Smart Lock on SmartPhone

Boyang Cai, Xuerui Yan

UCLA

Outline

- Background
- Related Work
  - Previous Research
  - Related Platform
- Core Methodology
- Implementations and Evaluations
- Performance
- Demo
- Future Work

Background

- Security feature becomes a important thing for phones
- Different recognition method rather than using password
  - Pin, Gesture, Face, Fingerprint, FaceID, Voice, Iris
- People are tired of logging into their phone every time they wake up their phone
- Some features has been used to reduce the times to log in
  - extended “wait time” after turn off the screen
  - Trusted places by android
  - Trusted devices by android
  - Voice match by android
  - On body detection by android
- Our goal: Minimize the time a user have to log in on the basis of true positive surrounding safety check.

Related Work

- Intelligent Display Auto-lock Scheme for Mobile Devices
  - dynamically adjust the unlock time period setting of an auto-lock scheme based on derived knowledge from past user behaviors, current user activities or current user environment.
- On Body Detection by Google
  - Once you unlock your phone, it will stay unlocked while you're holding it or it's in your pocket.
  - Once you set it down, it will lock again.
- On-Body Location of a mobile phone
  - detect the body movement based on the sensor of the phone.
- Smartphone based data mining for fall detection
  - Use supervised machine learning to classify fall from usual daily activity such as sitting and jumping
- Implicit Authentication through Learning User Behavior
  - record user daily activity and behavior and compare it with the input to reduce the log in times

TensorFlow Lite

- a lightweight solution of TensorFlow
- TensorFlow is designed to provide a Machine Learning algorithm for different platform. (released on Feb 2017)
- example: image recognition, voice recognition...
- enables on-device machine learning inference with low latency and a small binary size.
- Just released in late 2017, only a small portion of library of TensorFlow is released
- Still under developer review so we will not use it for now

A person riding a motorcycle on a dirt road.

Using Scikit-Learn In AWS Lambda

- Not supported to save and retrieve the trained model (e.g. model.pkl) for future real-time prediction. Each time it is called, it has to train the model again.
- Not supported all native C libraries, thus affecting numpy, scipy, and scikit-Learn. Need to handle all related dependencies and configurations.

Core Methodology

Data Collection

- Collecting large amount of relevant data from the built-in sensors of smartphone, and form as evaluation datasets.
- Set up the automation data collection feature that can collect data in the backend every 30 seconds.
- User marking label (safe vs unsafe)
- Take advantage of the Google Spreadsheet and Google Script to store the updated collected data.

Core Methodology

Data Analysis

- Utilize elaborate machine (supervised) learning algorithms to analyze and train the datasets as different features in order to generate **Real-time Safety Check Model**.
- Use the trained model to real-timely receive the new data from the environment and evaluate the safety of surrounding environment for the phone.
- The result of safety check will collaborate to manage the screen lock on smartphone in real time.

Machine Learning Algorithms Selection

- Random Forest
- K-nearest neighbors (KNN)
- Naive Bayes
  - Gaussian NB
- Logistic Regression (logistic loss function + SGD)

Feature Selection

Original Features

Server time	
Local day	weekday/weekend
Local time	hour
acceleration in x axis of phone	m/s^2
acceleration in y axis of phone	m/s^2
acceleration in z axis of phone	m/s^2
calculated total acceleration	m/s^2
Latitude	degree
Longitude	degree
Accuracy	the radius of the accuracy circle in meters
Altitude	meter
Speed	m/s
Wifi Mac Address	N/A
Wifi SSID	N/A
Wifi Signal Level	N/A
Provider	location provider (network/gps)
Bluetooth	N/A

Label

- Safety (binary)

