

Enhancing Malware Detection Using Stacked BiLSTM with Attention Mechanism: A Deep Learning Approach

MSc Research Project
MSc Cybersecurity

Srinivasan Masilamani
Student ID: 21159904

School of Computing
National College of Ireland

Supervisor: Arghir Nicolae Moldovan

National College of Ireland
MSc Project Submission Sheet
School of Computing

StudentName: Srinivasan Masilamani
Student ID: 21159904
Programme: Msc Cyber Security **Year:** 2023-2024
Module: Msc Academic Internship
Supervisor: Arghir Nicolae Moldovan
Submission Due Date: 25 April 2024
Project Title: Enhancing Malware Detection Using Stacked BiLSTM with Attention Mechanism: A Deep Learning Approach
WordCount:...5652 **Page Count :** 19

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:Srinivasan Masilamani.....
Date:25/05/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
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.	<input type="checkbox"/>

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):	

Enhancing Malware Detection Using Stacked BiLSTM with Attention Mechanism: A Deep Learning Approach

Srinivasan Masilamani

x21159904

Abstract

The detection of malware offers quite a few challenges due to the ever-evolving nature of threats in the field of cybersecurity obviously. Traditional methods of malware detection methods already struggling to keep pace due to the rapidly changing landscape of malicious software. This study introduces a novel approach with the help of deep learning techniques which will enhance malware detection efficiency and address the prior work limitations like traditional methods which are already facing challenges. There are four models which are going to be performed in this project ie., Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM) networks and Stacked BiLSTM with an Attention Mechanism. These models are more effective in identifying complex patterns and dependencies found in Portable Executable (PE) files, which are frequently used as vectors for the spread of malware. Using an extensive dataset that includes both malware and benign PE files that are sourced from reliable repositories, this study guarantees a strong training framework. While this project tests all the four models, and after comparing the results, stacked BiLSTM with attention mechanism wins the accuracy margin. This study fills a critical gap in malware detection by leveraging advanced machine learning techniques to enhance cybersecurity defenses and mitigate the risks posed by evolving malware threats.

Keywords: Malware detection, Deep learning, Stacked BiLSTM, Attention mechanism, Cybersecurity

1. Introduction

Malware is a malicious type of software which means any type of software that is designed to cause harm intentionally to a server, PC, computer, any kind of network etc (Sharma et al., 2021). Malware could be of different forms such as worms, trojans, viruses, ransomware, spyware, adware et. Its main objective is to prevent data theft or system disruption due to any type of financial fraud (Butt et al., 2020). Now malware detection is the detection of malicious software. It is a process which is used to classify and identify malicious types of software within a computing system. It includes use of a different type of technologies and techniques as well for analyzing various files, network traffic, and system behavior for detecting the presence of malware and mitigating its impact.

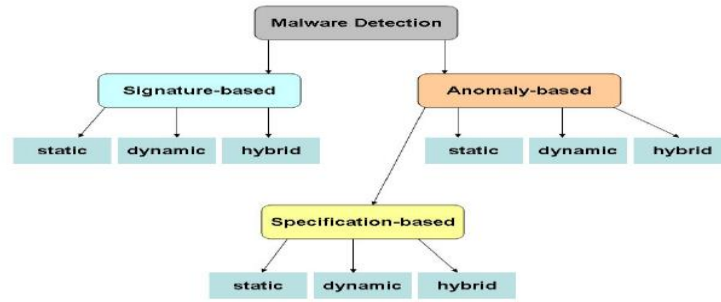


Figure 1.1: Malware Detection Techniques (Idika and Mathur, 2007)

Malware detection in Portable Executable (PE) files, usually found in Windows operating systems which is used to focus mainly on identifying malware within executable files (.exe), dynamic link libraries (.dll) (Azeez et al., 2021), and other executable formats. PE files contain executable types of code, resources, and metadata, making them a common target for malware authors seeking to compromise systems and networks.

The Project analyzed various file properties of course, found various things in it like file size, header information, section headers, import/export tables, and embedded resources, as well as dynamic analysis methods, including sandboxing and code things, are commonly used to detect malware in PE files. In addition to being effective tools for malware detection, machine learning and deep learning techniques have made it possible to create models that are trained on big datasets and can certainly identify PE files in a reliable manner like if it is spam or benign based on their characteristics and behavior.

1.1 Aim of the Study and Contribution:

This study aims to use deep learning models and to enhance the effectiveness and accuracy of this malware detection project of Portable Executable (PE) files. This study has been used mainly four deep learning models of course which includes CNN, LSTM, BiLSTM and Stacked BiLSTM with Attention Mechanism. The dynamic and developing nature of malicious software mostly makes traditional methods of malware detection not suitable of course, requiring the investigation of new strategies. With the use of deep learning architectures, including Stacked BiLSTM with Attention Mechanism, the project hopes to overcome some of the drawbacks of traditional type of models and get superior precision in malware identification and classification. By giving a more reliable and effective method of identifying malware, the research is looking to advance the field of cybersecurity and improve the general security posture of networks and systems. this incorporation of BiLSTM based model with attention mechanism which further enables to focus on relevant parts of input sequences while neglecting the irrelevant information.

- **Use of Attention mechanism with Stacked BiLSTM** : this novel approach of incorporating and utilizing the strength of stacked BiLSTM with attention mechanism. Stacking layers of BiLSTM enhances the model's capacity which will learn patterns and dependencies, while attention mechanism will help the project to selectively attend the most informative part of the input.

- **Enhanced Feature:** By using the Attention mechanism, the research can effectively extract and give priority to critical features of the input sequence, which will lead to improved classification of benign and malicious software.
- **Improved Accuracy:** Comparing the Evaluation result of the benchmark datasets, the effectiveness of Stacked BiLSTM with attention mechanism is demonstrated in achieving higher accuracy in this classification problem.

1.2 Research objectives:

The research objectives of this study are as follows:

1. To investigate and analyze traditional methods of malware detection to understand their limitations and shortcomings in addressing obviously malware threats.
2. To do a comparative analysis of Stacked BiLSTM which is this projects work and with Prior work.
3. To develop and implement novel deep learning architectures, such as Stacked BiLSTM with Attention Mechanism, and main aim of course to achieve superior accuracy and robustness compared to traditional methods.

1.3 Research Questions :

The research questions addressed in this study are as follows:

1. How does the proposed novel deep learning architecture, Stacked BiLSTM with Attention Mechanism, perform in terms of malware detection compared to traditional methods in terms of accuracy?

