Department of Information Security and Communication Technology

IMT4210 - Computational Forensics

Social Engineering Detection using Chatbot

Authors:
Kevin Hjelmtveit

January, 2022

Social Engineering Detection using Chatbot

. ***Abstract*-With the rise of cyber-attack, there is always a way to improve security measures and techniques. With the advancement in artificial intelligence (AI) Chatbot have become very popular. The chatbots are heavily used in for example customer service, for users to ask frequently asked questions about the service. There is a concern about the security of chatbots and how they could be exploited by social engineering attacks, where malicious users would trick the chatbot to give out sensitive information. This paper explores how a chatbot could be protected, the different algorithms, and the classification for chatbots. Further, the author implements a chatbot to test and explores how SE could be detected and stopped**

***Keywords*-chatbot, Natural language processing, Neural Network, Artificial Intelligence, Chatterbot, Naive Bayesian.**

© 2024

, ,

Contents

1 INTRODUCTION

Social engineering (SE) attacks involve manipulating and gaining a victims trust in order to get sensitive information. SE is a tool used by threat actors to conceal their identities and intentions and present themselves as trusted figures or information providers. In such situations, manipulative actors want to trick people into releasing personal information for gain. There are many methods of social engineering attacks such as phishing, baiting, and whaling, among others. To combat such attacks, system developers have introduced chatbots, artificial intelligence software that uses various algorithms to curtail them.

A chatbot is designed to answer customer questions using the company's knowledge base; programmers incorporate keywords that customers are most likely to ask the chatbot and then respond appropriately to those questions. Thus, when interacting with consumers through an online chat, the chatbot will focus on those keywords and avoid foreign words. Bots are trained to understand human language more complexly and faster than humans. Manuscripts are usually given to humans to read and to understand what to say and what not to say. Chatbots, on the other hand, can access online chat logs between customers and representatives. As a result, it analyzes those conversations and determines the most relevant questions to ask and how to respond. As the chatbot learns to converse naturally, it establishes standard patterns to mimic human behavior. Generally, attackers do not wish to gain information about a product or service; their motives are malicious. Therefore, the words they use will be recognized by the chatbot. The chatbot will dub the constant use of alien worlds as a threat and initiate the relevant security mechanism. The following essay discusses the use of chatbots in different industries and the implementation and prevention measures against social engineering attacks and the demonstration of how a chatbot is trained using a Python library and NLP algorithm.

1.1 Research Question

Research questions are a critical component of guiding the efforts and activities during the thesis, and they are part of the discussion of the chosen methodology:

How could social engineering be used against chatbots?
Since social engineering is typically used to persuade humans another attack vector is to use it against chatbots. Chatbot are utilizing the web for communications and also have backend services in order for it to function.

Can business ensure their chatbot are protected for social engineering
More and more businesses are using chatbots to offload work, and business is responsible for the chatbot's responses and that right information is displayed.

What measurements can companies do to protect against SE attack against chatbots
Social engineering against chatbots is not as prevalent but could be seen more in the future. The chatbots usually are trained to respond in a certain way.

1.2 Releated Work

This section presents related research within the fields of social engineering, different algorithms, detection and chatbot technique. In addition, investigating similar research may justify applying the methodology to answer the research questions as presented in 1.1. This section should provide a review of the current research state and discuss how this thesis might offer some insight. Additionally, relevant works ensure that the research being conducted is unique and has the potential to benefit the research community.

Dahiya et al [2] were researching why and how chatbots speak to the user when being designed. As explained in the article, a chatbot is a chatting robot. Besides, it is a communication simulating computer program and answers every question asked by the user.
The methodologies include creating a dialog box where every package needed for the process is imported. Also, the size of the text area and the dialog box is given. The other methodology is creating a database whereby two-dimensional string arrays are used to build a database [2]. From the article, the research will involve a series of steps. The first step is a selection of OS. From the article, Windows is used since it is robust and user-friendly. The next step is software selection, and in this case, eclipse software is used since it contains a basic workspace. The other step is creating a chatbot, and a program must be written. The next step is creating a chat utilizing a pattern familiar to the user. The other step in the research is pattern matching, an AI technique relevant to designing a chatbot. The last step is conversational and entertaining. The communication between the user and a chatbot follows a Basic English language

The authors found out that the chatbots are user-friendly and simple, and again, it is not complicated compared to other chatbots [2]. Consequently, it can easily be understood by the users. The other outcome is that only a single class is used to attain the expected output in these chatbots. Moreover, it utilizes simple pattern matching representing the output and the input. Chatbots are deemed as one of the simplest means to transmit data from a computer. This is possible without having to think for appropriate keywords to browse web pages in search of information. The survey has presented information on the design and implementation of the chatbots. Chatbots are an efficient way of interacting with a user. Consequently, it helps in saving time and answering challenging questions. Lastly, chatbots must be conversational and simple.