Feature Selection

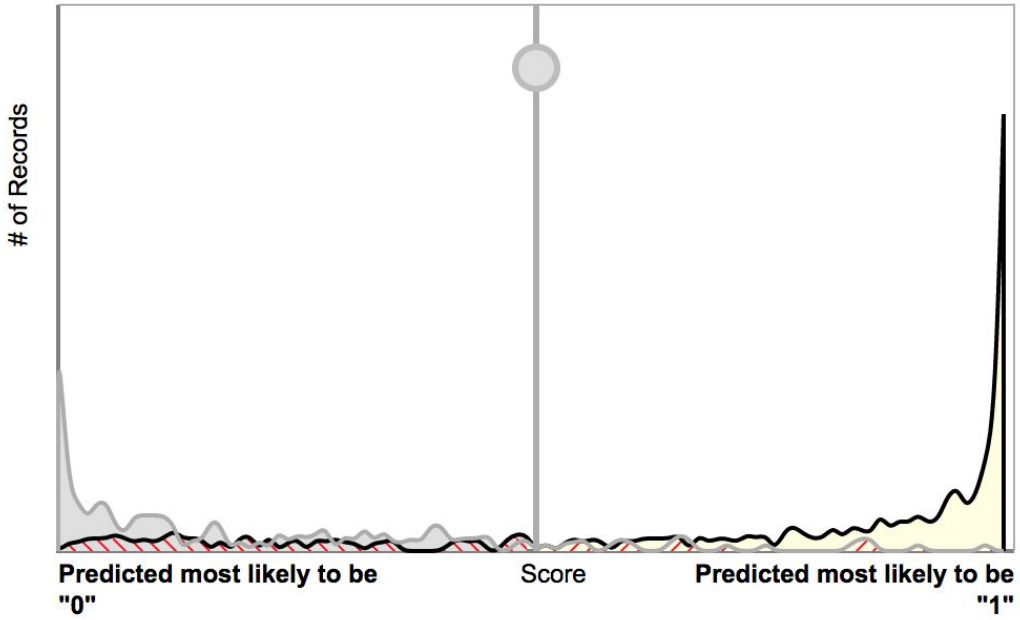
Selected Features

- Local day
- Local time
- Calculated total acceleration
- Latitude
- Longitude
- Accuracy
- Altitude
- Speed
- Wifi Mac Address
- Wifi SSID
- Wifi signal level
- Location provider

Local Training with collected datasets and selected features

Model	Model Accuracy Score
Random Forest	**0.910**
KNN	0.834
Gaussian NB	0.848
Logistic Regression	**0.947**

**Use scikit-learn as machine learning library in python for local training to select models for implementations. Best performances when train/test size: 0.7/0.3, with 10-fold cross validation.*



Analysis of Logistic Regression

Sensitivity and Specificity

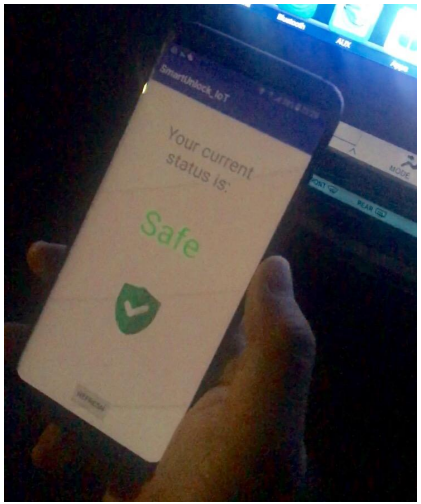
	True	False
Positive	73.4%	1.9%
Negative	21.7%	2.9%

Implementations and Evaluations

We implement two architectures of Smart Lock Systems

- Android App + Server (RPi) + AWS IoT + Random Forest
- Android App + Amazon Machine Learning + Amazon S3 + Logistic Regression

Diagram of Android App + Server (RPi) + AWS IoT + Random Forest

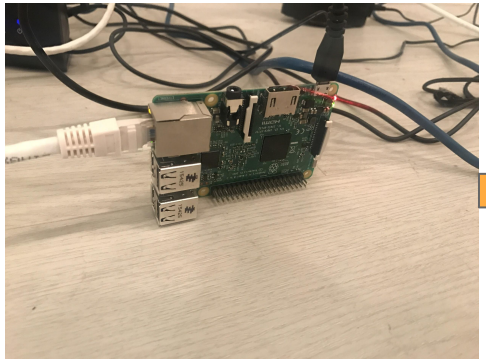


Android App

Environment Data

JSON by MQTT

Safety Check Result



Server (RPi)