1.5 Research Gaps:

There are still a number of research gaps in the field of malware detection, despite giving very good developments in the field. First off, all, the research is on the effectiveness of deep learning architectures which is of course designed for malware detection tasks—like Stacked BiLSTM with Attention Mechanism—is lacking. Although traditional type of techniques has given good and encouraging outcomes, their capacity to adjust to quickly changing virus iterations is still restricted. The diversity and size of existing datasets might not be sufficient to train reliable machine learning models that can successfully generalize to real-world situations. Moreover, there is a knowledge vacuum about the actual implementation of machine learning models for real-time virus detection in applications. Further research is necessary to fully understand the issues that the incorporation of machine learning into applications brings in terms of scalability, efficiency, and user experience. In order to improve cybersecurity defenses against emerging threats and advance the state-of-the-art in malware detection, it will be essential and important to address these research gaps.

2. Literature Review

2.1 Traditional Methods of Malware Detection

There are several traditional methods which had been used in malware detection including Signature-based detection, Heuristic analysis, Behavior-based detection, Sandbox analysis, Anomaly detection, etc.

The spread of malware in modern computing presents a serious problem, which drives academics to come up with creative ways to strengthen computer security. To tackle this threat, several approaches have been developed, including behaviour-based and signature-based detection methods. (Botacin et al., 2022) presents HEAVEN, a framework that combines hardware and software to speed up signature-based malware detection in real-time. It achieves a 100% detection rate without any false positives and of course reduces the amount of CPU cycles needed for scanning. Similarly, (Fortino et al., 2023) introduces SigIL, which focuses again and do signature scanning on intermediate languages, making it possible to detect malware on different devices. The complexities of signature-based malware detection are explored in (Goyal and Kumar, 2020) and (Jalilian et al., 2020), where (Goyal and Kumar, 2020) looks at both static and dynamic features for classification obviously using machine learning classifiers and (Jalilian et al., 2020) suggests a of course novel approach which is based on N-gram distribution and a Top K approach for file classification, both of which show promise for malware detection. Besides this, (Assegie, 2021) presents an enhanced KNN model derived from API call sequence analysis, with a remarkable 98.17% detection accuracy for malware.

Moreover, (Suryati and Budiono, 2020) concentrates on heuristic-based approaches designed especially for Android devices, which target the Android OS. Similarly, (Yunmar et al., 2024) explores Android malware detection which basically do expose the drawbacks of current hybrid solutions and promoting on-device detection and better usability. Meanwhile, (Li et al., 2021) provides a more comprehensive overview of heuristic-based malware detection techniques, stressing the necessity of precise detection in obviously various malware kinds.

Table 2.1: Comparison Table on Traditional methods of malware detection

Study	Methodology	Features	Algorithm	Main Contribution
(Botacin et al., 2022)	Hardware and software combination (HEAVEN)	Hardware-assisted signature matching, branch pattern history	Machine learning	Accelerated real-time malware detection with reduced CPU cycles

(Fortino et al., 2023)	SigIL: Signature scanning on Intermediate Language	Intermediate representation of binaries	Machine learning	Cross-device malware detection independent of architectural specifics
(Goyal and Kumar, 2020)	Signature-based malware detection	String features for static analysis, nonrepetitive consecutive API calls for dynamic analysis	Machine learning classifiers (k-Nearest Neighbors, Gaussian Naive Bayes, Multi Naive Bayes, Decision Tree, Support Vector Machine, Random Forest)	Detailed exploration of both static and dynamic features for classification
(Jalilian et al., 2020)	Static signature-based malware detection	Opcode and binary file signatures	N-gram distribution, Top K approach	Proposal of a novel signature-based detection method
(Assegie, 2021)	Optimized KNN model for signature-based malware detection	API call sequences	K-nearest Neighbor (KNN)	Development of an efficient KNN-based model for real-time intrusion detection
(Suryati and Budiono, 2020)	Heuristic-based malware detection for Android	-	Hybrid approach integrating static and dynamic methods	Identification of strengths and limitations in hybrid detection solutions
(Yunmar et al., 2024)	Survey on hybrid Android malware detection	-	-	Examination of challenges and opportunities in existing hybrid detection approaches
(Li et al., 2021)	Survey on feature extraction methods of heuristic malware detection	Features extraction techniques	-	Overview of feature extraction methodologies and associated advantages/challenges

2.2 Advanced Methods of Malware Detection

There are various machine and deep learning models which are considered as advanced methods in malware detection for PE files. Bai et al. (2014)'s suggested approach makes use of the PE file's format information through extensive static analysis and mining. In this paper, all of the PE files' features were retrieved for the PE header file, where the classification approach was used to improve the features' completeness and decrease their dimensionality for accuracy. To differentiate between malware and benign software, a classification system was used to train the chosen attributes.

Furthermore, (Qiang et al., 2022) suggests a strong dynamic analytic technique for malware identification that makes use of particular fine-grained behavioral characteristics, like control flow traces, and combines CNN and LSTM networks. Their suggested classifier of course shows robustness against hostile samples and packing approaches, and it is effective in identifying known as well as unknown malware variants. For example, (Demirci, 2022) suggests using generative pre-trained transformer-based (GPT-2) and stacked bidirectional long short-term memory (Stacked BiLSTM) deep learning language models for detecting malicious code. A stacked BiLSTM-based deep learning language model for malware detection is also introduced in (Demirci, 2022), which uses assembly instructions taken out of Portable Executable (PE) files. To attain high detection accuracy, they use Sentence Level Analysis Models (SLAM) and Document Level Analysis Models (DLAM). Another study given by (Aslan et al., 2021) who suggested a Convolutional Neural Network (CNN) model based on deep learning (DL) that classifies malware in PE binary files. To improve classification performance, the CNN model makes use of fusion feature sets and large-scale learning approaches.

3. Research Methodology

3.1 Methodology

Our approach is of course presenting structured way in understanding PE file malware classification in this project. The process will involve business understanding first which is used to enhance malware detection in PE files with the help of classification. Then it comprises collecting PE file datasets and exploring them, cleaning them then preprocessing data. Then four DL models are going to be implemented which are, CNN, LSTM, BiLSTM and stacked BiLSTM with an attention mechanism.

1. **Business Understanding:** The study's main focus for the business knowledge phase is of course the necessity of strengthening defenses against malware attacks that target Windows systems. The objective is to create a advanced type of detection system that can quickly detect and eliminate threats concealed as Portable Executable (PE) files, given the increase in malicious software.