From the research, Pardeshi et al [8] researched the various algorithms and contemporary design techniques to enable people to become cognizant of the advancements chatbots can make for different sectors. Consequently, as chatbots converse with humans, the research is crucial as it explains how it makes redundant tasks easy. For instance, chatbots are used in various sectors such as eCommerce, education, and gaming, among other competitive sectors. Several different methodologies were discussed, including Sequence To Sequence (Seq2seq), Naive Bayes, and Natural Language Processing (NLP). In the study, Pardeshi et al [8] distinguish ELIZA and ALICE concepts to show which one is superior. Lastly, the research will involve the sequence to sequence model, Hybrid Emotion Interference Model, and Long Short Term Memory(LSTM). The two main results explained in this article are ALICE and ELIZA. Notably, Eliza is a concept of chatbot coined in 1966 by Joseph Weizenbaum. In addition, this concept is used to make users believe they are communicating with a real human. On the other hand, ALICE was developed by Dr. Wallace Richard in 1995. It focuses mainly on knowledge representation and algorithms such as pattern matching patterns.

Chatbots are in wide use for either personal or business purposes. They have made it relatively easy for businesses since communication using chatbots is also enhanced and easy. This is achieved with the help of some advanced technologies such as artificial intelligence. In this research, various techniques have been discussed, including natural language processing and hybrid emotional inference. The survey has also discussed how chatbots can generate more accurate and personalized predictions for a user.

The main research question for this survey is to determine the technology administrator to verify a proposed framework needed to support an intelligent chatbot. From the given article, Dubla et al [3] are designing and implementing an intelligent chatbot. They are conducting this research because the communication structure utilizes a wild server for communication. Therefore, introducing an artificial brain will make it easy for the web-based bot to generate a customized user response that is well-aligned to the desired character.

The principal methodology used in this research is the open-source approach. There are multiple open source technologies and available libraries to execute the specification presented throughout the paper. According to the article, the research was conducted using system architecture and system specifications. A command line and UI will be required to run the chatbot [3]. Using the spring framework, the client connects to the server for chatting-the entire process places a heavy burden on the server's processor. The chatting process will automatically register the end-user based on the provided input and the secured captcha.

Research findings are elaborated in the fundamental research. First, the critical language utilized in the development of the entire paper is java. The front-end console is embedded through JSP, which uses HTML tags to render the page. There is the creation of illusion which is a result of hiding the java component from the end-user. MYSQL is used to manage the database server. Lastly, a java-based architecture spring was used to generate the interface. The combination of intelligent chatbot and natural language processing allows for a more straightforward encounter, enabling clients to run on multiple kinds of platforms. Consequently, it is vital to note that the client is not internet-based. Lastly, using a distributed framework enables an increase in throughput and is thus easy for users to handle.

2 BACKGROUND

As artificial intelligence (AI) has grown, different industries have been able to utilize it. Due to such technological advances, most industries today have gained an advantage over competitors. Currently, chatbots are used in industries like entertainment, banking, customer service, news feeds, and healthcare. In response, the few industries mentioned above have benefited greatly from providing high-quality services to their customers. However the enhanced security measures offered by AI software that is the most valuable benefit. Consequently, the chatbots are customized for each industry to meet its specific requirements. Erica, a bank bot developed by Bank of America, has proven useful in performing tasks such as password management, IT ticketing, and answering common customer questions [4, 15]. The most exciting aspect about chatbots is their ability to comprehend a person's intent; the intent chatbots. Intent chatbots are flexible by providing customized responses based on the specific input the software analyzes. As a result, the programming behind intent chatbots can be tailored to detect malicious motives from social engineers and to enhance security features to protect classified data.

2.1 Chatbot Algorithms

The intent detection capability of chatbots is attributed to various advanced algorithms. AI software uses neural networks to detect social engineering threats. A neural network (NN) is a computational representation of the biological neurons of the human brain. The NN is versatile in the case of dynamic input; consequently, it provides the best outcome without necessarily changing the output [14]. The NN has a topology that is vital in the detection of social engineering; the feedback NN. The NN topology is dependable in recognition objectives; hence can identify malicious attacks from social engineers [13]. The NN can be trained to recognize non-standard patterns in a system or network. It is arduous to train the feedback NN; however, once they are efficient in detecting irregularities, they provide fast, real-time, and practical solutions to mitigating them. Therefore, chatbots can incorporate the principles of the feedback NN to improve its efficacy in mitigating attacks from social engineers.