ML Safety Check Model

AWS IoT

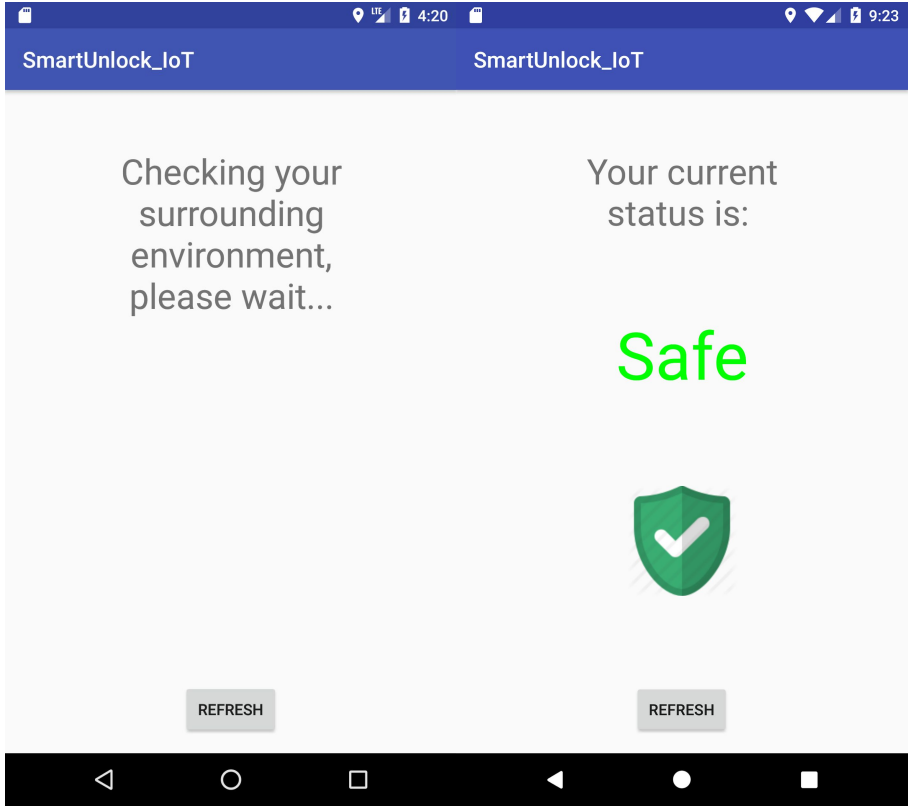
APP/UI Design

One Activity: MainActivity

Multiple thread that running continuously to collect different data:

- Wifi Monitor Thread
- Location Listener Thread
- Bluetooth Monitor Thread

Send data to IoT server once the app is open (represent a mock for wake up the phone)



Backend Logic

Server Initialization ➡

Server subscribing topic **phoneToRPi** ➡

Receiving phone check request via JSON in MQTT ➡

Server loading data from JSON to **Safety Check Model** and predict ➡

Return/publish the safety check result in JSON to topic **rpiToPhone** that the phone is subscribing

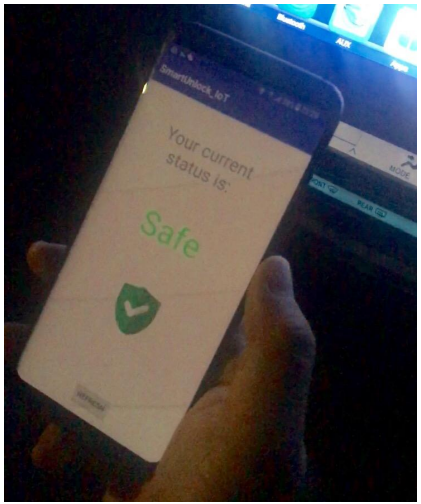
Advantages:

- Flexible for more different type of machine learning based models
- Low cost

Disadvantages:

- Weak Scalability
- Latency
    - Server computing power dependent

Diagram of Android App + Amazon Machine Learning + Amazon S3 + Logistic Regression