Security analysts, system administrators, and software engineers who are in charge of protecting sensitive data and corporate assets are examples of stakeholders.

2. **Data Understanding:** To fully understand the data for this project, a thorough type of analysis and examination for the PE file dataset have been derived from the supplied CSV file is very very essential and required as well. This will definitely look at the attributes that the PEfile library was able to extract in order to learn more about the traits of both malicious and benign PE files. Developing efficient detection algorithms requires an understanding of the distribution of features and the frequency of various malware variants.
3. **Data Preparation:** The third phase is data preparation which includes of course several steps which ensure that the deep learning models have the dataset which should be well suited. First, this includes fixing any missing values, eliminating outliers, and resolving data quality concerns to clean the dataset. Then, to make model training easier, preprocessing operations including scaling numerical features and encoding categorical variables are carried out. To balance the dataset and ensure a fair representation of both classes, methods such as SMOTE oversampling are used in light of the possible class imbalance of course between benign and malicious samples. Techniques for feature selection can also be used to determine which features are most important for classifying malware.
4. **Modelling:** For this project, modeling will choose, optimise, and train deep learning models to classify PE files which would be either harmful or benign. BiLSTM (Bidirectional Long Short-Term Memory) and stacking BiLSTM with attention processes are the two main models that are taken into consideration. These models are selected based on their capacity to efficiently extract sequential patterns from PE file characteristics, an essential skill for identifying sophisticated malware behaviors. The models are trained on the preprocessed dataset using the TensorFlow and Keras libraries, with hyperparameters adjusted and architectures optimized for best results. Metrics like accuracy, loss, and validation scores are used to track the model's performance during training in order to guarantee convergence and avoid overfitting.
5. **Evaluation:** This project's evaluation includes a thorough analysis of how well the trained deep learning models classified PE files as harmful or benign. A range of evaluation criteria are employed, such as specificity, sensitivity, confusion matrices, accuracy, loss, and classification reports. The models' capacity is of course to accurately classify samples, recognize real positives and negatives, and reduce false positives and negatives is revealed by these metrics.
6. As an Additional work in this project, we could further utilize the Flask framework to incorporate the trained deep learning model into an intuitive online application. Users can upload PE files to the

application to get real-time infection predictions. After a file is received, the program will use the “pefile” library to extract pertinent features, which are then sent to the trained model for categorization. After that, the algorithm predicts the file's likelihood of being benign or malicious, giving consumers immediate feedback on the possible threat level. To enable ongoing development and response to changing threats, monitoring technologies are also used to track program performance and user interactions.

3.2. Libraries Imported

In this project, there are several libraries which have been used and imported for different and varied tasks which include model development, preprocessing purpose, evaluation stage, visualization tasks and all. The imported libraries comprise a few for data manipulation like numpy and pandas, for data visualization, we are using seaborn and matplotlib, for building up deep learning models we have tensorflow and keras, scikit-learn for preprocessing techniques and evaluation metrics, learn for addressing class imbalance using SMOTE oversampling, and plotly for interactive visualization. In addition to these libraries, this study is going to deal with sys and os modules as well which have been imported for system-related operations pickle and JSON for serialization and deserialization of Python objects, and warnings module to suppress specific warnings during runtime. These libraries collectively provide the necessary tools and functionalities to explore, preprocess, train, evaluate, and deploy machine learning models for the classification of PE file malware.

3.4 Data Preprocessing:

In this study, this phase of the project is data preprocessing which is quite important and plays a vital role in preparing the dataset for doing evaluation and model training. There are several processing techniques, but this study is going to perform label encoding which converts the categorical type of variables into the numerical type of representations. This will also make it suitable for deep learning algorithms and machine learning algorithms as well. In addition to that, given the imbalance between the number of benign and malicious PE files, we are going to employ the Synthetic Minority Over-sampling Technique (SMOTE) which is applied to address the class imbalance by generating synthetic samples for the minority class (malicious files) to balance the dataset. After that, this will ensure that ML models should be trained on a balanced type of dataset and on comprehensive as well, thereby improving their ability to generalize and accurately classify both benign and malicious files. Furthermore, there are some other preprocessing steps such as handling missing values, scaling numerical features, and potentially selecting relevant features based on feature importance analysis may also be performed to further enhance model performance.

3.5 Data Splitting (Training and Testing the Model):

After the data preprocessing the next phase is the data splitting phase which will divide the dataset in the ratio of 80:20. Training, validation. This will ensure that for the training of models, an adequate amount of portion of data should be allocated while also providing separate subsets for validation and evaluation to assess the model's performance effectively. The 'Malware' column, which acts as the target variable indicating the classification label of each PE file, is separated from the other features to complete the feature-target split. The 'Malware' column have been of course removed from the dataset to extract the features (X), leaving only the attributes needed for model training and prediction.

3.6 Dataset Description

Table: Details of dataset Used

Dataset Name:	Benign & Malicious PE Files
Number of Files:	dataset_malwares.csv (6.4 Mb)
Number of Rows:	19611
Number of Features:	78
Missing Values:	0

The dataset which this study is going to perform on Benign and Malicious Portable Executable (PE) files. This dataset has been sourced from the Kaggle repository having the URL <https://www.kaggle.com/datasets/amauricio/pe-files-malwares>. This dataset of course contains a collection of Portable Executable (PE) files which represent both benign instances and malicious instances. Built as a component of a research project centered on malware detection and machine learning, it is not a valuable tool for researching and creating defense strategies against online dangers. Using the pefile Python module, extracted features from each PE file reveal details about the structure and properties of the executable files. The features will obviously and of course cover a broad range of characteristics, such as metadata associated with the PE file header, such as 'e_magic', 'e_cblp', 'e_cp', and 'e_crlc', which indicate characteristics such as file size and signature, as well as details regarding memory allocation and file sections. The characteristics "Machine," "NumberOfSections," and "SizeOfOptionalHeader," which provide information about the executable's architecture, segmentation, and optional header size, are other noteworthy features. Additionally, the dataset contains very very important attributes like "AddressOfEntryPoint," "TimeDateStamp," and "ImageBase," which offer geographical and temporal type of details regarding memory structure and file execution.

3.7 List of Models:

There are various deep-learning models which have been used in this paper to classify malware found in PE files. A list of the models utilized in this report is provided below:

1. **Convolutional Neural Network (CNN):** The CNN model is firstly used in this study which is a deep learning model to enhance malware detection. It has been trained on diverse dataset of Portable Executable (PE) files, it identifies intricate type of patterns indicative of malicious behavior.

