Malicious Network Activity Detection using BERT

Project Team

Aamir Ahmad Khan	P17-6150
Mubariz Ahmad Khan	P18-0010
Muhammad Saad Hassan	P17-6137

Session 2017-2021

Supervised by

Dr. Mohammad Nauman

Department of Computer Science

National University of Computer and Emerging Sciences
Peshawar, Pakistan

Dec, 2021

Student's Declaration

We declare that this project titled " *Malicious Network Activity Detection using BERT*", submitted as requirement for the award of degree of Bachelors in Computer Science, does not contain any material previously submitted for a degree in any university; and that to the best of our knowledge, it does not contain any materials previously published or written by another person except where due reference is made in the text.

We understand that the management of Department of Computer Science, National University of Computer and Emerging Sciences, has a zero tolerance policy towards plagiarism. Therefore, We, as authors of the above-mentioned thesis, solemnly declare that no portion of our thesis has been plagiarized and any material used in the thesis from other sources is properly referenced.

We further understand that if we are found guilty of any form of plagiarism in the thesis work even after graduation, the University reserves the right to revoke our BS degree.

Aamir Ahmad Khan Signature: ____________________

Mubariz Ahmad Khan Signature: ____________________

Muhammad Saad Hassan Signature: ____________________

____

Verified by Plagiarism Cell Officer
Dated:

Certificate of Approval

The Department of Computer Science, National University of Computer and Emerging Sciences, accepts this thesis titled *Malicious Network Activity Detection using BERT*, submitted by Aamir Ahmad Khan (P17-6150), Mubariz Ahmad Khan (P18-0010), and Muhammad Saad Hassan (P17-6137), in its current form, and it is satisfying the dissertation requirements for the award of Bachelors Degree in Computer Science.

Supervisor

Dr. Mohammad Nauman Signature: ____________________

____

Mashal khan

FYP Coordinator
National University of Computer and Emerging Sciences, Peshawar

____

Dr. Hafeez ur Rehman

HoD of Department of Computer Science
National University of Computer and Emerging Sciences

Acknowledgements

First of all, we are thankful to the Almighty that He bestowed His generous blessings upon us in a shape of sound health and well-being which were the primary touchstone for the accomplishment of this project.We are also thankful to our parents for their support and attention that we needed. They encouraged us at each stage of life and firmly supported us through our thin and thick, for which we are highly indebted to them.We acknowledge our deepest and profound gratitude to our worthy and valuable supervisor, Dr. Mohammad Nauman for his effort, hard work, timely corrections and valuable suggestions. We are extremely thankful and indebted to him for sharing his expertise, sincere and valuable guidance and continuous encouragement which enabled us to pursue this project.We take this opportunity to express gratitude to all of the Department faculty members for their help and support.

Aamir Ahmad Khan
Mubariz Ahmad Khan
Muhammad Saad Hassan

Abstract

The primary aim of this project is to devise a system or a mechanism which will be able to identify whether the Network Activity of a certain node connected to a Network is Malicious or not. For this purpose, the solution will be a Deep Learning based solution. Deep Learning is a major sub-field of Machine Learning. Deep Learning is a type of Machine Learning which mimics the way the Human Brain gains knowledge and thus it helps in training a model based on provided feedback. Malicious and Normal Network Activity will be used as fine-tuning data of the Neural Language Model, Since the advent of the design of Network Traffic, it's essence is a Network Language, hence State-of-the-art model, BERT is the deep Neural Model used as it's the most appreciable Model for this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces what the project is all about, the motivation behind doing this project, the objectives and scope of the project.

1.1 Introduction

With the outbreak of the Corona Virus, Work Places, Businesses and Institutions around the globe have shifted to the idea of Working from Home, which has enabled people around the world to stay connected via the Internet. The rise of data being shared across the internet has further given Cyber-Criminals all the more reason to Attack and Intercept the shared information. Due to increased public awareness about the importance of keeping online transactions and documents secure, Cyber-Criminals have also modified their network attack strategies making it difficult for traditional anomaly detection systems to effectively analyze and identify abnormal traffic, hence we propose a Deep Learning based solution, which analyses Network Activity sequences to classify whether the certain node is Malicious or Benign.

This project is about a malicious activity detection system, that will capture real time network traffic stream pass it through a Deep Neural Model, which analyses Net-work Activity sequences to classify whether the certain node is Malicious or Benign.

1.2 Motivation

Many global organisations have turned their attention to the Importance of Cyber-Security.

Many Research Organisations have also demonstrated success in Malicious Network Activity Detection and Malware Detection on the basis of Deep Learning Models. 'MMM-Malware Detection on Highly Imbalanced Data through Sequence Modeling' uses BERT and LSTM Network to learn Android Malware API Call Sequences, and have been successful in achieving high accuracy scores thus proving Neural Language Models can be used to achieve high score in various Security tasks.

The research of Radfordet al. [5] describes a frequency-based model for applying unsupervised anomaly detection to cybersecurity applications on the computer network traffic data.

The work of Hanet al. [6] proposes an attention model that can process network traffic streams with an adjustable length to detect payload-based attacks.

