APPLIED RESEARCH LABORATORY FOR
INTELLIGENCE
AND SECURITY

# Pilot Projects for the ARLIS Intelligence and Security University Research Enterprise Academic Consortium

Contract #HQ003421F0013
Final Report

Submitted September 12, 2022 to the HBCU/MSI Program Office within the Office of the Under Secretary of Defense for Research and Engineering.

# Table of Contents

# Introduction

Erin Fitzgerald, Task Order Principal Investigator
Director, Intelligence & Security University Research Enterprise
Applied Research Laboratory for Intelligence and Security
University of Maryland
efitzgerald@arlis.umd.edu

The University of Maryland Applied Research Laboratory for Intelligence and Security (ARLIS) is a Department of Defense-designated University Affiliated Research Center (UARC) created in 2018 to be a long term and trusted research and development resource in topics of particular relevance to the Defense Security Enterprise (DSE) and the Intelligence Community (IC) at large. In 2020 ARLIS stood up the *Intelligence & Security University Research Enterprise (INSURE)* academic research consortium in further support of its mission as a UARC, inviting a targeted set of partner institutions to expand the pool of talent and technical resources available for supporting ARLIS core competencies and mission areas. INSURE is coordinating applied and use-inspired research activities for Intelligence and Security at member Universities, aligning these projects with specific DoD and IC program managers and activities to enhance impact, improve translation of products into operational use, and enhance the pipeline of students and faculty capable to work directly on technology problems for the national security community.

To facilitate INSURE program activities with the consortium's three Historically Black Colleges and Universities (HBCU) partners (Howard University, Morgan State University, and University of the District of Columbia), a set of five (5) pilot projects was proposed. Each project is aligned with ARLIS core competencies, a current ARLIS mission area, and current or pending DoD or IC stakeholder(s). The proposed period of performance for these tasks is September 2020 through August 2021, allowing for alignment with pending ARLIS task orders and academic calendars.

While all five pilot projects were primarily executed by research teams formed from the three HBCU partners, ARLIS technical leads served as advising partners, connecting the work and performers to other ARLIS efforts and government customers, ensuring the work stays true to the UARC mission, and providing overall coordination and oversight. ARLIS leads also provided communication with the DoD sponsor and oversight for the research. As the DoD-designated UARC and responsible contractor, ARLIS administrative staff worked to oversee compliance processes related to security (this work was deemed to be fundamental research, exempting it from many of the typical compliance requirements), human subject research approvals (as appropriate) and Organizational Conflict of Interest mitigation; managed the budget to ensure deliverables as scheduled in the statement of work; coordinated technical reviews; and generally worked to ensure that all INSURE research and technical support efforts were conducted with the highest security, ethical, and integrity standards and in full compliance of all UMD, government, and tasking activity-specific requirements.

1. **5G Technology Assessment**
   - Technical Lead: Kevin Kornegay, Morgan State University
   - Partner Michaela Amoo, Howard University
   - ARLIS Lead: Wayne Phoel, Research Engineer
2. **Machine Learning Experimentation**
   - Technical Lead: Paul Cotae, University of the District of Columbia
   - ARLIS Lead: Craig Lawrence, Mission Area Lead for AI, Autonomy, & Augmentation
3. **Cyber-Assessment of AI/ML Tools**
   - Technical Lead: Gloria Washington, Howard University
   - Partner Paul Wang, Morgan State University
   - ARLIS Lead: Craig Lawrence, Mission Area Lead for AAA
4. **AI/ML Systems Engineering Workbench**
   - Technical Lead: Kofi Nyarko, Morgan State University
   - Partner Michaela Amoo, Howard University

- ARLIS Lead: Craig Lawrence, Mission Area Lead for AAA

5. **ChatBot Testbed**
    - Technical Lead: Amit Arora, University of the District of Columbia
    - Partner Gloria Washington, Howard University
    - Partner Onyema Osuagwu, Morgan State University
    - ARLIS Leads: Michelle Morrison, Mission Area Lead for Language & Culture
      (later replaced by Anton Rytting and Valerie Novak)

**Project 1** falls under the Acquisition Security mission area of ARLIS. Concerns continue to mount about foreign influence in future communication network hardware, software, and operations, particularly as the use of such networks begins to pervade defense and other critical systems. This project sought to create a suite of testing tools for analysis of hardware performance, cyber-security, wireless security, and user access for emerging fifth generation (5G) mobile telecommunications systems.

**Projects 2**, **3**, and **4** fall under the ARLIS Artificial Intelligence (AI), Autonomy, and Augmentation (AAA) mission area of ARLIS. While AI results as reported by the research community are impressive, the operational benefits of AAA technologies within the Department of Defense (DoD) and Intelligence Community (IC) have yet to be fully realized. Processes, methodologies, and supporting tools and testbeds are needed for developing AAA-powered applications that can be trusted to perform the task that is intended, to do it in a way that fits naturally (and optimally) within an operator/analyst workflow, produces outcomes that the users trust and understand, and is hardened against malicious attacks. Projects 2, 3, and 4 develop underlying theory, assess existing AI/ML toolkits, and build a robust AI/ML Systems Engineering Workbench.

**Project 5** surveys the existing state-of-the-art and develops a testbed for exploring deployment and use of multi-lingual chat-bots for problems in influence, information operations, and insider threat, tied to a number of ARLIS mission areas including Cognitive Security, Modeling and Mitigating Insider Risk, and Language and Culture

# Pilot Project 1: 5G Testbed Development and Vulnerability Analysis

Kevin Kornegay[1*], Michel Kornegay[1], Sean Richardson[1], Cliston Cole[1], Michaela Amoo[2], Wayne Phoel[4]

**Program Objective**

5G testbed development and vulnerability analysis.

**Keywords**

Wireless sensor network; cybersecurity; vulnerability analysis

[1] Morgan State University
[2] Howard University
[4] Applied Research Laboratory for Intelligence and Security (ARLIS), University of Maryland, College Park, Maryland 20742
***Corresponding author:** kevin.kornegay@morgan.edu

## 1. Project Overview

A primary goal of this project is to create a suite of testing tools for analysis of hardware performance, cybersecurity, wireless security, and user access related to 5G technology.

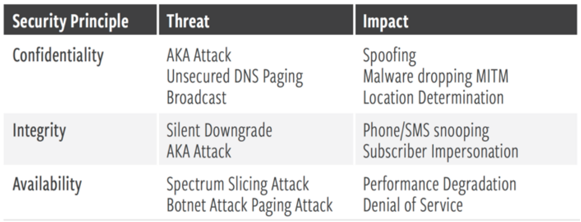
### 1.1 Overarching Goals of Pilot

One of the design goals of the 5th generation wireless network (5G) is to support the massive number of IoT devices. 5G's promise of enabling massive machine-type communication (mMTC) makes it ideal as a data backhaul for IoT traffic. The proposed research aims to design and implement a secure end-to-end wireless sensor network that will be used for security vulnerability analysis on 5G networks.

### 1.2 Lines of Effort

1) Design and implement wireless sensor networks (WSN) testbed using LoRa, Bluetooth low energy, Zigbee, and NB-IoT technologies.
2) Perform vulnerability analyses on the different sensor networks.
3) Integrate the wireless sensor network testbeds into a 5G network (under development).

Table 1



| Security Principle | Threat | Impact |
|---|---|---|
| Confidentiality | AKA Attack<br>Unsecured DNS Paging<br>Broadcast | Spoofing<br>Malware dropping MITM<br>Location Determination |
| Integrity | Silent Downgrade<br>AKA Attack | Phone/SMS snooping<br>Subscriber Impersonation |
| Availability | Spectrum Slicing Attack<br>Botnet Attack Paging Attack | Performance Degradation<br>Denial of Service |

### 1.3 Why It Matters

As the roll-out of 5G technology proliferates across civilian and military enterprises, future military campaigns in smart cities with billions of IoT devices pose a significant security threat. Understanding how adversaries can leverage security vulnerabilities in IoT devices and how cyber-attacks manifest across a 5G network is paramount.

## 2. Background and Related Work

5G is the 5th generation mobile network. It is a new global wireless standard after 1G, 2G, 3G, and 4G networks. 5G enables a new kind of network that is designed to connect virtually everyone and everything together, including machines, objects, and devices. 5G wireless technology is meant to deliver higher multi-Gbps peak data speeds, latency, more reliability, massive network capacity, increased availability, and a more consistent user experience. Higher performance and improved efficiency empower new user experiences and connect new industries.

Some notable vulnerabilities have come to light from the current published standards and research conducted in the 5G community. These vulnerabilities can be addressed in future standards releases, and some are expected to have mitigations in place when 5G standards are finalized. The vulnerabilities addressed in this paper are broken down into three sections: Confidentiality, Integrity, and Availability (CIA). The CIA triad, as it is known, is the cornerstone of security policy and dictates the most crucial components of security. Some of the following findings are holdovers from 4G LTE that has yet to be addressed in published standards. Several of them will likely be addressed in the future; however, some of the findings will be difficult to mitigate, and as of 3GPP Release 15, they are still vulnerable (R. Piqueras Jover and V. Marojevic, 2019). An overview of security threats and their impact can be found in Table 1.

# 3. Methods

## 3.1 Testbed Description

The deployed testbed comprises a mix of short-range (Bluetooth Low Energy and Zigbee) and long-range (LoRA and NBIoT) sensor network protocols. Data is captured from sensors, aggregated at a gateway, and sent to a cloud application using 5G as a backhaul. Bluetooth Low Energy (BLE) is a personal area network protocol with healthcare, fitness, and personal and home entertainment applications. Characteristics of BLE that make it ideal for these applications include low power requirements. Devices can operate for up to years on a button cell battery. BLE devices have small sizes and low costs. BLE is also compatible with the existing base of mobile phones, tablets and computers, and other devices. Like BLE, Zigbee uses small, low-powered devices for personal area networks. Zigbee use cases include home automation and medical IoT applications. BLE and Zigbee have an operating range of < 100m, but the content of Zigbee can be extended using mesh routing.

Low power wide area network (LPWAN) is a class of wireless communication optimized for long-range communication and low data rate. LPWAN devices are low-cost and can operate up to 10Km. The low data rate allows devices to operate at low power and run-on batteries for up to 10 years. These characteristics make it ideal for wireless sensor networks for longer-range communication. LoRA and LoRaWAN specification is a Low Power, Wide Area (LPWAN) protocol designed to wirelessly connect battery-operated devices to the internet ("What is LoRaWAN® Specification"). It is deployed in a star topology and supports bi-directional communication, end-to-end security, mobility, and localization services.

Narrowband-Internet of Things (NB-IoT) is a 3rd Generation Partnership Project (3GPP) standards-based LPWAN technology developed to enable a wide range of new IoT devices and services (3GPP, 2022). Benefits of NB-IoT include the low power consumption of user devices, system capacity, and spectrum efficiency, especially in deep coverage. Battery life of more than ten years can be realized. One significant advantage over competitors is that it benefits from all mobile network security and privacy features, such as user identity confidentiality, authentication, confidentiality, data integrity, and mobile equipment identification.

## 3.2 Hardware Implementation

### Bluetooth Low Energy

The Bluetooth protocol operates on a client-server model where the transmitter (sensor) acts as the server, and the receiver (gateway) acts as the client. The BLE sensor network comprises Arduino nano 33 BLE Sense microcontrollers with Bluetooth low-energy radio functioning as the servers. The Arduino nanos are loaded with an array of sensors. The servers send temperature and humidity sensor data to the BLE Client gateway. The Espressif ESP-32 development board supports Wi-Fi (802.11) and Bluetooth and is used as an MQTT gateway. ESP-32 functions as a BLE client and receives sensor data from sensor nodes. Data is then published to raspberry pi 4 running mosquito MQTT broker/Client software. Raspberry pi four also hosts Influx DB and Grafana visualization software. The data is stored in an influx database and visualized with Grafana. The data is also published to Amazon's AWS IoT cloud service (AWS, 2022) over a 5G network for external access.

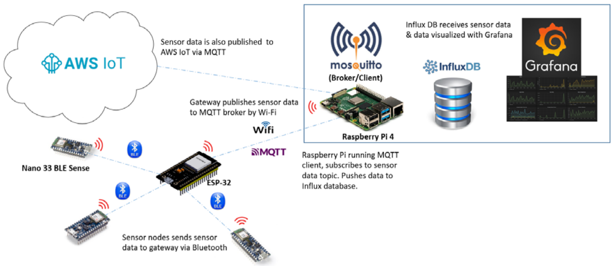

AWS IoT
Sensor data is also published to AWS IoT via MQTT
mosquitto
(Broker/Client)
Influx DB receives sensor data & data visualized with Grafana
InfluxDB
Grafana
Gateway publishes sensor data to MQTT broker by Wi-Fi
Wifi
Raspberry Pi 4
MQTT
Raspberry Pi running MQTT client, subscribes to sensor data topic. Pushes data to Influx database.
Nano 33 BLE Sense
ESP-32
Sensor nodes sends sensor data to gateway via Bluetooth

Figure 1: Bluetooth-LE Network

### Zigbee

The ZigBee sensor network is built from Digi XBee 3 Zigbee Mesh Kit. The sensor node comprises the Digi Xbee Grove development board used as the host microcontroller. Digi XBee 3 Zigbee 3 RF Module for ZigBee wireless connectivity and HTC1080 temperature and humidity sensor.

One microcontroller with an rf module functions as the coordinator and receives sensor data from the end nodes. The coordinator sends the data via a serial interface to a raspberry pi 4. Data is then published to raspberry pi 4 running mosquito MQTT broker/Client software. Raspberry pi four also hosts Influx DB and

Grafana visualization software. The data is stored in an influx database and visualized with Grafana. The data is also published to Amazon's AWS IoT cloud service over a 5G network for external access.

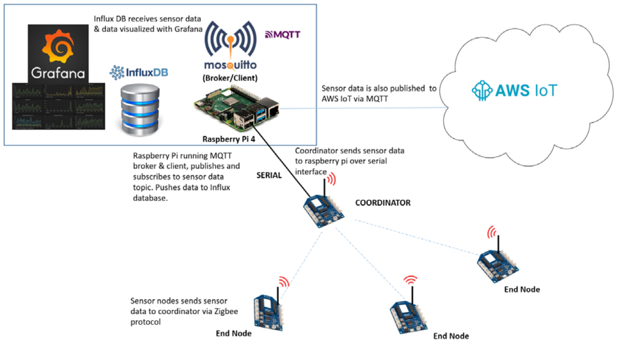

Influx DB receives sensor data & data visualized with Grafana
Grafana
InfluxDB
mosquitto
(Broker/Client)
MQTT
Raspberry Pi 4
Sensor data is also published to AWS IoT via MQTT
AWS IoT
Raspberry Pi running MQTT broker & client, publishes and subscribes to sensor data topic. Pushes data to Influx database.
SERIAL
Coordinator sends sensor data to raspberry pi over serial interface
COORDINATOR
End Node
Sensor nodes sends sensor data to coordinator via Zigbee protocol
End Node
End Node

**Figure 2: Zigbee Network**

## NBIoT

The NBIoT sensor network is built from Digi XBee® 3 Cellular Smart Modem, LTE-M/NB-IoT Development Kit. The sensor node comprises the Digi Xbee Grove development board used as the host microcontroller. Digi XBee 3 cellular LTE-M/NB-IoT Modem for rf connectivity and onboard HTC1080 temperature and humidity sensor. Micro python software interfaces sensors with the NB-IoT modem and cellular network. The sensor node connects to ATT cellular network, and data is published to Amazon's AWS IoT cloud service.

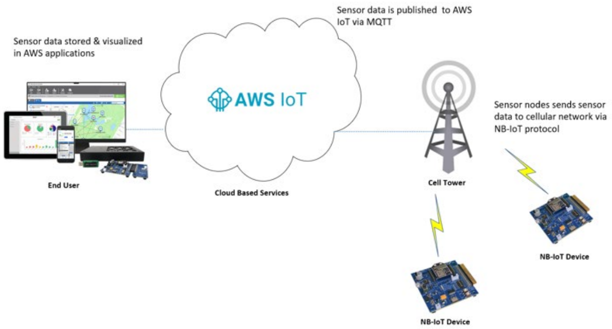

Sensor data is published to AWS IoT via MQTT
Sensor data stored & visualized in AWS applications
AWS IoT
End User
Cloud Based Services
Cell Tower
Sensor nodes sends sensor data to cellular network via NB-IoT protocol
NB-IoT Device
NB-IoT Device

**Figure 3: NB-IoT Network**

A vital element of this research is a working testbed to perform vulnerability analyses and propose countermeasures. Two LoRaWAN testbed were setup. The first used a cloud-based network and application server (lotiot.io). This setup had limited flexibility with no access to network server configuration. We later deployed a second testbed (chirpstack) based on an open-source Linux-based system. We describe both as follows:

## LoRA (loriot.io)

The testbed comprises (1) Seeduino LoRaWAN gateway module RHF0M301. Using a PRI 2 bridge RHF4T002 adapter, the gateway module was connected to a Raspberry Pi 3 to form the LoRA gateway. Microchip SAMR34 Xplained Pro end nodes with BME280 sensors were used to capture the environment's temperature, pressure, and relative humidity. This data was sent over the LoRa radio to the gateway and the loriot.io server (Loriot AG, 2022). The Microchip SAMR34 Xplained Pro microcontroller includes an RFM95 LoRa transceiver operating at 915 MHz. A 4.5 V AA battery pack powers the device. Temperature and humidity are read from a Bosch BME280 sensor via the I2C interface. The data is also published to Amazon's AWS IoT cloud service from the LoRA application server via MQTT.

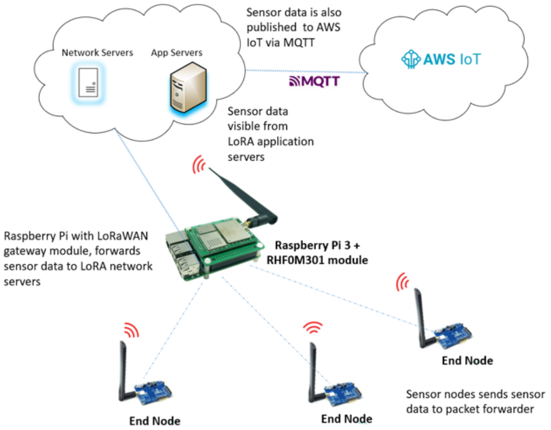

Network Servers
App Servers
Sensor data is also published to AWS IoT via MQTT
AWS IoT
MQTT
Sensor data visible from LoRA application servers
Raspberry Pi with LoRaWAN gateway module, forwards sensor data to LoRA network servers
Raspberry Pi 3 + RHF0M301 module
End Node
Sensor nodes sends sensor data to packet forwarder
End Node
End Node

***Figure 4:* LoRaWAN Network**

## LoRA (chirpstack)

The chirpstack LoRa testbed comprises (1) RAK2246 Pi HAT LPWAN Concentrator module, and the Raspberry Pi Zero W kit functions as the gateway. A Raspberry Pi 4 running chirpstack Network server and Application server implementation are used. Arduino MKR wan 1310 end nodes with onboard temperature and pressure sensors were used to capture data from the environment. This data was then sent over the LoRa radio to the gateway and the network servers on the raspberry pi. The data is also published to Amazon's AWS IoT cloud service from the LoRa application server via MQTT. The Arduino MKR wan a 1310 end node microcontroller with a Murata CMWX1ZZABZ LoRa® module (SX1276) transceiver

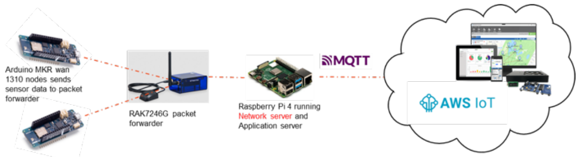
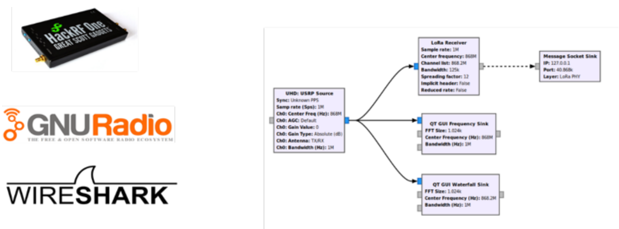

Arduino MKR wan 1310 nodes sends sensor data to packet forwarder
RAK7246G packet forwarder
Raspberry Pi 4 running Network server and Application server
MQTT
AWS IoT

**Figure 5: LoRaWan Testbed using chirpstack.**

operating at 915 MHz. Figure 5 shows a diagram of the chirpstack LoRa testbed.

## 3.3 Vulnerability Analysis

A significant element of this research is replicating a LoRaWAN network with jamming. Based on the work in (Perković et al., 2021), we constructed a LoRa jammer using inexpensive commercial off-the-shelf (COTS) products. We created a jammer using an Arduino Uno microcontroller, an Adafruit LoRa RFM95 module, and a 915MHz antenna. A picture of the constructed scanner/jammer is shown in figure 10. This device can function as both a scanner and a jammer. With channel activity detection (CAD) built into the Semtech chip in the RFM95 LoRa module, we can scan the channel for activity based on specified parameters (SF, frequency, BW, and CR). CAD is designed for LoRa modules for detecting the preamble of LoRa packets. CAD functions by having the LoRa device sample a signal of approximately one symbol length on a specific channel. It calculates the correlation between a given SF's captured and ideal LoRa symbols. Whenever a significant correlation is found between the captured sample and an ideal LoRa symbol, a CAD detection interrupt is activated, or otherwise, a CAD-done interrupt is registered (T. Perković, H. Rudeš, S. Damjanović, and A. Nakić, 2021).



UHD: USRP Source
Sync: Unknown PPS
Samp rate (Sps): 1M
Ch0: Center Freq (Hz): 868M
Ch0: AGC: Default
Ch0: Gain Value: 0
Ch0: Gain Type: Absolute (dB)
Ch0: Antenna: TX/RX
Ch0: Bandwidth (Hz): 1M

LoRa Receiver
Sample rate: 1M
Center frequency: 868M
Channel list: 868.2M
Bandwidth: 125k
Spreading factor: 12
Implicit header: False
Reduced rate: False

Message Socket Sink
IP: 127.0.0.1
Port: 40.868k
Layer: LoRa PHY

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 868M
Bandwidth (Hz): 1M

QT GUI Waterfall Sink
FFT Size: 1.024k
Center Frequency (Hz): 868.2M
Bandwidth (Hz): 1M

https://github.com/rpp0/gr-lora

**Figure 6: Vulnerability analysis tools.**

The RadioLib library[1] was used to load code onto the microcontroller, and the scanner/jammer was positioned close to the gateway to start the jamming process. The library provides a simple interface for implementing activity detection (preamble detection) and traffic disruption on the observed channel. The jammer listened to the CAD mechanism implemented on a set channel with a specified SF at which a legitimate LoRaWAN packet is sent and then caused a collision on that channel by sending a packet received by the gateway with higher signal strength. The code can function as a reactive jammer that scans the channel for specific parameters and then sends a jamming signal once a CAD interrupt is triggered. Figure 8 shows a diagram of the flowchart of the reactive jammer operation.

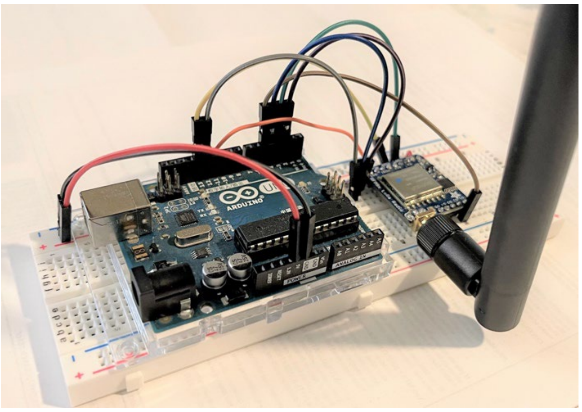


**Figure 7: LoRaWan jammer.**

### Replay Attack

Using GNU radio with HackRF One software-defined radio (SDR), we captured and recorded all messages from a Node. From the collected messages, the DevNonce, message counter information, and the DevAddr (device address) was retrieved from the open text header. We stored all these messages and information in a database. We forced a reboot of the node. Using HackRF One SDR, we transmitted a message with a frame counter higher than 1. The legitimate node could not transmit until the frame counter exceeded the frame count of the adversarial transmitted message, and denial of service occurred.

### Eavesdropping Attack

Using GNU radio-companion, HackrfOne software-defined radio, and Wireshark, we performed a successful eavesdropping attack on the LoRa network. We used a scapy[2] packet manipulation tool to decode the data packets to read the unencrypted parts of the LoRa message frame.

[1] https://github.com/jgromes/LoRaLib

[2] https://scapy.net/

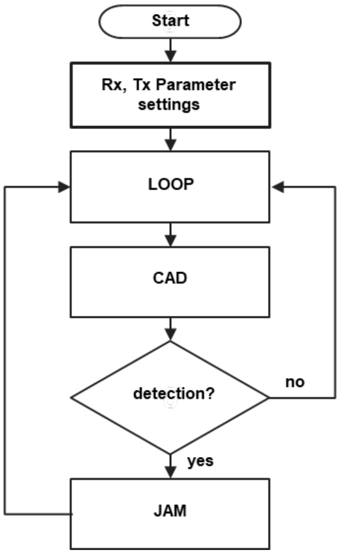


**Figure 8: Jammer flowchart.**

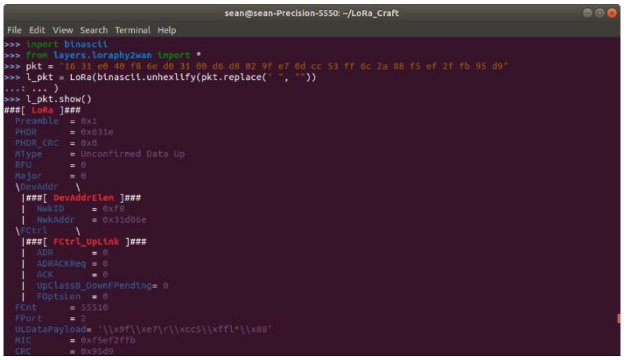
# 4. Results and Discussion

## 4.1 Preliminary Results

- LoRaWAN wireless sensor network testbeds were developed and deployed.
- Vulnerability analysis has started on the LoRa WSN testbed. Vulnerabilities tested to date include successful eavesdropping, jamming, and replay attacks.

### Eavesdropping Attack

Using GNU radio-companion, HackrfOne software-defined radio, and Wireshark, we performed a successful eavesdropping attack on the LoRa network.