2. **Long Short-Term Memory (LSTM):** The LSTM model, has been used in this study to bolster malware detection. This model addresses the shortcomings of traditional detection methods, which will enhanced accuracy in identifying evolving malware threats.
3. **BiLSTM (Bidirectional Long Short-Term Memory):** The full form of BiLSTM is Bidirectional Long Short-Term Memory. It is a neural network architecture and it is of course a variation of Long Short-Term Memory (LSTM) that can process input sequences both forward and backwards. This deep learning model is of course useful for sequential type of data, like PE file characteristics, since it integrates with bidirectional processing, which allows it to capture dependencies in both past and future things. Its ability to pick up long-term dependencies and remember data over lengthy periods is essential for identifying complex patterns that point to virus behaviour.
4. **Stacked BiLSTM with Attention Mechanism:** This model combined with an attention mechanism, which used to extend the BiLSTM architecture. By enabling the model to concentrate on relevant type of segments of the input sequence, attention mechanisms improve the model's capacity to identify significant characteristics and filter out unimportant noise. This model may learn hierarchical representations of the input data by stacking many BiLSTM layers and integrating attention mechanisms, which may enhance classification performance. Which is the reason behind why we chose this model as our novel approach.

These models were selected based on their capacity to handle sequential data efficiently and to represent complex patterns found in PE file features. The goal of the classification system is obviously achieve high accuracy and robustness in detecting malware in PE files by utilizing deep learning architectures such as BiLSTM and stacked BiLSTM with attention methods.

3.8. Designing Our Workflow

So, basically this section is on design specifications of the workflow which will of course provides a complete overview for the criteria and requirements needed to construct a malware classification system that uses deep learning models. It provides a basic overview of the system's design, functionality, and technical specifications, acting as a blueprint for the entire project. The project's goals and scope have been established, which highlights the necessity of improving cybersecurity protocols by accurately identifying malware in PE files. It lists the important parties who are involved, such as software developers, system administrators, and security analysts, whose specifications and input will influence the design of the system.

The section then goes on to explain the system architecture, including all of the parts and how they work together. A number of modules, including ones for data preparation, model training, evaluation, and deployment, are included in the design of the system. Every module has various and separate responsibilities, including preprocessing and cleaning the dataset, training deep learning models, assessing the performance of the models, and integrating the generated models into applications. The technical specifications for every module, including the frameworks, libraries, and implementation tools, are also described in the design specification. To handle class imbalance, for example, the data

preprocessing module needs libraries like pandas, numpy, and scikit-learn for data manipulation and preprocessing methods like SMOTE. Similarly the model training module builds and trains CNN, LSTM, BiLSTM and stacked BiLSTM models using deep learning frameworks like TensorFlow and Keras.

The section of course do outlines each module's input and output specifications. While the model training module needs preprocessed feature vectors and associated malware labels, the data preprocessing module accepts input in the form of a dataset with PE file features extracted using the pefile library. Performance measures including accuracy, loss, confusion matrices, and classification reports are produced by the evaluation module and are essential for determining the efficacy of the model.

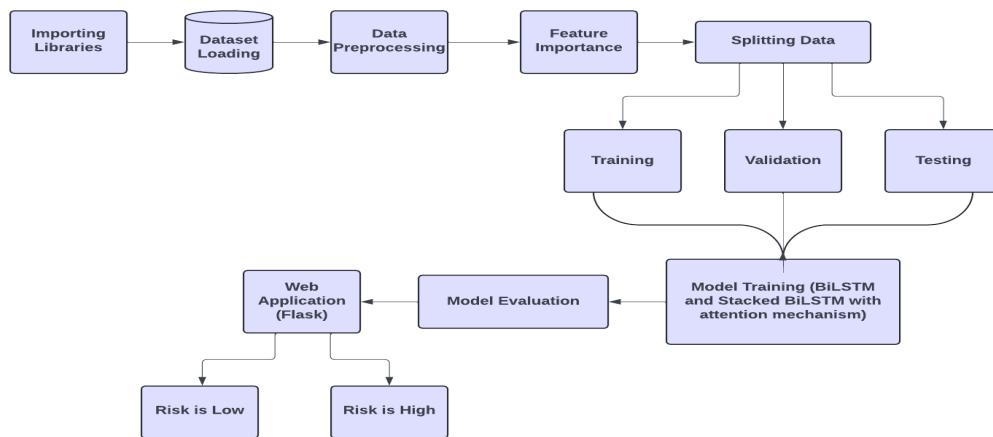


Figure 3.1: Proposed Workflow

4. Implementation

4.1 CNN

A Convolutional Neural Network (CNN) is a deep learning architecture which has been designed for processing and analyzing of course grid-like structured data, such as images, time series, and sequences (Dong et al., 2021). It consists multiple layers, including convolutional, pooling, and fully connected layers, each serving a specific purpose in feature extraction, dimensionality reduction, and classification. At its core, a CNN leverages convolutional layers has been there which is used to detect local patterns and features within the input data through a process called convolution. This CNN model consists of two Conv1D layers with 32 filters each and a kernel size of 2, followed by Rectified Linear Unit (ReLU) activation functions to introduce non-linearity.

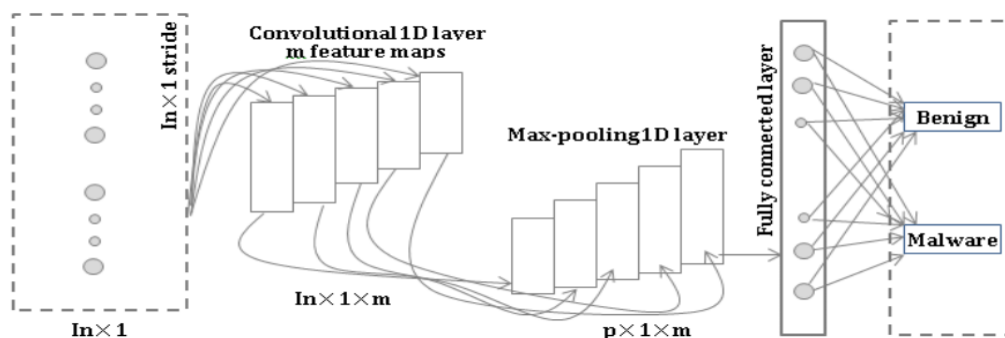


Figure4.1 :CNN architecture (Vinayakumar et al., 2019)

4.2 LSTM

A Long Short-Term Memory (LSTM) model is a type of recurrent neural network (RNN) architecture which has been designed to address the gradient problem in sequential data (Sherstinsky, 2020). The LSTM architecture comprises input, forget, and output gates, along with a memory cell that regulates the flow of information (Staudemeyer and Morris, 2019). This LSTM model comprises several layers, starting with an LSTM layer with 8 units and `return_sequences=False`, indicating that it returns only the output of the last timestep and not the full sequence. The input shape is determined by the dimensions of the training data (`X_test`) with a single feature.