'Malware Detection by Analysing Network Traffic' deals with the problem of detecting malware on the basis of encrypted https traffic analysis on the basis host address,timestamps and data volume information of aggregated packets. The neural language model used is LSTM.

As with each passing day, Cyber Criminals and constructing new and improved methodologies to manipulate a Network, New and Advanced Techniques must be used to secure a Network, and with the success of Neural Language Models on Malware Detection, gives us all the more reasons to use Neural Language Model for the classification of Network Activity.

1.3 Objectives

First thing first and that is the collection of data to be feed to model. There are many resources which can provide data which they have collected from real case scenarios.

So we will be using dataset of Intrusion Detection Evaluation Dataset (CIC-IDS2017) and (CIC-IDS2012).

It is known to everyone that anomaly-based intrusion detection approaches are greatly suffering from consistent and accurate performance evolutions because of lack of reliable test and validation datasets. Intrusion Detection Systems (IDSs) and Intrusion Prevention Systems (IPSs) are the most efficient and important defensive tools for sophisticated and ever-growing attacks on network. (CIC-IDS2012) and CICIDS2017 are the most reliable and up-to-dated datasets with most frequent anomaly attacks and also benign. One can say that these datasets are the replica of real world data as it includes highly imbalanced ratio of anomaly and benign. These datasets also contain the results of the network traffic analysis, this is done through CICFlowMeter. CICFlowMeter consists of time stamp, source, and destination IPs, source and destination ports, protocols and attack in pcap files which is converted into CSV files to be fed to python.

The main focus of these datasets is generate realistic background traffic and to generate such a realistic traffic they built the abstract behaviour of 25 users based on the HTTP, HTTPS, FTP, SSH, and email protocols.

They identified 11 criteria which are very important for building a reliable benchmark data-set. No other previous IDS dataset had covered all of the 11 criteria, these 11 criteria are: Complete Network configuration, Complete Traffic, Labelled Dataset, Complete Interaction, Complete Capture, Available Protocols, Attack Diversity, Heterogeneity, Feature Set and MetaData.

1.3.1 Data Pre-processing

When data is collected pre-processing is needed so that we can use it for our what is required.

We will directly use Pre-trained Model (BERT) because training model from scratch will require huge amount of resources and time so that is why a pre-trained model will be used. Once pre-trained model is set, datasets of ISCX-2012 and ISCX-2017 are fed to that model (BERT) as to fine-tune the data by that pre-trained model. Both these datasets (ISCX-2012 and ISCX-2017) from the 'Canadian Institute for Cybersecurity' are very famous and widely used.

CIC-2012 was designed for injecting four types of attacks i.e. including internal penetration attacks, HTTP DoS attacks, DDoS attacks and brute force attacks. while CIC-2017 was designed for network traffic with the most common attack families i.e. brute force attacks, heart-bleeding attacks, botnets,DDoS attacks, and Web attacks.

In our case classification is done through Transmission Control Protocol (TCP), a single TCP is comprised of five tuple i.e. source-IP, source-port, destination-IP, destination-port and Internet protocol. Those Packets with these same five tuple are called as TCP stream.

1.4 Project Scope

The scope of the project is that, a node with a Deep Neural Model deployed on it, will be connected to a certain network. The node will capture real-time-stream-data as input, process and parse the raw real-time-stream-data such that it is suitable to be fed to our model. As a Trained Model will be deployed on the node, The model will be able to classify whether the activity in the Network is Malicious or not. If the activity is found to be Malicious, The Administrator will be notified to the mitigate the certain Malicious Node.

Chapter 2

Review of Literature and Proposed System

This chapter provides a comprehensive literature review of the related work to this project, the methodology and our proposed system for Malicious Network Activity Detection using BERT.

2.1 Dataset Literature Review

With exponential growth in the size of connected networks and developed applications, the important increasing of the potential injury which will be caused by launching attacks is turning into obvious. Meanwhile, Intrusion Detection Systems (IDSs) and Intrusion hindrance Systems (IPSs) area unit one among the foremost vital defense tools against the delicate and ever-growing network attacks. thanks to the dearth of adequate dataset, anomaly-based approaches in intrusion detection systems are suffering from exact processess, analysis and evaluation. There exist variety of such datasets like DARPA98, KDD99, ISC2012, and ADFA13 that are employed by the researchers to guage the performance of their projected intrusion detection and intrusion hindrance approaches. Supported the study of 11 on the market datasets since 1998, several such datasets are outdated and unreliable to use. a number of these datasets suffer from lack of traffic diversity and volumes, a number of them don't cowl the variability of attacks, whereas others anonymized packet data and payload that cannot replicate the present trends, or they lack feature set and meta-data.[4]

2.1.1 ISCX-IDS CIC 2012 Dataset

ISCX2012 has been divided into two profiles, An Attacker-Profile and a Benign-Profile, using which multi-stage attack scenarios are carried out. The dataset contains the network traffic for Simple-Media-Transfer-Protocol (SMTP), Secure-Shell-Protocol (SSH), Internet-Message-Access-Protocol (IMAP), Post-Office-Protocol 3 (POP3) and File-Transfer-Protocol(FTP) with full packet payload. However, there are no traces of HTTPS in this dataset, which consists of 70-percent of today's modern network traffic.