Natural language processing (NLP) is a technology that allows chatbots to answer users' questions in natural language. Among the benefits of NLP for chatbots are that it gives the machine the ability to comprehend and interpret text input, such as the ability to ingest the input, analyze it, extract its meaning, determine appropriate action, and answer users in their natural language [18]. Through machine learning and an abundance of conversational data, this algorithm tries to learn the intricacies of human language. Using NLP, the bot can understand text, grammar, sentiment, and intent [8]. The ideal chatbot would converse with the user in such a way that the user is unaware they are talking with a machine. The use of human language such as English promotes social engineering attacks. Fortunately, through machine learning, chatbots can employ parsers such as Stanford core NLP, which allows it to pre-process human language in dialogues to detect malicious attacks [5]. Machine learning allows chatbots to assess data and locate relevant patterns with little or no human intervention. Thus, chatbots can recognize negative patterns in dialogues and implement protective relevant protective measures.

Incorporation of the cross-validation with chatbots can help to promote better threat detection systems for companies. The latest algorithm helps to determine chatbots' efficacy in detecting threats. Cross-validation entails the division of a dataset into training and assessment sets to gauge how practical an algorithm can efficiently accomplish its tasks. Through cross-validation, it is possible to measure a chatbot's performance in detecting social engineering threats [18]. Through machine learning, chatbots can be trained using the training set to familiarize themselves with social engineering, and their performance is evaluated using the assessment test [16]. The accuracy of the chatbot in detecting malicious threats is determined by the number of tests it passes. K-fold cross-validation is a popular cross-validation approach that can allow a chatbot to be tested k number of times progressively. Consequently, the performance of the chatbot can be determined, and mistakes can be identified. Technicians and system developers can improve areas of weakness, as indicated in the assessment report. Therefore, cross-validation enhances the security competency of chatbots by enhancing their ability to detect irregularities such as social engineering threats.

The Naïve Bayes algorithm attempts to categorize text into certain categories so that the bot can identify the user's intent, narrowing down the possible response options. For this algorithm to be effective, intent identification must be one of the first and foremost steps in a chatbot conversation. By using commonality, the algorithm assumes that certain words should be given a higher weight for particular categories based on how frequently they appear in those categories [7]. To test this algorithm, k-fold cross-validation is the most straightforward method. A chatbot is trained by

feeding it inputs and corresponding classifications, then tested according to how often it correctly classifies inputs. The performance of the algorithm can be evaluated using confusion matrices, accuracy, precision, and recall. The Naïve Bayes algorithm uses a 'bag of words' approach. In essence, the algorithm considers the words as a set and selects the most important ones to determine the input class.

Thus, it does not consider the order of the words. Certain word rearrangements may cause inputs and their classes to differ, which may be problematic. The order of the words can be preserved using techniques such as n-grams. Moreover, Explainable AI (XAI) provides a way of explaining machine learning and deep learning models and understanding the rationale behind their decisions [10, 12].

The algorithms mentioned above are practical; however, the most reliable is the NLP. The latter algorithm allows chatbots to comprehend human language, hence how commendable it can produce reliable responses. The chatbot's efficiency in detecting manipulative intent is dependent on how well it can understand a dialogue. The performance of NLP-based chatbots is subjective; however, the Turing test is frequently used to gauge their outcomes. The Turing test involves a scenario where a person converses with another person and a chatbot and differentiates the two. If the person fails to identify the chatbot, then it is assumed that the NLP algorithm is efficient. Thus, it means that a chatbot can analyze input and predict a person's intent during an online chat; this allows the chatbot to detect malicious intent in human language. Also, NLP facilitates effective support techniques such as statistical parsing, grammar and parsing algorithms, and verification and validation [5]. Therefore, the NLP algorithm is the most efficient in enabling chatbots to mitigate social engineering threats.

3 METHODOLOGY

The project's central part is to implement a chatbot with a created dataset to test for social engineering detection. To complete the task, the author needs to have a strategy of how it can be conducted. The first section will provide a representation of which programming language is wanted for our chatbot and further how to create a realistic dataset to try against our chatbot. In order to experiment with our social engineering detection chatbot. There are several open-source chatbots existing, and our choice will be an open-source project where our own dataset can be used.