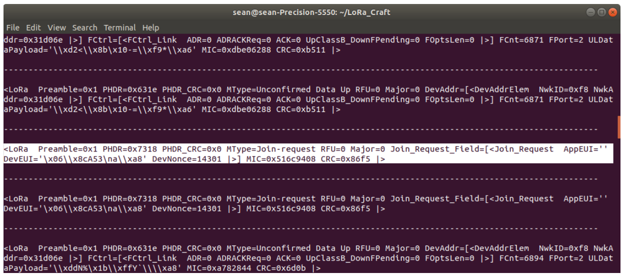

sean@sean-Precision-5550: ~/LoRa_Craft
File Edit View Search Terminal Help

```
>>> import binascii
>>> from layers.loraphy2wan import *
>>> pkt = "16 31 e0 40 f8 6e d0 31 00 d6 d8 02 9f e7 0d cc 53 ff 6c 2a 88 f5 ef 2f fb 95 d9"
>>> l_pkt = LoRa(binascii.unhexlify(pkt.replace(" ", ""))
...: ... )
>>> l_pkt.show()
###[ LoRa ]###
  Preamble  = 0x1
  PHDR      = 0x631e
  PHDR_CRC  = 0x0
  MType     = Unconfirmed Data Up
  RFU       = 0
  Major     = 0
  \DevAddr   \
   |###[ DevAddrElem ]###
   |  NwkID     = 0xf8
   |  NwkAddr   = 0x31d06e
  \FCtrl     \
   |###[ FCtrl_UpLink ]###
   |  ADR       = 0
   |  ADRACKReq = 0
   |  ACK       = 0
   |  UpClassB_DownFPending= 0
   |  FOptsLen  = 0
  FCnt      = 55510
  FPort     = 2
  ULDataPayload= '\\x9f\\xe7\r\\xccS\\xffl*\\x88'
  MIC       = 0xf5ef2ffb
  CRC       = 0x95d9
```

**Figure 9: LoRa packet capture.**

### Replay Attack

Using GNU radio-companion, HackrfOne software-defined radio, and Wireshark, we performed a successful replay attack on the LoRa network.

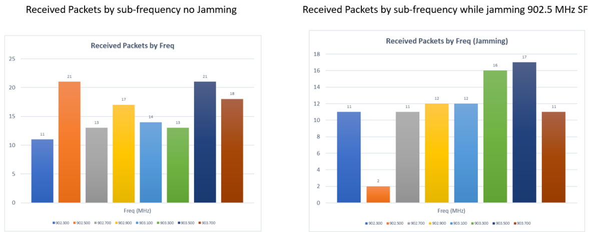

sean@sean-Precision-5550: ~/LoRa_Craft
File Edit View Search Terminal Help

```
ddr=0x31d06e |>] FCtrl=[<FCtrl_Link  ADR=0 ADRACKReq=0 ACK=0 UpClassB_DownFPending=0 FOptsLen=0 |>] FCnt=6871 FPort=2 ULDat
aPayload='\\xd2<\\x8b\x10-=\\xf9*\\xa6' MIC=0xdbe06288 CRC=0xb511 |>

-------------------------------------------------------------------------------------------------------------------------

<LoRa  Preamble=0x1 PHDR=0x631e PHDR_CRC=0x0 MType=Unconfirmed Data Up RFU=0 Major=0 DevAddr=[<DevAddrElem  NwkID=0xf8 NwkA
ddr=0x31d06e |>] FCtrl=[<FCtrl_Link  ADR=0 ADRACKReq=0 ACK=0 UpClassB_DownFPending=0 FOptsLen=0 |>] FCnt=6871 FPort=2 ULDat
aPayload='\\xd2<\\x8b\x10-=\\xf9*\\xa6' MIC=0xdbe06288 CRC=0xb511 |>

-------------------------------------------------------------------------------------------------------------------------

<LoRa  Preamble=0x1 PHDR=0x7318 PHDR_CRC=0x0 MType=Join-request RFU=0 Major=0 Join_Request_Field=[<Join_Request  AppEUI=''
DevEUI='\x06\\x8cA53\na\\xa8' DevNonce=14301 |>] MIC=0x516c9408 CRC=0x86f5 |>

-------------------------------------------------------------------------------------------------------------------------

<LoRa  Preamble=0x1 PHDR=0x7318 PHDR_CRC=0x0 MType=Join-request RFU=0 Major=0 Join_Request_Field=[<Join_Request  AppEUI=''
DevEUI='\x06\\x8cA53\na\\xa8' DevNonce=14301 |>] MIC=0x516c9408 CRC=0x86f5 |>

-------------------------------------------------------------------------------------------------------------------------

<LoRa  Preamble=0x1 PHDR=0x631e PHDR_CRC=0x0 MType=Unconfirmed Data Up RFU=0 Major=0 DevAddr=[<DevAddrElem  NwkID=0xf8 NwkA
ddr=0x31d06e |>] FCtrl=[<FCtrl_Link  ADR=0 ADRACKReq=0 ACK=0 UpClassB_DownFPending=0 FOptsLen=0 |>] FCnt=6894 FPort=2 ULDat
aPayload='\\xddN%\x1b\\xffY`\\\\xa8' MIC=0xa782844 CRC=0x6d0b |>
```

**Figure 10: LoRa joins request packet.**

### Jamming Attack

Using a low-cost jammer set to the specified frequency and spreading factor, jam signal when preamble detected.


Received Packets by sub-frequency no Jamming

Received Packets by sub-frequency while jamming 902.5 MHz SF 10.

**Figure 11: LoRa jamming attack.**

We must secure IoT devices and communication systems to connect IoT systems. We integrated the IoT testbed with AWS. We also performed a comprehensive security vulnerability analysis on LoRaWAN to determine its viability as an IoT enabler. We built a low-cost scanner/jammer to simulate real-world jamming attack scenarios. We have proposed a machine learning-based LoRa jammer detection system and countermeasure to prevent denial of service (DOS) and other attacks in LoRaWAN networks. The proposed countermeasure is to detect jamming and to instruct the LoRaWAN network server to automatically switch to a new subset of frequencies once jamming is detected. Preliminary experimental results show that machine learning can detect and evade harmful jamming signals in LoRa-based wireless sensor networks. This research will illustrate that cloud-based services like AWS can be an effective tool in implementing security measures on resource constraints IoT devices.

## 5. Future Directions

This research focuses on the security of LPWAN technologies, emphasizing LoRaWAN wireless sensor networks (WSN). To date, we performed vulnerability analysis, and AWS implementation of jamming countermeasure is ongoing.

Future work will integrate IoT testbeds into a pending 5G testbed. The current IoT testbeds use ethernet traffic backhaul from the gateway to the network servers. We will eventually replace the backhaul with a future 5G testbed. 5G mobile networks are also an IoT enabler, and there are still open security questions regarding how 5G will integrate with IoT. The 5G network testbed setup is ongoing, with tentative completion in 2022. Once we complete the 5G network testbed setup, we will integrate the 5G testbed into the existing IoT networks. We will then recreate various attacks on the IoT network to see how the attacks are manifested in the 5G network.

The 5G network by itself may have undiscovered security weaknesses. Questions about user data security and securing control plane information are open. We intend to test the 5G network by designing and implementing various attacks. One attack that we plan to implement is a botnet attack on the 5G testbed.

# Pilot Project 2: Machine Learning Experimentation

Dr. Paul Cotae[3*], Dr. Anteneh Girma[3], with ARLIS Lead Craig Lawrence[4]

**Program Objective**

To find a data preparation model and model configuration that gives good or great performance.

**Keywords**

Machine Learning; Cybersecurity; Planning algorithm

[3] The University of the District of Columbia
[4] Applied Research Laboratory for Intelligence and Security (ARLIS), University of Maryland, College Park, Maryland 20742
***Corresponding author:** pcotae@udc.edu

## 1. Project Overview

The full research Project 2 “Machine Learning Experimentation” was executed at the University of the District of Columbia under the supervision of PI Dr. Paul Cotae and Co-PI Dr. Anteneh Girma supervising six PhD students.

Our Machine Learning (ML) experimentation project objectives include discovering a simple approach to plan and manage the ML and Artificial Intelligence (AI) experiments. Students and faculty would be exposed to advanced custom AI/ML hardware, experiment design, data preparation and curation**.** To find a data preparation model and model configuration that gives good or great performance for ML, AI, and Security University Research, we carefully planned and managed the order and type of experiments that we run.

### 1.1 Overarching Goals of Pilot

The overarching goals of this ARLIS pilot project are:

- Use the descriptive statistics and plots for exploratory data analysis, fit probability distributions to data, generate random numbers for Monte Carlo (MC) simulations, and perform hypothesis tests.
- Develop ML predictive models with classification algorithms, including Decision Trees, K-Nearest Neighbors, and naive Bayes.
- Test regression and classification algorithms to draw inferences from data and build predictive ML models.
- Consider cooperative multi agent decision making in centralized and decentralized environments and to deliver anytime planning algorithms with a cost factor by using Max-Plus algorithms.

### 1.2 Lines of Effort

Our results are given in sub-projects 1-3 for ML and sub-projects 4-6 for AI and Decision Making (DM) processes in cyber security research.

2.1. Ransomware Attack Detection on the Internet of Things Using Machine Learning Algorithm
2.2. Malware Detection Model on a Cyber-Physical System Using Artificial Intelligence and Machine Learning
2.3. Detecting DDoS Attacks in Software-Defined Networks Through AI Machine Learning
2.4. Scalable Real-Time Multiagent Decision Making Algorithm with Cost
2.5. A Scalable Real-Time Distributed Multiagent Decision Making Algorithm
2.6. A Hybrid Cost Collaborative Multiagent Decision Making Algorithm with Factored Value Max Plus

### 1.3 Why It Matters

The Intelligence and Security Communities should care about data because that is our most valuable asset. Before building and deploying ML models, we need to make sure that an end-to-end data management system is in place. The success and sustainability of ML initiatives are mainly dependent on how well we govern, monitor, and manage data on a real-time basis. We have learned how the development of ML algorithms helps to address different process automations, business predictions, and product innovations. Many data researchers, however, now find themselves unable to effectively analyze a greater amount of data from a rising number of sources in a secure manner.

The application of ML and AI is a great advantage to enhance various domains of cyber security to provide analysis-based approaches for the detections of catastrophic cyber-attacks and countermeasures. While researching the advancement of cybersecurity solutions, many researchers are exploiting different tools and mechanisms of Machine Learning and Artificial Intelligence (ML/AI) to keep the integrity, confidentiality, and availability of information assets and to effectively respond to sophisticated cyber-attacks.

We are observing the rise of artificial intelligence applications not only creating AI products for automating tasks, but also for creating products that enhance traditional cybersecurity methods. With the exponential growth of AI, we see an increasing number of cybersecurity tasks being automated. Some widely used ML/AI tools to enhance cyber security include early attack detection, threat identification, network monitoring, and enhancing threat alert systems, to name a few.

# 2. Background and Related Work

Machine Learning (ML) is an application of the Artificial Intelligence (AI) that enables a system to learn from data rather than through explicit programming. ML is a technique that lets the computer “learn” with provided data without thoroughly and explicitly programming every problem. ML enables classification and prediction based on known data and achieves high accuracy and reliability, which makes it more likely to help cyber network administrators get a correct decision about threats. In recent years, machine learning has been applied into intelligence and security to improve their performance and offer satisfactory cybersecurity results.

ML is sub-categorized into three types: supervised learning (where we have a data set which acts as a teacher and its role is to train the model or the machine), unsupervised learning (where the model learns through observation and finds structures in the data), and reinforcement learning (where the agent can interact with the environment and discover the best outcome).

An ML approach usually consists of two phases: training and testing. Often, through experiments, the following steps are performed: a) Identify class attributes (features) and classes from training data, b) Identify a subset of the attributes necessary for classification (i.e., dimensionality reduction), c) Learn the model using training data, d) Use the trained model to classify the unknown data.

A machine learning experiment pipeline can be broken down into three major steps. These include data collection, data modeling (where modeling refers to using a ML algorithm to find insights within our collected data), and deployment. In our research project regarding ML, we designed and conducted ML experiments in Python programming language and Matlab software platforms. We considered the supervised and unsupervised machine learning algorithms, including support vector machines (SVMs), boosted and bagged decision trees, k-nearest neighbor, k-means, k-medoids, hierarchical clustering, Gaussian mixture models, and hidden Markov models.

### Cyber-Physical Systems

LOE 2.2 deals with cyber-physical systems, building on a larger body of work. For intelligence and security communities, Cyber-physical systems (CPSs) is an umbrella term that includes systems such as robotics, machine automation, industrial control systems (ICSs), process control systems, supervisory control and data acquisition (SCADA) systems, the Internet of Things (IoT) (Houbing Song, Security and Privacy in Cyber-Physical Systems, 2019), and the industrial Internet of Things (Chauduri 2019). Cyber-physical systems (CPS) are used on a large scale in the modern industrial system (Sharmeen, Huda, & Abawajy, 2019).

Nowadays, the use of CPS, which is the interconnection of multiple devices, and connections of devices to humans, are showing rapid growth (Zewdie & Girma, 2020). That said, CPS devices are not typically updated or given security patches at the frequency of other cyber systems, leaving a security vulnerability as well as a research opportunity (Cotae and all 2021, 2022).

# 3. Methods and Results for Subprojects

A team at UDC consisting of six PhD students supervised by the PI Dr. Paul Cotae and Co-Pi Dr. Anteneh Girma has been working on the following six research subprojects.

## 3.1 Ransomware Attack Detection on the Internet of Things Using Machine Learning

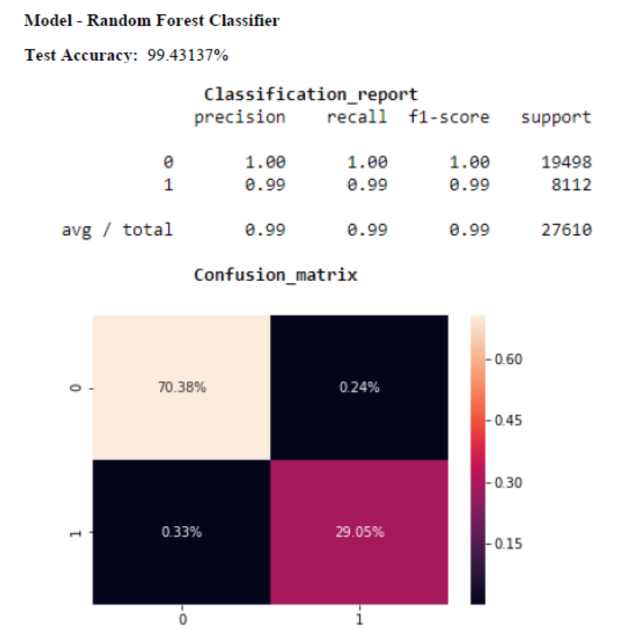
LEAD IN TEXT

### Approach

Once the data set was identified, the following steps were executed accordingly.

#### *Data Preprocessing*

The loaded data set is preprocessed and grouped by name and other attributes. In this stage, data preprocessing includes data cleaning, instance selection,



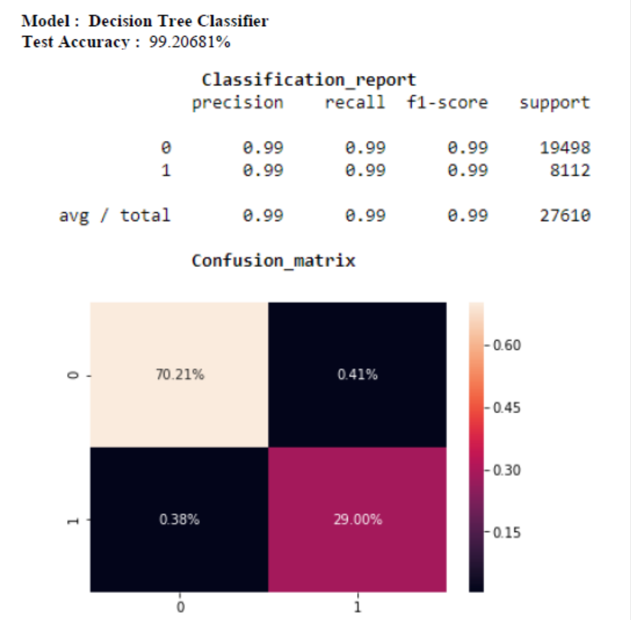
Model - Random Forest Classifier

Test Accuracy: 99.43137%

```
            Classification_report
             precision    recall  f1-score   support

          0       1.00      1.00      1.00     19498
          1       0.99      0.99      0.99      8112

avg / total       0.99      0.99      0.99     27610
```

Confusion_matrix

Figure 2 Random Forest Classifier



Model : Decision Tree Classifier
Test Accuracy : 99.20681%

```
            Classification_report
             precision    recall  f1-score   support

          0       0.99      0.99      0.99     19498
          1       0.99      0.99      0.99      8112

avg / total       0.99      0.99      0.99     27610
```

Confusion_matrix

Figure 1.. Decision Tree Classifier

normalization, feature extraction, and selection. Moreover, the product of data preprocessing is the final training set. The imported Python libraries organize the data set in a format that the machine learning model can explore.

### *Data Normalization*

The loaded data set needs to normalize for exploratory analysis to be conducted. Analysis and derived insight from the analyst can be done. In this regard, we cleaned data by removing duplicates, marking missing values, and imputing missing values.

### *Data Labeling*

The input variable for the feature extraction is split into the x-axis and the y-axis. The split sets are trained around the x- and y-axis for testing.

### *Feature Extraction*

The data set is grouped according to a classifier called legitimate. From the classified data, legitimate data is denoted by 1. There were 41,323 in the count. Malicious data denoted as 0, and there were 96,724 in the count. The derived vectors, legitimate and malicious, form the basis of feature extraction model on the data set.

## AI and ML Models:

### *Random Forest Algorithm*

This research primarily uses Random Forests (RF) machine learning algorithms to get good predictive performance, low overfitting, and easy interpretability. Random forest is an applicable model for binary, categorical, and numerical features. It improves bagging because it decorrelates the trees with the introduction of splitting on a random subset of the feature. It means that at each split of the tree, the model considers only a small subset of features rather than all the model's features. From the given data set of available features n, a subset of m features (m=square root of n) is selected at random. While we are using RF, it requires little pre-processing, and the data does not need to be rescaled or transformed.

The model is great with high-dimensional data since we work with subsets of data, split into smaller data groups based on the data features that are named a decision tree. Fig. 1 shows how we used a small set of data that only has data points under one label. Reducing the number of features and creating new features in a data set from the existing ones is known as Feature Extraction. The new reduced set of features summarizes most of the information contained in the original set of features.

### *Decision Tree Algorithm*

In this research, we also consider Decision Tree (DT) to benefit from its advantages. DT lays out the problem so that all options can be challenged and allow us to analyze the possible consequences of a decision fully. Moreover it provides a framework to quantify the values of outcomes and the probabilities of achieving them. A decision tree classifier can use different feature subsets and decision rules at different stages of classification. (Results in Fig. 2.)

### Performance Analysis

To evaluate and validate the performance of the proposed ransomware detection classifier, i.e., the Decision Tree model, we used different parameters such as accuracy, sensitivity selectivity, and specificity from the Decision Tree model derived. False-positive and false-negative rates are derived too.

**Table 1 Data set Features**

| | |
|---|---|
| RUSER | Real user id. The textual or decimal representation |
| PPID | Select parent process by process id |
| UID | User id number |
| PID | User id |
| PGRP | Process group id |
| %CPU | CPU utilization of the process in ##.# format |
| %MEM | Memory usage of the process |
| VSZ | Total virtual memory size in bytes |
| TIME | Total accumulated CPU utilization time for the process |
| SIZE | Memory size in kilobytes |
| Legitimate | Labeled as 1 if the process is legitimate. Labeled as 0 if the process is malware |

### Data set and Data Description

We got our testing data from Kaggle. The total number of rows in a data set was 138;047: 41323 were legitimate files, and the remaining 96724 were malware. The derived vectors, legitimate and malicious, form the basis of the feature extraction model on the data set.

This Data set is extracted from the proc virtual file system. The data.csv file contains the process samples from the Ubuntu Desktop environment. Thus, data has the following features given in Table 1.

**Implementation hardware and software**

To implement AI/ML models, we used Jupyter Notebooks and Google Colab. Both tools (applications) helped us with comparison and measuring efficiency of the result. Google Colab provides high GPUs to run our code better. To manage our data set, we used both FileZilla Server and FileZilla Host as needed. Both tools were downloaded and configured on our machine. A FileZilla Server is a server that supports FTP and FTP over TLS which provides secure encrypted connections to the server.

**Research outcomes**

The focus of Subproject 2.1 is to prove that AI/ML should become conventional in cybersecurity applications to protect against cyber-attacks. Therefore, our research focused on a seasonal malware attack called ransomware. Such malware ransomware variants have increased from time to time, and the counter-defense mechanism for such an attack has been very critical. Thus, the outcomes of this research show the importance and need of integrating AI/ML and information security to achieve the best cybersecurity practices to secure IoT systems and organizational data.

Our classification and detection AI/ML Random Forest model and Decision Tree model showed better accuracy. In both models, this research achieved more than 99% detection accuracy. We believe this result contributes a great deal to academia. Our research results analysis and discussions will raise more academic awareness of the need for machine learning techniques to be integrated into IoT-connected device networks.

Finally, we would like to point out that our proposed AI/ML solution is limited to reporting ransomware incidents to the system user and does not automatically counter the ransomware attacks. It lays the foundation for further academic research and industrial innovation to stop and counter detected ransomware attacks effectively and automatically.

### Deliverables for this subproject:

Based on our experiment, we delivered the following model accuracy that achieved maximum detection accuracy results as given in Table 2.

**Table 2 Maximum Detection Accuracy Results**

| Model | Accuracy |
|---|---|
| Random Forest (RF) Classifier | 0.994314 |
| Decision Tree (DT) Classifier | 0.992068 |

This research was presented at the 24th International Conference (HCI2022) on June 26, 2022 and published before the end of August 2022.

## 3.2 Malware Detection Model on a Cyber-Physical System Using Artificial Intelligence and Machine Learning

**Project Deliverable**

One of the significant deliverables of our subproject is the accuracy result**.** In our machine learning experimentation, with a given data set of 2,426,573 rows and 25 columns, we got the following accuracy result. Thus, Random Forest, Decision Tree, Ada boost, and Extra Tree classifier have 100% accuracy. The K-Neighbors classifier has 99.64% accuracy, the SGD Classifier has 91.24% accuracy, and the Gaussian NB Classifier has 78.87% accuracy.

This research paper has been accepted for presentation and publication at the 62nd IACIS Annual International Conference in Las Vegas, Nevada in October 2022.

### Approach

Once the data set has been identified the following steps have been executed accordingly.

### *Data Preprocessing*

In this stage, data preprocessing includes data cleaning, instance selection, normalization, feature extraction, and selection.

### *Preprocessing Steps*

BoTNeTIoT data set contains nine IoT device traffics sniffed using Wireshark in a local network before applying the preprocessing steps. The following data preprocessing steps have been done on the captioned data set:

1. Adding feature names: the feature names are added to each data set column.
2. Dropping of six empty column features: unnecessary columns are deleted from the data set.
3. Replacing empty ports with 0: two NA values were filled with 0.
4. Dropping non-required features: features not required for the ML model are removed from the data set.
5. Encoding objects to categorical values: three features, "HpHp_L0.1_pcc," "Attack," and "label," are factorized to encode the object as a categorical variable.

The preprocessing steps implemented on the BoTNeTIoT data set reduced the number of features to twenty-five. Such preprocessed data is then used to train the ML model.

**Data Classification and prediction process:** See Fig.3.

### *Random Forest Classifier*

A random forest algorithm consists of many decision trees. The ‘forest’ generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms. This classifier is more accurate than the decision tree algorithm. Moreover, it provides an effective way of handling missing data and can produce a reasonable prediction without hyper-parameter tuning. Finally, in every random forest tree, a subset of features is selected randomly at the node’s splitting point.

### *Decision Tree Classifier*

The main advantage of the decision tree classifier is its ability to use different feature subsets and decision rules at different stages of classification.

### *K-Neighbors Classifier*

We used KNN for multiclass classification. Therefore, if the data consists of more than two labels or if you are required to classify the data in more than two categories, then KNN can be a suitable algorithm.

### *AdaBoost Classifier*

AdaBoost is best used to boost the performance of decision trees on binary classification problems. AdaBoost can be used to boost the performance of any machine learning algorithm. These are models that achieve accuracy just above random chance on a classification problem. The most suited and common algorithm used with AdaBoost are decision trees with one level. Because these trees are so short and only contain one decision for classification, they are often called decision stumps. Each instance in the training data set is weighted.

The initial weight is set to:

$$\text{Weight }(xi) = 1/n$$

**Equation 1**

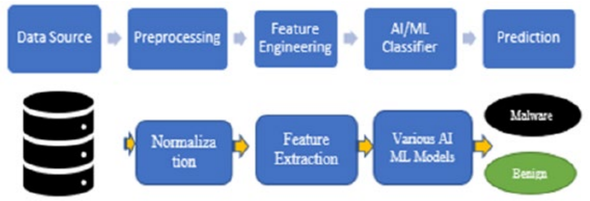


**Figure 3. Data Classification and Prediction process**

where $x_i$ is the $i$th training instance and $n$ is the number of training instances.

### *Stochastic Gradient Descent (SGD) Classifier*

In Subproject 2.2, we used Stochastic Gradient Descent because as we can read from the previous text, SGD allows minibatch (online/out-of-core) learning. Therefore, it makes sense to use SGD for large-scale problems where it's efficient. Additionally, SVM or logistic regression will not work if you cannot keep the record in RAM. On the other hand, SGD classifier continues to work.

### *Extra Trees Classifier*

This Classifier implements a meta estimator that fits several randomized decision trees (a.k.a. extra-trees) on various sub-samples of the data set and uses averaging to improve predictive accuracy and control over-fitting.

### *Gaussian Naïve Bayes*

A Gaussian Naive Bayes algorithm is a special type of Naïve Bayes algorithm. It is specifically used when the features have continuous values. It is also assumed that all the features are following a Gaussian distribution i.e., normal distribution. Bayes' theorem is based on conditional probability. The conditional probability helps us calculate the probability that something will happen, given that something else has already happened.

$$P(A/B) = P(B/A) * P(A) / P(B)$$

**Equation 2**

## Data set and Data Description

For this project, we also used Kaggle's data set. This data set was originally in a CSV format, "Comma-Separated List" for tabular data. The malware data set, which is the most recent data set, contains nine IoT devices traffic sniffed using Wireshark in a local network using a central switch. It includes two Botnet attacks (Mirai and Gafgyt). The data set contains twenty-three statistically engineered features extracted from the .pcap files. Seven statistical measures were computed (mean, variance, count, magnitude, radius, covariance, correlation coefficient) over the time window of 10 seconds with a delay factor equal to 0.1 (Kaggle). For our research, we used the IoT data set for Intrusion Detection Systems (IDS) from Kaggle.

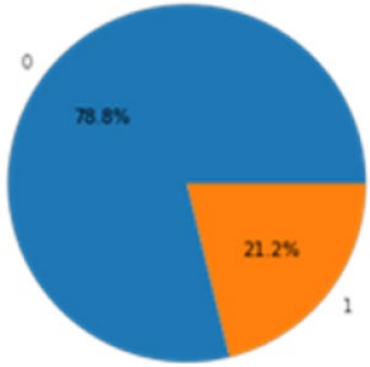


**Figure 4. Classification of malware and benign files**

BoTNeTIoT-L01 is a data set integrated with all the IoT device's data files from the detection of IoT botnet attacks (BoTNeTIoT) data set. This latest version reduced the redundancy of the original data set by choosing the features in the 10 second time window only. In the data set class label, 0 stands for attacks, and 1 stand for normal samples.

**Implementation tools- Software and Hardware used**

To implement AI/ML models, we used Jupyter Notebooks and Google Colab. Both tools (applications) helped us in comparison and measuring efficiency of the result. Google Colab provides high GPUs to run our code better. To manage our data set, we used both FileZilla Server and FileZilla Host as needed. Both tools have been downloaded and configured on our machine. FileZilla Server is a server that supports FTP and FTP over TLS which provides secure encrypted connections to the server.

Alternatively, we used Win Zip to compress our data files since it takes up less storage space and can be transferred to other comp uters quicker than uncompressed files.