2.1.2 ISCX-IDS CIC 2017 Dataset

Two Networks were designed and implemented to create a vast data-set. An Attack Network and a Victim Network is a secure Network with a firewall, router, switches. The Attack-Network is a separate network and contains various operating system's such as Kali and Windows 8.1, which allowed performing different Attack Strategies.

The Victim-Network is a high secure infrastructure with Firewall, Router, switches and most of the common operating systems along with an agent that provide the benign behaviors on each PC. The Attack-Network is a completely separated infrastructure designed by a router and switch and a set of PCs with public IPs and different necessary operating systems for executing the attack scenarios.

2.1.3 Attacks Performed

- **Brute-Force-Attack:**
  Brute-Force-Attacks are one of the most common Attacks, It is usually performed for cracking passwords but can also be used to discover hidden content in a web application.

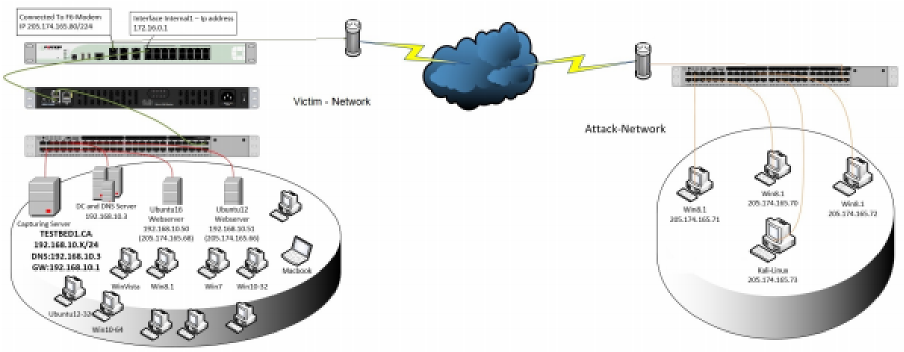


Figure 2.1: Testbed Architecture

- **Heart-Bleed-Attacks:**

  Heart-Bleed-Attacks come from a bug in OpenSSL cryptography library, That bug is used by Attackers by sending a malformed heartbeat request with a small payload but with the large length to the Victim, which can normally be a server to evoke a response.

- **Botnet:**

  A botnet owner can use a number of Internet-connected devices to perform tasks such as data-theft, sending spam data and can also the Attack to access the device and connection.

- **DoS Attack:**

  The attacker floods the victim's machine with a large number of requests in order to overload the system so that valid requests aren't fulfilled.

- **DDoS Attack:**

  DDOS Attack occurs when multiple systems, flood the bandwidth or resources of a victim.Such an attack is often the result of multiple compromised systems (for example, a botnet) flooding the targeted system with generating the huge network traffic.

- **Web Attack:**

  This attack types are coming out everyday, because individuals and organizations take security seriously now. We use the SQL Injection, which an attacker can create a string of SQL commands, and then use it to force the database to reply the information, Cross-Site Scripting (XSS) which is happening when developers don't test their code properly to find the possibility of script injection, and Brute Force over HTTP which can tries a list of passwords to find the administrator's password.

- **Infiltration Attack:**
  The infiltration of the network from inside is normally exploiting a vulnerable software such as Adobe Acrobat Reader. After successful exploitation, a backdoor will be executed on the victim's computer and can conduct different attacks on the victim's network such as IP sweep, full port scan and service enumerations using Nmap.

2.1.4 Dataset Analysis

80 traffic features have been extracted using CICFlowMeter.(which is a flow basedfeature extractor and can extract 80 features from a pcap file) The 80 extracted features are tested using the Random Forest Regressor in order to select the smallest feature set for each attack. Then, the selected feature's performance and accuracy are examined with sevon common Machine learning algorithms. The quality of the evaluated data-set is comparing with mistake and criticisms of other synthetically created datasets.

For the detection of **DDoS attacks** the best common features are Flow Inter arrival time related features such as Min, Mean, Max and also the Flow Duration.

For the detection of **Heart bleed attack** the best common features are the Flow Duration, Sub-flow Forwarding (Fwd) and Backwarding bytes along with packet length features like Standard Deviation of the backward packets and length of forward packets are most influential.

For the detction of **SSH-Patator** and **FTP-Patator** the best common features are the shown Initial window bytes along with some flags such as ACK, Psh and SYN.

For the detction of **Bot Attack** the best common features are Subflow F.Bytes, Total Len F.Packets and F.Packet Len Mean.

For the detction of **DoS GoldenEye** the best common features are B.Packet Len Std, Flow

IAT Min, Fwd IAT Min and Flow IAT Mean.

For the detection of **DoS Slowhttp** the best common features are Flow Duration, Active Min, Active Mean and Flow IAT Std.

For the detection of **Infiltration Attack** the best common features is forwarding sub flow-bytes and forwarding packets length along with the duration of the flow and Mean of active time.

Dataset related paper is ’Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization’[4]

2.2 BERT and other Neural Networks

BERT stands for Bidirectional Encoder Representations from Transformers, it is a transformer-based machine learning technique for natural language processing pre-training developed by Google. BERT was created and published in 2018 by Jacob Devlin and his colleagues from Google.

