BL(u)E CRAB

Bluetooth Low Energy Connection Risk Assessment Benchmarking

by

Dylan Christopher Conklin

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science
in
Computer Science

Thesis Committee:
Primal Pappachan, Chair
Roberto Yus
Bart Massey
Nirupama Bulusu
Wu-chang Feng

Portland State University
2026

Abstract

The usage of Bluetooth Low Energy (BLE)-based tracker devices for stalking has become a salient privacy concern. Detecting unwanted or suspicious trackers is challenging due to their cross-platform compatibility issues, inconsistent detection methods, and lack of an industry-wide standard for detecting malicious devices. *BL(u)E CRAB*, Bluetooth Low Energy Connection Risk Assessment Benchmarking, scans data and generates risk factors about nearby devices to classify them as suspicious or not. These risk factors include the number of encounters the user had with a device, the duration of time a device has been near the user, the distance a device has traveled with the user, the number of areas the device appeared in, the device's proximity to the user, and the stability of the device's signal strength. After collecting this information, *BL(u)E CRAB* uses one of several classifiers adapted to these risk factors to determine whether a device is suspicious or not. We have integrated a multitude of new device classifier methods, including single- and multi-dimensional clustering methods. We evaluated these classifiers against existing methods using a diverse dataset of BLE tracker data in various real-world scenarios. The benchmark results show the efficacy of different classifiers in identifying suspicious BLE trackers. We also developed a working prototype of *BL(u)E CRAB* that is easy to use, customizable, and can easily integrate other classifiers.

Dedicated to my wife, Carly Conklin, whose endless love and support has made this academic journey possible.

Acknowledgments

Thank you to my wife, Carly Conklin, for your encouragement, patience, and unwavering support throughout my academic journey at Portland State University. I would never have made it through school without you working so hard to support us.

Thank you to Primal Pappachan for being the best advisor I could ever ask for. Your guidance and mentorship throughout this research project challenged my assumptions and changed the way I think about problems. You created a healthy learning environment that encouraged collaboration, discussion, and communication that led to significant breakthroughs in this project.

Thank you to Orobosa Ekhator for collaborating with me on this project. Your contributions have been invaluable in a way that significantly altered the direction of this project for the better. Your constant experimentation and willingness to explore new ideas made clear the direction we should take at many crucial junctions.

Thank you to everyone in the DIPr Lab for your collaboration and support on our respective projects. Although I don't always understand you guys' presentations right away, the ideas and strategies shared with you all have been a joy to experience.

Thank you to the committee for your feedback and collaboration. I had different experiences with each committee member, but each of you has contributed to the project and my academic journey in a significant way, which is why I am proud to have all of you listed as committee members for this project.

Thank you to my family for your support. Without my parents, grandparents, and siblings, I would not be the person I am today.

Thank you to my wife's family for your support and for letting us stay with you while I attend school.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Motivation

The motivation for this project stems from the increasing prevalence of Bluetooth Low Energy (BLE) tracker devices and their associated object location networks. Ensuring the safety of users and non-users is an essential feature as these devices are integrated into daily life. *BL(u)E CRAB* aims to address potential shortcomings in BLE-based object location networks to notify users of suspicious trackers and improve the overall security of potential stalking victims using BLE technology.

Due to their cost-effectiveness, small size, and ease of use, BLE trackers have become increasingly popular for locating lost items. However, these features also make them effective tools for bad actors to track individuals without their consent [7]. The ability to discreetly place a BLE tracker on a person or their belongings poses significant safety concerns to users, as well as challenges for manufacturers to implement effective stalking prevention measures in these networks.

Although the implementation details of proprietary object location networks are not publicly available, there are known limitations of their current implementations, including attacks on the servers that facilitate communication throughout the network [30, 33]. Many services cannot adequately notify users of potential stalking threats if the suspicious tracker does not participate in their network [4, 9, 16, 20], or the measures taken may not be able to identify malicious

trackers in a reasonable amount of time [31]. Publicly available methods deployed in open source applications use static thresholds to evaluate suspicious behavior [17–19], which limits their efficacy in evaluating the activity of suspicious devices.

The primary objective of this project is to develop a system that can efficiently detect and mitigate stalking threats posed by BLE trackers. *BL(u)E CRAB* seeks to provide a robust solution that enhances user safety and maintains user privacy.

1.2 Methodology

The application was designed to scan for Bluetooth Low Energy (BLE) devices and collect relevant data such as the user's location, device identifiers, signal strength, and timestamps. The data collection process included deploying the application in various environments to ensure a diverse dataset. We also used the publicly available dataset provided by BLE-Doubt [18]. We used the data to benchmark the classifiers implemented in the *BL(u)E CRAB* application. The benchmarking suite was used to evaluate the baseline and novel classifiers based on their F1 scores.

1.3 Research Contributions

This thesis offers the following research contributions:

1. We identified several limitations and challenges associated with implementing a BLE tracker risk assessment model in practical applications.

2. We developed a novel risk assessment framework that accurately assesses the risk Bluetooth Low Energy (BLE) trackers pose to users based on various risk factors.

3. We developed a mobile app that implements our risk assessment framework to notify users of suspicious BLE trackers in real-time.

4. We developed a framework for evaluating the efficacy of suspicious device classifiers in predicting the risk that BLE trackers pose using identified risk factors.

5. We evaluated the efficacy of our approach with 7 classifiers in 14 third-party datasets and 3 first-party datasets. The multi-dimensional clustering approach outperformed the baseline approaches across many datasets, while the single-dimensional approaches showed good results but fell short of the baselines.