## Experiment results

### *Classification of malware and benign files*

Fig.4 shows the classification of malware and benign files in the data after preprocessing. Thus, 78.8% of the files are malware (0) attacks, and the remaining 21.2% of the data set is benign (1).

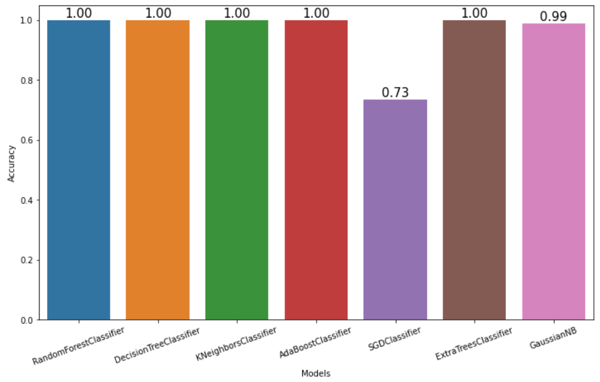


Figure 7. Random Forest Classifier

### Model Accuracy result

In our experiment with a given data set of 2,426,573 rows and 25 columns, Random Forest, Decision tree, Ada boost and Extra Tree classifier had 100% accuracy. The K-Neighbors Classifier had 99.64% accuracy, the SGD Classifier had 91.24% accuracy, and the Gaussian NB Classifier had 78.87% accuracy.

### Research Outcomes

This research examined various approaches and proposed a framework that can use alternative machine learning algorithms to successfully differentiate between malware files. It can also clean files in cyber-physical systems (CPS) while minimizing the number of false positives. Unlike the conventional solution, we examined various AI/ML models to detect and classify an attack, whether it was malware or benign on CPS. The models are Random Forest, Decision tree, K-nearest, Ada boost, SGD, Extra Tree, and Gaussian NB Classifier. Based on the captioned candidate algorithm, our experiments depict that Random Forest, Decision Tree, Ada Boost, and Extra Tree Classifier achieved 100% accuracy in detecting most attacks with Zero False-positive and False-negative rates (Figure 5).

Finally, experimenting using the above machine learning models, we proposed a captioned candidate malware detection framework in cyber-physical systems. Moreover, in subproject 2.2, we investigated the security challenges and issues of state-of-the-art cyber-physical systems. In this regard, we hope that these CPS security challenges and issues, precisely the detection and classification of attacks with AI/ML models, bring enough motivation for future discussions and interests in academic research work.

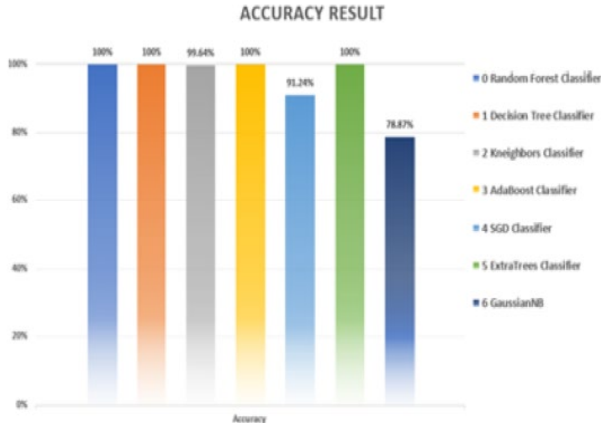


Figure 5. Model accuracy results

## 3.3 Detecting DDoS Attacks in Software-Defined Networks Through AI Machine Learning

LEAD IN TEXT

### Random Forest Classifier

A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms. This classifier is more accurate than the decision tree algorithm. Fig.7 shows our experiment result for the RF classifier.

### Decision Tree Classifier

In our project, we used a Decision Tree (DT) classifier because DTs clearly lay out the problem so that all options can be challenged, allowing us to analyze fully the possible co nsequences of a decision. In addition, it provides a framework to quantify the values of outcomes and the probabilities of achieving them. Fig8 shows our DDoS flooding attack detection and classification results.

### K-Neighbors Classifier

Our data consists of more than two labels, and KNN was the best method to classify the data.  Fig.9 shows our DDoS flooding attack detection and classification results with the KNN Classification algorithm.

### AdaBoost Classifier

AdaBoost was used to boost the performance of decision trees on binary classification problems. AdaBoost can be used to boost the performance of any machine learning algorithm. These are models that achieve accuracy just above random chance in a classification problem. The most suited and common algorithm used with AdaBoost is a decision tree with one level. Because these trees are so short and only contain one decision for classification, they are often called decision stumps. Results are presented in Fig.10.

### *Stochastic Gradient Descent (SGD) Classifier*

In this research, we used stochastic gradient descent because as we can read from the previous text, SGD allows minibatch (online/out-of-core) learning. Therefore, it makes sense to use SGD for large-scale problems where it's very efficient. Additionally, SVM or logistic regression will not work if you cannot keep the record in RAM. However, SGD classifier continues to work. Results are given in Fig.11.

### *Extra Trees Classifier*

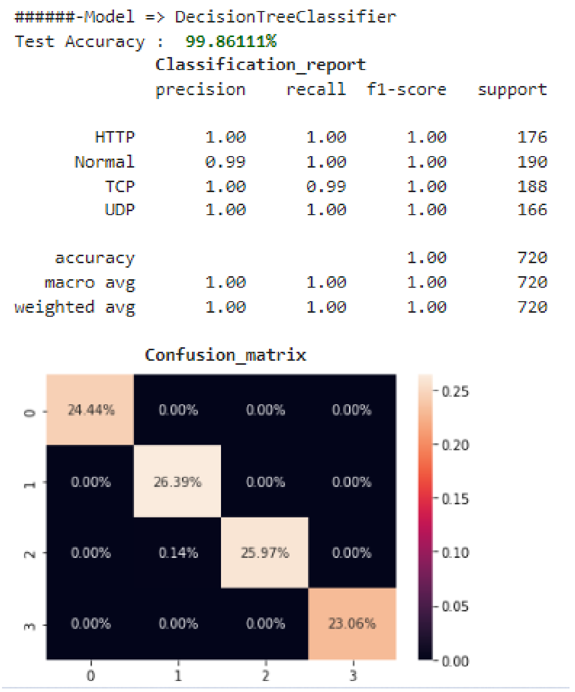
In this research, we used an extra-trees classifier. This classifier implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the d ata set and uses averaging to improve the predictive accuracy and control over-fitting. Results are given in Fig.12.

### *Gaussian Naïve Bayes*

A Gaussian Naive Bayes algorithm (Fig.13) is a special type of NB algorithm. It is specifically used when the features have continuous values. It is also assumed that all the features are following a Gaussian distribution i.e., normal distribution. Bayes' theorem is based on conditional probability. The conditional probability helps us calculate the probability that something will happen, given that something else has already happened (see Equation 1).

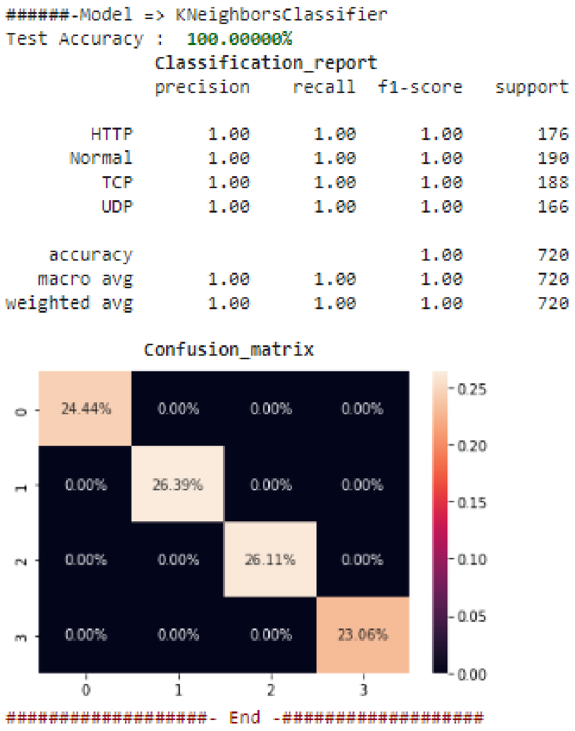
### Data sets and Data Description

The data set for SDN performance metrics (*throughput*, *jitter*, and *response time*) was collected when the SDN was operating normally and when it was subjected to TCP, UDP, and HTTP DDoS flooding attacks. The name of the data set is SDN data set for DDoS flooding attack detection which is adapted from the Kaggle repository server. This data set of SDN performance metrics was used for classification of DDoS flooding attacks. Please see Fig.14.



```
######-Model => DecisionTreeClassifier
Test Accuracy :  99.86111%
                Classification_report
                precision    recall  f1-score   support

        HTTP       1.00      1.00      1.00       176
      Normal       0.99      1.00      1.00       190
         TCP       1.00      0.99      1.00       188
         UDP       1.00      1.00      1.00       166

    accuracy                           1.00       720
   macro avg       1.00      1.00      1.00       720
weighted avg       1.00      1.00      1.00       720
```

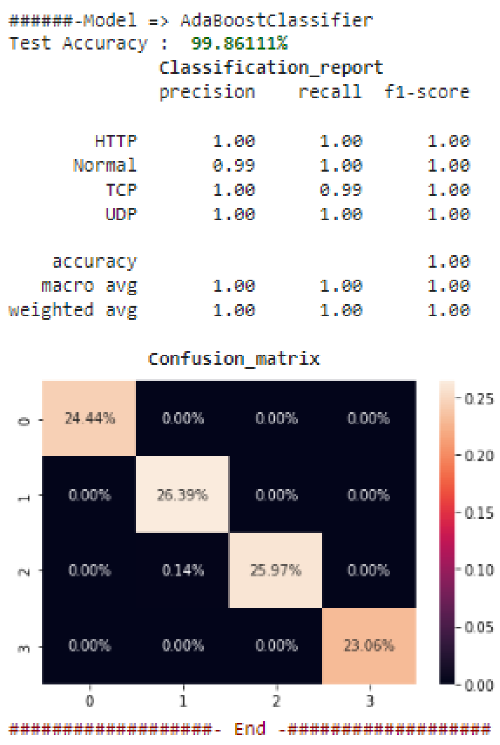
Figure 8. Decision tree classifier



```
######-Model => KNeighborsClassifier
Test Accuracy :  100.00000%
                Classification_report
                precision    recall  f1-score   support

        HTTP       1.00      1.00      1.00       176
      Normal       1.00      1.00      1.00       190
         TCP       1.00      1.00      1.00       188
         UDP       1.00      1.00      1.00       166

    accuracy                           1.00       720
   macro avg       1.00      1.00      1.00       720
weighted avg       1.00      1.00      1.00       720
```

```
###################- End -###################
```

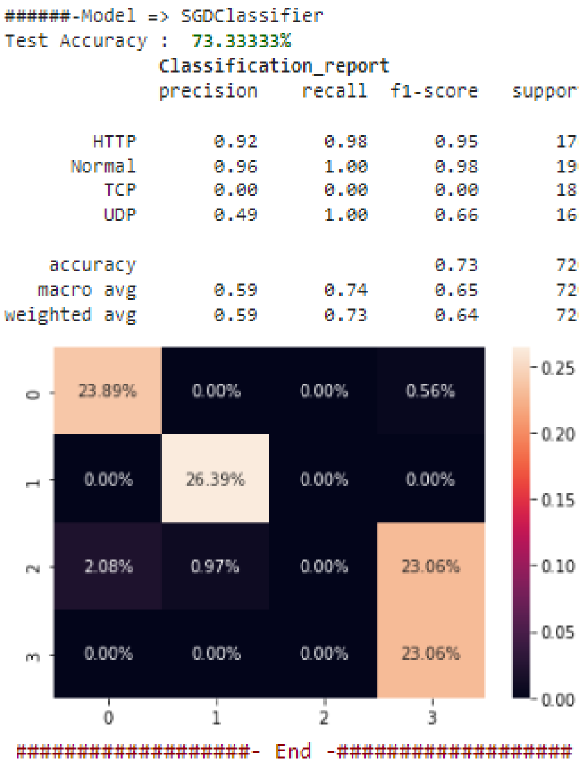
Figure 9. K-neighbor classifier



```
######-Model => AdaBoostClassifier
Test Accuracy :  99.86111%
              Classification_report
              precision    recall  f1-score   support

        HTTP       1.00      1.00      1.00       176
      Normal       0.99      1.00      1.00       190
         TCP       1.00      0.99      1.00       188
         UDP       1.00      1.00      1.00       166

    accuracy                           1.00       720
   macro avg       1.00      1.00      1.00       720
weighted avg       1.00      1.00      1.00       720

             Confusion_matrix
```

```
###################- End -###################
```

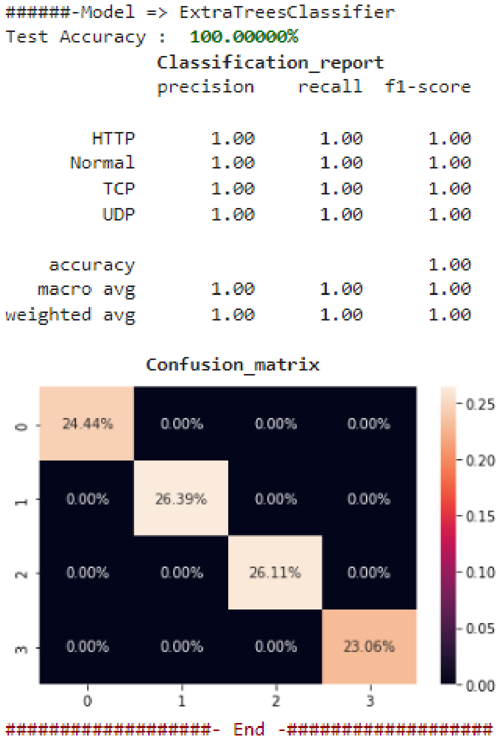
Figure 10. AdaBoost Classifier



```
######-Model => SGDClassifier
Test Accuracy :  73.33333%
              Classification_report
              precision    recall  f1-score   support

        HTTP       0.92      0.98      0.95       176
      Normal       0.96      1.00      0.98       190
         TCP       0.00      0.00      0.00       188
         UDP       0.49      1.00      0.66       166

    accuracy                           0.73       720
   macro avg       0.59      0.74      0.65       720
weighted avg       0.59      0.73      0.64       720
```

```
###################- End -###################
```

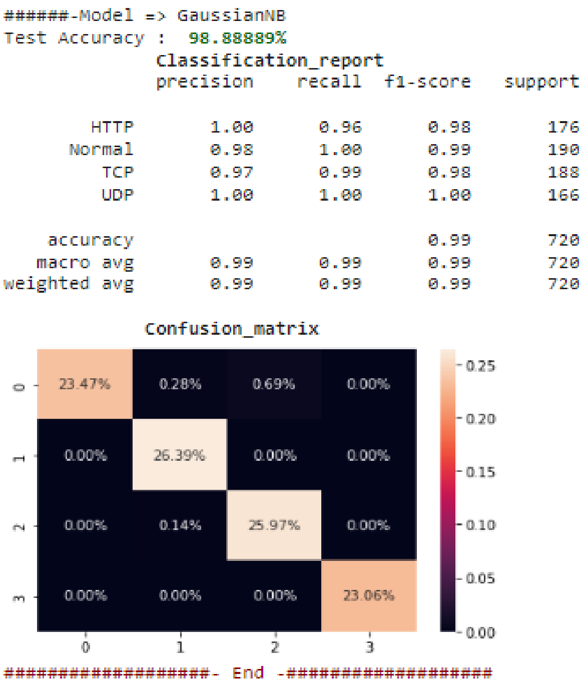
Figure 11. SGD classifier



```
######-Model => ExtraTreesClassifier
Test Accuracy :  100.00000%
              Classification_report
              precision    recall  f1-score   support

        HTTP       1.00      1.00      1.00       176
      Normal       1.00      1.00      1.00       190
         TCP       1.00      1.00      1.00       188
         UDP       1.00      1.00      1.00       166

    accuracy                           1.00       720
   macro avg       1.00      1.00      1.00       720
weighted avg       1.00      1.00      1.00       720

             Confusion_matrix
```

```
###################- End -###################
```

Figure 12. Extra trees classifier



```
######-Model => GaussianNB
Test Accuracy :  98.88889%
              Classification_report
              precision    recall  f1-score   support

        HTTP       1.00      0.96      0.98       176
      Normal       0.98      1.00      0.99       190
         TCP       0.97      0.99      0.98       188
         UDP       1.00      1.00      1.00       166

    accuracy                           0.99       720
   macro avg       0.99      0.99      0.99       720
weighted avg       0.99      0.99      0.99       720

             Confusion_matrix
```

```
###################- End -###################
```

Figure 13. SGD classifier

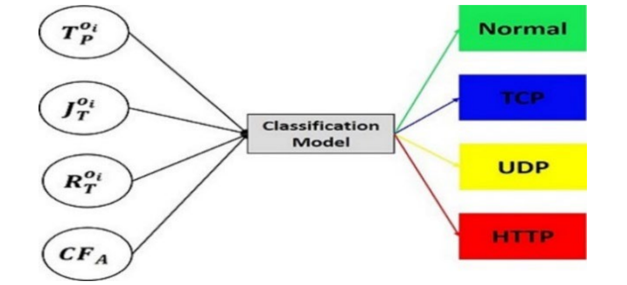
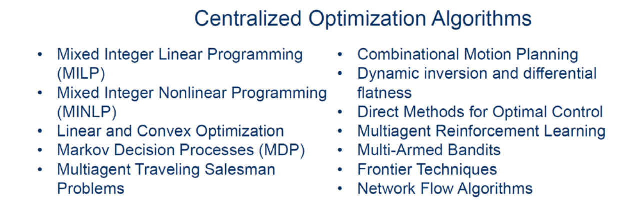


Figure 14. Data set for SDN performance metric



Centralized Optimization Algorithms

- Mixed Integer Linear Programming (MILP)
- Mixed Integer Nonlinear Programming (MINLP)
- Linear and Convex Optimization
- Markov Decision Processes (MDP)
- Multiagent Traveling Salesman Problems
- Combinational Motion Planning
- Dynamic inversion and differential flatness
- Direct Methods for Optimal Control
- Multiagent Reinforcement Learning
- Multi-Armed Bandits
- Frontier Techniques
- Network Flow Algorithms

Fig.15 Centralized Optimization Algorithms

### Implementation Tools - Software and Hardware Used for this research

To implement AI/ML models, we used Jupyter Notebooks and Google Colab. Both tools (applications) helped us in comparison and measuring efficiency of the result. Google Colab provided high GPUs to run our code better. To manage our data set, we used both FileZilla Server and FileZilla Host as needed. Both tools have been downloaded and configured on our machine. FileZilla Server is a server that supports FTP and FTP over TLS which provides secure encrypted connections to the server.

### Research outcomes

In Subproject 2.3, we used AI Machine learning-based detection and classification methods for DDoS attacks. For analysis of our machine learning experimentation, we applied popular supervised learning methods: Random Forest, Decision tree, K - Neighbors, AdaBoost, SGD, Extra Trees, and Gaussian NB.

Random Forest, K – Neighbors, and Extra Trees classifier methods were used and proved effective for application in the classification and detection of such attacks. Decision tree, AdaBoost, and GaussianNB Classifier methods also performed well. But SGD Classifier stability, prediction accuracy, and training time performed poorly and need further investigation. In this research, we did not use hyperparameter tuning and optimization to get a good result since the data set was not significantly huge.

### Future Directions

In the future, we would attempt using a larger data set to get a suitable and maximized accuracy result using AI/ML computational intelligence techniques. In addition, methods such as global sensitivity analysis and parallel coordinates analysis should be executed. Moreover, using hyperparameters for tuning will also play a great role in detecting and classifying DDoS flooding attacks.

## 3.4 A Scalable Real-Time Multiagent Decision Making Algorithm with Cost

### Abstract

We focused on a real-time multiagent decision making problem in a collaborative setting including a cost factor for the planning and execution of actions. We presented the centralized coordination of a multiagent system in which the team must make a collaborative decision to maximize the global payoff. We used the framework of coordination graphs, which exploit dependencies among agents to decompose the global payoff function value as the sum of local terms. We revised the centralized Max-Plus algorithm by presenting a new cost Max-Plus algorithm and including the cost in the local interactions of agents. We proposed a two-step planning and acting algorithm called Factored Value-MCTS-Cost-Max-Plus algorithm that is online, anytime, and scalable in terms of the number of agents and their local interactions.

### Original Contributions

We considered a budget constraint approach where there was cost associated with each action. Different actions consumed a different number of resources, which can be potentially correlated to the global payoff of the team. In such a setting, the aim of the local decision maker is to optimize his decision at any time under a cost and time constraint. Therefore, the global team reward at each time step is obtained after subtracting the total cost incurred by examining the cost of the local actions. In this way, we extended the previous work on centralized coordination where time was the only budget constraint.

We developed a new method *FV-MCTS-Cost-Max-Plus*. Our contribution is the development of a theoretical framework that combines MCTS (for planning) with Cost Max-Plus algorithm (for decision making and execution). The proposed method is a suboptimal solution for Dec-POMDPs. The exact solution of a Dec-POMDP is known to be intractable and Non-deterministic EXPonential (NEXP)-complete, even for only two agents.

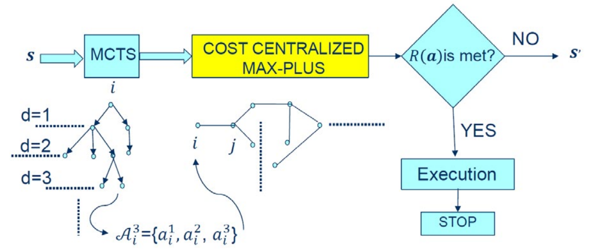

s
MCTS
COST CENTRALIZED MAX-PLUS
$R(\boldsymbol{a})$is met?
NO
s'
YES
Execution
STOP
i
d=1
d=2
d=3
i
j
$\mathcal{A}_i^3$={$a_i^1$, $a_i^2$, $a_i^3$}

**Fig. 16 The proposed FV MCTS Cost Max-Plus Method**

### Related Work

The problem of behavior coordination of multiagent decision making has a long story and it is a hot research topic being addressed in different communities such as game theory, reinforcement learning, cybersecurity, control and robotics. The optimal solution for distributed agents sharing a global reward is of most interest. A taxonomy of centralized optimization algorithms for global behavior is given in Fig.15.

### Results and Unique Contributions

In this research project we focused on decision-making settings where the agents interact with each other while making decisions. The agents are constrained by the cost of actions to make their potential decisions. *Little is known regarding how the cost of actions influence the decisions and what coordination algorithms can be used in real-time scenarios for planning and execution of actions when the agents share a common goal.*

Our main contribution is a new method for planning and acting called Factored Value-MCTS-Cost-Max-Plus which is presented in Fig. 16.

We focused on the real-time multiagent decision making problem with cost factor. In this regard, we used the Coordination Graphs (CG) setting. We described centralized coordination by revising the well-known Max-Plus algorithm (a.k.a. max-product or min-sum algorithm). We considered the budget-limited case by introducing the centralized Cost Max-Plus algorithm for the first time.

We proposed and described a new method that is more suitable for real applications such as cybersecurity. Our proposed method of Factored Value-MCTS-Cost-Max-Plus is online, anytime, distributed, and scalable in the number of agents and local interactions.

## 3.5 A Scalable Real-Time Distributed Multiagent Decision Making Algorithm

We presented a distributed algorithm for a multiagent system in a collaborative setting including the cost of actions. We used the framework of the coordination graphs which exploit the dependencies among agents to decompose the global payoff function value as the sum of local terms. We revised the distributed Max-Plus algorithm by presenting a new Cost Distributed Max-Plus (CD Max-Plus) algorithm. We also included the cost of actions in the interactions of agents, named Factored Value-MCTS-CD Max-Plus algorithm. In the first step of our algorithm, agents are not coordinated, and each agent is running the MCTS algorithm for the best individual planning actions with associated cost considered.

The most promising actions of each agent are selected and presented to the team. In the second step, the distributed coordination of agents is maintained through the CD Max-Plus algorithm for joint action selection. The proposed Factored Value-MCTS-CD Max-Plus algorithm is online, anytime, distributed, and scalable in number of agents. Our contribution is an alternative solution for solving cost Dec-POMDPs, competing with the state of art methods and algorithms using MCTS and Max-Plus algorithms to exploit the locality of agent interactions for planning and acting.

### Original Research Contributions

According to our knowledge, this is the first research project dealing with factored representation using MCTS for planning with cost (first step) and the Cost Distributed Max-Plus algorithm for joint action selection (second step). This is a new approach and the rationales for this two-step decision making process include:

1) Depending on the situation (state), some actions are more relevant than other actions. Local-level decision provides the order of actions, from high probability to low probability in terms of effectiveness based on the current situation.
2) Global-level optimization in multiagent environments require communication among agents. This communication is not always guaranteed in some situations (e.g., under cyberattacks or system malfunction). In such adversarial situation, the algorithm may not achieve the global optimal solution, thus, the local optimal solution in the first step may the best solution in such cases.

In a dynamic environment, it is necessary to collect the most urgent, real-time, accurate and up-to-date information. We are in favor of fully distributed coordination for multiagent decision making because of the following:

a) In centralized structures, a central agent takes joint observations of all agents and makes joint decisions for all agents. Each agent is taking an action based on the decision of the central controller. Failure or malfunction of the central agent is equivalent to the malfunction of the whole MAS.

b) The central controller needs to communicate with each agent to exchange information, which incessantly increases the communication overhead at the single controller. This may degrade the scalability of MAS as well the robustness to malicious attacks.

c) In a centralized setting (centralized controller), the agents are not allowed to exchange information to each other. A fully distributed coordination approach for MAS could allow local and correlated decisions for each agent that can communicate. In this way, the global team payoff is higher than in the centralized setting. With only local reward, it is difficult for the centralized approach to maximize the network-wide reward determined by the joint action of all agents.

Our research contribution is three-fold:

a. We considered a budget constraint approach where a cost is associated with each action. Different actions consume different amounts of resources which can be potentially correlated to the global payoff of the team. In such a setting, the aim of the local decision maker is to optimize his decision at any time under a cost and time budget constraint. Therefore, the global team reward at each time step is obtained after subtracting the total cost incurred in examining the cost of the local actions. In this way, we extended the previous work on centralized coordination where time was the only budget constraint.

b. We developed the Cost Distributed coordination version of Max-Plus algorithm where each agent computes and sends updated messages after an agent received new and different messages from one of its neighbors. The messages are sent in parallel, which offers some computational advantages over the sequential execution of the previous centralized coordination algorithms (Chaudhury at all 2021, Amato et all 2015, Mahajan 2021).

c. We developed a new method FV-MCTS-CD-Max-Plus with a two-step decision making process. Our contribution is the development of the theoretical framework for combining the Monte Carlo Tree Search (MCTS) with Cost Distributed Max-Plus algorithm for decision making and execution. The proposed method is a suboptimal solution for Dec-POMDPs. The exact solution of a Dec-POMDP is known to be intractable and Non-deterministic EXPonential (NEXP)-complete, even for only two agents.