4.3 BiLSTM

This study is going to be performed on Bidirectional Long Short-Term Memory (BiLSTM) neural network architecture. This architecture is of course going to be implemented for the classification of PE files. To classify PE file malware, a Bidirectional Long Short-Term Memory (BiLSTM) neural network architecture is used in this study. The BiLSTM model's architecture consists of multiple layers that are planned to extract and process characteristics from the input data (Aslan et al., 2021). With eight units, the first layer is a bidirectional LSTM layer that is set up to return sequences to maintain temporal information. This type of layer will of course captures dependencies and patterns within the data, it is also extracted from the PE files of sequential features. This layer adds non-linearity to the model and most importantly it improves its representational ability. The output tensor is transformed into a one-dimensional vector by adding a Flatten layer after the dense layer. This one-dimensional vector is then passed through another dense layer with eight units and ReLU activation. To avoid overfitting, a Dropout layer is also included, which randomly drops a portion of the units during training. Its dropout rate is 0.5.

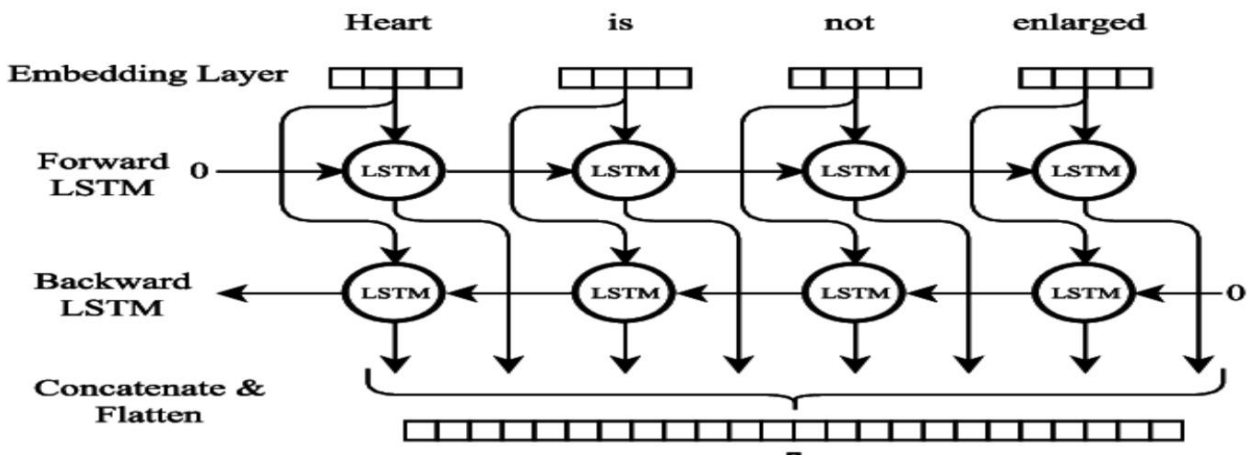


Figure 4.2: BiLSTM architecture (<https://paperswithcode.com/method/bilstm>)

4.4 Stacked BiLSTM with Attention Mechanism

The next model is Stacked BiLSTM with Attention Mechanism. It is a hybrid model that combines two different types of architectures. This architecture is a very very polished neural network which is designed for handling sequential types of data which includes PE files which has been extracted its features. This Stacked BiLSTM with Attention Mechanism architecture of course contain multiple types of layers whose main purpose to serve the processing of the input data and capturing it. The Attention mechanism used to improve the model's capacity to

identify significant characteristics and omit unimportant data (Niu et al., 2021). Various input sequence segments are given weights by this method in real-time, according on how relevant they are to the current job. This approach is definitely implemented by the `Attention` class defined in the code. Initially, it sets up the trainable weights $\{W\}$ and $\{b\}$, which are then utilized to calculate the attention scores for every component in the input sequence (Altaheri et al., 2023).

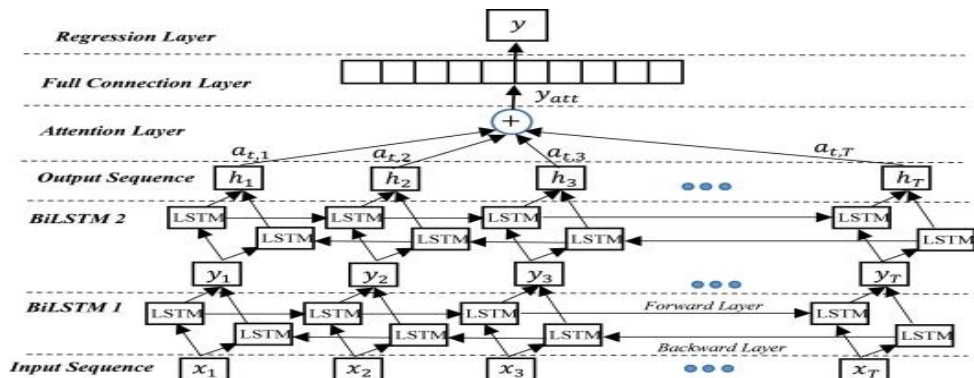



Figure 4.3: Stacked BiLSTM with Attention Mechanism Architecture (Rathore and Harsha, 2022)

5 Evaluation

5.1: CNN Results:


Classification Report :

	precision	recall	f1-score	support
0	0.52	1.00	0.69	1306
1	1.00	0.09	0.17	1322
accuracy			0.54	2628
macro avg	0.76	0.55	0.43	2628
weighted avg	0.76	0.54	0.43	2628

