spam and were correctly predicted as not spam by the model.

False Negatives (FN): The value 8 indicates the number of emails that were actually spam but were predicted as not spam by the model.

After Considering the drift:

Accuracy: The overall accuracy of the model can be calculated as (TP + TN) / (TP + TN + FP + FN), which is (0 + 293) / (0 + 293 + 0 + 8) = 0.973 or approximately 97.3%. This indicates that the model correctly classified 97.3% of the emails in the test set with drift.

Precision: It is a measure of how many of the predicted spam emails were actually spam. In this case, since there are no true positives (TP = 0), precision cannot be calculated using the standard formula (TP / (TP + FP)), as division by zero is not possible.

Recall (Sensitivity or True Positive Rate): It indicates how many of the actual spam emails the model correctly predicted. It is calculated as TP / (TP + FN), but since TP is 0, recall cannot be calculated using this formula.

F1-Score: The F1-score, which balances precision and recall, cannot be calculated in this case due to the absence of true positives (TP = 0).

The confusion matrix for the above two models suggests that the spam prediction by LSTM model (99.66%) is high compared to GRU model (97.3 %).

6.3 Discussion

The use of RNNs, such as Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU), for modelling sequential data in email text is significant as they have ability to capture temporal dependencies, making them useful for detecting patterns in email conversations and detecting

evolving spam tactics. These models are efficiently able to predict spam emails with and without drift with high accuracy but with the realistic datasets which contain emails with drift can provide a more accurate representation of the challenges faced in practical applications. By using a realistic, complex dataset with drift, we can gain insights into how the machine learning algorithms perform in challenging conditions. This type of evaluation will be better for deployment in practical applications and help to develop models that are robust and adaptable to changing data distributions.

7 Conclusion and Future Work

Overall, both deep learning algorithms were quite successful in predicting spam emails, both with and without drift. However, LSTM performed remarkably well on spam datasets with drift. In future research, this work can be expanded by exploring different deep learning techniques with complex mechanism like attention mechanisms within RNNs or Transformer-based models, Transfer learning can be used for prediction task. Attention can help the model focus on relevant parts of the text and adapt to varying email content patterns over time. Consider using transformer-based models like BERT or its variants (RoBERTa, DistilBERT) to leverage contextual information from the entire email text. These models have shown exceptional performance on natural language processing tasks and can capture intricate relationships in the data. Experiment with different model architectures, increased complexity and adding more layers or neurons to the model can be researched.

References

Vinitha, V.Sri., Renuka, D.K. and Kumar, L.A. (2023). Long Short-Term Memory Networks for Email Spam Classification: pp.176-180.

AbdulNabi, I. and Yaseen, Q. (2021). Spam email detection using deep learning techniques, Procedia Computer Science: 184, pp.853–858.

Rahman, S.E. and Ullah, S. (2020). Email Spam Detection using Bidirectional Long Short-Term Memory with Convolutional Neural Network: pp.1307 - 1311.

Mujtaba, G., Shuib, L., Raj, R.G., Majeed, N. and Al-Garadi, M.A. (2017). Email Classification Research Trends: Review and Open Issues: 5, pp.9044–9064.

Zi Hayat, M., Basiri, J., Seyedhossein, L., & Shakery, A. (2010). Content-based concept drift detection for email spam filtering: pp.531–536.

Delany, S.J., Cunningham, P., Tsymbal, A. and Coyle, L. (2005). A case-based technique for tracking concept drift in spam filtering. Knowledge-Based Systems, 18(4-5), pp.187–195.