6. We prescribe measures manufacturers can take to mitigate the risks associated with BLE tracking networks.

2 Background

This chapter provides background information on the technologies and concepts relevant to this thesis, covers the basics of Bluetooth Low Energy (BLE) and object location networks, and provides summaries of related work in the domain of BLE tracking technologies. This section focuses in particular on the approaches taken by previous research to address these limitations.

2.1 Bluetooth Low Energy (BLE)

Bluetooth Low Energy (BLE) is a wireless personal area network technology designed for short-range communication with low power consumption [13]. BLE is a separate protocol from Bluetooth Classic, and these protocols have almost no compatibility with each other, despite communicating over the same 2.4 GHz radio frequency. BLE is compatible with many types of devices, including smartphones, tablets, laptops, and IoT devices. BLE is also widely used in small battery-powered devices due to its low power consumption, making it ideal for tracking devices that need to operate for months or years without battery replacement.

BLE devices communicate by sending advertising packets at regular intervals that contain information about the device, such as access address data, protocol data, and a payload [29]. The payload includes UUIDs, service data, and manufacturer-specific data [32]. *BL(u)E CRAB* is able to access this information through an API to gather data about the device name, device type, manufac-

turer, transmission strength, and metadata that can be used to identify suspicious devices.

2.2 Object Location Networks

Object location networks are systems that leverage a network of devices to determine the location of objects, often using wireless communication technologies such as Global Positioning Systems (GPS) and Bluetooth Low Energy (BLE) [7]. These networks are commonly used to help users locate lost or misplaced items.

High-power devices, such as smartphones, tablets, and computers, make up the backbone of object location networks. These devices are typically equipped with stronger networking capabilities that can determine their exact location using GPS. High-power devices typically use more battery power to perform networking tasks, and are intended to be used for more than just object location [30]. High-power devices can also participate in object location networks to update their location so that their owners can track those devices as well. There are some commercially-available trackers that use 4G LTE and GPS to provide location updates to their associated object location network [5], but these devices are more expensive to use and require more battery power to use.

Low-power BLE trackers that take advantage of these networks can be attached to small items, such as keys, wallets, or bags. Low-power tracking devices communicate with nearby high-power devices in the network to relay their location information back to the tracker's owner [11]. To do this, a low-power device will

advertise packets containing service information correlated with the object location network. When the tracker finds a high-power device that participates in that network, the tracker will share data that the high-power device will forward to the server responsible for maintaining the object location network, along with the location of the high-power device. The server then updates the tracker's location, allowing the owner to see where their item is located.

The widespread adoption of high-power devices and object location networks provides accurate and reliable location information for lost items [7]. Some of the most common tracker devices are AirTags, Tiles, and SmartTags. A recent study observed that AirTags and other Find My-compatible devices were the most widely used tracking devices, accounting for more than 70% of all tracked devices observed. Tile devices were the second most used tracking device [7].

2.3 Tracker Services

Numerous tracker services have been developed to help users locate their belongings using BLE technology. These services typically involve a combination of centralized servers to post device location updates, high-powered mobile devices with applications that work together with servers to keep device location data up-to-date, and low-powered trackers to provide object location [11]. Table 2.1 lists some of the most popular tracker services and their respective manufacturers.

Proprietary services such as Find Hub, Find My, SmartThings Find, and Tile are compatible only with devices that participate in their services. Apple and

Google have partnered on an industry specification to address unwanted tracking by supporting tracking alerts for Find My devices on Android devices [10]. Chipolo does not run their own object location network. Instead, they make trackers for use with Find My or Find Hub. Table 2.2 compares existing tracker devices and their compatibility with popular tracker networks. Third-party applications such as AirGuard and *BL(u)E CRAB* are not limited to interaction with devices participating with a particular service and are compatible with all device types, making them more versatile for detecting various tracker devices, including custom devices.

Based on the information in Table 2.2, the ability of proprietary tracking services to detect unwanted tracking is limited to first-party devices or third-party devices that participate in the service. This closed ecosystem approach limits the ability of users to detect trackers from other manufacturers or custom devices that may be used for malicious purposes. Users may be particularly vulnerable to custom devices that mimic legitimate devices in the network [30], so having a detection service that can detect trackers of all types of devices is crucial for user safety.

Apps that interact with tracker devices or participate in object location networks often implement features found in tracker finder apps, security apps, or both. Finder apps focus on helping users locate their own devices, while security apps focus on alerting users to the presence of unknown trackers in their vicinity. Apps such as Find My and Tile implement both functionalities, while apps such as *BL(u)E CRAB* and AirGuard [19] focus solely on security features to help improve

user safety when other applications may fail.

Service	Developer
AirGuard [19]	TU Darmstadt
Find Hub [27]	Google
Find My [8]	Apple
SmartThings Find [21]	Samsung
Tile [4]	Life 360

Table 2.1: Existing tracker services and their developers

Tracker Device	Manufacturer	First-Party	Find My	Find Hub
Chipolo [15]	Chipolo		✓	✓
AirTag [2]	Apple	✓	✓	
SmartTag [3]	Samsung	✓		
Tile [6]	Life 360	✓		

Table 2.2: Existing tracker devices and network compatibility

2.4 Threat Model

We assume the adversary, or the person planting the suspicious device on a user, is interested in tracking a user using only commercially-available off-the-shelf BLE trackers (e.g., AirTag, Tile, SmartTag, etc.). These tracking devices are frequently used for malicious purposes such as stalking a person or their preferred mode of transport [7]. The adversary utilizes an object location network (Section 2.2) that connects to the smartphone of the tracked entity or other nearby smartphones. Although many custom-made devices can be built by experts to circumvent security features supported in consumer apps, we assume that perpetrators will only utilize off-the-shelf BLE trackers by any commonly-used object location service. Non-

BLE-based devices, such as GPS-based trackers, are not considered in the threat model of *BL(u)E CRAB*.