2.2.1 LSTMs Vs Transformers

A transformer is a deep learning model that adopts the mechanism of attention, differentially weighing the significance of each part of the input data. It is used primarily in the field of natural language processing and in computer vision.

2.2.2 LSTMs

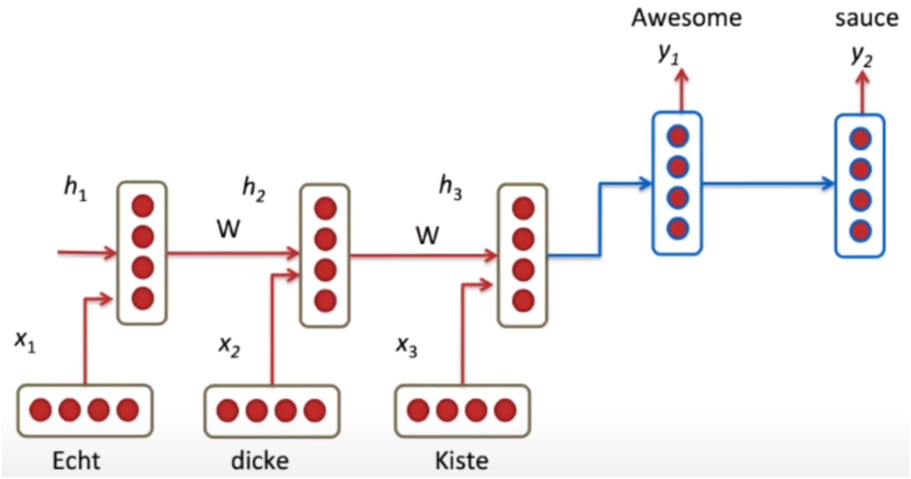


Figure 2.2: LSTMs Modal - Long Short Term Memory

Transformer Neural Network architecture was initially created to solve the problem of language translation, this was very well received until this point LSTM networks had been used to solve this problem, But they have some problem themselves:

- LSTM Networks are slow to train

- Words are passed and generated sequentially

It can take several number of time steps for the neural net to learn and its not really the best of capturing the true meaning of words. Yes, even bi-directional LSTMs because even here they are technically learning left to right and right to left context separately and the concatenating them so the true context is slightly lost.

2.2.3 Transformers

Transformer architecture addresses some of these concerns:

- Transformers are fast to train
- Learn words in both directions simultaneously

Let's see the transformer in action, say we want to train this architecture to convert English to French. The transformer consist of two key components; an encoder and a decoder. The encoder takes the English words simultaneously and it generates embeddings for every word simultaneously. These embeddings are vectors that encapsulate the meaning of the word, similar words have closer numbers in their vectors. The decode takes these embeddings from the encoder and previously generated words of translated French sentences anf then it uses them to generate the next french word and we keep translation one word at a time until the end of the sentence is reached.

What makes this conceptually so much appealing than so LSTM cell is that we can physically see a separation in tasks, the encoder learns What is English? What is context? where the decoder learns how do English words relate to French words? **Both of them even separately have some underlying understanding of language**, and its because of this understanding that we can pick apart this architecture and built systems that understand language.

We stack the decoders and we get the **GPT transformer architecture** conversely, if we stack just the encoders we get **BERT - Bidirectional Encoder Representations from Transformers**.