Figure 5.1: Classification report where accuracy : 54%

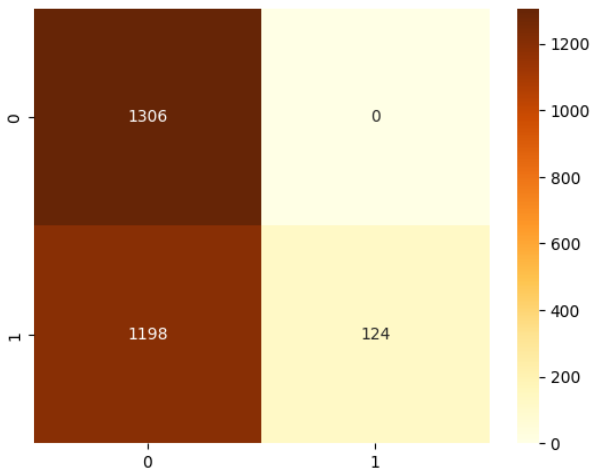


Figure 5.2: Confusion matrix

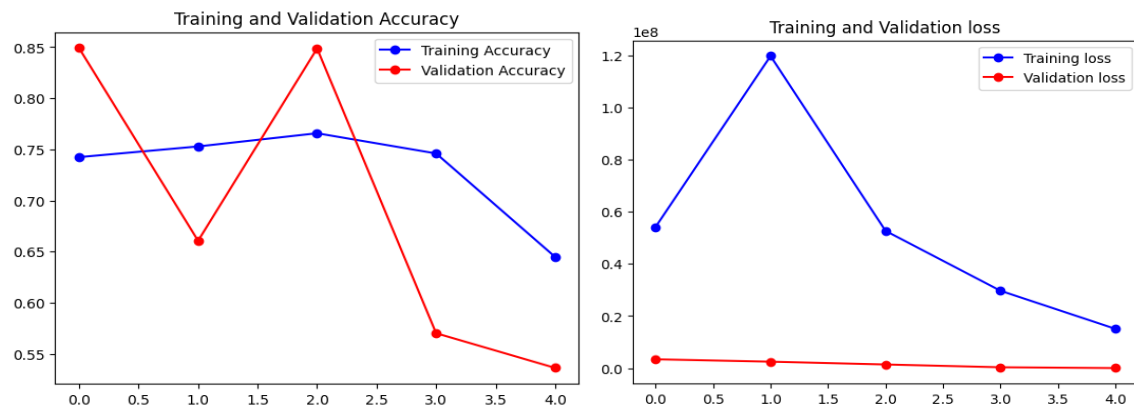


Figure 5.3: Accuracy and Loss Graph

This CNN model has been achieved very poor accuracy with 54% which indicates and has classified accurately PE files as malicious or benign.

5.2: LSTM Results:

Classification Report :				
	precision	recall	f1-score	support
0	0.98	0.90	0.94	1306
1	0.91	0.98	0.94	1322
accuracy			0.94	2628
macro avg	0.94	0.94	0.94	2628
weighted avg	0.94	0.94	0.94	2628

Figure 5.4 : Classification report where accuracy : 94%

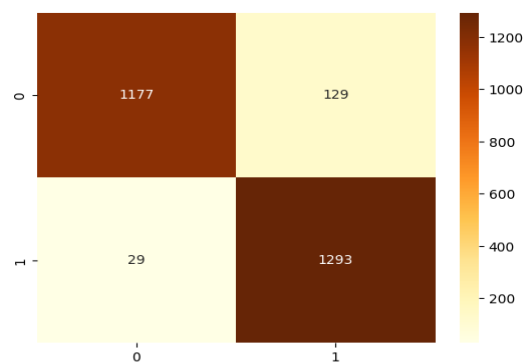


Figure 5.5 Confusion matrix

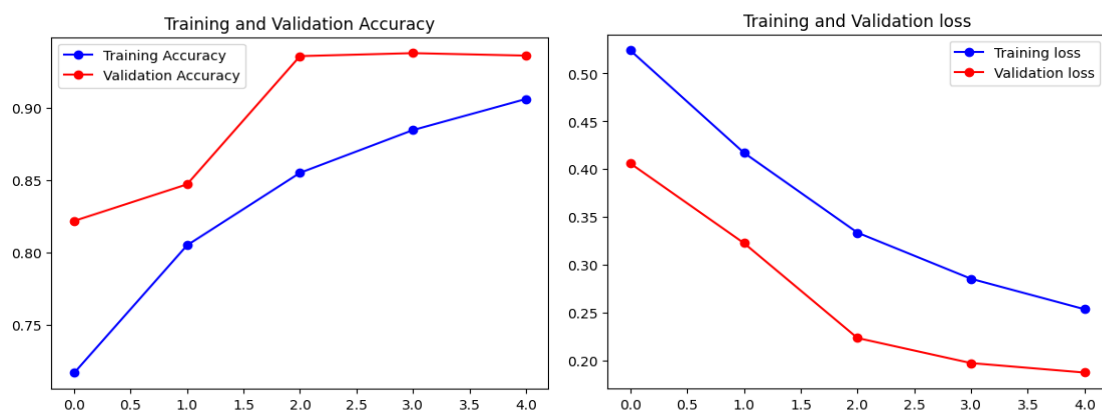


Figure 5.6: Accuracy and Loss Graph

This LSTM model has been achieved quite good accuracy with 94% which indicates and has classified accurately PE files as malicious or benign.

5.3: BiLSTM Results :

Classification Report :				
	precision	recall	f1-score	support
0	0.98	0.92	0.95	1277
1	0.93	0.98	0.95	1351
accuracy			0.95	2628
macro avg	0.95	0.95	0.95	2628
weighted avg	0.95	0.95	0.95	2628