The tracking device utilizes an object location network (Section 2.2) that utilizes the location of the victim's smartphone to report it as its own location. We trust the object location network provider (e.g., Apple, Google) and believe that they do not actively collude with the adversary, though the adversary utilizes the provider's legitimate infrastructure for malicious ends. We assume that our adversary hasn't modified the hardware or software of the tracker device and utilizes the manufacturer specific application to track its location. The adversary has physical access to the tracker only during the initial setup and placement phase. Post-placement, the adversary has remote access to location updates via the companion app but cannot communicate directly with the tracker.

"Suspicious tracker" refers to any device that exhibits unusual behavior compared to other devices in the dataset. Risk factors are measurable indicators of the threat a suspicious tracker poses to the user, such as the duration of time a device has traveled with the user and the distance a device has traveled with the user. Suspicious trackers will have elevated values for one or more risk factors, and *BL(u)E CRAB* includes a variety of classifiers as a means to identify suspicious devices. We do not make assumptions about where the tracker is hidden or the environment about the user, instead letting the classifier account for these variables.

We do not attempt to prevent location tracking via a network of beacons that may be able to re-identify the user. Beacon-based object location networks may

be deployed at a small-scale in a hospital, office building, or any other facility that has a network of permanently-placed beacons that can identify devices' locations via re-identification. These types of object location networks rely on high-power beacon devices that passively sniff known identifiers in a fixed location and update a registry with specific device locations.

2.5 Related Work

In this section, we provide a summary of previous work involving BLE trackers. The differences between these existing methods and BLUE CRAB are then discussed.

2.5.1 AirGuard

AirGuard [22] is a mobile application that detects nearby BLE trackers by scanning for BLE advertisements. The application collects data on nearby devices, including their MAC address (or other available identifiers), RSSI, and advertising data. The application then uses these data to identify potential trackers by checking for a duration of time of at least 30 minutes near the user, at least three detections, a minimum distance of 400 meters traveled with the user, and if there has been a security alert for the device within the past seven hours. If a device meets these criteria, the user is immediately notified of the suspicious device.

The researchers reverse-engineered the Find My network's tracking detection algorithm to create a classifier that mimics the behavior of the Find My network's

tracking detection algorithm. Due to the high time threshold value used by AirGuard, *BL(u)E CRAB* uses part of this reverse-engineered classifier as a baseline to compare the performance of *BL(u)E CRAB*'s classifiers.

AirGuard focuses on detecting suspicious trackers based on static time and distance traveled with user thresholds, which may not be effective in all scenarios. *BL(u)E CRAB* uses a multitude of classifiers that analyze risk factors using dynamic thresholds, allowing for a more adaptable approach to identify suspicious devices. The approach of AirGuard to classifying suspicious devices is similar to *BL(u)E CRAB*, but AirGuard and *BL(u)E CRAB* each provide features that the other does not. For example, AirGuard provides advanced features such as native notifications, background scanning, and a mechanism for finding suspicious trackers when devices are identified as a suspicious device. *BL(u)E CRAB* provides advanced features for report generation and data analysis on Bluetooth tracker datasets, the option to use multiple classifiers to identify suspicious devices, and data management features for researchers.

2.5.2 BLE-Doubt

BLE-Doubt [14] proposes a Topological Classification algorithm to quickly detect suspicious BLE trackers. The proposed classifier uses ϵ-connectedness to group the locations of devices over time and flag devices with high ϵ-connectedness. The researchers compare their proposed Topological Classification method with three baseline methods: a duration classifier that flags devices that have traveled with

the user for longer than some time threshold, a diameter classifier that flags devices that have traveled with the user farther than some distance traveled with the user threshold, and a hybrid classifier that flags devices that were flagged in both the duration classifier and the diameter classifier.

All three baselines had good recall, a low false negative rate, and a high false positive rate, leading to low accuracy in devices classification due to overly cautious behavior. The topological classifier had slightly worse recall and a worse false negative rate, but better accuracy due to the lower false positive rate. This study concludes that the topological classifier is better suited for detecting maliciously deployed trackers due to its lower false positive rate, which leads to fewer non-suspicious devices being flagged as suspicious.

BLE-Doubt focuses on detecting suspicious trackers using a single classifier based on a topological classifier and ϵ-connectedness to cluster device locations over time. *BL(u)E CRAB* implements a similar approach to calculate various risk factors, but dynamically determines the thresholds used and expands the range of risk factors that can be used to identify suspicious devices. BLE-Doubt is also not a commercially-available product like AirGuard is, instead focusing on the efficacy of their topological classifier.

2.5.3 Please Unstalk Me

This study attempts to quantify the prevalence of stalking via Bluetooth trackers and identify effective countermeasures [7]. The researchers found that stalking

via Bluetooth trackers is a significant problem, with nearly 20% of the surveyed participants reporting being tracked via these devices. Women were the most likely to be victims of stalking by male perpetrators, while men were slightly less likely to be victims of stalking but were targeted by both male and female perpetrators at a nearly equal rate.