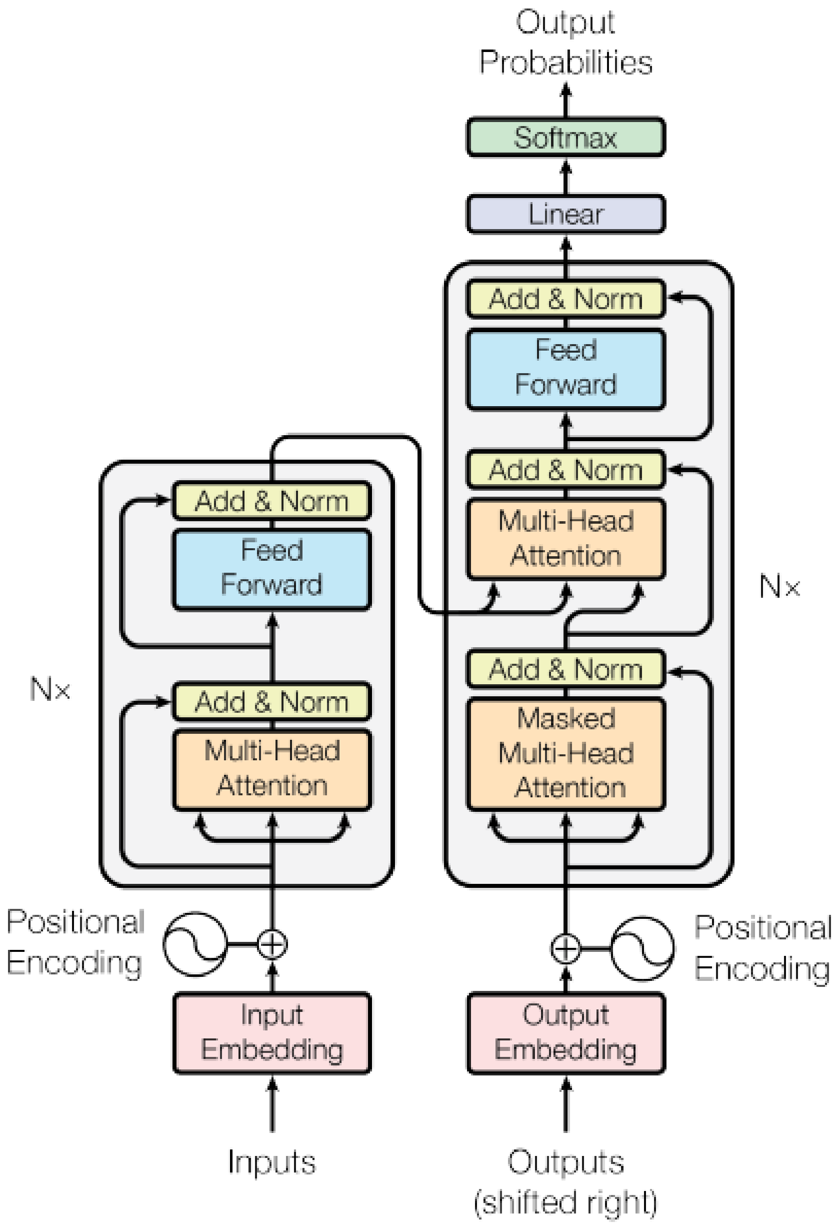


Figure 2.3: The Transformer - model architecture.

2.2.4 BERT

A bi-directional Encoders Representation from Transformers which is exactly what it is.

We can use BERT to learn:

- Neural Machine Translation

- Question Answering
- Sentiment Analysis
- Text summarization

Turns out all of the problems require the understanding of language so we can train BERT to understand language and then fine-tune BERT, depending on the problem we want to solve, as such train of BERT is done in two phases:

- Phase 1: Pretrain BERT to understand language
- Phase 2: Fine tune BERT to learn specific task

This figure show pretraining and fine-tuning phases of Deep Neural Mode, BERT

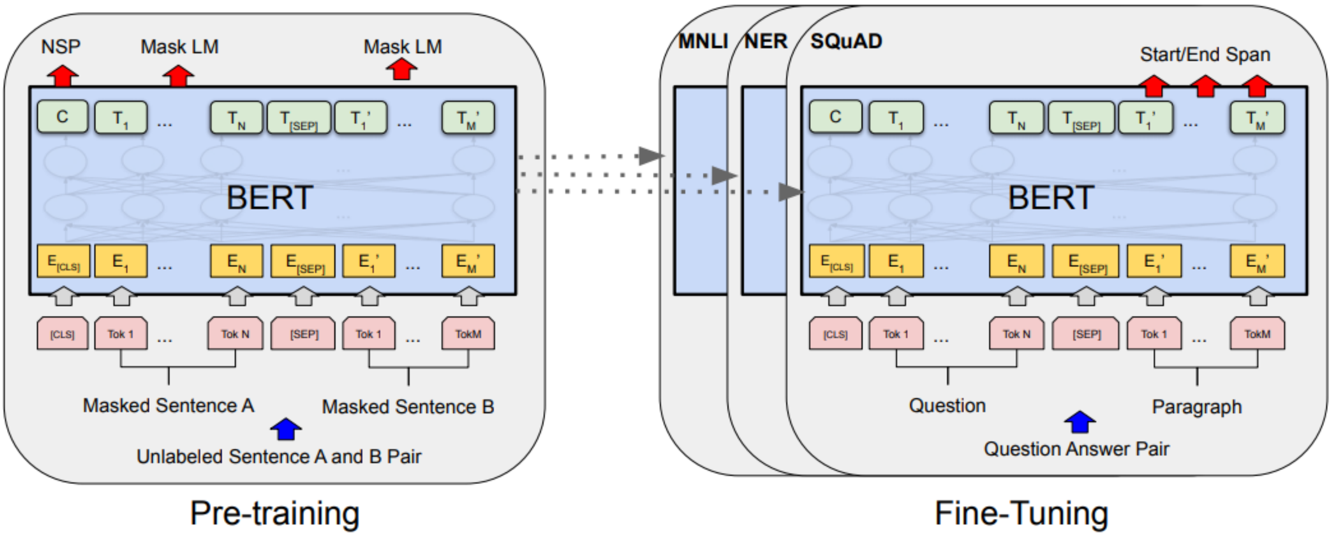


Figure 2.4: Overall pre-training and fine-tuning procedures for BERT.

Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

2.2.5 Pre-Training:

The goal of pre-training is to make BERT learn "What is language?" and "What is context?", BERT learns language by training on two unsupervised tasks simultaneously, they are:

- Masked Language Model (MLM)
- Next Sentence Prediction (NSP)

We will not dive deep in pretaining of BERT as we will pick the pretained BERT model and fine-tune it.