Figure 5.7: Classification report where accuracy: 95%

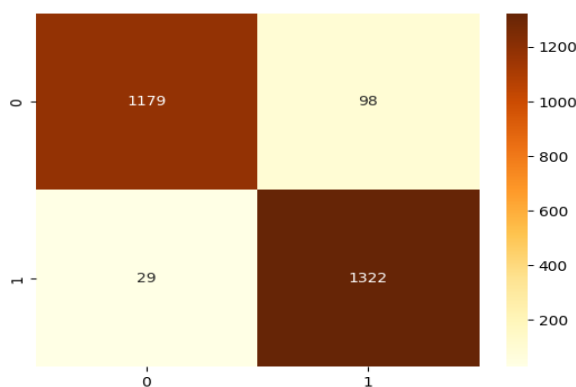


Figure 5.8: confusion matrix

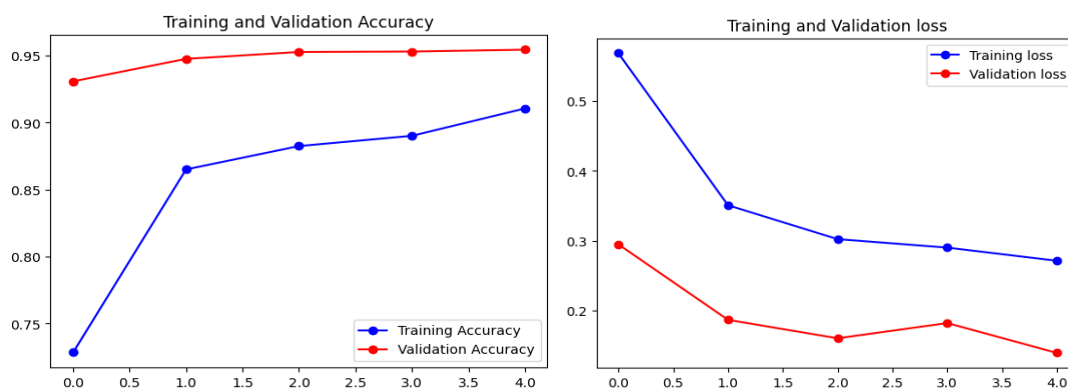


Figure 5.9: Accuracy and Loss Graph

This Bidirectional Long Short-Term Memory (BiLSTM) model has been achieved quite good accuracy with 95% which indicates good effectiveness and has classified accurately PE files as malicious or benign. By utilizing this type of bidirectional architecture, it captures some sequential patterns with good ability within the data, this model showed quite good strong performance.

5.4: Stacked-BiLSTM with Attention Mechanism:

Classification Report :					
	precision	recall	f1-score	support	
0	0.99	0.96	0.97	1339	
1	0.96	0.99	0.97	1289	
accuracy			0.97	2628	
macro avg	0.97	0.97	0.97	2628	
weighted avg	0.97	0.97	0.97	2628	

Figure 5.10: Classification report where accuracy: 97%

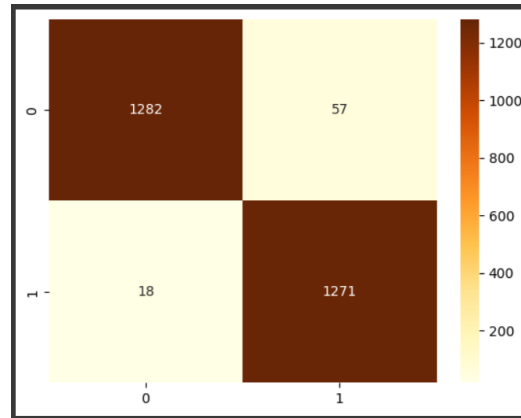


Figure 5.11: confusion matrix

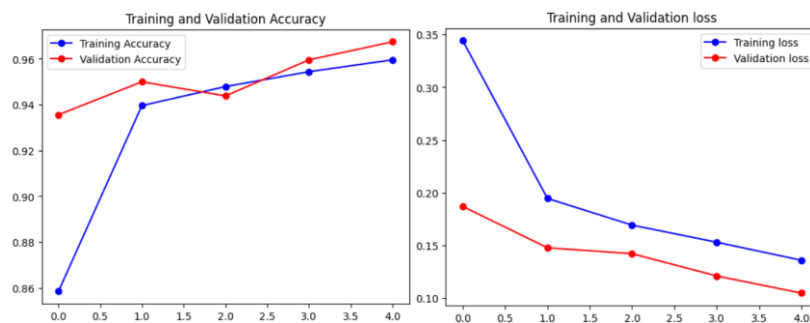


Figure 6.12: Accuracy and Loss Graph

This Stacked BiLSTM with Attention Mechanism has achieved a noteworthy and good accuracy of 0.97, surpassing the BiLSTM model's performance. This suggests that the hybrid architecture achieves better classification performance by efficiently capturing complex patterns in the input data by utilizing both BiLSTM layers and attention methods. The Stacked BiLSTM with Attention Mechanism model, the better performing of the two models examined in this project, combines the advantages of BiLSTM for sequential data processing and attention mechanisms for feature extraction. This allows the model to distinguish between benign and malicious PE files with greater accuracy.

5.5 Classification Performance of Deep Learning Models

This section of classification performance of deep learning models which includes two models has been assessed using recall, f1 score, and precision type of metrics. F1-score, precision, and recall metrics were used to evaluate the classification performance of deep learning models. Precision, recall, and F1-score for the malware (1) and benign (0) classes in the BiLSTM model were 0.98/0.92/0.95 and 0.93/0.98/0.95, respectively. In contrast, the Stacked BiLSTM with Attention Mechanism scored higher—0.99/0.96/0.97 for malware and 0.96/0.99/0.97 for benign—for both categories. With better precision and recall for both classes, the Stacked BiLSTM with Attention

Mechanism outperformed the other model, proving that it is capable of correctly identifying benign and malicious PE files and so on for other models.

Table 5.1: Accuracy Table of all 4 models

Models	Accuracy
CNN	54%
LSTM	94%
BiLSTM	95%
Stacked BiLSTM with Attention Mechanism	97%

5.6 Comparative Analysis:

In comparative analysis, the Stacked BiLSTM with Attention Mechanism, our best model, outperforms prior work, achieving an accuracy of 97%. While the CNN-BiLSTM from prior studies achieved a comparable accuracy of 95.7%, our BiLSTM model achieved 95%. The significant improvement in accuracy by the Stacked BiLSTM with Attention Mechanism underscores its effectiveness in capturing complex patterns and dependencies within the data, highlighting its superiority over traditional models. This demonstrates the potential of advanced deep learning architectures for enhancing malware detection capabilities.

Table 5.3: Comparison of Deep Learning Models with prior work

Model	Accuracy
CNN-BiLSTM (Prior work)	95.7%
Stacked BiLSTM with Attention Mechanism (Best Model)	97%

6 Conclusion and Future Works:

6.1 Conclusion:

In conclusion this study is presenting and working in malware detection domain with the help of deep learning models. The dataset which has been employed in this report having collection if various malicious and benign Portable

Executable (PE files). This dataset has been sourced from Kaggle and their link is already in the dataset description section. PE file classification is done in a methodical manner through the project workflow, which includes data loading, cleaning, visualization, preprocessing, model training, assessment. To categorize PE files as benign or malicious, a variety of deep learning models were investigated, such as CNN, LSTM, BiLSTM and Stacked BiLSTM with Attention Mechanism. With an accuracy of 97%, the Stacked BiLSTM with Attention Mechanism surpassed the BiLSTM model, which had an accuracy of 95%. The latter proved to be more successful in correctly identifying risks present in executable files, exhibiting higher precision, recall, and F1-score for both malware and benign classifications. When a deep learning model like Stacked-BiLSTM with attention mechanism is integrated with a Flask web application, users may submit executable files and get instant safety feedback through an easy-to-use interface. When a file is uploaded, the program runs it through the trained model and returns a classification result that says whether or not the file contains malware. Users are of course notified of the danger involved in the file they have submitted; a low-risk designation means the file is safe to use. With the help of this web application, users can evaluate executable files' security before interacting with them, which is a useful tool for improving cybersecurity protocols.