To help remedy this problem, the researchers recommend that smartphone manufacturers integrate more effective tracking detection mechanisms into their devices by default. Ideally, this would involve the creation of a set of standards that manufacturers should follow when developing these devices and services. Creating such a standard and enforcing compliance would take years and is not guaranteed to succeed due to the disparate nature of these devices and the ability to create custom devices with abnormal behavior. They also recommend the creation of stalking prevention apps that can identify devices independently of any existing standard.

Researchers also recommend a privacy-by-design approach when developing these devices to limit the ability of device owners to misuse tracking devices for stalking. This could include reducing the device's location accuracy being reduced after being away from the owner for a certain period of time, or implementing stalking prevention features on the manufacturer side. Devices that never or seldom rotate their MAC address, such as those that have a low power mode, could also be re-identified over long periods of time, which could be used to alert perpetrators that their victim is nearby. Increasing the frequency of MAC address rotation could help mitigate this issue by making it more difficult for stalkers to re-identify

devices placed on their victims.

This research explores the severity of Bluetooth trackers being used to stalk people and highlights the urgency needed for effective detection mechanisms. *BL(u)E CRAB* addresses this need by providing a comprehensive solution for detecting suspicious BLE trackers using multiple classifiers and risk factors. Although researchers recommend that smartphone manufacturers integrate tracking detection mechanisms into their devices, *BL(u)E CRAB* provides an independent solution that can be used across various devices and platforms without relying on manufacturer cooperation.

2.5.4 DeTagTive

BLE devices avoid re-identification by rotating their MAC addresses. This behavior is crucial for the privacy of the device owner, but this feature also makes detection difficult. DeTagTive proposes an algorithm to detect rotating MAC addresses using non-changing parameters in device metadata, including signal strength and advertisement information [17].

Researchers try to prevent a device from being misclassified as a new device when the MAC address changes while in range of the scanner. This is done using the device's signal strength (RSSI) and advertisement intervals. This method does not require software changes, cooperation from device manufacturers, or special hardware beyond what is found on a cell phone.

The researcher's algorithm links rotated MAC addresses associated with devices

to track the length of time a device is near the scanner. This association is made by comparing the unchanging characteristics of the device (such as RSSI strength and advertised data) to devices currently connected to the scanner. The signal strength of the device remains strong as long as it is near the scanner, and the advertisement data do not change. When a device disappears, the algorithm looks for a potential match using a score calculated as the sum of the absolute value of the difference between the mean RSSI strength, the RSSI standard deviation, and the mean advertising interval. The lowest match score among all the candidates is linked as the new device.

DeTagTive focuses on linking rotating MAC addresses to track devices over time using unchanging parameters in the device metadata. This research provides a solution to this problem with the intention of helping identify suspicious devices but does not attempt to build a comprehensive device classifier. *BL(u)E CRAB* provides a more comprehensive approach to detecting potential threats and uses a MAC address change detection feature similar to DeTagTive.

2.5.5 Who Tracks the Trackers?

Many object location networks have introduced features to prevent BLE trackers from being used to track people, but in this study, researchers demonstrate that it is possible to create custom devices that can participate in these networks while circumventing security measures that would alert users to its presence. This offensive approach differs from the approach of *BL(u)E CRAB* and the aforemen-

tioned works, which are defensive strategies that use unmodified devices to identify trackers. This study focuses on Apple's Find My network and AirTags, but similar techniques could be used to attack other object location networks. The researchers developed three techniques to avoid detection by the Find My network's tracking detection algorithm.

Bit Flipping: The researchers observed that certain bits in the messages sent out by lost AirTags revealed hardware information about the tracker's battery. An unwanted tracking device can avoid triggering an alert by modifying its bit information to mimic the iPhone's format and by modifying how the network reacts to signals from the device.

Rotating Keys: The researchers observed lost AirTags advertising repeated messages over a long period of time, but custom devices can rotate keys to send location reports, which can be re-constructed to provide the user's location history without triggering an alert. This attack works by increasing the amount of keys sent to the server to prevent alerts from being triggered.

Generating Keys on Device: A custom device could generate keys on device to prevent keys from being repeated. This attack is similar to the rotating key attack but would ensure that no keys are repeated, making it robust to repeated key detection.

This research explores server-side vulnerabilities of object location networks by exploiting the service with a custom device. This is one example of an offensive approach to tracking that the service running the object location network is unlikely to detect. The techniques presented in this research highlight the need for

robust detection mechanisms that can adapt to evolving threats, and on-device detection methods could be a way to mitigate a similar threat.

3 Features

This chapter describes the features that *BL(u)E CRAB* offers. These features include the ability to customize the app experience to advanced use-cases, adjust the parameters used when identifying suspicious devices, and manage datasets collected within the app. Each feature is designed to enhance the user experience and provide flexibility for a variety of use-cases in a user-friendly way.