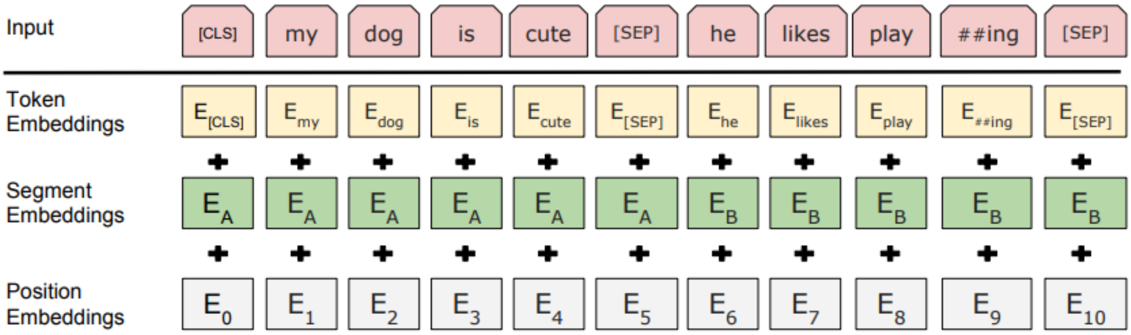

Input
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2.5: BERT input representation.

The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

The segment and position embeddings are required for temporal ordering since all these vectors are fed in simultaneously into BERT and language models need this ordering preserved.

2.2.6 Fine-tuning:

The goal is to make BERT learn "How to use language for specific task?". We can now further train BERT on very specific tasks, for example let's take Q and A. All we need to do is replace the fully connected output layers of the network , with a fresh set of output layers that cab basically output the answer to the question we want. Then we can perform supervised training using a Question answering dataset, it won't that long since its only the output parameters that are learned from scratch, the rest of the model parameters are just slightly fine-tuned. As a result training time is fast. We can do this for any NLP (Natural Language Processing) problem that is replace the output layers and then train with a specific dataset. (see Figure 2.5)

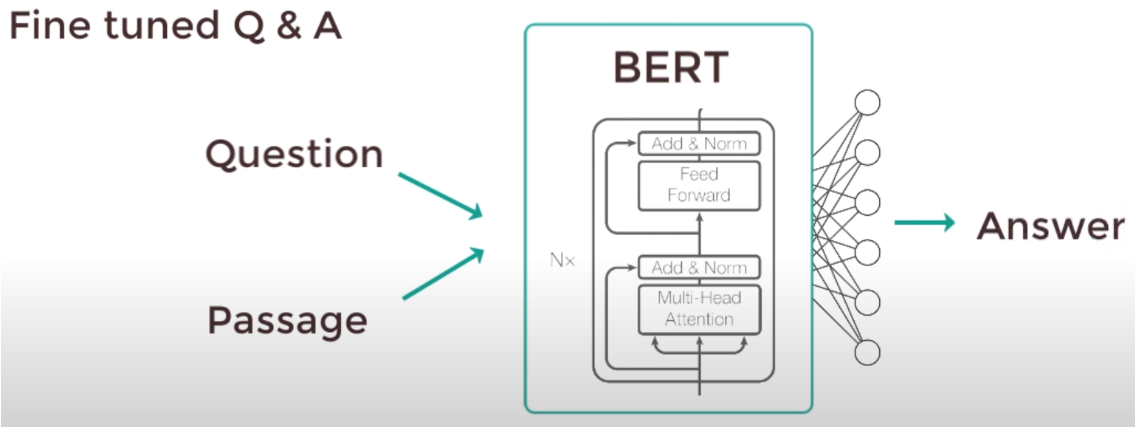


Figure 2.6: Fine-tuning BERT

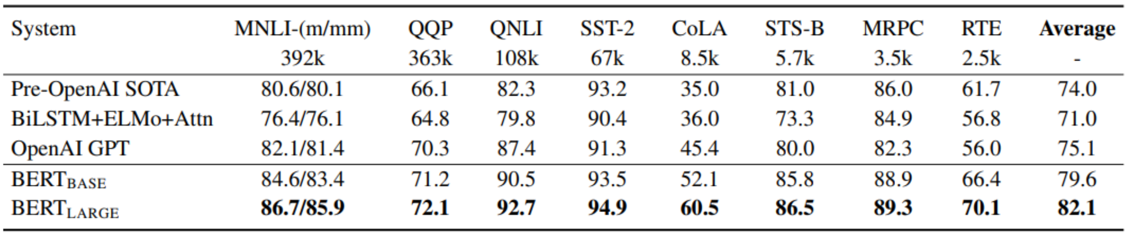
2.2.7 Summary

We pretrain BERT with Mask Language Modelling and Next Sentence Prediction, for every word we get the token embedding (see Figure 2.4) from the pre trained wordpiece embeddings, add the position and segment embeddings to account for the ordering of the inputs, these are then pass in two the BERT which under the hurd is the stack of transformer encoders and it output the bunch of word vectors for Mask Language Modelling and a binary value for a Next Sentence Prediction.

Word Vectors are then converted into distribution to train using **cross entropy loss**, once training is complete, BERT was some notion of language, its a language model. The next step is fine-tuning phase; where we perform a supervised training depending of the task we want to solve and this should happen fast.

2.2.8 BERT Performance

The BERT SQuAD that is Stanford question and answer model only takes about 30 minutes to fine tune from a language model for a 91 percent performance.