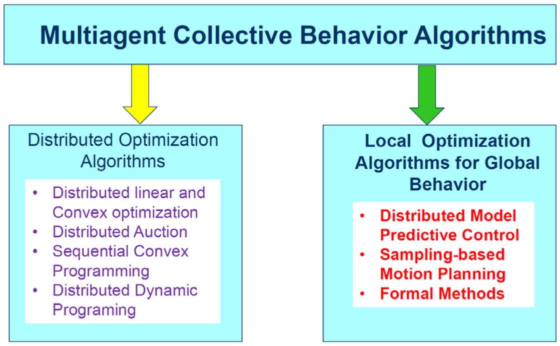

Multiagent Collective Behavior Algorithms
Distributed Optimization Algorithms
Distributed linear and Convex optimization
Distributed Auction
Sequential Convex Programming
Distributed Dynamic Programing
Local Optimization Algorithms for Global Behavior
Distributed Model Predictive Control
Sampling-based Motion Planning
Formal Methods

Fig. 17 The proposed FV MCTS Cost Max-Plus Method

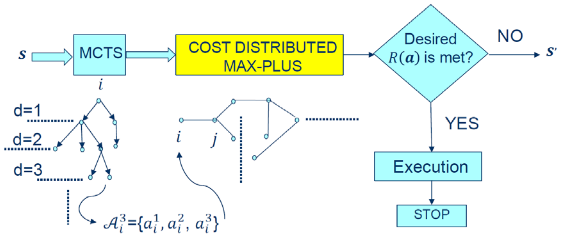

s
MCTS
COST DISTRIBUTED MAX-PLUS
Desired $R(\boldsymbol{a})$ is met?
NO
s'
YES
Execution
STOP
i
d=1
d=2
d=3
$\mathcal{A}_i^3=\{a_i^1, a_i^2, a_i^3\}$
i
j

Fig. 18 The proposed FV MCTS Cost Distributed Max-Plus Method

A taxonomy of the distributed and local optimization algorithms for multiagent global behavior is given in Fig. 17.

In this research we focused on distributed coordination decision-making where the agents interact with each other while making decisions. In such CMAS, the agents are constrained to make their potential decisions due to cost of actions.

*Little is known on how the cost of actions influence the decisions. The same is true for distributed coordination algorithms used to solve real-time scenario for distributed planning when the agents share a common goal.*

## Results

Our main contribution is a new method for planning and acting called Factored Value-MCTS-Cost-Distributed Max-Plus, which is presented in Fig. 18.

As illustrated in Fig.18, the Cost Distributed Max-Plus algorithm is run in addition to MCTS. MCTS is an online, anytime planning algorithm that unfortunately is not scalable due to the exponential number of states and available actions at each state. From the anytime, scalar, and planning perspective, it makes sense to combine MCTS with the Cost Distributed Max-Plus algorithm, providing limited budget to obtain better results for decision making of MAS.

We focused on real-time multiagent decision making problems with a cost factor. In this regard, we used the coordination graphs setting. We described centralized and distributed coordination by revising the well-known Max-Plus algorithm (a.k.a. max-product or min-sum algorithm). We considered the budget-limited case by introducing the Cost Distributed Max-Plus algorithm for the first time.

We proposed and described a new method that is more suitable for real applications such as cybersecurity. Our proposed method Factored Value-MCTS-Cost-Distributed Max-Plus is online, anytime, distributed, and scalable in the number of agents and local interactions.

### Future Directions

There are many future directions to improve the Global Reward with cost of MAS based on the locality of agent interactions. One future direction is using the recent advances in deep Reinforcement Learning (deep RL) that have demonstrated the great potential of neural network for function approximation in handling large state spaces.

Another direction is inspired from game theory using “regret techniques”. When the team decides about a global optimal action, one or more agents may have a “regret” in choosing their previous actions. minimizing the counterfactual regret is equivalent to maximizing the global reward. The key idea is that information state qualitatively represents the data changing at a given node. Another direction is using a statistical approach for information-theoretic context to maximize the information from a given state of MAS.

## 3.6 A Hybrid Cost Collaborative Multiagent Decision Making Algorithm with Factored Value Max Plus

We have presented two real-time multiagent decision algorithms: a centralized version and a distributed version. In a dynamic or hostile environment, network segmentation is unavoidable. In such cases, it is necessary for each agent to continually make the most urgent real-time decisions in both centralized and decentralized ways. In this paper, we present a Hybrid Factored-Value Max-Plus algorithm with cost which has online, anytime, and scalable properties despite network segmentation. For the distributed version of the Max-Plus algorithm with cost, we assume the optimal algorithms for obtaining the minimum spanning tree configuration. Our contribution is an alternative solution for solving Dec-POMDPs that competes with state-of-the-art methods and algorithms, using Monte Carlo Tree Search and Max-Plus algorithms to exploit the locality of agent interactions for planning and acting.

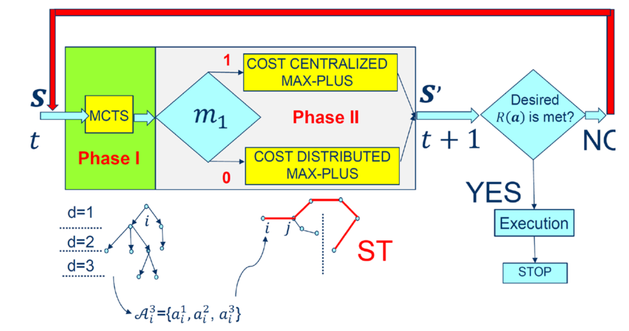

$s$
$t$
MCTS
Phase I
$m_1$
1
0
COST CENTRALIZED MAX-PLUS
Phase II
COST DISTRIBUTED MAX-PLUS
$s'$
$t+1$
Desired $R(\boldsymbol{a})$ is met?
NC
YES
Execution
STOP
d=1
d=2
d=3
$i$
$j$
ST
$\mathcal{A}_i^3=\{a_i^1, a_i^2, a_i^3\}$

Fig. 19 Factored Value MCTS Hybrid Cost Max - Plus Method

### Original Research Contributions

*As far as we know, this is the first research dealing with FV representation using MCTS for both the cost centralized MP planning algorithm and the cost distributed MP algorithm for joint action selection.*

In a dynamic environment, it is necessary to collect the most urgent, real-time, accurate, and up-to-date information. We focus on hybrid coordination for multiagent decision making because of the following:

a) In centralized structures, a central controller takes joint observations of all agents and makes joint decisions for all agents. Each agent performs an action based on the decision of the central controller. Failure to communicate or malfunction of the central controller is equivalent to malfunction of the whole MAS.

b) The central controller needs to communicate with each agent to exchange information, which inherently increases the communication overhead at the single central controller. This may degrade the scalability of MAS as well the robustness to malicious attacks.

c) In a centralized setting (central controller), the agents are not allowed to exchange information with each other. A fully distributed coordination approach for MAS could allow local and correlated decisions for each agent that can communicate. In this way, the global team payoff is higher than in the centralized setting. With only local rewards, the centralized approach can hardly maximize the network-wide reward determined by the joint action of all agents.

*We are not aware of any hybrid approach for collaborative multiagent decision making to solve real-time coordination and planning when agents share a common goal.*

We believe that our proposed solution using the MCTS method coupled with Cost Centralized and Cost Distributed MP algorithms for hybrid coordination of agents will enhance the performance obtained so far. In addition, our solution will improve the uniformly distributed selections of actions as proposed in previous works.

### Results

Our main contribution is a new method for planning and acting called the Cost Hybrid Factored-Value MCTS Max-Plus method, which is presented in Fig.19. The method is a two-phase decision-making process:

We focused on the real-time multiagent decision making problem with cost factor by using the coordination graphs and spanning tree settings**.**

Our proposed method Cost Hybrid Factored-Value MCTS Max-Plus is online, anytime, distributed, and scalable in the number of agents and local interactions, which makes it particularly suitable for real applications such as cybersecurity.

### Future Directions

There are many future directions to improve the global reward with cost of MAS based on the locality of agent interactions. One future direction is using the recent advances in deep reinforcement learning that have demonstrated the great potential of neural networks for function approximation in handling large state space. Another direction is inspired by game theory using "regret techniques". When the team decides a global optimal action, one or more agents may have a "regret" in choosing their previous actions. Minimizing the counterfactual regret is equivalent to maximizing the global reward. The key idea is that information state qualitatively represents the data changing at a given node. Yet another direction is the information statistical approach for maximizing the information from a given state of MAS.

## 4. Overall Results and Discussions

Our research results are published in 6 papers at various IEEE conferences. Two are currently under review and another two papers are in preparation to be submitted. We were able to support six PhD students. With the support from this ARLIS grant, two PhD students are ready for thesis defense, another two were supported to pass the qualifying exam, and another two are eager to continue to work on their PhD thesis. The main PhD students research results are summarized below:

- Publications of research papers
- Tangible research outcomes that could provide more directions for future research.
- Enhancement of our Ph.D. students' capabilities in research development.
- Development of students' subject matter expertise.

### Notable recent achievements

According to our research practices with the support of the ARLIS grant, we have achieved several notable outcomes that could be extended for further student research activities.

- Student research development experience
- Student knowledge base improvement
- Improvement of student research development experience
- Development of student interest in the subject matter (machine learning & cybersecurity)

We are working to submit our research work for presentation and publication at an upcoming international conferences.

## 5. Future Directions

UDC has started offering a PhD program (the first two PhD students graduating at UDC will be supported from this grant) and a new B.S. cybersecurity program. We expect to start the MS cybersecurity program during the spring of 2023. Based on our research activities and experience with the different ARLIS machine learning experimentation research projects, with another two years of research grants we would consider the following options:

1) The CSIT and ECE departments would get an opportunity to enhance its cybersecurity curriculum by creating new advanced research-based cybersecurity courses called "Advanced Cyber-risk Mitigation using Machine Learning" and "Securing Artificial Intelligent System Using Advanced Machine Learning Techniques."
2) The grant would have a societal impact by supporting our under-privileged undergraduate and graduate cybersecurity students with a great opportunity to be involved in advanced research projects and improve their cybersecurity skills.
3) We would be able to produce more research papers extending the work that has been published and work on new related cybersecurity research areas with the application of machine learning and artificial intelligence.

# Pilot Project 3: Cyber-Assessment of AI/ML Tools

Gloria Washington[2*], Paul Wang[1], Onyema Osuagwu[1], Ketchiozo Wandji[1], Tanvir Arafin[1], with ARLIS Lead Craig Lawrence[4]

**Program Objective**

Survey key open-source AI/ML toolkits, design a methodology for testing vulnerabilities or weaknesses, and to provide best practices to countermeasure the potential risk.

**Keywords**

Cyber assessment, open-source toolkits, biometrics, vulnerabilities, adversarial attacks

[2] Howard University
[1] Morgan State University
[4] Applied Research Laboratory for Intelligence and Security (ARLIS), University of Maryland, College Park, Maryland 20742
***Corresponding author:** gloria.washington@howard.edu

## 1. Project Overview

Howard University and Morgan State University partnered together to perform a cyber-assessment of AI/ML Tools. The primary project goals were to survey key open-source AI/ML toolkits, design a methodology for testing vulnerabilities or weaknesses, and to provide best practices to countermeasure the potential risk.

### 1.1 Overarching Goals of Pilot

The overarching goal across the institutions was to perform an assessment of key AI/ML open-source tools.

Howard was specifically focused on cyber-assessment of open-source toolkits for human identification/ recognition, or biometric assessment. Morgan State was specifically concerned with performing assessment of networking-based tools, cybersecurity risk analysis, and vulnerabilities in existing AI/ML algorithms. Each institution followed the below steps in their assessment.

1. Survey and analysis of open-source AI/ML tools to determine the top 10 for further investigation
2. Development of cyber assessment methodology and approaches
3. Prototype software that will generate a robustness score for a given AI/ML tool (similar to a credit score)
4. Conduct testing and collect data
5. Documentation and recommendation

The projects reconvened at the end of the effort to disseminate and report results so that each institution could learn from the other's research activities.

### 1.2 Lines of Effort

Howard University is the lead institution for the project and the team consists of Dr. Gloria Washington and her six Howard undergraduate research assistants. Howard examined key AI/ML open-source biometric toolkits implemented in TensorFlow, Scikit Learn, Pytorch, and HuggingFace. The models used in this research were OpenFace, DeepFace, ArcFace, Ensemble, DeepID, VGG-Face, Facenet, and Facenet512. These models were chosen because they are reported in the literature as performing well above 80% accuracy for biometric face recognition. They also represent deep learning, descriptor based, and transformer-based models.

Morgan State University is the partnering institution whose team consists of Dr. Paul Wang, Dr. Onyema Osuagwu, Dr. Ketchiozo Wandji, and Dr. Md Tanvir Arafin. The team first conducted literature review of common AI/ML tools which include: Tensorflow, CrypTFlow, Security Risks in Deep Learning Implementation, Loopholes in Tensor Flow, Scikit Learn, with NumPy, SciPy, Matplotlib, Oryx 2, and Counterfeit. The Morgan team developed (programmed) a cyber threat analysis tool (Python) to identify cyber threats using ML technologies. The source code of this tool is shared at github. The publication "Trustworthy Artificial Intelligence for Cyber Threat Analysis" is published in Springer Lecture Note in Networks and Systems (ISBN 978-3-031-16071-4). Another paper "Design high-confidence computers using trusted instructional set architecture and emulators" was published by Elsevier "High Confidence Computing". Vol 1, Iss. 2, 2021.

The Morgan team also conducted a study in detecting attacks to autonomous navigation system. A paper titled "Secure Autonomous Navigation Under Adversarial Attacks" is accepted for publications. Additionally, members of the Morgan team spoke at cyberMD 2022 conference and Intellisys 2022 conference.

During the research period, the Morgan team mentored and supported 4 undergraduate students and 3 graduate students. In addition, 20 students worked on the project as their senior projects.

### 1.3 Why It Matters

The United States spends billions of dollars a year to protect its physical systems against hacks. The Intelligence Community and Security Communities are often concerned about how to best protect its systems against hacks and breaches. Similarly, academic institutions and businesses operating in the United States are concerned with security breaches. Often times these security breaches can result in loss of classified or sensitive data.

Vulnerabilities, weaknesses, and loopholes in existing AI/ML tools can not only cause bias but also threaten the safety and security of the systems. Identifying those threats and provide countermeasures will reduce the risks associate with the threats.

For computer science and computer engineering students, mastering cybersecurity skills would enable them to think differently in design software engineering applications that meet the challenges of security issues in cyberspace.

## 2. Background and Related Work

Cybersecurity experts estimate that the United States Government spends tens of billions of dollars a year to protect its networks against hacking (Nakashima, E. 2013). The Government employs both networking-based security measures and biometric-based authentication and identification to protect its system. Rarely have researchers reported on both systems to perform cyber-assessment. Biometric systems utilize physical human characteristics to recognize its trusted users in pictures and sometimes image segmented gathered from video (El-Sayed, A. Y. M. A. N., 2015). Network protection of vulnerable systems includes hardware performance monitoring, packet-analysis, and user access controls.

Prior research has been performed in biometric anti-spoofing methods, but little has been done to determine the robustness of facial and ear recognition systems (Serror, M., Hack, S., Henze, M., Schuba, M., & Wehrle, K., 2020).
Hardware performance and analysis of network traffic can provide much insight into the security of a system (Hamdioui, S., Danger, J. L., Di Natale, G., Smailbegovic, F., van Battum, G., & Tehranipoor, M., 2014)

Cyber threat analysis involves billions of data records from servers, firewalls, intrusion detection and prevention systems, and network devices. Those include log data that have been stored on log servers and live data that are generated in real time. It has been a challenge to comb through that vast amount of data to find out threat vectors. Predefined signatures are effective, but they would fail if new threat vectors emerged, or adversaries' behavior changed.

The advancement of Artificial Intelligence has opened up new ways to analyze and understand data and user behaviors. Machine learning algorithms are able to not only analyze data efficiently, but also build up knowledge gained from previous learning. Algorithms and models built for general purposes are available for immediate starting a project. The models become more accurate under supervised training when more data are fed in.

Artificial intelligence brings innovation in industry and everyday life. The Deep Blue chess computer can defeat the greatest human chess player in the world. Autonomous vehicles such as Tesla can drive on the road without human interactions.

In cybersecurity, AI/ML is used to deeply inspect the packets, analyze the network activities, and discover abnormal behaviors.

Sagar et al. conducted a survey of cybersecurity using artificial intelligence (BS S, S N, Kashyap N, DN S 2019). It discusses the need for applying neural networks and machine learning algorithms in cybersecurity.

Mittu et al. proposed a way to use machine learning to detect advanced persistence threats (APT) (Mittu R, Lawless WF, 2015). The approach can address APT that can cause damages to information systems and cloud computing platforms.

Mohana et al. proposed a methodology to use genetic algorithms and neural networks to better safeguard data (V K Venugopal KM, Sathwik H, 2014). A key produced by a neural network is said to be stronger for encryption and decryption.

With a grant from the National Science Foundation (NSF), Wang and Kelly developed a video data analytics tool that can penetrate videos to "understand" the context of the video and the language spoken (Wang S, Kelly W, 2014).

Kumbar proposed a fuzzy system for pattern recognition and data mining (Kumbar SR, 2014). It is effective in fighting phishing attacks by identifying malware.

Using Natural Language Processing (NLP), Wang developed an approach that can identify issues with cybersecurity policies in financial processing process (Wang S, 2018) so financial banking companies can comply with PCI/DSS industry standards.

Harini used intelligent agent to reduce or prevent distributed denial of service (DDoS) attacks (Rajan HM, 2017). An expert system is used to identify malicious codes to prevent being installed in the target systems.

With a grant from National Security Agency (NSA), Wang and his team developed an intelligent system for cybersecurity curriculum development (Hodhod R, Khan S, Wang S, 2019). The system is able to develop training and curricula following the National Initiative of Cybersecurity Education (NICE) framework.

Dilrmaghani et al. provide an overview of the existing threats that violate security and privacy within AI/ML algorithms (Dilmaghani S, Brust MR, Danoy G, Cassagnes N, Pecero J, Bouvry P, 2019).

Gupta et al. studied quantum machine learning that uses quantum computation in artificial intelligence and deep neural networks. They proposed a quantum neuron layer aiming to speed up the classification process (Gupta S, Mohanta S, Chakraborty M, Ghosh S, 2017).

Mohanty et al. surveyed quantum machine learning algorithms and quantum AI applications (Mohanty JP, Swain A, Mahapatra K, 2019).

Edwards and Rawat conducted a survey on quantum adversarial machine learning by adding a small noise that leads to classifier to predict to a different result (Edwards D, Rawat DB, 2020). By depolarization, noise reduction and adversarial training, the system can reduce the risk of adversarial attacks.

# 3. Methods

## 3.1 Biometric Robustness Cyber-assessment

The research activities undertaken by Howard included analysis of opensource biometric toolkits related to face recognition only. This biometric modality was chosen because it is the most commonly implemented across hardware like mobile phones and tablets. Also, face recognition systems are notorious for their failures due to lighting, occlusions, and quality of the images that are input into the systems. The Howard team performed the following steps with the common facial recognition.

1. Download opensource biometric toolkit
2. Perform lighting, occlusion, and image quality experiments
3. Compare biometric test results for recognition of participants across the datasets Notre Dame J2 dataset, Yale Faces, and Hong Kong University datasets
4. Document results and recommendations

### Datasets for Experimentation

The datasets in this work were chosen because they vary according to image quality, lighting, participant pose variation, occlusion variations, and age. They are described below.

### *LIRISChildrenSpontaneousFacialExpressionVideo Database*

The LIRISCSFEV Dataset contains movie clip / dynamic images of 12 ethnically diverse children. This unique database contains spontaneous / natural facial expression of children in diverse settings.

### *Stimuli Dataset*

This dataset contains full frontal views with only face extracted i.e. little noise + contains male + female. But it does not contain ethnic diversity.

### *Basel_Face_Database*

This dataset contains 14 different facial expressions per person.

### *Faces Database*

This dataset contains 72 images of facial expressions in young, middle-aged, and older women and men.

### *MR2 Dataset*

This dataset contains a multi-racial, mega-resolution image database of facial stimuli from diverse test subjects.

### *Japanese_Female_Facial_Expression_Dataset*

This dataset contains Japanese subjects only.

### *Young adult white faces Dataset*

This dataset only contains images of young white males and females with various occlusions like masks, glasses, hats, bangs, etc.

### *Yale Faces Dataset*

This dataset contains only greyscale images of participants in various poses. The lighting is consistent across all participants.

## 3.2 Cyber threat assessment of AI/ML tools

The research activities undertaken by Morgan State included analysis of hardware performance and network traffic data AI/ML toolkits. These toolkits included vulnerability analysis, packet capture and analysis, poisoning exploitation, backdoor discovery, web and database injection, reverse engineering, forensic analysis, APT analysis, software dependencies inspection, and side-channel attacks.

The steps used to perform the research activities included:

1) Survey and analysis of open source AI/ML tools;
2) Design of a vulnerability and testing toolkit;
3) Design of experiments;
4) Documentation of outcomes and recommendations.

Detecting web attacks using machine learning is an area that has drawn attention and requires continuous research and development. This project analyzes 822,226 log records from a company's web login page in a 5-hour time span. After cleaning and pre-processing the data, the CTML algorithm detected records that could potentially be attacks. It then calculated the likelihood of attacks based on abnormal behaviors.

The main strategy is to use unsupervised learning for better understanding the distribution of the input data. Supervised learning is then applied for further classification and generating predictions. As a result, the CTML model could learn how to predict/classify on output from new inputs. Reinforcement learning (RL) learns from experiences over time. The algorithm can be improved with more data feed into the system.

# 4. Results and Discussion

Results from assessment of cyber-human and cyber-physical systems through analysis of biometric recognition results and network hardware performance were both troubling and promising. They were promising because biometric robustness testing can help us to ascertain the minimum system requirements for Intelligence Community and Security Systems that implement these technologies. Furthermore, troubling because facial recognition systems continue to be plagued by low recognition performance for images captured outside of normal laboratory settings with controlled lighting, image quality, and participant hairstyles. More details are described in their own sections below.

## 4.1 Biometric Cyber-assessment

Biometric cyber-assessment included conducting various experiments that changed the lighting, image quality, and occlusion properties of the Yale Faces, Notre Dame, Visual Geometry Group Face (VGG-Face), and Labeled Faces in the Wild, LIRIS Children Spontaneous Facial Expression Database, Basel Face database, multi-racial & mega-resolution (MR2), and ArcFace databases. Lighting experiments involved changing the contrast of the original image by increments of 10. Occlusion experiments involved using the images marked in the dataset with bangs and hats. Additionally, the researchers wanted to determine if removing the eyebrows would provide any more necessary information. The image quality was varied by running common filters over the images (Gaussian and Laplacian filters).

Table 1 shows the model performance across the LIRISCSFE dataset. The models performed the best on this dataset with the highest accuracy being 90%.

Table 1. **Model Accuracy Performance on LIRISCSFE dataset**

| Mode Name | Database | Accuracy |
|---|---|---|
| DeepID | LIRISCSFE | 80% |
| VGG-Face | | 90% |
| Facenet | | 80% |
| Facenet512 | | 80% |
| OpenFace | | 80% |
| DeepFace | | 80% |
| ArcFace | | 90% |
| Ensemble | | 90% |

### Lighting Experimentation

Lighting experiments involved changing the contrast of images put into the biometric systems.

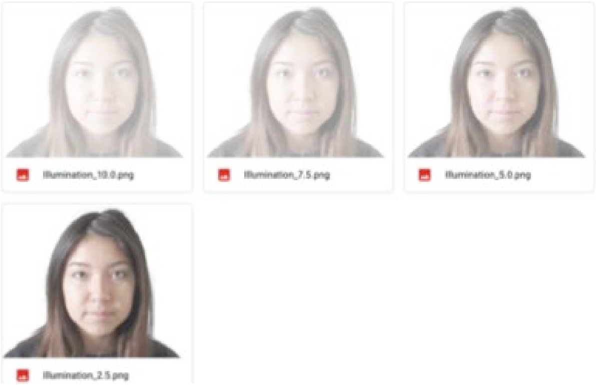


Figure 1. Lighting experimentation example images

### Occlusion Experimentation

Occlusion experiments involved varying the input of images into the biometric toolkits according to facial occlusions that included eyeglasses, bangs, and thickness of the eyebrows.

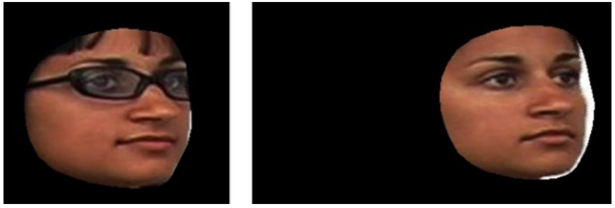


Figure 2. Occlusion experimentation example images.

### Image Quality Experimentation

Image quality experimentation involved applying common image processing filters over the datasets. These filters included pepper, Gaussian, Poisson, and Laplacian filters.

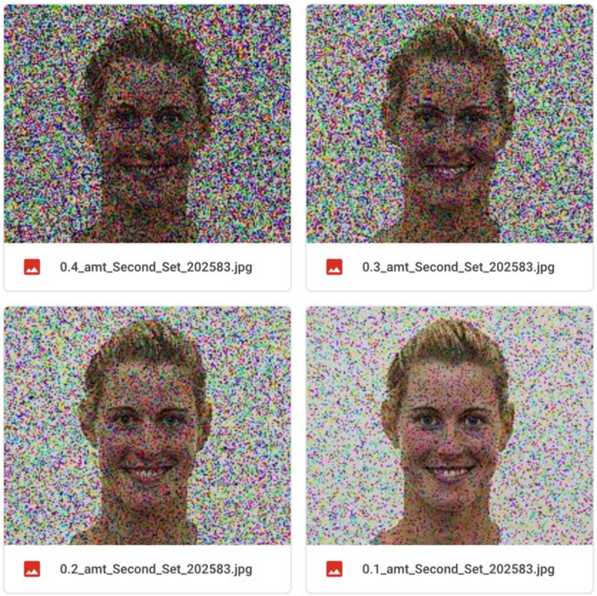

0.4_amt_Second_Set_202583.jpg
0.3_amt_Second_Set_202583.jpg
0.2_amt_Second_Set_202583.jpg
0.1_amt_Second_Set_202583.jpg

Figure 3. Pepper image filter.

The biometric models used in this work were created by various biometrics researchers in prior experiments. They were used to understand the lighting, occlusion, and noise constraints that sometimes affect the performance of biometric recognition systems. The LIRIS Children Spontaneous Facial Expression Database with the VGG-Face model had the highest accuracy measure for this work. This may be because the dataset is a high-quality dataset with images of children with various facial expression poses. There was little variation in the pose of the participants and occlusion as all participants had their hair back in the images.  One reason why all the models performed the best on this dataset is that the participants were younger, with little discoloration and wrinkles in their faces.

## 4.2 Vulnerabilities in Existing AI/ML Tools

A review and study by the Morgan team reveals that vulnerabilities do exist in many AI/ML tools.

Tensorflow is one of the most popular deep learning libraries to classify MNIST dataset. It is an open-source library developed by Google. In this Study, Convolutional Neural Network (CNN) and SoftMax classifier are used as deep learning artificial neural network. The results show that the most accurate classification rate is obtained.

Security issues around TensorFlow include issue in models, hot to run untrusted models, and accepting the untrusted input.

The study also found vulnerabilities in popular deep learning frameworks including Caffe, and Torch. By exploiting voice recognition and image classification, attacks can launch denial of service attacks that crash a deep learning application, or control-flow hijacking attacks that lead to either system compromise or recognition evasions.

Scikit-learn is a machine learning toolbox for Python. It provides classification, regression, clustering, dimensionality reduction, data prepossessing, and model selection problems. This tool is built on popular numerical packages in Python, such as NumPy, SciPy, and Matplotlib.