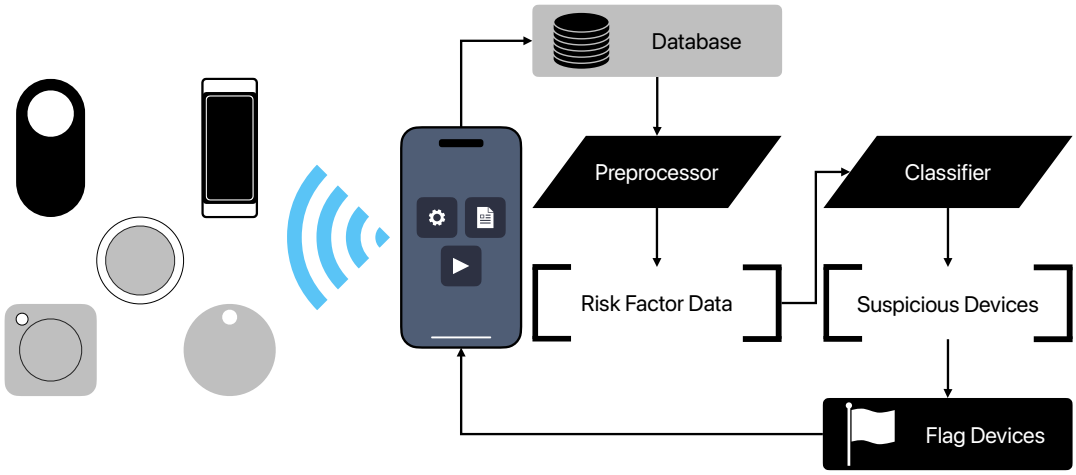

Figure 3.1: Architecture of *BL(u)E CRAB*

3.1 Implementation Details

BL(u)E CRAB is a mobile app developed using the Flutter framework. Flutter is a cross-platform development platform that can deploy applications to desktop

devices (e.g. Windows, macOS, and Linux), mobile devices (e.g. Android and iOS), and the web. Due to limitations in development time, limited platform support in third-party APIs, and the nature of the problem *BL(u)E CRAB* attempts to solve, *BL(u)E CRAB* is only supported on Android, iOS, and macOS.

We chose Flutter because of the flexibility it provides for cross-platform support and user experience. However, it does not provide out-of-the-box support for advanced platform features such as native notifications and background scanning. Native notifications could potentially be added, but we chose not to prioritize it due to the lack of background scanning, when it would most often be used.

Figure 3.2: Screenshots of *BL(u)E CRAB*: scanner view (left), report view (left center), device detail view (right center), and map view (right)

3.2 App Modes

BL(u)E CRAB has many features to offer that may be essential for users with specific use cases, but many of the more advanced features are not useful for casual users. *BL(u)E CRAB* attempts to balance user-friendliness and feature demands, so it uses app modes to modify behavior and expose features. None of the app modes affect the parameters that users can modify in the settings.

App Mode	Core Features	Share Report	Generate Reports	Delete Dataset	Load Sample Data	Live Dataset Statistics
Default	✓					
Developer	✓	✓	✓	✓		
Demo	✓			✓	✓	✓
Data Collection	✓	✓				

Table 3.1: Bonus features enabled in each app mode

Default Behavior and Features - The default behavior of the app includes starting and stopping the scan, changing the parameters in the settings, and viewing suspicious device data. *BL(u)E CRAB* will search for devices and periodically check if there are any suspicious devices in the dataset.

Developer Mode - The developer mode includes everything in the default behavior, but also enables buttons to delete scan data, share scan data, and run tests. Information about the test suite can be found in Section 3.10. The Developer mode is intended for researchers and developers who want to generate detailed reports about the collected datasets.

Demo Mode - The demo mode includes everything in the default behavior

but displays buttons to delete existing scan data and load a sample dataset to allow users to quickly test the app. During scanning, the app will display information about the dataset, including the number of devices, data points, and suspicious devices in the dataset. The demo mode is intended to demonstrate how the app works with potential new users, researchers, and contributors.

Data Collection Mode - Data collection mode enables the scanning device to export and share collected data and disables the classifier. Data collection mode is intended for users who want to collect datasets as efficiently as possible and do not care about being alerted to suspicious devices near them.

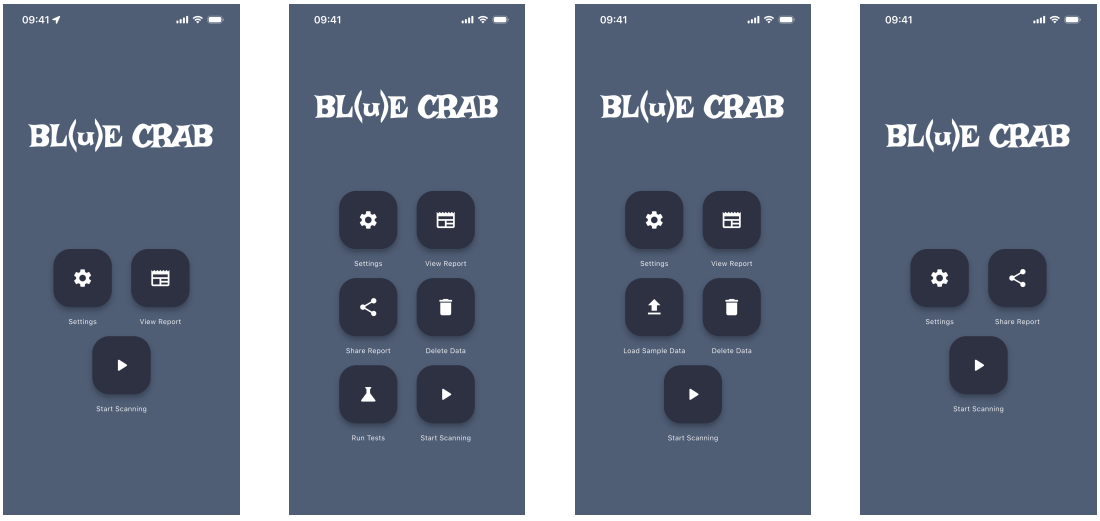

Figure 3.3: Screenshots of the four app modes in *BL(u)E CRAB*: Default (left), Developer (left center), Demo (right center), and Data Collection (right)

3.3 Customization

BL(u)E CRAB allows the user to customize the time and distance parameter thresholds used to calculate risk factors to better suit their environment and use case. The default values are set to 10 seconds and 30 meters, respectively, but users can adjust them based on their specific needs. The time parameter is used to help reconstruct the user's path for drawing on the map and calculating the risk factors. The distance parameter is used to filter out data that are near safe zones and to calculate the distance between locations for the area count risk factor.