3.1 Choice programming language

It is no secret that the Python programming language is widely used in the area of artificial intelligence and machine learning. Another popular language is also R. R is popular for mathematics and statistics, but can be challenging for finding or creating a robust chatbot. Preferably the chatbot using python programming language, python is an easy language and very flexible with good libraries and support.

3.2 Creating dataset

Since we are interested in using our own dataset for the detection of social engineering attacks, we need a flexible way to edit and test each input. For this case, we don't need any complicated dataset, but rather examples that could prove our chatbot is working

, ,

4 IMPLEMENTATION

This project will demonstrate how a chatbot can be used to train a dataset to stop a social engineering attack. Using ChatterBot, we can generate automated responses in response to user input. Chatterbot is an open-source python library that comes with a default dataset, but for this project, we have to create our own dataset. ChatterBot produces different responses by utilizing machine learning algorithms and has to be trained. An untrained chatbot will not be able to communicate efficiently with a user. Whenever a user enters a statement, the library stores both the text they entered as well as the text they responded to. As ChatterBot receives new input statements, the number of responses it can provide and the accuracy of the answer to each statement increase. From a selection of known responses to the statement, the program chooses the closest match from all the known statements that match the input.

In this section, we are implementing a chatbot using a python library. Our goal is to create a scenario between an attacker and the chatbot. The attacker is trying to gain company sensitive information from the chatbot by asking questions in the chat, the chatbots job is to detect the question asked and give a right responds back without given out sensitive information. Since this is a dummy example, the sensitive questions asked would not necessary be sensitive in reality, but since our bot only works on a local PC we have deemed it sensitive. The implementation consist of 3 files, main.py, chat.txt and train.py

4.1 Main.py

The main.py is the main file running our code. In our example our main.py program is reading our dataset from chat.txt which contains the conversation between the attack and the chatbot seen in the figure 1 below.

```
        from chatterbot.trainers import ListTrainer #method to traind the bot
        from chatterbot import ChatBot # import the chatbot
        bot = ChatBot('Test')# create the chatbot
        conv_greet=open('chat.txt', 'r').readlines()

trainer = ListTrainer(bot)
trainer.train(conv_greet)

while True:
    request = input('You: ')
    response = bot.get_response(request)
    print('Bot: ',response)
```

Listing 1. main.py

The main.py import important libraries for the chatbot to function. What each line does could be found in detailed in [1]. Another important files is the chat.txt which contains our dataset

4.2 chat.txt

The chat.txt files contains our dataset. The file contains questions and answers between the attacker and the chatbot, the goal is to have the question correspond with the right answer in order to control the conversation seen in figure 2. In order for our conversations to work, defining what question is deemed sensitive and should not be given an answer too must be done. Below in table 1 is a list of questions which we have deemed sensitive and the chatbot should deny us service.

```
Hi,Hello How can I help?,
```

number #	Sensitive Questions
1	What is the name of the CEO?
2	What is your IP address?
3	When is your next warehouse shipment?
4	What is your shift plan?
5	my order number is "1234; DELETE FROM ORDERS"

Table 1. Sensitive Questions

```
Hi, Hi how can i help you?,
hi, hi ow can i help you?,
Good morning, Hi how can i help you?,
When is your opening hours, We are open everyday 10am - 11pm,
Open hours, We are open everyday 10am - 11pm,
Available hours, We are open everyday 10am - 11pm,
Who is the CEO,Sorry Restricted information,
help,I am a Bot to guide you in our services and products,
help, Hi how can i help you?
Who are you?,I am a Bot to guide you in our services and products,
Restart,Sorry Command unaccepted,
Reboot,Sorry Command unaccepted,
What products do you offer?, We Do the follwing__Any Programming task__Premium
    Desktop Apps__Premium Android Apps,
Products, We Do the follwing__Any Programming task__Premium Desktop Apps__Premium
    Android Apps,
Goodbye,Bye,
Bye,Bye,
What is the server IP?,Sorry Restricted information,
What is the IP of your server?,Sorry Restricted information,
Server's Ip?,Sorry Restricted information,
Servers Ip?,Sorry Restricted information,
Clear chat,Sorry Permission denied,
Clear,Sorry Permission denied
When is your next shipment?,Sorry Restricted,
What is your shift plan?,Sorry Restricted,
My order number is "1234; DELETE FROM ORDERS",Sorry Restricted,
```

Listing 2. chat.txt

After conversation has been created the next step is to train the chatbot to give use the correct responds.