Android App

Environment Data

Amazon Cognito Sync

Safety Check Result

Amazon Machine Learning

amazon web services™

Retrieve data and train

amazon web services™ **S3**

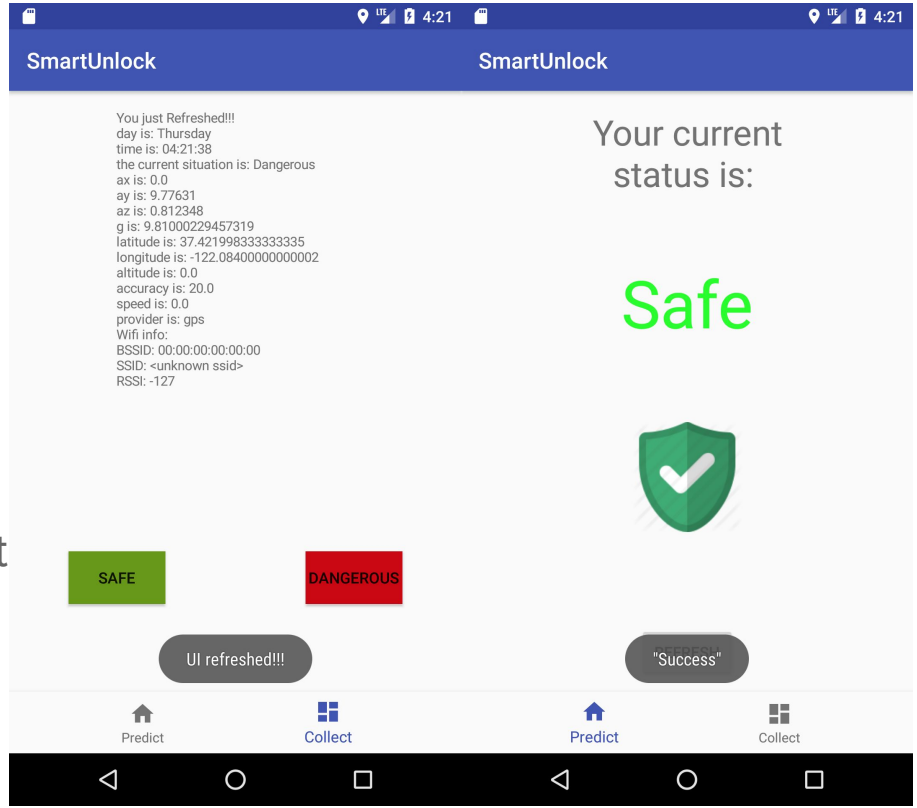
APP/UI Design

Similar to the APP design for IoT app but adding the data collection features.

Send data to AWS ML server instead using Http connection

Data Collection part:

User need to choose whether his current status is a “Save” status or a “Dangerous” status



Backend Logic

Storing preprocessed data into Amazon S3 ➡

Selecting features and parameters for building model and training data ➡

Amazon ML creating real-time prediction endpoint ➡

Amazon ML receiving phone check HTTP request through Amazon ML Predict API ➡

Amazon ML predicting safety result by pre-build **Safety Check Model** and return the result to phone with HTTP response

Advantages:

- Easy deployment and good scalability
- Low latency
- High reliability and security

Disadvantages:

- Pay by number of predictions and hourly reserved capacity
- Limited ML models in the lib

Performance

The true positive rate means the situation is safe and the prediction result is safe

SmartLock

Check for location detection:

at home: 40/40 shows a successful result

in car: 38/40 shows a successful result

(the bluetooth is not connected for the rest of two)

Check for movement detection:

on body: 2/40 shows a successful result

true negative rate:

outside: 32/40

Original Phone

Check for location detection:

at home: 39/40 shows a successful result

in car: 38/40 shows a successful result

(the bluetooth is not connected for the rest of two)

Check for movement detection:

on body: 39/40 shows a successful result

true negative rate:

outside: 40/40

Future Work

- Train model in real time based on user's real-time data.
- Adding the context states of the data as new feature in datasets for more accurate prediction.
- Fix basic features, and customize other features for user demand.
    - user can deselect those features that they thought are too private to provide. The program will analysis based on the feature that user provided.

First Demo

In home: https://vimeo.com/261200606

In private car: https://vimeo.com/261200767

In strange room: https://vimeo.com/261200785

**video password: ece209as*

Second Demo

References

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7153935&tag=1
Intelligent Display Auto-lock Scheme for Mobile Devices by Nai-Wei Lo
https://link.springer.com/content/pdf/10.1007/978-3-642-41674-3_63.pdf
Smartphone Based Data Mining for Fall Detection: Analysis and Design by Abdul Hakim
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6258136
Design and Implementation of an On-body Placement-aware Smartphone by Kaori Fujinami
https://www.ncbi.nlm.nih.gov/pubmed/22205862
Machine learning methods for classifying human physical activity from on-body accelerometers
https://www.sciencedirect.com/science/article/pii/S1877050917302065
Smartphone Based Data Mining for Fall Detection: Analysis and Design
http://elaineshi.com/docs/isc.pdf
Implicit Authentication through Learning User Behavior
https://serverlesscode.com/post/deploy-scikitlearn-on-lamba/
Using scikit-learn in AWS Lambda
https://docs.aws.amazon.com/machine-learning/latest/dg/requesting-real-time-predictions.html
Requesting Real-time prediction
https://www.androidpolice.com/2015/03/20/trusted-butts-new-on-body-detection-smart-lock-mode-in-android-seems-to-be-hitting-some-devices/
New “On-Body Detection” Smart Lock Mode in Android Seems To Be Hitting Some Devices

Q&A