3.4 Safe Zones

There may be data that the user wants to exclude from analysis based on known safe locations, such as their home or workplace. *BL(u)E CRAB* allows users to define safe zones to exclude from data analysis. Safe zones are defined using GPS coordinates and include the area around the location with the radius defined by the user in the app settings. When the user is within a safe zone, *BL(u)E CRAB* will still collect data but will filter the relevant data when evaluating suspicious devices. Users can add safe zones in the app settings.

3.5 MAC Address Rotation Detection

BLE devices may periodically change their MAC addresses to prevent tracking and enhance user privacy [17]. This behavior, known as MAC address rotation

or randomization, can complicate the process of classifying devices over longer periods of time, because we cannot reliably identify devices based on their MAC addresses alone. The platform on which the app runs may not provide information about whether a device has changed its MAC address and may not even allow developers to access the MAC address value at all [12, 28], instead opting to use a unique identifier to reference the device. *BL(u)E CRAB* refers to this identifier in place of the MAC address when performing MAC address rotation detection, because peripheral devices appear as new devices in both cases.

Before classifying devices, *BL(u)E CRAB* attempts to link devices that have rotated their MAC addresses using the DeTagTive algorithm [17]. This algorithm analyzes the average RSSI values, the standard deviation of the RSSI values, and the transmission frequencies of candidate devices to identify pairs of devices that are likely to be the same physical device using different MAC addresses.

The algorithm first identifies devices that may have changed their MAC addresses by selecting devices with a strong signal that have been near the user for some duration of time and have stopped emitting packets. Then it identifies devices that could potentially be the same device under a new identifier. Potential matches are found by selecting devices that started emitting packets shortly after the original device stopped and that have a match score below the threshold, indicating that the two devices have similar RSSI values and durations between packets emitted.

BL(u)E CRAB improves this method by adding an additional filter when selecting candidate devices to only consider devices that have metadata similar to

the original device, including device name, manufacturer data and service UUIDs. This helps reduce incorrect matches when finding devices that may have similar signal characteristics but are different types of devices.

Our preliminary results suggest that these changes improve MAC address rotation detection by reducing the number of faulty connections. In one of our datasets, the original algorithm made one correct connection and thirty-eight false connections. By adding metadata matching, we reduced the false connections to twelve. In another dataset, the original algorithm made one correct connection and eight false connections, but our improvements reduced the false connections to two.

The pseudocode for our variant of the DeTagTive algorithm is provided in algorithms 1, 2 and 3. 1 selects devices that have exhibited a strong signal, have been near the user for some duration of time, have stopped emitting packets, and have metadata to identify them. It then uses 2 to find potential matches for each candidate device, and merges the matching devices. 2 identifies potential matches for a candidate device by selecting devices that started emitting packets shortly after the candidate device stopped, have short advertisement durations, and have similar metadata. It then calculates a match score for each potential match using 3 and selects the device with the lowest match score as the match. 3 calculates the match score for a candidate device and a potential match by comparing their average RSSI values, standard deviation of RSSI values, and average transmission durations. The lower the match score, the more likely it is that the two devices are the same physical device under different identifiers.

Algorithm 1: DeTagTive

Constants : D (All devices in dataset)

$d_{strong} \leftarrow D.where(traceIsStrong)$;

$d_{long} \leftarrow D.where(traceIsLong)$;

$d_{ended} \leftarrow D.where(traceEndedRecently)$;

$d_{hasMetadata} \leftarrow D.where(traceHasMetadata)$;

$d \leftarrow d_{strong} \cap d_{long} \cap d_{ended} \cap d_{hasMetadata}$;

for $d_{cand} \in d$ **do**

  $d_{match} \leftarrow findMatch(d_{cand})$;

  if $\exists d_{match}$ **then**

    $d_{match} \leftarrow \text{merge}(d_{cand}, d_{match})$;

  end

end

Algorithm 2: FIND_MATCH

Input : d_{cand} (candidate device)

Constants : D (All devices in dataset)

$match \leftarrow$ NULL ;

$d_{inWindow} \leftarrow D.where(startsInWindow)$;

$d_{shortAdvs} \leftarrow D.where(hasShortAdvertisements)$;

$d_{sameMeta} \leftarrow D.where(hasSameMetadataAs(d_{cand}))$;

$d \leftarrow d_{inWindow} \cap d_{shortAdvs} \cap d_{sameMeta}$;

for $d_{match} \in d$ **do**

  $score \leftarrow matchScore(d_{cand}, d_{match})$;

  if $\exists match$ **then**

    if $score < matchScore(d_{cand}, match)$ **then**

      $match \leftarrow d_{match}$;

    end

  else

    $match \leftarrow d_{match}$;

  end

end

return $match$;

Algorithm 3: MATCH_SCORE

Input : d_{cand} (candidate device's data points)

: d_{match} (device's data points)

Output : The match score for d_{cand} and d_{match}

Constants : D (All devices in dataset)

$RSSI_{avg} \leftarrow avgRSSI(d_{cand}) - avgRSSI(d_{match})$;

$RSSI_{stdDev} \leftarrow stdDevRSSI(d_{match}) - stdDevRSSI(d_{cand})$;