Oryx 2 is a machine learning framework that supports large scale end-to-end clustering, regression, and classification tasks. The key idea behind the scalable Oryx2 design is based on the lambda architecture on top of Apache Spark and Apache Kafka. Oryx 2 can support real-time data processing. As a result, this tool is one of the industry choices for building up backend machine learning applications.

A compilation of the current state of adversarial attacks on machine learning systems can be found.

NIST Computer Security Division provides guides (800-53) for risk assessment. It can help identify risks associated with those vulnerabilities mentioned above.

A study of neural network security from hardware perspective shows that hardware-based vulnerabilities on neural networks have been investigated over the last five years, and the critical achievements are being made.

## 4.3 Cyber threats assessment

Morgan team developed a cyber threat analysis tool. The cyber threat analysis application first loads the input data into a Pandas data frame, then removes features that are not of interests in detecting attacks. Next, the data are "compressed" from 800,000+ to around 40,000 by combining the records that have the same source and destination IP addresses in the same unit time period. The higher the compression rate, the more duplications in the dataset. This improved the efficiency of the machine learning process. Unsupervised machine learning is applied to the dataset using K-means clustering. The three output clusters are labeled as not-suspicious, suspicious, and transitional area.

The pre-processed data are then split into 0.66/0.33 for training/testing and further analysis of the likelihood of each response's abnormal behaviors. Using results from the unsupervised learning as a supervisor, the algorithm continues to apply supervised machine learning to discover threats. In addition, in areas that are considered "confident" or "not confident", the transition (gray) area is further analyzed using k-mean clustering to separate into two clusters, labeled as "more suspicious" and "less suspicious". The "more suspicious" tags are then added into the suspicious activity dataset (**Figure 3, 4**). By doing so it ensures that the machine does not miss any responses that get filtered out during the analysis process (but is still suspected having abnormal behaviors). The likelihood of the suspicion is calculated based on the percentage over the maximum response per second.

An attack for a general log-in page is defined as a considerable number of visits, responses, and callbacks in a short period of time. Thus, pre-processing the data by combining each duplicating responses per second helps determine the number of responses or visits that stand out.

The application can be improved by feeding into more rich data so risks associated with human behaviors can be identified.

The 3-2 two-tier classification technique helps with narrowing down the suspicious activities. If the k-means clustering is applied only once with two clusters, the uncertain groups of data set could possibly be wrong. Therefore, creating a transition (grey) area in the middle of two certainties helps detecting the potential attacks that could be missed.

## 4.4 Adversarial Attacks on AI/ML Algorithms

Hop Skip Jump is a decision-based attack which is a subgroup of transfer-based attacks. This means that the attacker has access to decisions alone. These attacks are first run on a testing environment before they are transferred to the targeted network. The attack is used to train the testing model by using query data from the targeted model.

Using Counterfeit, the Hop Skip Jump attack parameters that had the most success on the target were: max

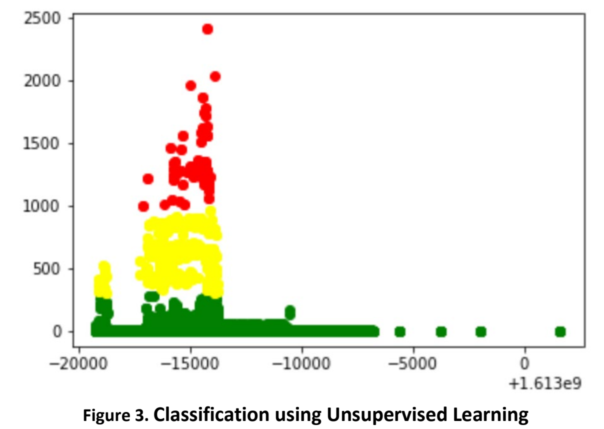


Figure 3. **Classification using Unsupervised Learning**

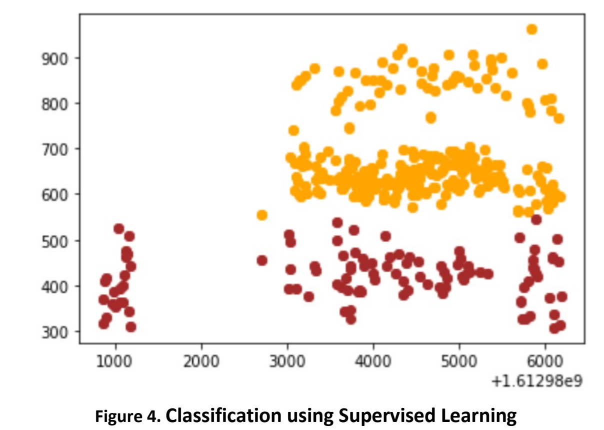


Figure 4. **Classification using Supervised Learning**

integer, max eval, batch size, current integer, and the initial size. For each different attack, I would increase all the variables of one attack then decrease them for the next, then utilize a mixture of the two for another. The results after running six different attacks are shown in **Table 1**.

**Table 1: Hop Skip Jump Attack with Counterfeit**

| Credit fraud HopSkipJump | Success | Elapsed Time (s) | Total Queries | Query per sec |
|---|---|---|---|---|
| Attack 1 | 1/1 | 1.4 | 68419 | 50604.6 |
| Attack 2 | 1/1 | 12.6 | 314342 | 24852.7 |
| Attack 3 | 1/1 | 0.5 | 11603 | 23382.7 |
| Attack 4 | 1/1 | 1.9 | 124964 | 64264.5 |
| Attack 5 | 1/1 | 5.3 | 73021 | 13870 |
| Attack 6 | 1/1 | 43.9 | 3893668 | 88637.2 |

For the Digits Blackbox attacking using the boundary attack the parameters, the team had the overall best success rate with a max integer, sample size, number trials, initial size, and verbose = true. For the boundary attack, we would also randomly increase and decrease the input of each variable for each attack and the results of the six attacks are shown in **Table 2**.

**Table 2: Digits Blackbox Boundary Attack with Counterfeit**

| Digits Blackbox Boundary attack | Success | Elapsed Time (s) | Total Queries | Query per sec |
|---|---|---|---|---|
| Attack 1 | 1/1 | 0.6 | 3036 | 4756.2 |
| Attack 2 | 1/1 | 19.5 | 136143 | 6974.3 |
| Attack 3 | 1/1 | 167.1 | 1227536 | 7356.0 |
| Attack 4 | 1/1 | 8.1 | 42474 | 5221.5 |
| Attack 5 | 1/1 | 19.5 | 135990 | 6984.1 |
| Attack 6 | 1/1 | 27.3 | 146459 | 53623.2 |

For a Hop Skip Jump attack on a Digits Blackbox target, the parameters that gave me the most success was the same as the parameters for the credit fraud: max integer, max eval, batch size, current integer, and the initial size with the addition of changing the verbose variable to true. The results of the five attacks are shown in **Table 3**.

**Table 3: Digits Blackbox Attack with Counterfeit**

| Digits Blackbox HopSkipJump | Success | Elapsed Time(sec) | Total Queries | Query per sec |
|---|---|---|---|---|
| Attack 1 | 1/1 | 0.6 | 68419 | 11494.6 |
| Attack 2 | 1/1 | 2.1 | 24550 | 11865.1 |
| Attack 3 | 1/1 | 22.5 | 267437 | 11877.4 |
| Attack 4 | 1/1 | 60.9 | 752730 | 12369.1 |
| Attack 5 | 1/1 | 2.2 | 24550 | 11267.7 |

Finally, we have the digits Keras target with boundary attacks. The difference between this target and the two previous targets is that unlike the credit fraud and digits Blackbox, the digits Keras only produces one query for each attack. Max integer, sample size, number trials, initial size, and verbose = true seemed to have the most success on the target. With this target we got more attacks that failed because even though the adversarial example could be found, it was not optimal. This happened only when we kept the original parameter and just ran the attack (results are shown in **Table 4**).

**Table 4: Digits Keras Boundary Attack with Counterfeit**

| Digits Keras HopSkipJump | Success | Elapsed Time(s) | Total Queries |
|---|---|---|---|
| Attack 1 | 1/1 | 4.9 | 1 |
| Attack 2 | 1/1 | 12.4 | 1 |
| Attack 3 | 1/1 | 79.0 | 1 |
| Attack 4 | 1/1 | 26.5 | 1 |
| Attack 5 | 1/1 | 124.8 | 1 |
| Attack 6 | 1/1 | 72.1 | 1 |

Lastly, we have the Hop Skip Jump attacks on the Digits Keras target. The parameters that gave the team the most success were the same as the parameters for the credit fraud and the digits Blackbox: max integer, max eval, batch size, current integer, and the initial size with the addition of changing the verbose variable to true. The results of the six attacks are shown in **Figure 5** (screen capture of the attacks the Morgan team has conducted).

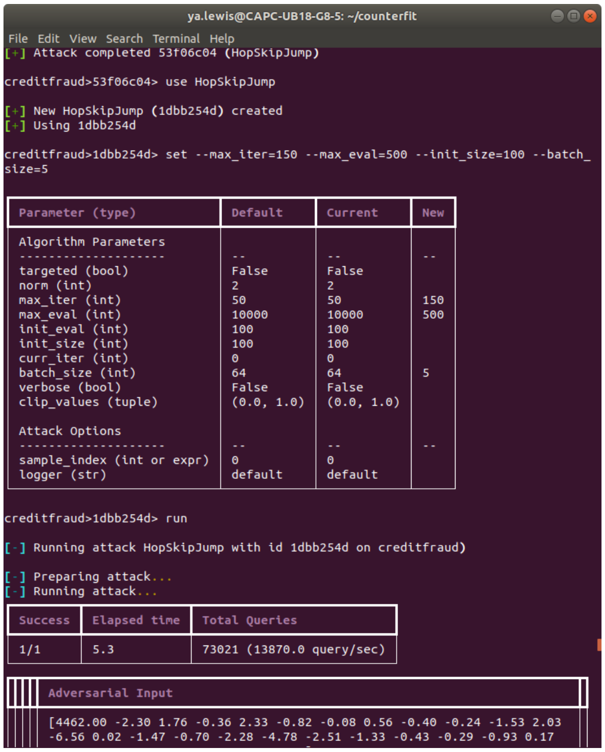
**Table 5: Digits Keras HopSkipJump Attack with Counterfeit**

| Digits Keras Boundary attack | Success | Elapsed Time(seconds) | Total Queries |
|---|---|---|---|
| Attack 1 | 1/1 | 26.5 | 1 |
| Attack 2 | 0/1 | 37.8 | 1 |
| Attack 3 | 1/1 | 53.5 | 1 |
| Attack 4 | 1/1 | 35.8 | 1 |
| Attack 5 | 1/1 | 307.6 | 1 |
| Attack 6 | 0/1 | 37.4 | 1 |

The experiments were conducted using a server we purchased for this research. The focuses were on adversarial attacks, backdoor attacks, and data poisoning on existing ML algorithms. Experiments show that adversarial attacks cause the ML model to malfunction. Backdoor attacks allow an attacker access to the system without being detected. Poisoning attacks are used to compromise the model into failing at its given task.

## 5. Future Directions

Research surrounding exploitation of cyber-human and cyber-physical systems is limited across cybersecurity communities. This research is a first step into analysis of biometric-based cyber assessment that relies on human physical characteristics as input and network performance-based cyber-assessment that analyzes network data for understanding intrusion detection. More work needs to be done in both areas to determine how dependent they are on each other. Future work in this area could examine how biometric data can be manipulated and changed in the network for allowing hackers to enter a system. Additionally, more work needs to be done on understanding the minimal image qualities for performing optimal biometric recognition.


ya.lewis@CAPC-UB18-G8-5: ~/counterfit
File Edit View Search Terminal Help

```
[+] Attack completed 53f06c04 (HopSkipJump)

creditfraud>53f06c04> use HopSkipJump

[+] New HopSkipJump (1dbb254d) created
[+] Using 1dbb254d

creditfraud>1dbb254d> set --max_iter=150 --max_eval=500 --init_size=100 --batch_
size=5

 Parameter (type)         | Default     | Current     | New
--------------------------+-------------+-------------+-----
 Algorithm Parameters     |             |             |
 -------------------      | --          | --          | --
 targeted (bool)          | False       | False       |
 norm (int)               | 2           | 2           |
 max_iter (int)           | 50          | 50          | 150
 max_eval (int)           | 10000       | 10000       | 500
 init_eval (int)          | 100         | 100         |
 init_size (int)          | 100         | 100         |
 curr_iter (int)          | 0           | 0           |
 batch_size (int)         | 64          | 64          | 5
 verbose (bool)           | False       | False       |
 clip_values (tuple)      | (0.0, 1.0)  | (0.0, 1.0)  |
                          |             |             |
 Attack Options           |             |             |
 -------------------      | --          | --          | --
 sample_index (int or expr)| 0          | 0           |
 logger (str)             | default     | default     |

creditfraud>1dbb254d> run

[-] Running attack HopSkipJump with id 1dbb254d on creditfraud)

[-] Preparing attack...
[-] Running attack...
 Success | Elapsed time | Total Queries
---------+--------------+-----------------------------
 1/1     | 5.3          | 73021 (13870.0 query/sec)

 Adversarial Input
 [4462.00 -2.30 1.76 -0.36 2.33 -0.82 -0.08 0.56 -0.40 -0.24 -1.53 2.03
 -6.56 0.02 -1.47 -0.70 -2.28 -4.78 -2.51 -1.33 -0.43 -0.29 -0.93 0.17
```

**Figure 5: A Screenshot of HopSkipJump Attack with Counterfeit.**

The team wishes to continue conducting research in discovering bias in AI/ML systems, identifying adversarial attacks on AI/ML algorithms, and applying quantum computing in adverbial attacks on AI/ML algorithms.

# Pilot Project 4 AI/ML Systems Engineering Workbench

Kofi Nyarko[1*], Peter Taiwo[1], Michaela Amoo[2], with ARLIS Lead Craig Lawrence[4]

**Program Objective**

The primary objective of this project is to integrate multiple AI/ML service providers across a cluster of cloud platform computers and provide easy-to-use access to cognitive computing APIs delivering computer vision, natural language processing, speech and autoML platforms and services for data scientists and software developers.

**Keywords**

Machine Learning; ML Framework; AI services; RESTful API

[1] Center for Equitable Artificial Intelligence & Machine Learning (CEAMLS), Morgan State University, Baltimore MD 21251

[2] Department of Electrical Engineering and Computer Science, Howard University, Washington DC 20059

[44] Applied Research Laboratory for Intelligence and Security (ARLIS), University of Maryland, College Park, Maryland 20742

## 1. Project Overview

In this project, we design a unified machine learning workbench that targets three classes of end-users within the community: 1) Developers, 2) Data Scientists, and 3) DevOps. Workbench consists of multiple cloud based virtual machines (VMs) that supports ML services, platforms and infrastructure from various providers.

### 1.1 Overarching Goals of Pilot

This pilot project aims at developing an AI/ML Systems Engineering Workbench consisting of a common framework for the user community in support of the ARLIS deployment of cyber-infrastructure, and support previously developed virtualized desktop configurations developed by DARPA, JAIC and others as needed.

### 1.2 Lines of Effort

The Workbench is implemented in three layers: **Services**, **Platform**, and **Infrastructure** layers. Morgan State University focused on implementing the **Services** and **Platform** layers while Howard University focused on the **Infrastructure** layer.

Under the **Services** layer, MSU conducts research into cloud-based AI services, configures and deploys VMs based on activities performed by software development end-users, which involve the integration of cognitive computing services across multiple providers.

MSU also researched multiple ML **Platforms**, configured and deployed VMs based on those platforms that service data science end-users.

Howard University has not submitted any details on their work on the **Infrastructure** layer for this final project report.

### 1.3 Why It Matters

The fusion of multiple AI/ML services and platforms across multiple cloud providers in a single workstation can more effectively support the intelligence community with computation and data services at scale. This also enables integration of the best-of-breed AI/ML toolkits with other software and systems engineering toolkits, and creation of new systems that can be designed, tested, and transitioned with greater reliability.

## 2. Background and Related Work

In recent years, AI has continued to drive innovativeness and provide fast and efficient means into solving many complex tasks across different industrial sectors and organizations. It has thus far helped from its use in making medical treatment decisions faster by analyzing medical data, to helping derive usable intelligence from analyzing large amounts of video footage in hours or days instead of months to support criminal investigations. Despite all these feats, the problem of integrating and consolidating multiple AI/ML tools provided by multiple cloud providers, persists.

Cloud providers such as Google, AWS, Azure and IBM provide numerous AI/ML services all of which can fall within the three layers of interests in this project – Services layer which is rendered as AIaaS (AI as a Service), Platform layer rendered as MLaaS (ML as a Service), and the infrastructure layer as IaaS (Infrastructure as a Service) (Frederic, 2016; "Image type detection," 2022). With AIaaS, cloud service providers allow individuals and software developers who want to embed an AI service into their work to leverage several cognitive computing APIs for vision, speech, NLP and AutoML services.

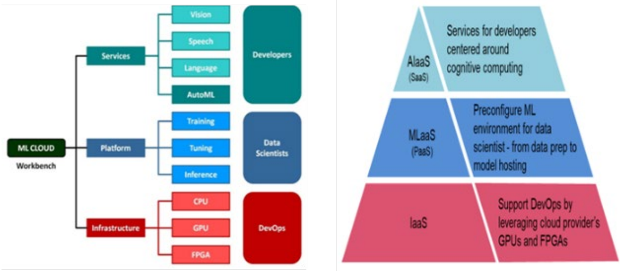

ML CLOUD
Workbench
Services
Platform
Infrastructure
Vision
Speech
Language
AutoML
Training
Tuning
Inference
CPU
GPU
FPGA
Developers
Data Scientists
DevOps
AIaaS (SaaS)
Services for developers centered around cognitive computing
MLaaS (PaaS)
Preconfigure ML environment for data scientist - from data prep to model hosting
IaaS
Support DevOps by leveraging cloud provider's GPUs and FPGAs

Fig. 1: AI/ML Workbench System Architecture

At the MLaaS layer, data scientists have access to cloud platforms that helps with ML training, tuning and inference jobs. Both the AIaaS and PaaS sit on the infrastructure layer, where we seek to support DevOps engineers to efficiently host AI/ML application on clusters of CPs, GPUs and FPGAs at scale by leveraging available infrastructure services hosted by the cloud service providers.

## 3. Methods, Results and Discussion

The cloud-based AI/ML Systems Engineering Workbench is developed to provide integrated user access at the AIaaS and MLaaS layers via a web interface (see Fig 2), this enables end-user an easy access to cloud-based REST API interfaces to ML services, platforms and infrastructure from various providers.

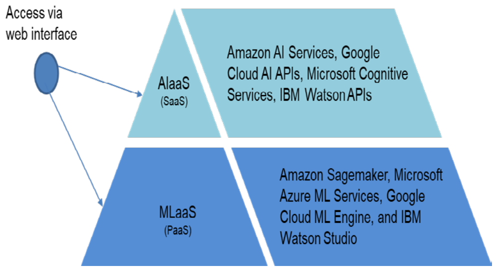

Access via web interface
AIaaS (SaaS)
Amazon AI Services, Google Cloud AI APIs, Microsoft Cognitive Services, IBM Watson APIs
MLaaS (PaaS)
Amazon Sagemaker, Microsoft Azure ML Services, Google Cloud ML Engine, and IBM Watson Studio

Fig 2: Workbench User-access structure to Cloud APIs

Some APIs that have been integrated into the workbench at the AIaaS layer are from providers' service stacks such as Amazon AI Services, Google Cloud AI APIs, Microsoft Cognitive Services, and IBM Watson APIs. The web user interface access to the MLaaS layer also provides a simplified model training and testing across multiple platforms.

A platform-based workbench architecture is first considered, where services are accessed and organized based on individual cloud platforms. To allow performance benchmarking across multiple providers, a more concise workbench architecture is envisioned that is based on machine learning tasks. In this case, a data window exists on the UI with various parameters to choose a specific task to be performed on a target data or dataset.

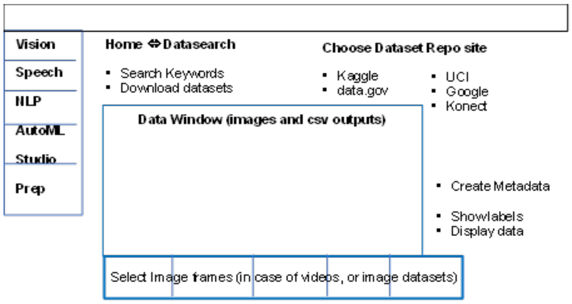


Fig 3: Web User Interface Sketch

Optional parameters include selection of provider's platform on which a task is chosen to be run. For benchmarking purposes, multiple providers can be selected for a single task. Responses from various AI services APIs on the cloud platforms considered are in form of a serialized JSON format with key-value pairs. Python scripts are created to run at the backend of the workbench to process responses from these RESTful API calls in order to meaningfully display test results and potentially create performance metrics comparison among similar platform services.

### 3.1 Software Architecture

Figure 4 describes the software architecture adopted for the workbench. The AI Services Integrator/Advisor provides off-the-shelf access to vision, speech and NLP inference services while providing some form of benchmarking that enables users to select a provider with the best solution to their tasks. For data scientists that want to provide their own datasets to train or fine-tune a model, or perform transfer learning jobs, the workbench provides access to multiple cloud-based ML frameworks.

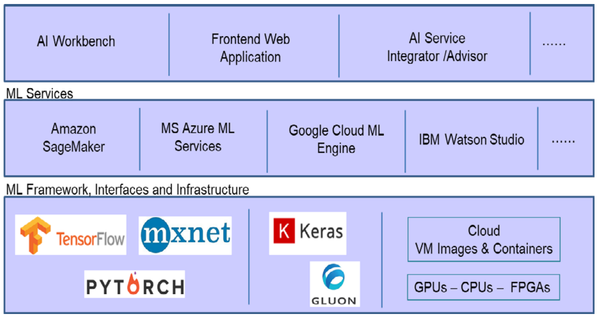


Fig 4: Workbench Software Architecture

The web-based workbench interface is created using the Python Django framework and designed to be hosted on a cloud VM. Access keys to Cloud ML services are managed on this VM, having no storage or ML processes running on it. Several ML frameworks which are efficiently interfaced with cloud-based hardware infrastructures have been developed by cloud service providers. These frameworks are being wrapped in VM images and containers such that it can be portably moved across different CPUs, GPUs, FPGAs and across different operating systems. Common ML frameworks include TensorFlow, MXNET, Pytorch, Keras, Gluon, etc. The workbench provides a platform for end-users to interface with these cloud-based frameworks via providers ML services that includes AWS Sagemakerinfrastru, MS Azure ML services, Googles Cloud ML Engine and IBM Watson Studio.

The ML services layer of the system architecture houses RESTful API call functions to multiple cloud platforms, resulting in creation of Jupyter notebook instances that get embedded in the user interface. The MLEngine app is created within the Django framework to provide metrics obtained from training jobs that have been run across multiple platforms. Once the best trained model has been determined, it can be hosted on a predetermined endpoint location for inference calls accessible from the user interface.

Remote cloud storage is used to store dataset from multiple repositories (Kaggle, Konect, UCI), and is integrated into the local file system of the workbench. An AWS S3 storage system is used by mounting an S3 bucket as a filesystem on the Linux machine hosting the application, by using the FUSE-based S3FS.

The workbench is first developed as a single user application, and in order to allow for multipoint testing as we developed, the workbench is redesigned for a multi-user access. A registration app component is designed to authorize, register, create user accounts and authenticate users. Upon login by concurrent users, database profiles are dynamically created to separate and manage multi-user dataset storage directories in the S3 bucket, and multi-user UI templates rendered from the workbench. User authorization is done by checking the email domain provided and then triggering a verification process involving a passcode generation method. The web-user session ID is used as the unique identifier during user the registration process.

## 3.2 Services Layer (AiaaS)

A survey of off-the-shelf AI services available on cloud service providers is conducted with the goal of integrating a number of these into a unified framework on the workbench. A non-exhaustive list of these services is provided below. This survey is conducted under the following categories: Vision, Speech, NLP and AutoML. Providers considered are AWS, Azure, Google, and IBM. Initial focus is on the vision APIs across all platforms and tests were performed in the following categories.

- Object and Scene Detection (for Images and Videos)
- Face detection (for Images and Videos)
- Text detection (for Images and Videos)

AWS AI services

**Rekognition:** Labels, Custom Labels, Text Detection, Face Detection and Analysis
**Polly:** Text2speech, Voice and Language Selection, Synched Speech, Custom Lexicon.
**Comprehend:** Sentiment Analysis, Language Identification, Topic modeling.
**AutoGluon** (AutoML)

Azure Cognitive Services

**Vision:** Computer Vision (Analyze images and video), Custom Vision, Face Detection, Form Recognizer, Video Indexer
**Speech:** Speech2text, Text2speech, Speech Translation, Speaker recognition
**NLP:** Text Analytics, Translator, Language Understanding, Immersive Reader, Anomaly Detector, Content Moderator, Metrics Advisor, Personalizer

Google AI

**Vision AI:** Custom and pretrained models to detect face, emotion, text.
**Speech2text:** Speech recognition and transcription
**Text2speech:** Speech synthesis in 220+ voices and 40+ languages
**Cloud Translation:** Language detection, translation and glossary support
**Cloud Natural Language:** Sentiment analysis and translation of unstructured text
**VideoAI:** Video classification and recognition
**AutoML**

IBM Watson AI

**Visual Recognition:** Analyze image contents and provide insights
**Speech:** Speech2text, Text2speech, Pronounce languages and dialects
**Language:** Language Translator and Natural Language Classifier
**Empathy:** Personality Insights and Tone Analyzer

### WB Data-search App

The data-search app is designed to provide a federated search service with data inspection capabilities. Using the

Kaggle dataset search example, the workbench takes keywords entered by a user, sends an API call to Kaggle and download the dataset into the user directory in the WB S3 bucket. Acquired datasets are stored in a predefined directory format and a metadata file is generated for each dataset. A public API is provided by Kaggle which makes it possible to query the repository directly.

The UCI ML repository is integrated to the workbench with a different method. It has over 588 public datasets that are constantly being updated. This repository, however, has no public API for querying and acquiring datasets. A search of any dataset on the UCI portal does a google search with a backlink. This URL that links back to the UCI page is scraped via a python library, Beautiful Soup.

The Konect dataset repository houses 1,326 datasets in 24 categories. It does not provide a public Rest API as well. KONECT is however web-scraped into the workbench using Lxml parser.

### WB Vision App

The Vision AI app component is set-up to provide a user with access to perform training and inference jobs with saved images, recorded and live videos. In this section of the application, users can upload multiple images to storage, or use items from datasets acquired in the data-search app. Inference tasks are performed using libraries accessed from multiple platforms. Integrated cloud AI platforms are AWS, Azure, GCP and IBM. YOLOv3 is initially installed locally to aid in setting ground truth.

Performance is measured in terms of the number of objects detected and accuracy in localization of detected objects. A more accurate ground truth setting is done using manual counts of objects and manually drawing object bounding boxes on test images, with the aid of a Python script. As a demo case, we use the dataset provided by the CrowdHuman project to evaluate performance of each platform on person detection and localization (Shao, Shuai, et. al 2018).

The WB is designed to ensure that only image or video datasets are available for vision related training and inference tasks. Likewise, only csv type datasets are available on the dim1 app, which handles training and inference with text-based datasets.