System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Figure 2.7: GLUE Test results

Scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.8 BERT and OpenAI GPT are singlemodel, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

2.2.9 Conclusion

Recent empirical improvements due to transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems. In particular, these results enable even low-resource tasks to benefit from deep unidirectional architectures. Our major contribution is further generalizing these findings to deep bidirectional architectures, allowing the same pre-trained model to successfully tackle a broad set of NLP tasks. In a little over a year, BERT has become a ubiquitous baseline in NLP engineering experiments and inspired numerous studies analyzing the model and proposing various improvements. The stream of papers seems to be accelerating rather than slowing down, and we hope that this survey will help the community to focus on the biggest unresolved questions.

2.3 Related Work

In the works of D. Yan, H. Takawale, R. Oak [6] has provided further proof to our hypothesis that Deep Neural Models can be used for a variety of Security tasks. Their aim was to use Android API Call Sequences, along with the a high level description of the activity a certain apk file has committed. The data-set was split into three non-duplicate datasets. 1) A huge large dataset which was used for Pre-training BERT Model; 2) A small labeled dataset that is used for training data for Malware Detection Task; and 3) A small dataset which can be used to test the model. As, the experiments are being carried out with highly imbalanced Data to mimic real-life-scenario; BERT seems to achieve a higher result than LSTM Networks, Achieving a F1 score of 0.882 on 1 percent Malware Data, where as LSTM was able to achieve a F1-score of 0.406 on 1 percent imbalanced Data.

The works of M.Guizani [8] has proposed a light-weight IOT Malware Detection Solution using Convolutional Neural Networks. The data was collected from a honey-pot and was labelled using the TotalVirus API, thus establishing the base-truth on which the experiments were carried out. The dataset was inclusive of very light malware which can only propagate through IoT devices which limited the variety of malicious binary files obtained. The Viruses collected were of Gafgyt or Linux.Mirai family. As for the goodware, samples of linux based software were used as a wide range of IoT's use Linux based software.

Other references are 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding'[5], 'Malware Detection by Analyzing Network Traffic with Neural Networks' [7], 'Attention Is All You Need'[2], 'A Lite BERT for Self-Supervised Learning of Language Representations'[9], 'Malware Detection by Analyzing Network Traffic with Neural Networks'[7], 'Malware detection on highly imbalanced data through sequence modeling'[8], 'Lightweight IoT malware detection solution using CNN classification'[1], and 'Sequence Aggregation Rules for Anomaly Detection in Computer Network Traffic'[3]

Chapter 3

Design and Specification

This chapter provides diagrams designed for this project as a part of documentation.

3.1 Use Case Scenario

- System
  - System Collect Data
  - Preprocess the Data
  - Testing Preprocessed-Data (on trained DL model)
  - Notify User/Admin (on presence of malicious activity)
- User
  - Quarantine/Ban Malicious Nodes.

3.2 Use Case Diagram

This is Use Case Diagram of Malicious Network Activity detection using a Deep Neural Network, BERT.