$RSSI_{avgTrans} \leftarrow avgTransDuration(d_{match}) - avgTransDuration(d_{cand})$;

$score \leftarrow |RSSI_{avg}| + |RSSI_{stdDev}| + |RSSI_{avgTrans}|$;

return $score$;

3.6 Classifying Devices

During scanning, *BL(u)E CRAB* periodically checks for suspicious devices in the dataset to alert users about potential unwanted tracking. *BL(u)E CRAB* provides a variety of device classifiers (Section 5), and allows the user to select the desired classifier in the settings. This classifier is used to obtain a list of identifiers associated with the devices in the dataset, which can then be displayed to the user. Devices that are marked as suspicious by the classifier are listed in the report view for further review by the user. Developers can create classifiers using any method they choose, but classifiers typically use at least one type of risk factor (Section 4) and at least one threshold. The thresholds used in classifiers can be static or dynamic. Static thresholds are values that never change during the lifetime of the

application, while dynamic thresholds are re-adjusted each time the classifier is invoked.

3.7 Data Collection

The user can begin scanning for nearby BLE devices by pressing the "Start Scan" button on the scanner view. This initializes the scanning process, enabling location tracking and listening for nearby devices. Location updates are disabled when the user is not scanning. The scan can be stopped at any time by pressing the "Stop Scan" button, which stops the scanning process, disables the location tracker, writes the collected data to persistent storage, and automatically redirects the user to the list of suspicious devices.

3.8 Data Storage

All scanned data are written to storage when the scan is stopped or when suspicious devices are detected. This includes the user's location history, as well as all the devices' identifiers, metadata, and scan data. The data is stored in a compact JSON format that allows for easy analysis in future sessions.

BL(u)E CRAB was initially designed to read the BLE-Doubt dataset so that we could use their datasets with our classifiers. The BLE-Doubt dataset separates device identifiers from detection data, resulting in duplicated MAC addresses, location values, and timestamps for each detection, as seen in Figure 3.4. However, storing large amounts of duplicate data posed a significant challenge due to memory

and API limitations. To remedy this, we designed our own data storage schema (Figure 3.5) to reduce duplication as much as possible. Our schema duplicates timestamps, but does not duplicate location data or MAC addresses.

The data storage schema *BL(u)E CRAB* uses trades off data storage space for increased processing time. When reading from our dataset, *BL(u)E CRAB* must reconstruct the location entries on a per-device basis by matching the timestamps in the device's detections entry to the most recent (i.e., "last seen") timestamp in the user's location history (i.e., "last seen at"). This process is more computationally expensive than reading the data as-is, but allows *BL(u)E CRAB* to operate within the API memory constraints. To avoid excessive file sizes, *BL(u)E CRAB* limits the number of entries to one per second by rounding the timestamps down to the nearest second. The format of the *BL(u)E CRAB* datasets is significantly smaller than the BLE-Doubt format, as shown in Figure 3.6, making data transfer and storage more efficient.

BL(u)E CRAB supports importing and exporting data in its own format, but only supports importing data in the BLE-Doubt format. A sample dataset in *BL(u)E CRAB* format can be found in appendix B.