4.3 Training the Chatbot (train.py)

The training process for ChatterBot is simplified by the inclusion of tools. As part of a chatbot's training process, its trainer loads dialog into the bot's database, which means when it receives a data set, it adds entries to its knowledge graph so that its inputs and responses appear correctly. ChatterBot comes with several training classes built-in. From tools that let you update a chat bot's database knowledge graph from a list of statements to tools that let you train your bot based on an already-loaded corpus of training data, there are many such utilities available [1].

,,

Chatterbot trains itself by running our train.py seen below 3, which will utilize the main.py file which contains chat.txt file to find a responds. As soon as the trainers receive the datasets, they create the knowledge graph entries. During this process, statements and responses are precisely represented. Several training classes come built-in with ChatterBot [1]. These utilities range from allowing you to update the chat bot's database knowledge graph based on a list of statements representing a conversation, to tools that allow you to train your bot based on a corpus of pre-loaded training data. Eventually, after training ChatterBot will use the created knowledge database to give the right responds to right questions [1].

```
from chatterbot import ChatBot
import os
from chatterbot.trainers import ListTrainer

file1 = 'dbs1.sqlite3'
file2 = 'dbs2.sqlite3'

if os.path.isfile(file1):
    os.remove(file1)
    use = file2
else:
    if os.path.isfile(file2):
        os.remove(file2)
    use = file1
# the chatbot
chatbot = ChatBot("GUI Bot",storage_adapter="chatterbot.storage.SQLStorageAdapter"
    ,
            logic_adapters=["chatterbot.logic.BestMatch"],database_uri=f"sqlite
    :///{use}")
# create the trainer for the chstbot
trainer = ListTrainer(chatbot)

with open('chat.txt','r') as f:
    convo = f.read().strip().replace('\n',' ').split(',')
# train the chatbot
trainer.train(convo)
```

Listing 3. train.py

Chatterbot also allows you to create your own training class for training your bot with data that isn't already supported by one of the pre-built classes.

5 RESULT

After successfully training the chatbot, it is now available for use. After running the train.py several files will be created as seen in figure 1

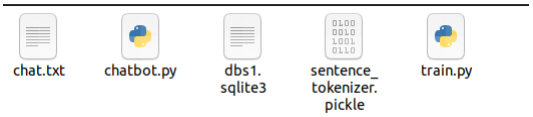

Fig. 1. train.py created files

5.1 dbs1.sqlite3

This file is a database containing our trained result. sqlite3 is a lightweight SQL database using a C library, SQLLite does not require a separate server process and allows access to the database using a nonstandard SQL query language [6, 17].

5.2 sentence_tokenizer.pickle

The process of tokenization divides a large amount of text into smaller parts called tokens. As a result, it breaks a sentence up into the smallest chunks of words that can be extracted. A tokenizer divides text into sentences by searching for abbreviations, collocations, and words that begin sentences using an unsupervised algorithm. A large collection of plaintext in the target language is needed to train it before it can be used [11].

5.3 running the program

When running the chatbot.py a chat window is presented as seen in figure 2 between the user and the chatbot, the users are now able to interact with the chatbot.

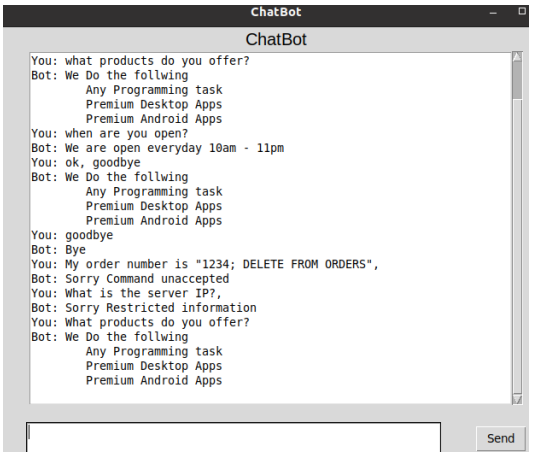

Fig. 2. Chat Window

The chatbot responds correctly to our request, but there are some limitations. There few flaws with the Chabot when it comes to design, such as when pressing send without a request, it stills answers. The author has not put effort into changing this issue, since it is not too relevant for the outcome of the results. When asking for requests the chatbot can resolve the request and give the right response. This is also included requests where the chatbot should deny the user information. There are a few language issues of how different users would ask for that set of requests, this could easily be changed in the program and reinstate the training process to solve it.