6.2 Future Works

To further increase the effectiveness and reach of the malware detection system, I need a number of extension obviously and upgrade opportunities as well which can be investigated in further work. First off, adding a bigger and more varied set of PE files to the dataset would improve the model's capacity to generalize to previously unseen data and accurately identify newly emerging malware types. Furthermore, adding more features that have been taken out of PE files—like byte-level n-grams, dynamic analysis features, or API call sequences—might enhance classification accuracy and offer more detailed depictions of malware activities. Additionally, by combining the predictions of several base models, investigating ensemble learning strategies like model stacking or boosting may improve the classification system's robustness and generalization skills. By using this method, of course the effects of individual model biases may be lessened and overall performance may be enhanced. This study should deploy transformer based models which can show promising results and may offer improved feature extraction which is capable for the analysis of PE files.

References

Sharma, N.A., Kumar, K., Raj, M.A. and Ali, A.S., 2021, December. A Systematic Review of Modern Malicious Softwares and Applicable Recommendations. In 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE) (pp. 1-6). IEEE.

Butt, U.J., Abbod, M.F. and Kumar, A., 2020. Cyber threat ransomware and marketing to networked consumers. In Handbook of research on innovations in technology and marketing for the connected consumer (pp. 155-185). IGI Global.

Idika, N. and Mathur, A.P., 2007. A survey of malware detection techniques. Purdue University, 48(2), pp.32-46.

Azeez, N.A., Odufuwa, O.E., Misra, S., Oluranti, J. and Damaševičius, R., 2021, February. Windows PE malware detection using ensemble learning. In *Informatics* (Vol. 8, No. 1, p. 10). MDPI.

Baldangombo, U., Jambaljav, N. and Horng, S.J., 2013. A static malware detection system using data mining methods. *arXiv preprint arXiv:1308.2831*.

Botacin, M., Alves, M.Z., Oliveira, D. and Grégio, A., 2022. HEAVEN: A Hardware-Enhanced AntiVirus ENgine to accelerate real-time, signature-based malware detection. *Expert Systems with Applications*, 201, p.117083.

Fortino, G., Greco, C., Guzzo, A. and Ianni, M., 2023, September. SigIL: A Signature-Based Approach of Malware Detection on Intermediate Language. In *European Symposium on Research in Computer Security* (pp. 256-266). Cham: Springer Nature Switzerland.

Goyal, M. and Kumar, R., 2020, October. The pipeline process of signature-based and behavior-based malware detection. In *2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA)* (pp. 497-502). IEEE.

Jalilian, A., Narimani, Z. and Ansari, E., 2020. Static signature-based malware detection using opcode and binary information. In *Data Science: From Research to Application* (pp. 24-35). Springer International Publishing.

Assegie, T.A., 2021. An optimized KNN model for signature-based malware detection. Tsehay Admassu Assegie. "An Optimized KNN Model for Signature-Based Malware Detection". *International Journal of Computer Engineering In Research Trends (IJCERT)*, ISSN, pp.2349-7084.

Suryati, O.T. and Budiono, A., 2020. Impact analysis of malware based on call network API with heuristic detection method. *International Journal of Advances in Data and Information Systems*, 1(1), pp.1-8.

Yunmar, R.A., Kusumawardani, S.S. and Mohsen, F., 2024. Hybrid Android Malware Detection: A Review of Heuristic-Based Approach. *IEEE Access*, 12, pp.41255-41286.

Li, N., Zhang, Z., Che, X., Guo, Z. and Cai, J., 2021. A survey on feature extraction methods of heuristic malware detection. In *Journal of Physics: Conference Series* (Vol. 1757, No. 1, p. 012071). IOP Publishing.

Qiang, W., Yang, L. and Jin, H., 2022. Efficient and robust malware detection based on control flow traces using deep neural networks. *Computers & Security*, 122, p.102871.

Demirci, D. and Acarturk, C., 2022. Static malware detection using stacked BiLSTM and GPT-2. *IEEE Access*, 10, pp.58488-58502.

Demirci, D., 2021. Static Malware Detection Using Stacked Bi-Directional LSTM (Master's thesis, Middle East Technical University).

Chaganti, R., Ravi, V. and Pham, T.D., 2023. A multi-view feature fusion approach for effective malware classification using Deep Learning. *Journal of Information Security and Applications*, 72, p.103402.

Aslan, M.F., Unlarsen, M.F., Sabanci, K. and Durdu, A., 2021. CNN-based transfer learning–BiLSTM network: A novel approach for COVID-19 infection detection. *Applied Soft Computing*, 98, p.106912.

- Niu, Z., Zhong, G. and Yu, H., 2021. A review on the attention mechanism of deep learning. *Neurocomputing*, 452, pp.48-62.
- Altaheri, H., Muhammad, G. and Alsulaiman, M., 2023. Dynamic convolution with multilevel attention for EEG-based motor imagery decoding. *IEEE Internet of Things Journal*.
- Rathore, M.S. and Harsha, S.P., 2022. An attention-based stacked BiLSTM framework for predicting remaining useful life of rolling bearings. *Applied Soft Computing*, 131, p.109765.
- Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P. and Venkatraman, S., 2019. Robust intelligent malware detection using deep learning. *IEEE access*, 7, pp.46717-46738.
- Dong, S., Wang, P. and Abbas, K., 2021. A survey on deep learning and its applications. *Computer Science Review*, 40, p.100379.
- Sherstinsky, A., 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404, p.132306.
- Staudemeyer, R.C. and Morris, E.R., 2019. Understanding LSTM--a tutorial into long short-term memory recurrent neural networks. *arXiv preprint arXiv:1909.09586*.
- Bai, J., Wang, J. and Zou, G. (2014), 'A malware detection scheme based on mining format information,' *The Scientific World Journal*, 2014