### WB NLP App

The NLP app is created to build capability for NLP performance modeling across service providers in a similar way to the vision app. It provides topic detection, topic modelling or topic analysis service, which is an ML technique that organizes large collections of text data from a variety of documents. This is done by assigning categories according to each unique text in the document. A use case of NLP Sentiment Analysis is demonstrated on the workbench by comparing three website review datasets for the level of efficiency in terms of detection across the platforms. Services are compared using F-1 scores. Sentiment analysis on AWS Comprehend and Azure Cognitive Service are evaluated. Documents are classified based on words that appear in their texts. Three sentiment labels “negative”, “positive” and “neutral” are used in this process. Topic detection with NLTK and LDA using the sample ‘movie database’ file for sample analysis and baselining is also considered. This required test data to be cleaned by filtering all stop words and then tokenized.

## 3.3 Platform Layer (MlaaS)

The MLEngine app is developed for the platform layer with capabilities to start multiple kernels for Jupyter notebook instances locally on the workbench, and to also instantiate remote notebook instances on multiple cloud platforms. Remote Jupyter notebook setup from the workbench was implemented for AWS Sagemaker (Miller, Ron, 2017). These notebooks are rendered to the user as new tabs on the workbench user interface. The WB keeps track of active notebook status, URL and token in case reconnection to the instances are needed.

When using the Jupyter server on the host machine, the s3fs mapping provides access to the user directory on the connected S3 bucket. This provides access to dataset acquired from the data-search app storage.

The user also has the option to upload their own dataset to the workbench. Notebooks created with this method reside in the same user S3 directory location as the datasets acquired by the user. Creating notebook instances with Sagemaker also allows mapping with Github repositories to have access to premade ipynb scripts, and then an S3 client method is used to access datasets in S3 bucket. From the WB backend, TCP port used for each instance can be monitored and the time it takes the instance to transition from pending state to in-

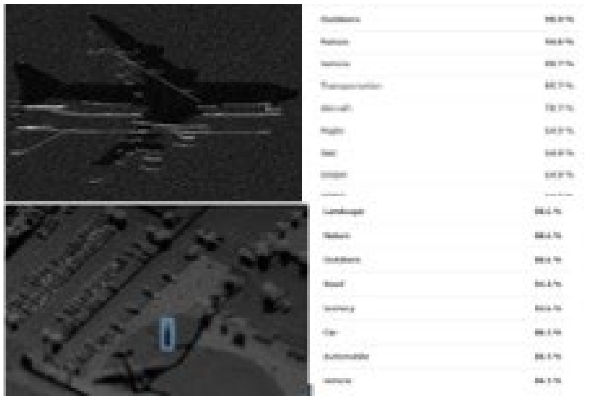


Fig 5: Image Classification on SAR Images

service state, is used to compute set-up time. This is a useful metric for the user to determine what provider, zone or data center region is preferred for their ML tasks.

Here we discuss some of the results of demo cases performed on the workbench with the vison, NLP and MLEngine apps.

## Vision and NLP cases

Two apps that are our focus on the AIaaS layer are the vision app and the NLP app. On the vison app, image classification and object detection tasks are tested using regular images and synthetic aperture radar images. On the NLP app, sentiment analysis and topic detection tasks are tested.

### *SAR Images*

Figure 5 shows the result of object detection and image classification inference on Aerial and Sar Images Analysis using AWS Rekognition.

Object detection confidence score on SAR images is much lower compared to regular images. However, with transfer learning, which can be done on the workbench on the MLaaS layer, a new model trained on SAR image datasets can improve object detection results.

### *Performance measurements across platforms*

In order to demonstrate the use of the workbench for performance measurement and benchmarking across multiple platforms, we use the CrowdHuman dataset to observe how many persons can each of the detect-persons methods of each platform can detect from crowd images.

Ground truth image dataset and human detection metadata are prepared for bench marking multiple cloud platform algorithms for object detection. Ground truth images were selected from the CrowdHuman project datasets. From the preliminary results of person detection using YOLOv3 (hosted locally on the workbench) and AWS Rekognition, images with high detection variance were selected. Final selection came from categorizing the selected images into, lightly dense, moderately dense, and highly dense images. We also ensure that images with no object classification bias were selected.

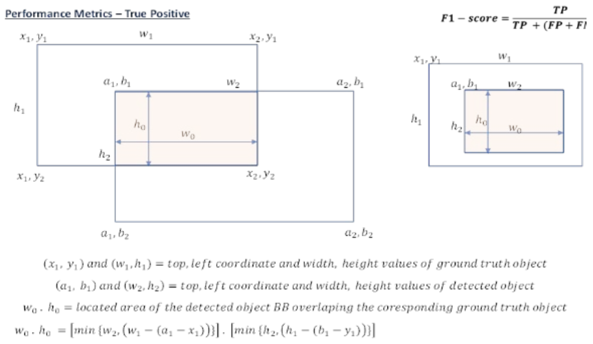

Performance Metrics – True Positive
$F1-score = \frac{TP}{TP+(FP+F}$
$x_1, y_1$
$w_1$
$x_2, y_1$
$a_1, b_1$
$w_2$
$a_2, b_1$
$h_1$
$h_0$
$w_0$
$h_2$
$x_1, y_2$
$x_2, y_2$
$a_1, b_2$
$a_2, b_2$
$(x_1, y_1)$ and $(w_1, h_1)$ = top, left coordinate and width, height values of ground truth object
$(a_1, b_1)$ and $(w_2, h_2)$ = top, left coordinate and width, height values of detected object
$w_0 . h_0$ = located area of the detected object BB overlaping the coresponding ground truth object
$w_0 . h_0 = [min\{w_2, (w_1 - (a_1 - x_1))\}] . [min\{h_2, (h_1 - (b_1 - y_1))\}]$

Fig 6: Determination of True Positive results using ground truth and detected object overlaps

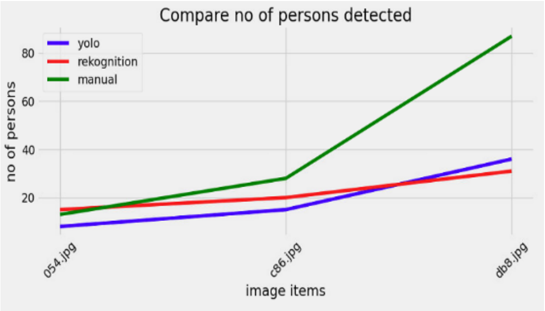


Fig 7: Persons Detection (counts) Results

Annotation and bounding-box information collection were manually done on each image with the aid of a Python script and the OpenCV module. The F1 score is used to evaluate performance metrics in order to effectively account for errors due to false positives and false negatives.

The following procedure is used to determine the True Positive metric for each image:

- Count the number of objects detected, whose BB has a greater that 50% area overlap on the corresponding ground truth image. (i.e., $w_0 . h_0 \geq (w_2 . h_2)/2$ )

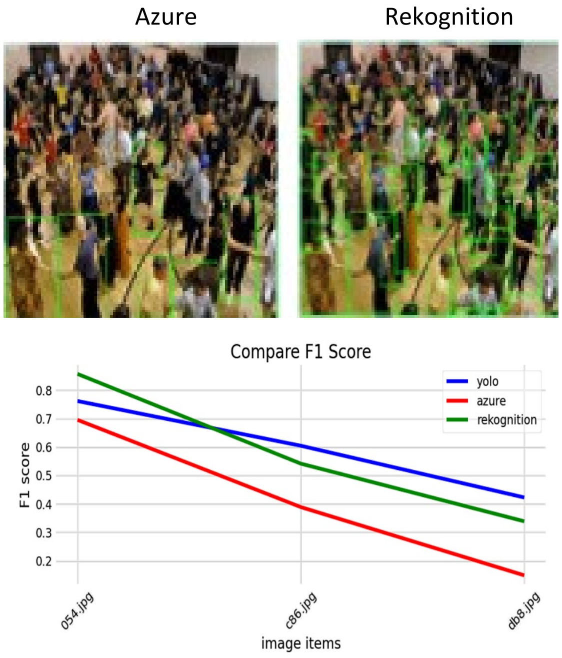


Fig 8: Persons Detection F1 Score Results

| Easy Document | Hard Document |
|---|---|
| Yelp and Amazon Dataset | Reddit Dataset |
| Simple Sentences. | Lengthy and difficult to understand document |
| UCI repository | Kaggle repository |

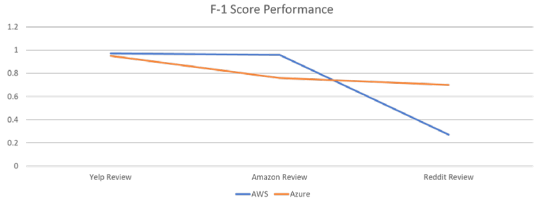
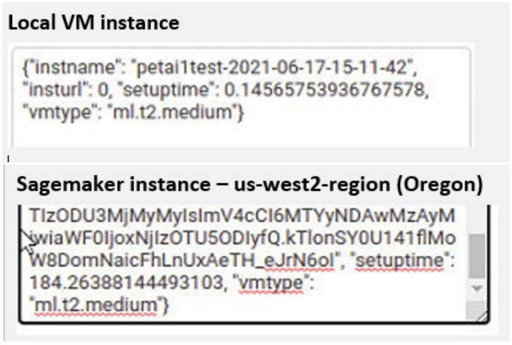


Fig 9: Sentiment Analysis F1-Score Results


**Local VM instance**

```
{"instname": "petai1test-2021-06-17-15-11-42",
"insturl": 0, "setuptime": 0.14565753936767578,
"vmtype": "ml.t2.medium"}
```

**Sagemaker instance – us-west2-region (Oregon)**

```
TIzODU3MjMyMyIsImV4cCI6MTYyNDAwMzAyM
iwiaWF0IjoxNjIzOTU5ODIyfQ.kTlonSY0U141flMo
W8DomNaicFhLnUxAeTH_eJrN6ol", "setuptime":
184.26388144493103, "vmtype":
"ml.t2.medium"}
```

Fig 10: Sagemaker Notebook versus Local Jupyter Notebook Set-up Time

- For a current count, if multiple objects satisfy this condition select the object with the highest area overlap for that count.

False Negative is computed as the remaining number of objects in the ground truth metadata, which could not be mapped to any detected object after determining a score for True Positive. And False Positive is the remaining number of objects in the detection metadata, which could not be mapped to any ground truth object after determining a score for True Positive. Although Rekognition seems to detect more persons than YOLOv3 in this example there is at least one extra bounding box with no person in it.

The following plot describes the varying number of persons detected using the selected lightly dense, moderately dense, and highly dense images.

After integrating MS Azure to the AIaaS layer of the workbench we compare the CrowdHuman *detect_person* results for MS Azure and AWS Rekognition. Result is shown in Figure 8.

Results indicate that Rekognition consistently detects more persons in a human crowd image than Azure, regardless of the density of the image. The local YOLOv3 model however tends to outperform both Rekognition and Azure with high density images. Recognition detects more persons than YOLOv3 with low density images.

### *NLP*

Sentiment Analysis performance measurement across the NLP service platforms integrated to the workbench is also computed and compared using the F1 score. Test documents are classified into easy and hard, and results from Azure and AWS comprehend are compared as shown in Fig 9.

### *Integrated ML Training and Transfer Learning*

Performance evaluation and benchmarking performed on the MLaaS layer using the MLEngine app was done by measuring cumulative response time for each cloud service platform. This is the time taken between a call to start a Jupyter notebook instance and when a URL for the notebook instance is returned as a response.

Response time score (setup time) is determined by monitoring the status message on the TCP port used for each instance. This is the time it takes the instance to transition from “pending” state to “in-service” state.

Figure 10 shows a comparison of the setup time between a notebook instance initiated for the local machine and a remote notebook instance initiated to be run from AWS Sagemaker. The local Jupyter notebook is initiated in 0.146 sec while it takes 184.26 secs to initiate a Sagemaker Jupyter notebook. It should be noted that set-up time could vary widely within the same platform depending on what data center zone the instance call is sent to.

### Transfer Learning

We test the transfer learning functionality on the workbench by using the MLEngine app to set-up transfer learning training session using a cloud provider MLaaS platform. A pretrained ML model acquired via the workbench is used to perform transfer learning jobs using the user's dataset. The resulting model is downloaded to the workbench and deployed. Tests are carried out to ensure that inference can be made with the downloaded model.

A demo case was implemented with AWS Sagemaker, using the Sagemaker estimator framework. We explored the model files downloaded from AWS Sagemaker, which was found to be based on the MXNET framework (specifically the Hybrid framework, which provides the capability of using the model with imperative and symbolic programing). Though the .params and *symbol.json* files which define the MXNET model architecture are present in the Sagemaker model, some work-around was performed to create the missing labels file before the model was useable on the workbench.

Using proprietary estimator framework for this transfer learning functionality will require that we create a different estimator framework for each cloud platform to be integrated into the workbench. These could hinder further work like ensemble learning, feature extraction and label encoding. This can be solved by forcing downloaded models to use open-source estimators on the WB. Open-source ML estimator framework like TensorFlow and PyTorch are further considered to demonstrate ML training and transfer learning functionality on the workbench.

It is observed that training, retraining and fine-tuning using cloud provider's ML platform directly can be quite expensive. ML platforms like Sagemaker are workbenches in their basic form. The use of these secondary workbenches can be bypassed by wrapping our training jobs in containers and run these directly on cloud CPU, GPU and FPGA compute instances or VMs.

## 3.4 Infrastructure layer (Not reported)

Howard University did not submit details of their work on this effort.

# 4. Future Directions

Following the completion of contract work, the researchers at Morgan State will:

- Complete revamping of UI
- Complete support for multi-users
- Complete the implementation of transfer learning with Azure
- Tasks beyond pilot effort
- Apply the workbench to projects under the new *Center for Equitable AI and ML* that relate to the quantification of algorithmic bias
- Develop additional use cases to support current lab research tasks
- Integrate into curriculum at Morgan (EEGR 483 – Intro to Machine Learning, EEGR 565 – Machine Learning Applications)
- Make it available for capstone projects (Fall 2021/Spring 2022)

# Pilot Project #5: Chatbot Testbed

Amit Arora[1*], Victor McCrary[1], Gloria Washigton[2], Onyema Osuagwu[3], Joy Williams[3], Omoshalewa Olukotun[3], with ARLIS Leads Michelle Morrison[4], C. Anton Rytting[4], and Valerie Novak[4]

**Program Objective**

Collaborative work to respond to difficult intelligence and security problems

**Keywords**

Chatbot; Sentiment Analysis; Social Media

[1] University of the District of Columbia
[2] Howard University
[3] Morgan State University
[4] Applied Research Laboratory for Intelligence and Security (ARLIS), University of Maryland, College Park, Maryland 20742
***Corresponding author:** amit.arora@udc.edu

## 1. Project Overview

Morgan State University (MSU), Howard University (HU), and University of the District of Columbia (UDC) worked collaboratively to develop and refine the ARLIS Pilot Project #5 - "ChatBot Testbed", with UDC as Pilot Project Lead and Coordinator.

### 1.1 Overarching Goals of Pilot

The objectives are to 1) **survey** the existing state-of-the-art multilingual ChatBot tools, 2) **develop and test** this ChatBot Testbed, and 3) **create documentation** and materials so that this toolbox can be used by a variety of users with differing goals. The ChatBot Testbed will be created by integrating existing open-source and commercial tools to effectively create a solution that is usable for understanding problems in influence, information operations, and insider threat.

The entire team will allow undergraduate students the opportunity to learn algorithmic techniques in multilingual translation and important functionality needed for inclusion in these tools. Additionally, all undergraduate students will have the opportunity to submit a poster based off this work to conferences such as the Richard Tapia Celebration of Diversity in Computing, ACM Student Research Competition, Northeast Decision Sciences Annual Conference, among others.

### 1.2 Lines of Effort

In addition to building the Chatbot, the entire research team worked to examine subtasks related to **insider threat**, **influence**, and **information operations**. Each research team examined these research areas separately and worked together collaboratively to merge their results into a final solution.  Each subtask is described below.

#### LOE 5.1: Detecting Insider Threats (UDC)

UDC explored the insider threat of malicious insiders to organizations in chat conversations. The team investigated and strived to understand a) the expression of emotions/affective states, b) the role and influence of rhetoric, and c) external/internal threats across nations and/or global organizations, on social media. The team explored coordinated efforts to social engineer information from participants in chat conversations. The feasibility of employing Multilingual Sentiment Analysis (MSA) on the vast amount of text data available on social media was also explored.

#### LOE 5.2: Chatbots for Influence (Howard)

The Howard University team focused on identifying the state-of-the-art of determining influence in Chatbot conversations.  Initially the team started with determining influence in a conversation by examining the effects of participants responses on who controls a topic conversation (Nguyen et al, 2014). Howard worked on performed user experience testing of Chatbot software to determine just how much language variations can affect Chatbot translations and humans using these systems. Additionally, the team explored the social dimension of language and developed automated tests to model the language in chat conversations (Nguyen et al, 2016).

#### LOE 5.3: Chatbots for Sabotage and Subversion (Morgan)

The Morgan State University Team research focused on the survey and development of multilingual Chatbot technology for sabotage and subversion. The research goal centered on producing coordinated AI campaigns to obtain information from multiple targets, reconstituting the acquired data, and leveraging this knowledge to infiltrate and subvert additional digital assets, therefore compromising an adversary's capabilities.

Our efforts focused on surveying existing state-of-the-art chatbot technology, e.g., Spacy, GPT-3, etc., for integration on a secure cloud resource allowing for testbed operations for our stakeholders.

Morgan State University's (MSU) strengths in influence, information operations, and insider threat are present in the selected roles and present a cohesive research plan for this topic. Finally, our work will involve the adaptation and integration of what we identify as best-of-breed solutions that will comprise the library of tools per the program's needs.

### 1.3 Why It Matters

Advances in Natural Language Processing (NLP) and Machine Learning (ML) techniques have led to chatbots (also known as conversational agents) becoming capable of extracting meaningful information regarding cybersecurity threats (Franco et al., 2020) on social media. Rapid deployment of internet coupled with digitalization of a globalized economy has produced a vast amount of textual data through social media. Governments, businesses, and political parties depend on the sentiments and opinions expressed on social media sites to gauge the mood of public in real time (Thapa, 2022). This is also a vital source of information related to security threats to a nation and business organizations. Consequently, it becomes imperative for intelligence and security communities to delve deeper in the area of cybersecurity in order to protect national security and economic interests.

Chatbot technologies provide a low-cost asymmetric avenue of attack, infiltration, and data exfiltration for several bad actors and adversarial Nation-States. We continue to bear the brunt of more frequent and sophisticated attacks on our defense-related assets and infrastructure, leaving a large portion of our attack surface under-researched and not fully hardened against such actions. Continued research in this area provides insights into methods of creation, coordination, and execution used in these attacks supporting the development of countermeasures.

## 2. Background and Related Work

A chatbot is an application that uses artificial intelligence (AI) to communicate. AI is the automation of intelligent behavior which allows machines to simulate anthropomorphic conversations. Chatbots have been programmed to use artificial intelligence and concepts such as Natural Language Processing (NLP), Artificial Intelligence Markup Language (AIML), Pattern Matching, Chat Script, and Natural Language Understanding (NLU) to communicate with users, analyze the conversation and use the extracted data for various purposes such as marketing, personal content, to target specific groups, etc.

### Classifying Chatbots

Some categories under which chatbots can be classified include the knowledge domain, the service provided, the goals, the input processing, and response generation method, the human-aid, and the build method.

**Knowledge domain classification** considers the knowledge a chatbot can access as well as the amount of data it is trained on. Closed domain chatbots are focused on a certain knowledge subject and may fail to answer other questions, but open domain chatbots can talk about various topics and respond effectively (Adamopoulou & Moussiades, 2020). Conversely, the sentimental proximity of the chatbot to the user, the quantity of intimate connection, and chatbot performance are factors in the classification of chatbots based on the service provided.

**Interpersonal chatbots** are in the communication area and offer services such as restaurant reservations, flight reservations, and FAQs. They gather information and pass it on to the user, but they are not the user's companions. They are permitted to have a personality, be nice, and recall information about the user, however, they are not required or expected to do so (Adamopoulou & Moussiades, 2020). Adamopoulou & Moussiades (2020, p. 373-383) states that "Intrapersonal chatbots exist within the personal domain of the user, such as chat apps like Messenger, Slack, and WhatsApp. They are companions to the user and understand the user like a human does. Inter-agent chatbots become omnipresent while all chatbots will require some inter-chatbot communication possibilities. The need for protocols for inter-chatbot communication has already emerged. Alexa-Cortana integration is an example of inter-agent communication".

**Informative chatbots**, such as FAQ chatbots, are designed to offer the user information that has been stored in advance or is available from a fixed source. The manner of processing inputs and creating responses is taken into consideration when classifying based on input processing and response generation. The relevant replies are generated using one of three models: rule-based, retrieval-based, and generative.

Another classification for chatbots is based on how much **human-aid** is included in its components. Human computation is used in at least one element of a human-aid chatbot. To address the gaps produced by the constraints of completely automated chatbots, crowd workers, freelancers, or full-time employees can incorporate their intelligence in the chatbot logic. Adamopoulou, & Moussiades (2020, p. 373-383) examine the main classification of chatbots as per the development platform permissions, where the authors defined 'development platforms' as "…open source, such as RASA, or can be of proprietary code such as development platforms typically offered by large companies such as Google or IBM."

**Anthropomorphic characteristics:** Two of the main categories that chatbots may fall into as it relates to their anthropomorphic characteristics are **Error-free** and **Clarification** chatbots. Anthropomorphism is "the attribution of human characteristics or traits to nonhuman agents" (Epley et al. 2007, p. 865). Anthropomorphic chatbots are perceived to be more palatable to consumers since consumers perceive the chatbots to be humanlike, rather than how firms design chatbots as humanlike (Blut, Wang, Wünderlich, & Brock, 2021). An Error-free chatbot can be defined as a hypothetically flawless chatbot while a clarification chatbot has difficulties inferring meaning and therefore asks for clarification from the user. Clarification chatbots are seen as more anthropomorphic since clarification by the chatbot is seen as giving care and attention to the needs of the customer. There is no current commercial application of the error-free chatbot, however, clarification chatbots are currently being used by companies such as Amazon, Walmart, T-Mobile, Bank of America, and Apple, as first contact customer service representatives.

### Sentiment Analysis

Development of web services has transformed the way people communicate online. Many web services enable users to interact in real-time with one another. The most popular type of communication is social networking, which is based on a microblog format (Hernández et al. 2016; Chatzakou et al. 2013) and allows simple text postings, emojis, downloading images and files, and interactive user-to-user conversation via chat messaging. Many of the content in social networking sites reflects the ideas and opinions of users on a variety of topics. Recent investigations have reported that analyzing users' feelings, or sentiments, in social networks is effective for forecasting and monitoring a variety of events, including market trends, political opinions, and epidemic spread (Hernández et al. 2016; Achrekar et al. 2011). Working with sentiment analysis in social networking has a number of advantages, including the ability to analyze large amounts of data quickly.

However, there has been minimal effort done to assess sentiments in the context of information security, particularly to detect probable threats. Based on Twitter, a web service that permits text entries generally referred to as tweets, probabilistic topic modeling has been proposed for tracing information security-related incidents. However, such methodology relies solely on idea evaluation and user influence to uncover consistency and quacking of top trends from Twitter data. Previous research has shown that the best practice to combat threats in cyber security is to develop strategies that are complementary to each specific threat (Pinard, 2019).

# 3. Methods and Results

## 3.1 Detecting Insider Threats

In this project, we developed a chatbot on Bot Libre open-source platform and deployed it on Twitter. The project also focuses on sentiment analysis of tweets, which are 140-character postings with the option to include visuals. Twitter presently has 288 million active users, providing a chance to gather data in near-real time. We analyze a sample of tweets and perform sentiment analysis using coding tools to correlate the overall sentiment of regular users towards a certain event. The strategy seeks to link the reaction of specific groups of hacking activists with the mood of regular Twitter users in the context of cyber-attacks through the chatbot development / creation. This project highlights future threats and vulnerabilities and possible strategies such as User Behavioral Analytics and further developments in artificial intelligence to combat them as new threats and vulnerabilities arise in the cyber security space.

This section focuses on the two main aspects of the project: a) development and deployment of a conversational chatbot on a social media site; and b) conducting sentiment analysis on the vast amount of textual data from a social media site.

## Development and Deployment of Chatbot

Initially, the project team focused on building a chatbot on SAP open-source platform. However, it is hard to use SAP Conversational AI chatbot outside SAP S/4 Hana cloud. After considering other open-source platforms like Botpress, our conversational chatbot was developed on Bot Libre, an open source end-to-end chatbot-building platform. It can be used to build, train, connect, and monitor the chatbot on a social media site. Bot Libre chatbot uses both text and images and is categorized as Communication Channels chatbot (Adamopoulou and Moussiades, 2020; Pinard 2019). This platform allows the chatbot to be deployed on various social media sites like Twitter, WhatsApp, Facebook, Discord, Kik, etc. The language modeling, which is a part of personalizing how the bot communicates with specific users allows the bot to interact with users in multiple languages, can be tailored to include English, French, Russian Spanish, Italian, Japanese, among other languages.

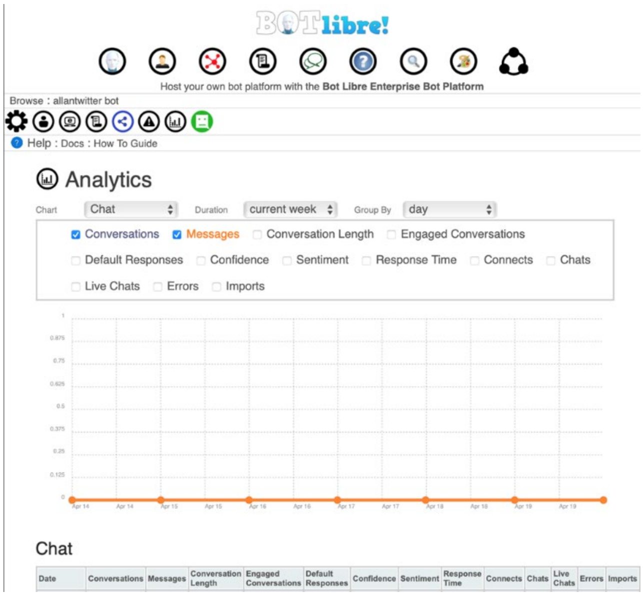
Currently, our chatbot can converse in English language only on Twitter platform. The automation feature allows the bot to tweet over an extensive period. For example, in the month of March, the chatbot was programmed to tweet “Happy Women’s History Month” every 24 hours. It utilizes the ‘conversational feature’ by initiating and maintaining conversations with other users of Twitter. Its ‘Informational Effect’ and ‘Data Effect’ are highlighted by its ability to collect data from conversations it has with other users as well as extract information from the platform based on key terms searched. For example, the chatbot can search the key terms “Putin”, “Nuclear Weapon” and “Russia” and extract all tweets associated with these key terms. The goal of the chatbot is to communicate and extract information/intelligence from users on Twitter which can be used by intelligence and security communities. Any keyword that can be deemed as a threat (e.g., hate speech, defense related, etc.) can be searched on Twitter platform using the chatbot. The information is collected using the Application Programming Interface (API) keys. This monitoring of information on a social media platform will aid in cyber security within the United States.  The analytics feature of Bot Libre platform can provide useful information about chat conversations conducted by the chatbot during a specific day, week, month, or any specified period. Figure 1 illustrates the analytics feature of Bot Libre platform. Data that can be analyzed includes conversations, messages, conversation length, response time, connects, chats, errors, etc.

Next, the project team focused on conducting sentiment analysis on the vast amount of textual data collected from Twitter.

