Lab3 :
VR Security & Attacks

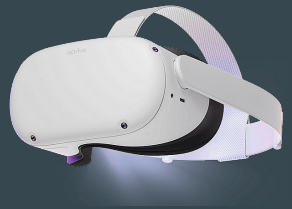
The Power Savvy Trio
Matt, Michelle, and Jose

User Identification Classifiers

- Tested different classification models for latency and accuracy
  - Gradient Boosting
    - Builds multiple weak decision trees where each new tree corrects the errors of the previous tree.
  - Histogram-Based Gradient Boosting
    - Discretizes features into bins, which can accelerate the training process and improve performance.
  - K-nearest Neighbors
    - Calculates the average of the k-nearest training samples to assign a new data point.
  - Random Forest
    - Constructs multiple decision trees and combines their predictions through majority voting.
  - Support Vector
    - Finds the optimal hyperplane or decision boundary that maximizes the margin between different classes

Introduction/Motivation

- With industry & users shifting towards metaverse, as well as technological advancements, sensors to capture highly specific data has increased tremendously
- Privacy and safety concerns are becoming more of an issue as sensor technology and models evolve
  - Identification concerns due to uniqueness of data
- Garrido et. al find that there is a growing discrepancy between security research and **implementation by industry**
- Inspired by devices such as the FitBit and Apple Watch, we decided to create a gesture based on fitness



Background Papers chosen

- Exploring the Unprecedented Privacy Risks of the Metaverse
  - Realize that there is a lack of attention to the fact regular user experience (even in just a few minutes in a game) allow for personal and identifiable information *without* user realizing
  - Conduct study where 30 users play a "escape room" style game → within 10-20 minutes in VR time, they extracted 25 personal attributes with **every user** claiming they were not aware *when* the attributes were measured
- A privacy-preserving approach to streaming eye-tracking data
  - Realize that Eye-tracking technology provide powerful tool to allow critical applications to function; however there has been no attempt to limit identification of users
  - Create a "gatekeeper" that reduces *substantially* (from 85% to about 30%) while sacrificing only about 1.5 degrees of error
- Going Incognito in the Metaverse
  - Realize that while Metaverse is increasing in popularity, there are many security tools from "typical" computing that VR doesn't have access too → no Incognito Mode for users!
  - Create a Unity Plugin "MetaGuard" allowing users to customize what level of anonymity they want. Importantly, authors needed additive **and** multiplicative noise to guarantee differential privacy

Methodology for gesture: Jumping Jacks!

- Have users do 10 jumping jacks per session
  - Where at each session users start with hands on their side before proceeding to do 10 *continuous* jumping jacks
- Allows for capture of *multiple* attributes based on the activity. Some interesting ones to note:
  - Wingspan
  - Height
  - Speed of jumping jacks
- Outside of the scope of this project, but it's interesting to consider fitness apps in light of a security context

User Identification Classifiers

- Tested different classification models for latency and accuracy
  - Gradient Boosting
    - Builds multiple weak decision trees where each new tree corrects the errors of the previous tree.
  - Histogram-Based Gradient Boosting
    - Discretizes features into bins, which can accelerate the training process and improve performance.
  - K-nearest Neighbors
    - Calculates the average of the k-nearest training samples to assign a new data point.
  - Random Forest
    - Constructs multiple decision trees and combines their predictions through majority voting.
  - Support Vector
    - Finds the optimal hyperplane or decision boundary that maximizes the margin between different classes

Latency Results

Classifier	Latency (s)
Gradient Boosting	287.546827
Histogram-Based Gradient Boosting	6.330578
K-nearest Neighbors	2.781285
Random Forest	38.177482
Support Vector	203.057954

Accuracy Results: Model Predictions

Test FIle	JO_01	JO_02	JO_03	JO_04	JO_05	MA_01	MA_02	MA_03	MA_04	MA_05	MI_01	MI_02	MI_03	MI_04	MI_05
GB	JO	JO	JO	JO	MA	MA	JO	MA	MA	MA	MI	MI	MI	MI	MI
HGB	JO	MA	JO	JO	JO	MA	MA	MA	MA	MA	MI	MI	MI	MI	MI
KNN	JO	JO	JO	JO	JO	MA	MA	MA	MA	MA	MI	MI	MI	MI	MI
RF	JO	JO	JO	JO	MA	MA	MA	MA	MA	MA	MI	MI	MI	MI	MI
SV	JO	JO	JO	JO	JO	MA	MA	MA	MI	MA	MI	MI	MI	MI	MI

Task 3: Adversary Detection

- Because kNN was our best model, we used it for adversary detection
- Adversary detection is based on a set threshold on the confidence level
- Notably, confidence varied across each user:
  - **Jose: >98%**
  - **Michelle: 86-96%**
  - **Matt: 86-93%**
- However, adversary data generally ranged from 40-70%
- Therefore, we chose a *threshold* of 90% confidence to accept
  - There was one *breakthrough case* where an adversary was identified as Matt with 85.16% confidence
  - We would prefer to have Michelle and Matt take longer to authenticate as opposed to an adversary gaining access

Task 3: Adversary Detection (ctd.)

- Average latency for detecting authenticated users was 152ms
- Average latency for detecting adversaries was 172ms
- With a threshold of 90%:
  - Jose's accuracy is 100%
  - Matt's accuracy is 60%
  - Michelle's accuracy is 80%
  - Adversary accuracy is 100%
- With a threshold of 86%:
  - Model is 100% accurate (based on the data)