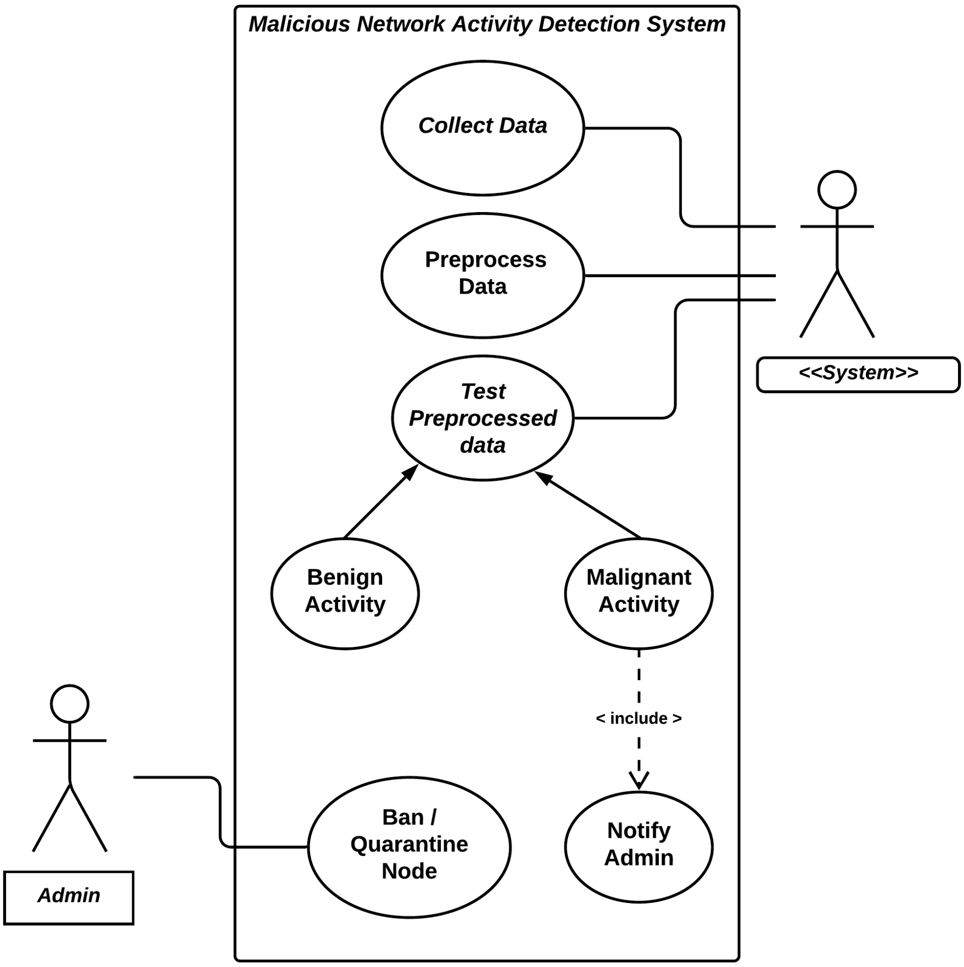


Figure 3.1: Use Case Diagram of Malicious Network Activity Detection

3.3 Activity Diagram

This is Activity Diagram of Malicious Network Activity detection using a Deep Neural Network, BERT

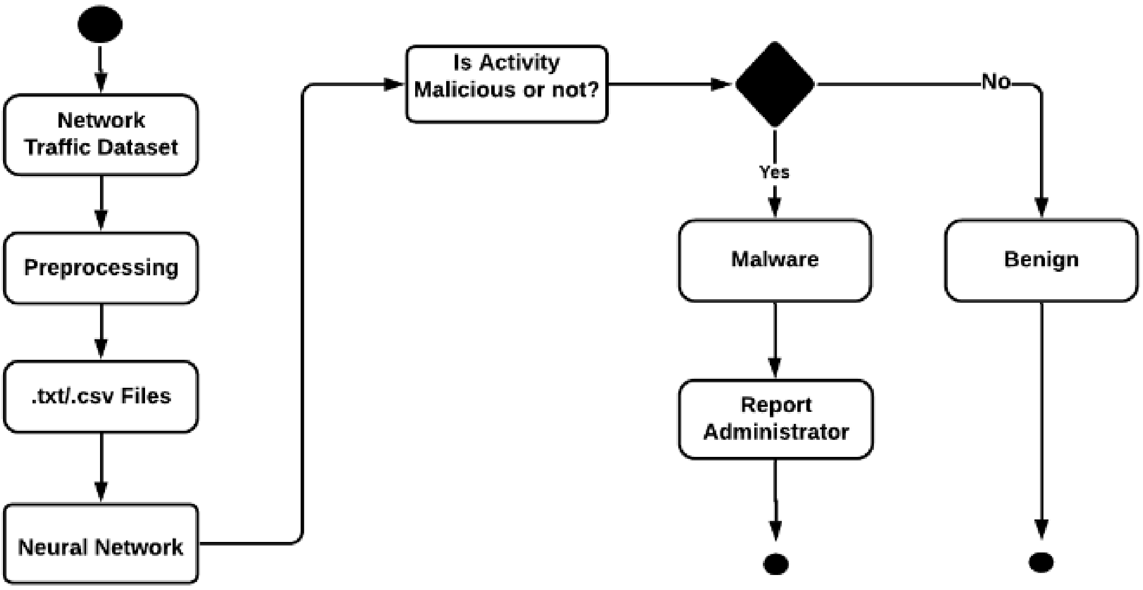


Figure 3.2: Activity Diagram of Malicious Network Activity Detection

Bibliography

[1] M. Guizani A. M. N. Zaza, S. K. Kharroub and K. Abualsaud. Lightweight iot malware detection solution using cnn classification. *arXiv [cs.CR]*, 2020.

[2] Niki Parmar Jakob Uszkoreit Llion Jones Aidan N. Gomez Ashish Vaswani, Noam Shazeer and Łukasz Kaiser. Attention is all you need. *arXiv:1706.03762v5 [cs.CL]*, 2017.

[3] and Shawn E. Davis Benjamin J. Radford, Bartley D. Richardson. Sequence aggregation rules for anomaly detection in computer network traffic. *arXiv:1805.03735v2 [cs.CR]]*, 2018.

[4] Arash Habibi Lashkari Iman Sharafaldin and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *4th International Conference on Information Systems Security and Privacy*, 2018.

[5] Kenton Lee Kristina Toutanova Jacob Devlin, Ming-Wei Chang. Bert: Pre-training of deep bidirectional transformers for language understanding. *Google AI Language*, 2019.

[6] Xuewen Zeng Luchao Han and Lei Song. A novel transfer learning based on albert for malicious network traffic classification. *ICIC International ©2020 ISSN 1349-4198*, 2020.

[7] Jiri Havelka Paul Prasse, Tomas Pevny. Malware detection by analyzing network traffic with neural networks. *IEEE Symposium on Security and Privacy workshop*, 2017.

[8] D. Yan H. Takawale R. Oak, M. Du and I. Amit. Malware detection on highly imbalanced data through sequence modeling. *in Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security - AIESec*, 2019.

[9] Sebastian Goodman Kevin Gimpel Piyush Sharma Zhenzhong Lan, Mingda Chen and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv:1909.11942v6 [cs.CL]*, 2020.