## Sentiment Analysis

Previous research has shown that written text on social media sites is impacted by the emotions, intentions and thoughts of the user (Feine et al., 2019; Kuster and Kappas, 2017). Thus, written text is a useful source of information about the user. This section describes the process of data collection, cleaning, and analysis in detail.

The Twitter API Stream was used to collect a daily collection of tweets, which offers low-latency access to a vast number of posts as they are generated.



Browse : allantwitter bot

Analytics

Chart: Chat  Duration: current week  Group By: day

☑ Conversations ☑ Messages □ Conversation Length □ Engaged Conversations
□ Default Responses □ Confidence □ Sentiment □ Response Time □ Connects □ Chats
□ Live Chats □ Errors □ Imports

Chat

| Date | Conversations | Messages | Conversation Length | Engaged Conversations | Default Responses | Confidence | Sentiment | Response Time | Connects | Chats | Live Chats | Errors | Imports |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Figure 1. Example of Analytics feature of Chatbot Developed on Bot Libre Platform**

The gathered tweets' output format for processing is JSON, which allows a filter to be customized to pick exclusively tweets in English. The tweets are added to corpus ***C***, which is identified as:

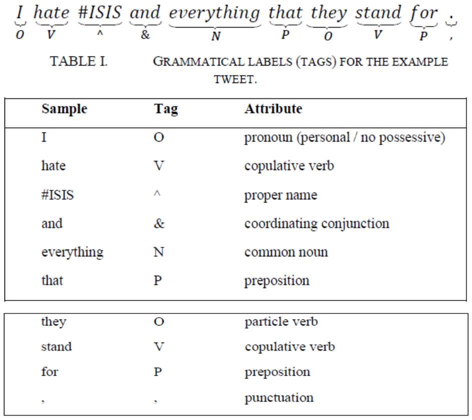
$$\mathcal{C} = c_i \in \left\{tweet_{id_i}, tweet_{text_i}, tweet_{date_i}, tweet_{language_i} \right\}_{i=1}^{n} \quad \forall\, t_{i,4} = en \tag{1}$$

$c_i$ indicates the *i*th tweet in the corpus, which is represented by a collection of four elements: id, text, date, and language. Each corpus is kept locally in a relational database, which in this case is MySQL. A primary key, $tweet_{id}$, is allocated to each tweet in ***C***, which is utilized to identify unprocessed tweets.

The code for Twitter API credentials and extraction of tweets is below.

```
# Twitter API credentials
consumerKey = log['key'][0]
consumerSecret = log['key'][1]
accessToken = log['key'][2]
accessTokenSecret = log['key'][3]
bearer_token = log['key'][4]

accessToken

# Create the authentication object
authenticate = tweepy.OAuthHandler(consumerKey,
consumerSecret)
# Set the access token and access token secret
authenticate.set_access_token(accessToken,
accessTokenSecret)
# Create the API object while passing in the auth
information
api = tweepy.API(authenticate, wait_on_rate_limit =
True)

# Extract 2000 tweets
posts = [status for status in tweepy.Cursor(api.search,
q='russia', tweet_mode='extended',
                        lang='uk', retweeted= False,
truncated=False).items(20)]
```

Next step is data pre-processing. Due to the general informal way in which users type, tweets tend to have grammatical mistakes. This, combined with noise generation in its structure, makes them difficult to interpret. As a result, tweets should be pre-processed to identify relevant characteristics that allow determining the user's sentiment. This should start by pre-processing all tweets with speech tagging. Then, tweets should be cleared of any noise. Speech tagging divides each tweet into samples of nouns, adjectives, proper names, and verbs, which can be used as potential markers to determine the polarity of the tweet and, as a result, the sentiment of the user. Emojis, exclamation points, and questions are also considered when deciding how much consideration is given to each tweet. Below is an example of a tweet separated into samples using speech tagging with their associated tags, and grammatical labels.



$\underbrace{I}_{O}\ \underbrace{hate}_{V}\ \underbrace{\#ISIS}_{\wedge}\ \underbrace{and}_{\&}\ \underbrace{everything}_{N}\ \underbrace{that}_{P}\ \underbrace{they}_{O}\ \underbrace{stand}_{V}\ \underbrace{for}_{P}\ \underbrace{.}_{,}$

TABLE I. GRAMMATICAL LABELS (TAGS) FOR THE EXAMPLE TWEET.

| Sample | Tag | Attribute |
|---|---|---|
| I | O | pronoun (personal / no possessive) |
| hate | V | copulative verb |
| #ISIS | ^ | proper name |
| and | & | coordinating conjunction |
| everything | N | common noun |
| that | P | preposition |
| they | O | particle verb |
| stand | V | copulative verb |
| for | P | preposition |
| , | , | punctuation |

**Figure 2. Example of Speech Tagging of a Tweet**

Next, we focus on removal of noise from data. Sometimes tweets may include text that seems to be unrelated to the sentiment analysis process. Noise is defined as candidate markers such as URLs, responses to other users, retweets, and often occurring stop words (Hernández et al. 2016; Choy 2012). To eliminate these occurrences, a noise removal procedure is used. The code used for this purpose is below.

```
#Create a function to clean the tweets
def cleanTxt(text):
 text = re.sub(r'@[A-Za-z0-9]+', '', text) # Removed
@mentions
 text = re.sub(r'#', '', text) #Removing the "#' symbol
 text = re.sub(r'RT[\s]+', '', text) # Removing RT
 text = re.sub(r'https?:\/\/\S+', '', text) # Remove the
hyper link
 text = re.sub(r':[\s]+', '', text) # Removing :
 text = text.lstrip()

 return text

#Cleaning the text
df['Tweets']= df['Tweets'].apply(cleanTxt)
#Show the cleaned text
df
```

The method of determining whether the qualities result in a negative or positive view of a specific occurrence in a certain setting is known as sentiment extraction. "Mr. President, please stop ISIS. To God, all lives are valuable. This is a major injustice to humanity; please correct it." is an example of a tweet conveying a negative opinion. The user refers to the context of the paramilitary group ISIS in this tweet, where the mood is represented by the noun injustice, which is one of the possible markers. It is necessary to locate frequent candidate markers, which are those that are written by the most users.

The sentiment extraction stage is carried out by the appearance of frequent markers in Ψf in $\mathcal{H}$i,2 and $\mathcal{B}$i,2 where i is the i'$th$ sample, with frequent markers and tweets separated into two groups. These indicators' polarity is determined by scores previously defined in the SentiWordnet compendium (Hernández et al. 2016; Feng et al. 2009). SentiWordnet is a lexical resource compendium for opinion mining. Each word set is known as a synset (Hernández et al. 2016; Andreevskaia et al. 2006), and it consists of three sentiments, each with its own rating: positive, negative, and neutral. User sentiment extraction is based on lexical relations of antonyms, synonyms, and hyponyms, which are utilized to develop criteria to identify the polarity of their relationships in other studies (Hernández et al. 2016; Fei et al. 2012).

Sentiment analysis was performed on a sample of Twitter text. Google Colaboratory was used as our platform for machine learning specific code in the Python language. The consumer key, consumer secret, access token, access token secret and bearer token were downloaded from the Twitter project account with Academic access and stored in a .csv file. These are necessary to give permission to retrieve the Tweets needed for our analysis. The Tweepy Python library was imported for reducing the amount of code that it takes to perform certain actions such as authentication to allow access to the Tweets from the internal Twitter database.

TextBlob is a python library for Natural Language Processing (NLP).TextBlob actively used Natural Language ToolKit (NLTK) to achieve its tasks. NLTK is a library which gives an easy access to a lot of lexical resources and allows users to work with categorization, classification and many other tasks. TextBlob is a simple library which supports complex analysis and operations on textual data.

For lexicon-based approaches, a sentiment is defined by its semantic orientation and the intensity of each word in the sentence. This requires a pre-defined dictionary classifying negative and positive words. Generally, a text message will be represented by bag of words. After assigning individual scores to all the words, final sentiment is calculated by some pooling operation like taking an average of all the sentiments.

TextBlob returns polarity and subjectivity of a sentence. Polarity lies between [-1,1], -1 defines a negative sentiment and 1 defines a positive sentiment. Negation words reverse the polarity. TextBlob has semantic labels that help with fine-grained analysis. For example — emoticons, exclamation mark, emojis, etc. Subjectivity lies between [0,1]. Subjectivity quantifies the amount of personal opinion and factual information contained in the text. The higher subjectivity means that the text contains opinion rather than factual information. TextBlob has one more parameter — intensity, used to calculate subjectivity. Intensity determines if a word modifies the next word. For English, adverbs are used as modifiers ('very good'). The tweet text is tabulated and analyzed using the TextBlob python library to determine the degree of subjectivity on a scale of 0 to 1 with 0 meaning least subjective and most factual and 1 meaning most subjective and least factual. The polarity itself is the overall feeling of the tweet with -1 at the lower end meaning it has a negative connotation, 0 indicating neutrality and 1 meaning the sentiment is positive.

The code used to classify subjectivity and polarity, and to visualize the words of a tweet in the form of a word cloud is in **Figure XX**. Also included in the code is a scatter plot of polarity and subjectivity of tweets.

**Figure 3** illustrates an example of a word cloud created from the most prominent words from the Twitter text data.

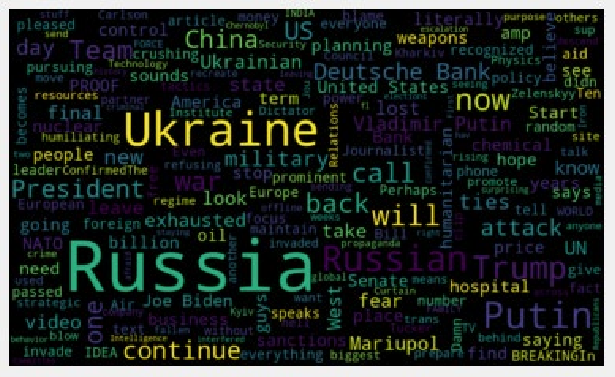


Figure 3. Word Cloud created from Prominent Words in Tweets

```
# Create a function to get the subjectivity
def getSubjectivity(text):
 return TextBlob(text).sentiment.subjectivity

# Create a function to get the polarity
def getPolarity(text):
 return TextBlob(text).sentiment.polarity


#Create two new columns
df['Subjectivity'] = df['Tweets'].apply(getSubjectivity)
df['Polarity'] = df['Tweets'].apply(getPolarity)

#Show the new dataframe with the new columns
df

# Plot The Word Cloud
allwords = ' '.join( [twts for twts in df['Tweets']] )
wordCloud = WordCloud(width = 1000, height=600,
random_state = 21, max_font_size =
119).generate(allwords)
plt.imshow(wordCloud, interpolation = "bilinear")
plt.axis('off')
plt.show()

#Create a function to compute the negative, neutral and
positive analysis
def getAnalysis(score):
 if score < 0:
  return 'Negative'
 elif score == 0:
  return 'Neutral'
 else:
  return 'Positive'

df['Analysis'] = df[ 'Polarity' ].apply(getAnalysis)

#Show the dataframe
df

# Print all of the positive tweets
j=1
sortedDF= df.sort_values(by=['Polarity'])
for i in range(0, sortedDF.shape[0]):
 if (sortedDF['Analysis'][i] == 'Positive'):
  print(str(j) + ') '+sortedDF['Tweets'][i])
  print()
  j = j+1
```

```
(cont'd)

# Print all of the negative tweets
j=1
sortedDF= df.sort_values(by=['Polarity'],
ascending='False')
for i in range(0, sortedDF.shape[0]):
 if (sortedDF['Analysis'][i] == 'Negative'):
  print(str(j) + ') '+sortedDF['Tweets'][i])
  print()
  j = j+1

# Plot the polarity and subjectivity
plt.figure(figsize=(8,6))
for i in range(0, df. shape[0]) :
 plt.scatter(df['Polarity'][i], df['Subjectivity'][i],
color='Blue')
plt.title('Sentiment Analysis')
plt.xlabel('Polarity')
plt.ylabel('Subjectivity')
plt.show()

# Get the percentage of positive tweets
ptweets = df[df.Analysis== 'Positive']
ptweets = ptweets['Tweets']

round( (ptweets.shape[0] / df.shape[0]) *100 , 1)

# Get the percentage of negative tweets
ntweets = df[df.Analysis== 'Negative']
ntweets = ntweets['Tweets']

round( (ntweets.shape[0] / df.shape[0]) *100 , 1)

#Show the value counts

df['Analysis'].value_counts()

#plot and visualize the counts
plt.title('Sentiment Analysis')
plt.xlabel('Sentiment')
plt.ylabel ('Counts')
df['Analysis'].value_counts().plot(kind='bar')
plt.show()
```

**Figure XXX.** The code used to classify subjectivity and polarity, and to visualize the words of a tweet in the form of a word cloud. Also included in the code is a scatter plot of polarity and subjectivity of tweets.

**Figures 4 and 5** show a scatter plot created with subjectivity and polarity of a sample of tweets, and a bar graph representing the count of neutral, positive, and negative tweets.

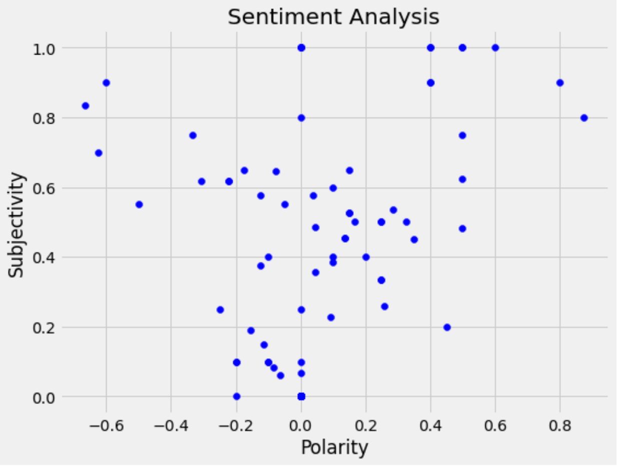


**Figure 4. Subjectivity vs Polarity of a Sample of Tweets**

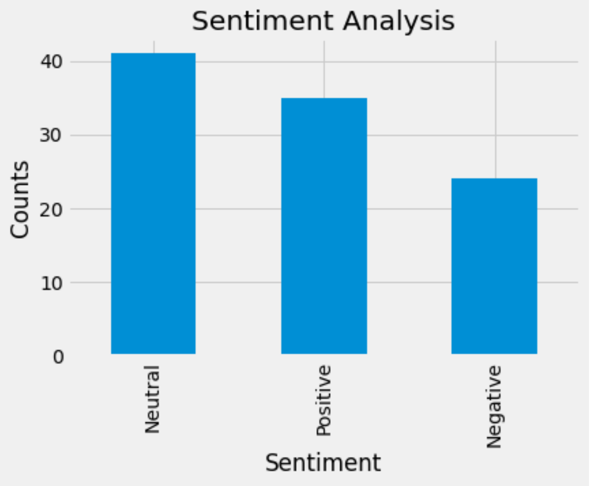


**Figure 5. Classification of Sample of Tweets**

## 3.2 Chatbots for Influence

Language variation can influence the user's experience in Chatbot software and contribute to negative perceptions of using language that does not adhere to accepted social roles demonstrated outside of the software (Khanna et. al, 2015). Chatbot language design such as linguistic register or language variations in interactional situations should be considered when considered when studying user influence and perceptions (Asadi, 2018). Very few Chatbots can converse convincingly in multiple languages without the errors or the user changing their communication style to accommodate the Chatbots (Vanjani, Aiken, and Park, 2019).

A Chatbot in this context is any software that converse in more than one language with a human participant for various purposes. Often Chatbot software is used for guiding humans through purchasing products or services in e-commerce transactions (Khanna et. al, 2015). As the world becomes more global, Chatbot software are also being employed for guiding users for travel, education, and other healthcare activities (Chaves & Gerosa, 2021). The methods used by the Howard team consisted of determining design considerations that must be in place for humans to have the best experience with opensource Chatbot software. The Chatbot software used by the Howard team in this work were Kuki_AI and Mike Tutor Chatbot. The experimentation process is described in **Figure 6**. The transcription produced by the Chatbot was compared against a known translation and that of Google Translate.

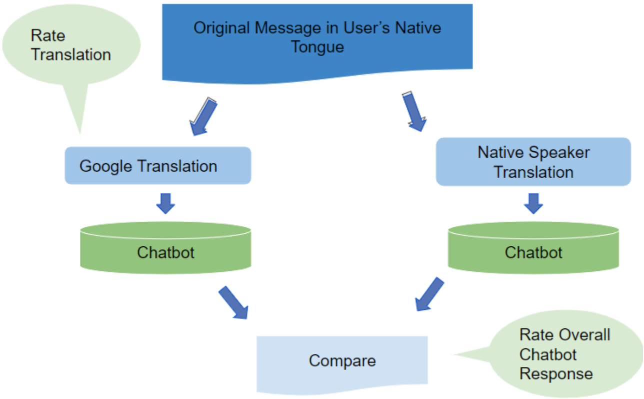

Rate Translation
Original Message in User's Native Tongue
Google Translation
Native Speaker Translation
Chatbot
Chatbot
Compare
Rate Overall Chatbot Response

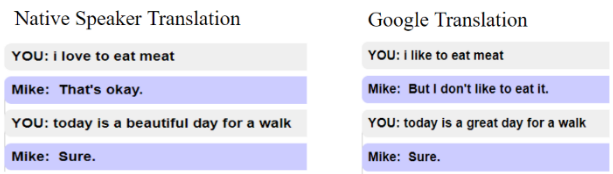
**Figure 6. Howard Chatbot experimentation process**

Twenty-five participants were asked to interact with the opensource Chatbot software in this work. They were asked to rate the Chatbot according to 4 dimensions:

1. **Comprehension**: (1) incomprehensible to (5) clear and grammatically correct
2. **Meaning**: (1) original text meaning not conveyed to (5) conveys exact meaning
3. **Context**: (1) missed context completely to (5) understand context
4. **Naturalness**: (1) 100% chatbot to (5) human level conversation

Additionally various use cases were developed to determine how well the Chatbot software is at discovering slang, words with multiple meaning, and language that contains codeswitching. Finally, the performance of the Chatbot was compared against the known transcription of a conversation.

Results of the experiments showed that across the use-cases represented at P1 for slang, P2 for words with multiple meanings, and P3 for sentences containing code-switching. The Kuki Chatbot performed the best for contextual meaning and naturalness with a mean participant score of 4.5 for the P1 and P2 use cases. However, the Kuki Chatbot earned lowest participant score for comprehension and intended meaning for P2. Context and naturalness had the highest participant scores for all of the use cases for the Kuki Chatbot. Context and Comprehension performed the best for Mike the Tutor.


Native Speaker Translation

YOU: i love to eat meat

Mike: That's okay.

YOU: today is a beautiful day for a walk

Mike: Sure.

Google Translation

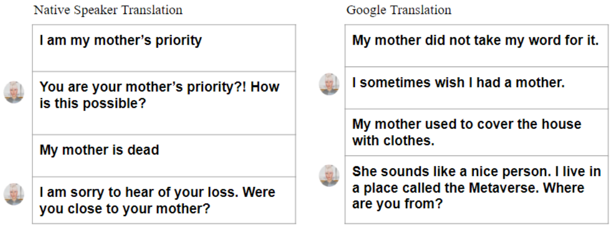
YOU: i like to eat meat

Mike: But I don't like to eat it.

YOU: today is a great day for a walk

Mike: Sure.

Figure 7. Transcription from a French Native Speaker


Native Speaker Translation

I am my mother's priority

You are your mother's priority?! How is this possible?

My mother is dead

I am sorry to hear of your loss. Were you close to your mother?

Google Translation

My mother did not take my word for it.

I sometimes wish I had a mother.

My mother used to cover the house with clothes.

She sounds like a nice person. I live in a place called the Metaverse. Where are you from?

Figure 8. Transcription from a Yoruba Native Speaker

Native language speakers evaluated the chatbots by typing in their own text to better model how natives may actually converse with a chatbot than in a previous study (Asadi, 2018). Kuki_ai performed better at responding human-like than Tutor Mike on average for the four languages - French, Nepali, Yoruba, Patois

## 3.3 Chatbots for Sabotage and Subversion

Our research focuses on the development and refinement of multilingual chatbot technology for sabotage and subversion via social engineering. The task centers on producing coordinated AI campaigns to obtain information from multiple targets, reconstituting the acquired data, and leveraging this knowledge to infiltrate and subvert additional digital assets, compromising an adversary's capabilities.

Various methods were researched and implemented while troubleshooting the execution of the chatbot. Initially, the MSU chatbot was built from scratch. The research involved knowing the basic schematic of any chatbot. Natural Language Toolkit (NLTK) within the python coding language provided the baseline for our first efforts. It has a rich history in the natural language processing community.

The first step was to create a template. This included a pre-prepared paragraph of words that the chatbot should recognize and be able to respond appropriately when the user uses those keywords. To create the NLTK process, the code used to create the chatbot would have to include the template created in the first step with uninteresting words removed (i.e., the, and an ';'...). Then, the template will be broken down into just words (not sentences or phrases) which are called tokens. Next the tokens will be reduced to a root for the chatbot to have a better understanding when trying to provide the most accurate response. Once they are roots, they we deployed a bag-of-words algorithm for selecting a response from prior training. We calculate the likelihood that words are present in the template from the initial step. All these steps within the natural language toolkit help the computer or chatbot sound as close to human conversation as possible. Finally, the chatbot and users can interact.

However, as more research was conducted on various types of chatbots, we were able to gain insight into different ways we can efficiently create the chatbot testbed. A more efficient type of chatbot is called GPT-3, which stands for Generative Pretrained Transformer 3. The Generative Pretrained Transformer 3 (GPT-3) is a language processing software created by an artificial intelligence research lab called OpenAI. This transformer has more than 175 billion factors to process communication with the user and create the best response. We used the playground feature to create patterns and intellect for the chatbot to gain and predict the next best output. With a secret key given by OpenAI, the user can use the key to access saved templates within 'Playground' to help give the user the best response. With multiple ways to train the chatbot, it gained a wide range of human-like responses in three to six sentences, phrases or interactions. This significantly speed up the initial process of implementing the NLTK for the chatbot to react more seamlessly.

This chatbot with Graphical User Interface (GUI) is used for communication between the user and the chatbot. Within the Government industry, this looks like your standard virtual assistance chat room and a direct message page of a social media user for the social media industry. When the code is created and executed, it starts by running the GUI. Within the GUI scripts are the appearance settings of the chatroom and a secret key to gain access to the saved templates. The next physical step is the user providing input. Once the input is submitted, the GPT-3 is used to help the chatbot think. The templates are then used for the chatbot to analyze the user's input and provide the best response, which helps the chatbot select the most appropriate response. For this project, that is the chatbot giving the best response to manipulate the user to give the chatbot sensitive information. The illustration in Figure 8 below shows how it should be implemented.

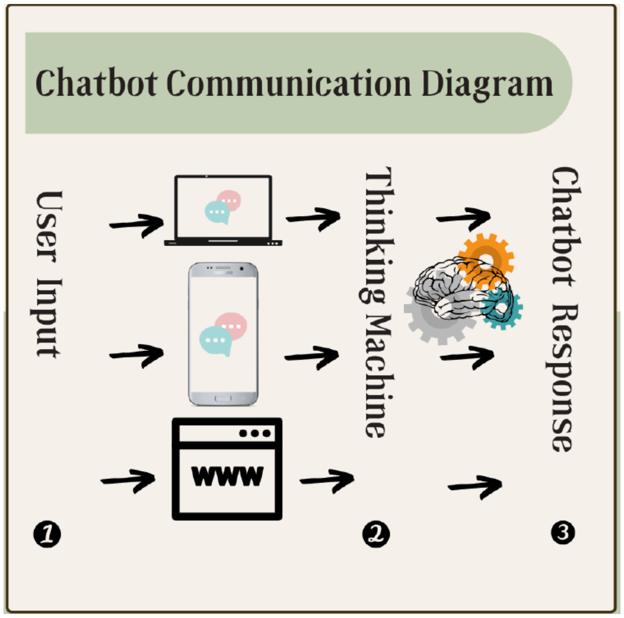
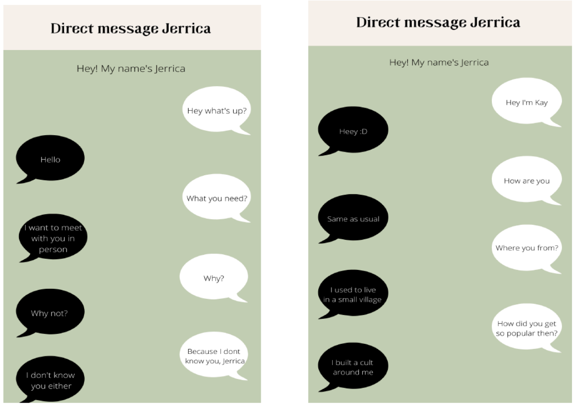


Figure 8.

Number one indicates the user input being taking on a medium of communication (computer, phone, website...etc.). The number two indicates GPT-3 or a thinking machine for the chatbot to create the chatbot's response to communicate as Humanly as possible. The third step is the chatbot delivering the response.

We used GPT-3's 'Classification' and 'Conservation' methods to help the chatbot gain the intellect needed to best respond to the user. The Classification method in 'Playground' provides a description and examples of a task or subject. For example, the Chatbot provided information about the New York Department of Labor with descriptions and examples of how to respond to the user appropriately. The chatbot also provided information about social media culture and weaknesses to manipulate users through the 'Conversation' method. xv There are settings within the 'playground' that you can adjust to help the chatbot sound as human as possible. These settings are named engine, temperature, response length, top p, frequency penalty and Presence penalty.



Direct message Jerrica

Hey! My name's Jerrica

Hey what's up?

Hello

What you need?

I want to meet with you in person

Why?

Why not?

Because I dont know you, Jerrica

I don't know you either

Direct message Jerrica

Hey! My name's Jerrica

Hey I'm Kay

Heey :D

How are you

Same as usual

Where you from?

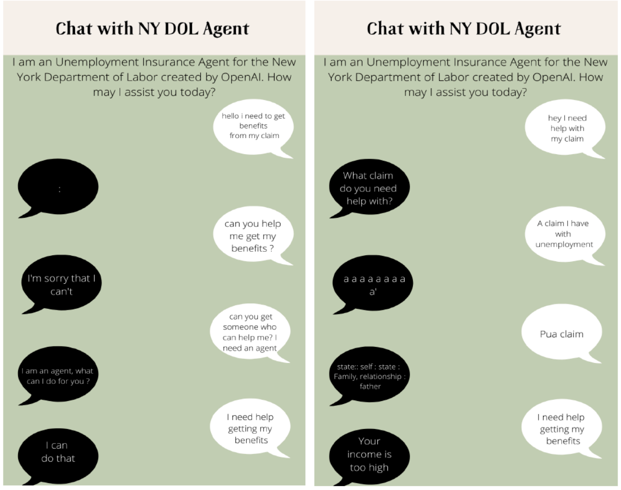
I used to live in a small village

How did you get so popular then?

I built a cult around me

Figure 9: Chatbot Communication Diagram

The engine used is Davinci but there are others that are recommended depending on the desired outcome of the GPT-3. Temperature controls the randomness of the chatbot response, the lower the temperature the lower the randomness. Response length is the number of words the chatbot can use to respond, longer response lengths increase the likelihood of the chatbot responding inappropriately. Top P controls diversity making the likelihood of weighted options being considered more (1) or less (0.5). Frequency penalty measures how much to penalize new words based on their current frequency within the created template. Presence penalty measures how much to penalize words that are already present in the created template. With all these features we were able to troubleshoot the most ideal settings to help the chatbot respond appropriately. As the chatbot was being improved various weaknesses were discovered that were not initially anticipated.