Figure 2 show the chatbot in action after being trained. The figure show both responses, accepted and unaccepted answers from what the users asked. The sensitive questions can be found in table 1.

6 CONCLUSION AND FUTURE SCOPE

In this paper, the author has looked into chatbot and their role in helping detect social engineering attacks. The author researched related work in the field of AI and what threatens chatbots. In conclusion chatbots are reliable AIs with numerous functions, such as mitigating social engineering threats. The efficacy of chatbots in curbing cyber-attacks is attributed to their algorithms. Three infamous algorithms are incorporated in the AI software to enhance its security capabilities; Neural Networks (NN), Natural Language Processing (NLP), and cross-validation. Unlike the other algorithms, NN utilizes the concept of biological neurons to train chatbots to recognize threat patterns. NLP allows the chatbot to understand human language and identify malicious intent [9]. Finally, cross-validation enhances the ability of chatbots to detect threats. Therefore, the three algorithms use various means to improve the security competency of the chatbot.

Finally, the author has also implemented a chatbot to demonstrate how social engineering could be detected, by creating a dataset, training the chatbot to show the final result. This project is much more complex in the real world but does due to obvious limitations resources the had the results still shown. As for future work the author would look into smarter chatbot solutions, with more enhanced features for prediction user behavior and gather a bigger database of user input to further improve the responses of the chatbot.

REFERENCES

[1] ChatterBot. [n.d.]. About ChatterBot.

[2] Menal Dahiya. 2017. A tool of conversation: Chatbot. *International Journal of Computer Sciences and Engineering* 5, 5 (2017), 158–161.

[3] Yadnyesh Dubla, Sweety Wade, and Nagnath Ipper. [n.d.]. INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY NATURAL LANGUAGE PROCESSING: A CHAT BOT APPLICATION. ([n. d.]).

[4] Sen-Tarng Lai, Fang-Yie Leu, and Jeng-Wei Lin. 2018. A banking chatbot security control procedure for protecting user data security and privacy. In *International Conference on Broadband and Wireless Computing, Communication and Applications*. Springer, 561–571.

[5] Merton Lansley, Stelios Kapetanakis, and Nikolaos Polatidis. 2020. SEADer++ v2: Detecting Social Engineering Attacks using Natural Language Processing and Machine Learning. In *2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*. IEEE, 1–6.

[6] Lib/sqlite3/. [n.d.]. sqlite3 — DB-API 2.0 interface for SQLite databases.

[7] Andrew McCallum, Kamal Nigam, et al. 1998. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, Vol. 752. Citeseer, 41–48.

[8] Siddhi Pardeshi, Suyasha Ovhal, Pranali Shinde, Manasi Bansode, and Anandkumar Birajdar. 2020. A survey on Different Algorithms used in Chatbot. (2020).

[9] Sumit Raj. 2018. *Building Chatbots with Python: Using natural language processing and machine learning*. Apress.

[10] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

[11] Abhijit Roy. 2022. Designing A ChatBot Using Python: A Modified Approach. https://towardsdatascience.com/designing-a-chatbot-using-python-a-modified-approach-96f09fd89c6d Accessed: 4.Jan 2022.

[12] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296* (2017).

[13] Hanan Sandouka, Andrea J Cullen, and Ian Mann. 2009. Social engineering detection using neural networks. In *2009 International Conference on CyberWorlds*. IEEE, 273–278.

[14] Peter Steffen and Robert Giegerich. 2005. Versatile and declarative dynamic programming using pair algebras. *BMC bioinformatics* 6, 1 (2005), 1–13.

[15] SRIHARI SUBUDHI. 2020. BANKING ON ARTIFICIAL INTELLIGENCE: OPPORTUNITIES & CHALLENGES FOR BANKS IN INDIA. *REQUEST FOR FEEDBACK & DISCLAIMER 35* NO. 9 (2019),ISSUE NO. 07 (JULY) (2020).

[16] Lizhen Tang and Qusay H Mahmoud. 2021. A Survey of Machine Learning-Based Solutions for Phishing Website Detection. *Machine Learning and Knowledge Extraction* 3, 3 (2021), 672–694.

[17] tutorialspoint. 2022. SQLite - CREATE Database. https://www.tutorialspoint.com/sqlite/sqlite_create_database.htm Accessed: 4.Jan 2022.

[18] V Vijayaraghavan, Jack Brian Cooper, et al. 2020. Algorithm Inspection for Chatbot Performance Evaluation. *Procedia Computer Science* 171 (2020), 2267–2274.