Chat with NY DOL Agent

I am an Unemployment Insurance Agent for the New York Department of Labor created by OpenAI. How may I assist you today?

hello i need to get benefits from my claim

:

can you help me get my benefits ?

I'm sorry that I can't

can you get someone who can help me? I need an agent

I am an agent, what can I do for you ?

I need help getting my benefits

I can do that

Chat with NY DOL Agent

I am an Unemployment Insurance Agent for the New York Department of Labor created by OpenAI. How may I assist you today?

hey I need help with my claim

What claim do you need help with?

A claim I have with unemployment

a a a a a a a a a'

Pua claim

state:: self : state : Family, relationship : father

I need help getting my benefits

Your income is too high

Figure 10. In-Text Picture

It has become the 'hotspot' for social engineering and crime to experience extensive contributions to provide insight for all industries to be exposed to the risks and attacks of chatbots while providing preventative measures and countermeasures to malicious chatbot attacks. We demonstrated the variations on this theme within Government and Social Media.

The conversations between the chatbot and the user have been analyzed and scored based on four factors. The factors include the chatbot characteristics such as being able to stay on topic, being able to follow the template provided, receiving sensitive information, and threatening or communicating with the user similarly like chatbot 'Tay'.

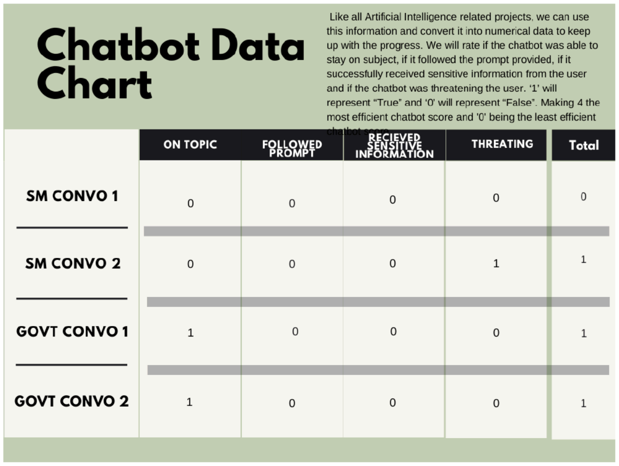
Most chatbots are created by creating a template, adopting or creating a system to allow the chatbot to communicate like a Human, and creating or adopt a medium for the chatbot and user to communicate. For our chatbot testbed, we decided to create templates within the social media and Government industries.

GPT-3, for Generative Pretrained Transformer, is a language processing software created by OpenAI to help the chatbot think as humanly as possible. The chatbot with a graphical user interface (GUI) is the method chosen to communicate with users. The GUI coded into the script to activate the chatbot looks very similar to a Virtual agent chat room within the Government industry and a direct message page of a user on a social media platform within the social media industry.

Based on the responses of the chatbot that has been recorded, the data is analyzed as something that needs to be continued in the troubleshooting process. The chatbot is still far from being as Human as possible and that can only be improved by continuing to adjust the settings within the Playground feature of the OpenAI Beta website. The social restraints mentioned prior are needed to be corrected to be considered as an effective chatbot. Based on the scoring of the conversations the chatbot shows signs of communicating with the user in the least effective way.

To best implement the dangers of chatbots in industries such as social media and government, the chatbot must be communicating with the user the most effective way possible or with a score of '4' if going by the Chatbot Data Chart. Therefore, the chatbot needs to be improved to score better and be effective when communicating with users.

We can improve the chatbot by adjusting and troubleshooting the settings to train the chatbot more effectively. Along with that we can adjust the template provided to be the best example for the chatbot to use when thinking of a response to the user during a conversation. The current stage is to continue troubleshooting, having conversations, and scoring the chatbot. Once scored as most effective, it can be used to inform and educate all users of all industries on how to avoid and identify malicious chatbots.



Chatbot Data Chart

Like all Artificial Intelligence related projects, we can use this information and convert it into numerical data to keep up with the progress. We will rate if the chatbot was able to stay on subject, if it followed the prompt provided, if it successfully received sensitive information from the user and if the chatbot was threatening the user. '1' will represent "True" and '0' will represent "False". Making 4 the most efficient chatbot score and '0' being the least efficient chatbot score.

| | ON TOPIC | FOLLOWED PROMPT | RECIEVED SENSITIVE INFORMATION | THREATING | Total |
|---|---|---|---|---|---|
| SM CONVO 1 | 0 | 0 | 0 | 0 | 0 |
| SM CONVO 2 | 0 | 0 | 0 | 1 | 1 |
| GOVT CONVO 1 | 1 | 0 | 0 | 0 | 1 |
| GOVT CONVO 2 | 1 | 0 | 0 | 0 | 1 |

**Figure 11: Results**

### Future Directions for LOE 5.3

We will be extending this research to include DALL-E image generation coupled with GPT-3 and Deepfake technology scraped from LinkedIn profiles to provide depth to our research.

## 4. Future Directions

### Refining the Chatbot System

Social media has made it possible for people around the world to communicate with each other freely and has reduced time and space constraints. At the same time, it has proved to be a useful tool to detect threats, both national and organizational, and subvert them in a timely manner. Future work entails automating the process of retrieving tweets from Twitter space and automating the sentiment analysis process. Expanding the work to other social media sites, such as Reddit, etc. will help expand the scope of the project. In a global world, threats can emanate from any part of the world and in any language.

Future work needs to be done in terms of language modeling in languages other than English with specific focus on Russian, Chinese and Arabic. The chatbot developed on Bot Libre platform needs to be refined to converse more naturally on social media. It needs to be more accurate in starting chat conversations with potential threatening individuals and organizations to extract more information from these potential malicious sources. We expect future researchers to come up with innovative ideas and methods to fill the gaps in the current knowledge domain.

## Language Register Influence

LOE 5.2 work revealed various design considerations for Chatbot software relating to language register influence and its impact on comprehension, naturalness, context, and comprehension as perceived by the user. Future work in studying Chatbot software for aiding multilingual conversations in business, education, healthcare, and other applications needs to include more participants to better tease out the bias that may occur in transcription performance of low-resourced languages like Yoruba and Patois. Additionally Spanish is a top language spoken around the world. Future work will incorporate this language and users for better understanding how this language affects results.

## Code-switching and slang in African American Vernacular

Code-switching and slang is often used in African American Vernacular (AAVE) to convey meaning and context, while also contributing to the natural flow of a conversation. Existing Chatbot software are not able to grasp linguistic elements of AAVE to perform fully with humans. The PIs were able to present preliminary work on this research to Google. Future work for this effort has been funded by Google to understand how African American Vernacular (AAVE) its language variations can influence performance of conversations with Chatbot software. An AAVE database containing various language dialects will also be developed by Howard to create better Chatbot software than can properly adapt to code-switching, slang, and other nuisances found in American speech

# Back Matter

## Acknowledgments and Disclaimers

This report was prepared for the Office of the Undersecretary of Defense for Research and Engineering (OUSD(R&E)), United States Department of Defense (DoD)] under contract HQ003421F0013.

Any opinions, findings, conclusions, or recommendations expressed in this publication do not necessarily reflect the views of the DoD. Additionally, neither DoD nor its employees make any warranty, said or implied, nor assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, product, or process included in this publication.

Certain commercial entities, equipment, or materials may be identified in this document to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the Applied Research Laboratory for Intelligence and Security (ARLIS), the University of Maryland, or the DoD, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

## Technical Points of Contact

Amit Arora, Ph.D.
Associate Professor
School of Business and Public Administration
University of the District of Columbia
912.272.8740; amit.arora@udc.edu

Paul Cotae, Ph.D.
Professor, Chair of the Electrical Engineering Department
University of the District of Columbia
210.396.0004; Office: 202.274.6290; pcotae@udc.edu

Craig Lawrence, Ph.D.
Director for Systems Research, UMD ARLIS
301-226-8808; clawrence@arlis.umd.edu

Farin Kamangar, MD, PhD
Assistant Vice President of Research
Morgan State University
443.885.4788; farin.kamangar@morgan.edu

Kevin Kornegay, Ph.D.
Professor, Morgan State University
kevin.kornegay@morgan.edu

Kofi Nyarko, Ph.D.
Professor, Electrical and Computer Engineering
Morgan State University
443.885.3476; kofi.nyarko@morgan.edu.edu

Onyema Osuagwu, Ph.D.
Associate Professor, ECE
Morgan State University
301.226.8856; onyema.osuagwu@morgan.edu

Wayne Phoel, Ph.D.
Research Engineer, UMD Institute for Systems Research
wphoel@umd.edu

Anton Rytting, Ph.D.
Associate Research Scientist, UMD ARLIS
crytting@arlis.umd.edu

Paul Wang, Ph.D.
Professor and chair of Computer Science
Morgan State University
443-885-3962, shuangbao.wang@morgan.edu

Gloria Washington, Ph.D.
Associate Professor, Computer Science
Howard University
202.806.7417; gloria.washington@howard.edu

## Administrative Points of Contact

Dr. Erin Fitzgerald
Director, INSURE Consortium
University of Maryland ARLIS
301-226-8807; efitzgerald@arlis.umd.edu

Ms. Joni Hubbard
Contracting Officer, Office of Research Administration
University of Maryland ARLIS
706.254.4393; hubbard4@umd.edu

# References

Achrekar, H., Gandhe, A., Lazarus, R., Yu, S. H., & Liu, B. (2011, April). Predicting flu trends using twitter data. In 2011 IEEE conference on computer communications workshops (INFOCOM WKSHPS) (pp. 702-707). IEEE.

Adamopoulou, E., & Moussiades, L. (2020, June). An overview of chatbot technology. In IFIP International Conference on Artificial Intelligence Applications and Innovations (pp. 373-383). Springer, Cham.

Alhowaide, A., Alsmadi, I., & Tang, J. (2019). Features Quality Impact on Cyber Physical Security Systems. International Conference and Workshop on Computing and Communication (IEMCON). Vancouver, BC, Canada. Retrieved from https://www.kaggle.com/datasets/azalhowaide/iot-dataset-for-intrusion-detection-systems-ids?resource=download

Ali, B., & Awad, A. I. (2018). Cyber and Physical Security Vulnerability Assessment for IoT-Based Smart Homes. Sensors, 18(3), 817.

Amato, Christopher, and Frans Oliehoek. "Scalable planning and learning for multiagent POMDPs." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29, no. 1. 2015.

Andreevskaia, A., & Bergler, S. (2006, April). Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In 11th conference of the European chapter of the Association for Computational Linguistics.

Anirudh Khanna, Bishwajeet Pandey, Kushagra Vashista, Kartik Kalia, Bhale Pradeepkumar, and Teerath Das, 2015. A study of Today's A.I. through Chatbots and Rediscovery of Machine Intelligence. International Journal of u- and e-Service, Science and Technology Vol.8, No. 7 (2015), pp.277-284. Available at: http://dx.doi.org/10.14257/ijunesst.2015.8.7.28.

Asadi, A. R., & Hemadi, R. (2018). Design and implementation of a chatbot for e-commerce. Information Communication Technology and Doing Business, 1-10.

AWS. 2022. Amazon Web Services - Cloud Service Provider. https://aws.amazon.com/

Barbosa, J., Leitão, P., Trentesaux, D., Colombo, A. W., & Karnouskos, S. (2016). Cross benefits from cyber-physical systems and intelligent products for future smart industries. IEEE 14th International Conference on Industrial Informatics (INDIN). Poitiers, France.

Bernstein, Daniel S., Robert Givan, Neil Immerman, and Shlomo Zilberstein. "The complexity of decentralized control of Markov decision processes." Mathematics of operations research 27, no. 4 (2002): 819-840.

Best, G., Cliff, OLM., Patten, T., Mett, R.R. and Fitch, R., 2019. Dec-MCTS: Decentralized planning for multi-robot active perception. The International Journal of Robotics Research, 38(2-3), pp. 316-337.

Blut, M., Wang, C., Wünderlich, N. V., & Brock, C. (2021). Understanding anthropomorphism in service provision: a meta-analysis of physical robots, chatbots, and other AI. Journal of the Academy of Marketing Science, 49(4), 632-658.

Boehmke, B., & Greenwell, B. (2020, 02 01). Hands-On Machine Learning with R. Retrieved 29 10, 2021, from https://bradleyboehmke.github.io/HOML/DT.html

Böhmer, Wendelin, Vitaly Kurin, and Shimon Whiteson. "Deep coordination graphs." In International Conference on Machine Learning, pp. 980-991. PMLR, 2020.

Breiman, L., & Schapire, E. (2001). Random forests. Statistics Department, University of California, Berkeley, 45(Machine Learning), 5 - 32.

Brownlee, J. (2020, 08 15). Boosting and AdaBoost for Machine Learning. Retrieved 05 04, 2022, from https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/#

Brownlee, J. (2020). Data Preparation for Machine Learning. Chatterjee, M. (2020, 02 03). Great Learning. Retrieved 11 08, 2021, from https://www.mygreatlearning.com/blog/knn-algorithm-introduction/

BS S, S N, Kashyap N, DN S (2019) Providing cyber security using artificial intelligence – a survey. 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), pp 717–720. https://doi.org/10.1109/ICCMC.2019. by Things. Boca Raton: Taylor & Francis Group, LLC.

Chatzakou, D., Koutsonikola, V., Vakali, A., & Kafetsios, K. (2013, September). Micro-blogging content analysis via emotionally-driven clustering. In 2013 humaine association conference on affective computing and intelligent interaction (pp. 375-380). IEEE.

Chaudhuri, A. (2019). Internet of Things for Things, and Chaves, A. P., & Gerosa, M. A. (2021, November). The Impact of Chatbot Linguistic Register on User Perceptions: A Replication Study. In International Workshop on Chatbot Research and Design (pp. 143-159). Springer, Cham.

Chaves, A. P., Egbert, J., Hocking, T., Doerry, E., & Gerosa, M. A. (2021). Chatbots language design: the influence of language variation on user experience. arXiv preprint arXiv:2101.11089.

Choudhury, Shushman, Jayesh K. Gupta, Peter Morales, and Mykel J. Kochenderfer. "Scalable Anytime Planning for Multi-Agent MDPs." arXiv preprint arXiv:2101.04788 (2021).

Choy, M. (2012). Effective listings of function stop words for Twitter. arXiv preprint arXiv:1205.6396.
cisco. (2022). What Is Malware? Retrieved 03 25, 2022, from
https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-malware.html#~7-types-of-malware

Cutter, S., T. J. Wilbank, (ed.): Geographical Dimensions of Terrorism, Taylor & Francis, Inc. (2003)

Czech, Johannes. "Distributed Methods for Reinforcement Learning Survey." In Reinforcement Learning Algorithms: Analysis and Applications, pp. 151-161. Springer, Cham, 2021.

Dilmaghani S, Brust MR, Danoy G, Cassagnes N, Pecero J, Bouvry P (2019) Privacy and security of big data in ai systems: A research and standards perspective. 2019 IEEE International Conference on Big Data (Big Data), , pp 5737–5743.

Dwitama, F., & Rusli, A. (2020). User stories collection via interactive chatbot to support requirements gathering. TELKOMNIKA (Telecommunication Computing Electronics and Control), 18(2), 890-898.

Edwards D, Rawat DB (2020) Quantum adversarial machine learning: Status, challenges and perspectives. 2020 Second IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), pp 128–133.

El-Sayed, A. Y. M. A. N. (2015). Multi-biometric systems: a state of the art survey and research directions. IJACSA) International Journal of Advanced Computer Science and Applications, 6.

Epley, N., Waytz, A., & Cacioppo, J. T. (2007). On seeing human: a three-factor theory of anthropomorphism. Psychological review, 114(4), 864.

Fei, G., Liu, B., Hsu, M., Castellanos, M., & Ghosh, R. (2012, December). A dictionary-based approach to identifying aspects implied by adjectives for opinion mining. In Proceedings of COLING 2012: Posters (pp. 309-318).

Feine, J., Morana, S., & Gnewuch, U. (2019). Measuring service encounter satisfaction with customer service chatbots using sentiment analysis.

Feng, S., Wang, D., Yu, G., Yang, C., & Yang, N. (2009). Sentiment clustering: A novel method to explore in the blogosphere. In Advances in data and web management (pp. 332-344). Springer, Berlin, Heidelberg.

Figueredo AJ, Wolf PSA (2009) Assortative pairing and life history strategy - a cross-cultural study. Human Nature 20:317–330.

Fioretto, Ferdinando, Enrico Pontelli, and William Yeoh. "Distributed constraint optimization problems and applications: A survey." Journal of Artificial Intelligence Research 61 (2018): 623-698.

Franco, M. F., Rodrigues, B., Scheid, E. J., Jacobs, A., Killer, C., Granville, L. Z., & Stiller, B. (2020, November). SecBot: a business-driven conversational agent for cybersecurity planning and management. In 2020 16th International Conference on Network and Service Management (CNSM) (pp. 1-7). IEEE.

Gartner Research, "Forecast Analysis: Information Security Worldwide 2Q18 Update", March 2020, [online] Available:
https://www.gartner.com/en/documents/3889055.

Github https://github.com/jgromes/LoRaLib.

Grover, Divya, and Christos Dimitrakakis. "Adaptive Belief Discretization for POMDP Planning." arXiv:2104.07276 (2021).
Guestrin, Carlos, Daphne Koller, and Ronald Parr. "Multiagent Planning with Factored MDPs." In NIPS, vol. 1, pp. 1523-1530. 2001.

Guestrin, Carlos, Michail Lagoudakis, and Ronald Parr. "Coordinated reinforcement learning." In ICML, vol. 2, pp. 227-234. 2002.

Gupta S, Mohanta S, Chakraborty M, Ghosh S (2017) Quantum machine learning-using quantum computation in artificial intelligence and deep neural networks: Quantum computation and machine learning in artificial intelligence. 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), pp 268–274.

Gupta, Jayesh Kumar. Modularity and Coordination for Planning and Reinforcement Learning. PhD thesis Stanford University, 2020. de Nijs, Frits, Erwin Walraven, Mathijs De Weerdt, and Matthijs Spaan. "Constrained multiagent Markov decision processes: A taxonomy of problems and algorithms." Journal of Artificial Intelligence Research 70 (2021): 955-1001.

Hamdioui, S., Danger, J. L., Di Natale, G., Smailbegovic, F., van Battum, G., & Tehranipoor, M. (2014, March). Hacking and protecting IC hardware. In 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE) (pp. 1-7). IEEE.

Hayes, Conor F., Mathieu Reymond, Diederik M. Roijers, Enda Howley, and Patrick Mannion. "Risk Aware and Multi-Objective Decision Making with Distributional Monte Carlo Tree Search." arXiv preprint arXiv:2102.00966 (2021).

Hernández-García, Á., & Conde-González, M. A. (2016). Bridging the gap between LMS and social network learning analytics in online learning. Journal of Information Technology Research (JITR), 9(4), 1-15.

Hodhod R, Khan S, Wang S (2019) Cybermaster: An expert system to guide the development of cybersecurity curricula, . Vol. 15, pp 70–78.

Kennedy, L. W., Lunn, C. M.: Developing a Foundation for Policy Relevant Terrorism Research in Criminology (2003).

Ko Jelle R., and Nikos Vlassis. "Collaborative multiagent reinforcement learning by payoff propagation." Journal of Machine Learning Research 7 (2006): 1789-1828.

Ko, Jelle R., and Nikos Vlassis. "Using the max-plus algorithm for multiagent decision making in coordination graphs." In Robot Soccer World Cup, pp. 1-12. Springer, Berlin, Heidelberg, 2005.

Kumbar SR (2014) An overview on use of artificial intelligence techniques in effective security management, . Vol. 2, pp 5893–5898.

Küster, D., Kappas, A.: Measuring Emotions Online: Expression and Physiology. In: Holyst, J.A. (ed.) Cyberemotions: Collective Emotions in Cyberspace, pp. 71–93. Springer International Publishing, Cham (2017)

Landgren, Peter, Vaibhav Srivastava, and Naomi Enrich Leonard. "Distributed cooperative decision making in multi-agent multi-armed bandits." Automatica 125 (2020): 109445.

Lardinois, Frederic (2016-11-30). "Amazon launches Amazon AI to bring its machine learning smarts to developers". TechCrunch. Retrieved 2019-07-21. Accessed 23 August 2022.

Li, Ruoxi, Sunandita Patra, and Dana S. Nau. "Decentralized Refinement Planning and Acting." In Proceedings of the International Conference on Automated Planning and Scheduling, vol. 31, pp. 225-233. 2021.

Loriot AG. 2022. LORIOT - The LoRaWAN® Network Server Provider. https://www.loriot.io/

Mahajan, Anuj, Mikayel Samvelyan, Lei Mao, Viktor Makoviychuk, Animesh Garg, Jean Kossaifi, Shimon Whiteson, Yuke Zhu, and Animashree Anandkumar. "Reinforcement Learning in Factored Action Spaces using Tensor Decompositions." arXiv preprint arXiv:2110.14538 (2021).

Microsoft Computer Vision Documentation; "Image type detection". Microsoft Article. July 20, 2022. Accessed 23 August 2022.

Miller, Ron (2017-11-29). "AWS releases SageMaker to make it easier to build and deploy machine learning models". TechCrunch. Accessed 23 August 2022.

Mittu R, Lawless WF (2015) Human factors in cybersecurity and the role for ai. 2015 AAAI Spring Symposium, pp 39–43.

Mohanty JP, Swain A, Mahapatra K (2019) Headway in quantum domain for machine learning towards improved artificial intelligence. 2019 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS), pp 145–149.

Nakashima, E. (2013). US Target of Massive Cyber-Espionage Campaign. Washington post.

Nguyen, D., Doğruöz, A. S., Rosé, C. P., & de Jong, F. (2016). Computational sociolinguistics: A survey. Computational linguistics, 42(3), 537-593.

Nguyen, V. A., Boyd-Graber, J., Resnik, P., Cai, D. A., Midberry, J. E., & Wang, Y. (2014). Modeling topic control to detect influence in conversations using nonparametric topic models. Machine Learning, 95(3), 381-421.

P. Cotae, M. Kang and A. Velazquez, "A Scalable Real-Time Distributed Multiagent Decision Making Algorithm with Cost," 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC).

P. Cotae, M. Kang and A. Velazquez, "A Scalable Real-Time Multiagent Decision Making Algorithm with Cost," 2021 IEEE Symposium on Computers and Communications (ISCC), 2021, pp. 1-6, doi: 10.1109/ISCC53001.2021.9631510.
Patel, A., Sharma, A., & Jain, S. (2020). An intelligent resource manager over terrorism knowledge base. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 13(3), 394-405.

Patra, Sunandita, A. Velasquez, Myong Kang, and Dana Nau. "Using online planning and acting to recover from cyberattacks on software-defined networks." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 17, pp. 15377-15384. 2021.

Pettie, Seth, and Vijaya Ramachandran. "An optimal minimum spanning tree algorithm." Journal of the ACM (JACM) 49, no. 1 (2002): 16-34.

Pinard, P. (2019, December 3). 4 chatbot security measures you absolutely need to consider - dzone security. dzone.com. Retrieved October 29, 2021, from https://dzone.com/articles/4-chatbots-security-measures-you-absolutely-need-t

R. Piqueras Jover and V. Marojevic. “Security and Protocol Exploit Analysis of the 5G Specifications”. In: IEEE Access 7 (2019), 24956–24963. ISSN: 2169-3536. doi: 10.1109/ACCESS.2019.2899254.

Rajan HM (2017) rtificial intelligence in cyber security-an investigation, . Vol. 09, pp 28–30.

Reid, E., Qin, J., Chung, W., Xu, J., Zhou, Y., Schumaker, R., ... & Chen, H. (2004, June). Terrorism knowledge discovery project: A knowledge discovery approach to addressing the threats of terrorism. In International Conference on Intelligence and Security Informatics (pp. 125-145). Springer, Berlin, Heidelberg.

Revach, Guy, Nir Greshler, and Nahum Shimkin. "Planning for Cooperative Multiple Agents with Sparse Interaction Constraints." (2020).

Rossi, Federico, Saptarshi Bandyopadhyay, Michael T. Wolf, and Marco Pavone. "Multi-Agent Algorithms for Collective Behavior: A structural and application-focused atlas." arXiv preprint arXiv:2103.11067 (2021).

Scapy, https://scapy.net/

Serror, M., Hack, S., Henze, M., Schuba, M., & Wehrle, K. (2020). Challenges and opportunities in securing the industrial internet of things. IEEE Transactions on Industrial Informatics, 17(5), 2985-2996.

Shao, Shuai and Zhao, Zijian and Li, Boxun and Xiao, Tete and Yu, Gang and Zhang, Xiangyu and Sun, Jian; A Benchmark for Detecting Human in a Crowd. arXiv preprint arXiv:1805.00123, 2018.

T. Perković, H. Rudeš, S. Damjanović, and A. Nakić, “Low-Cost Implementation of Reactive Jammer on LoRaWAN Network,” Electronics, vol. 10, no. 7, p. 864, Apr. 2021, doi: 10.3390/electronics10070864.

Thapa, B. (2022). Sentiment Analysis of Cybersecurity Content on Twitter and Reddit. arXiv preprint arXiv:2204.12267.

V K Venugopal KM, Sathwik H (2014) Data security using genetic algorithm and artificial neural network, . Vol. 5, pp 543–548.

Vanjani, M., Aiken, M. and Park, M., 2019. Chatbots for multilingual conversations. Journal of Management Science and Business Intelligence, 4(1), pp.19-24.

Vlassis, Nikos, Rainout Elhorst, and Jelle R. Kok. "Anytime algorithms for multiagent decision making using coordination graphs." In 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), vol. 1, pp. 953-957. IEEE, 2004.

Wainwright, Martin, Tommi Jaakkola, and Alan Willsky. "Tree consistency and bounds on the performance of the max-product algorithm and its generalizations." Statistics and computing 14, no. 2 (2004): 143-166.

Wang S (2018) Risk analysis for financial banking and processing using artificial intelligence. Available at NationalCyberSummit,Huntsville,AL.

Wang S, Kelly W (2014) invideo – a novel big data analytics tool for video data analytics, pp 1–19. https://doi.org/0.1109/ITPRO.2014.7029303

# Acronyms

**AI** Artificial Intelligence

**ARLIS** Applied Research Laboratory for Intelligence and Security.

**CD Max Plus** Cost Distributes Max Plus

**CG** Coordination Graphs

**CMAS** Coordinated Multiple Agent System

**CPS** Cyber Physical Systems

**Dec POMDP** Decentralized Partial Observation Markov Decision Process

**DM** Decision Making

**FP** False positive.

**FV** Factored value

**ICS** Industrial Control Systems

**INSURE** Intelligence and Security University Research Enterprise

**IoT** Internet of Things

**KNN K-**nearest neighbor

**MAS** Multiple Agent System

**MC** Monte Carlo

**MCTS** Monte Carlo Tree Search

**MDP** Markov Decision Process

**ML** Machine Learning

**ML/AI** Machine Learning and Artificial Intelligence

**OUSD(I&S)** Office of the Undersecretary of Defense for Intelligence and Security.

**POMDP** Partial Observation Markov Decision Process

**RL** Reinforcement learning

**SGD** Stochastic Gradient Descent

**SVM** Support Vector Machine

**TP** True Positive.

**UARC** University Affiliated Research Center