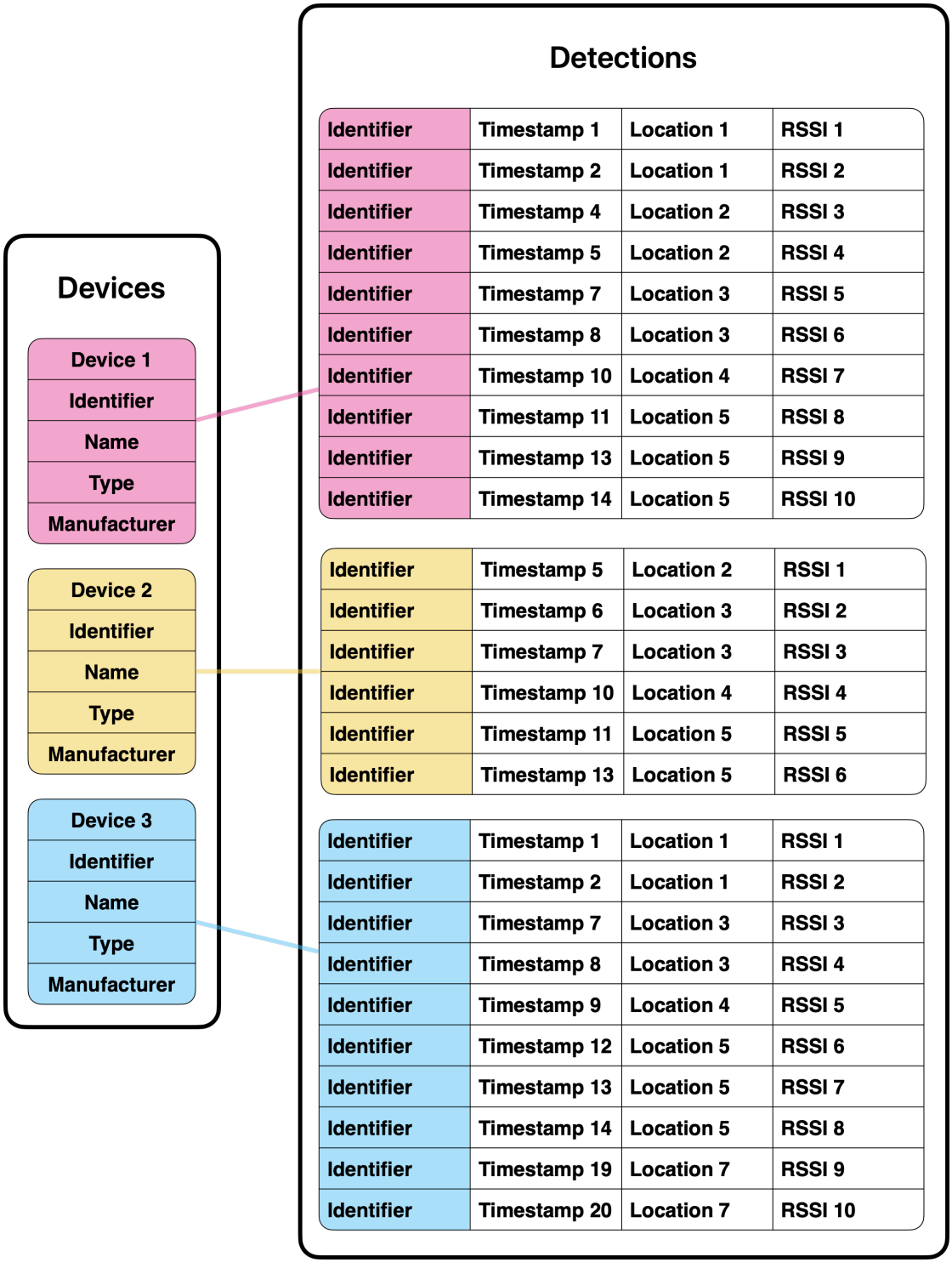

Figure 3.4: Simplified model of BLE-Doubt dataset

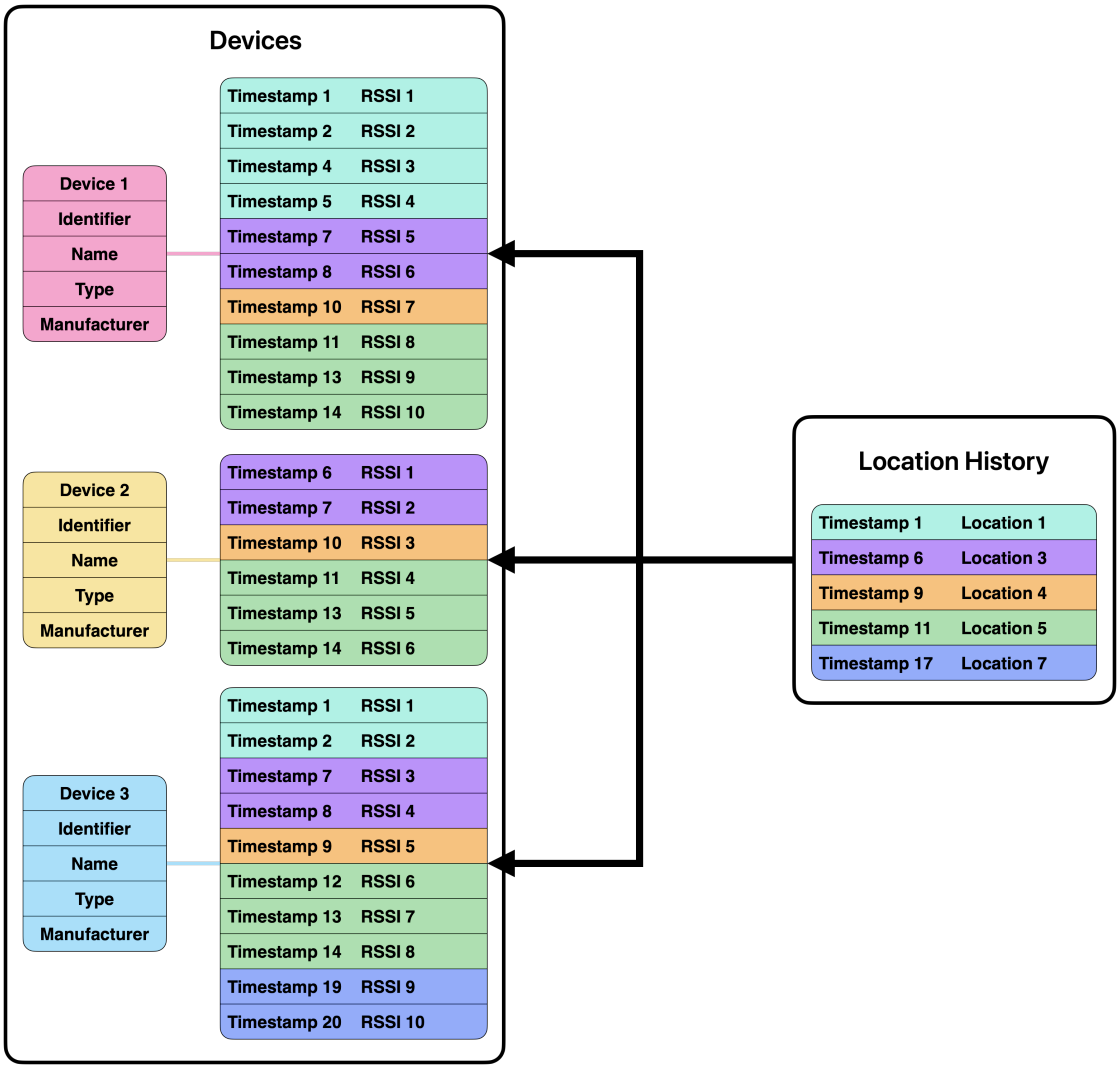

Figure 3.5: Simplified model of *BL(u)E CRAB* dataset

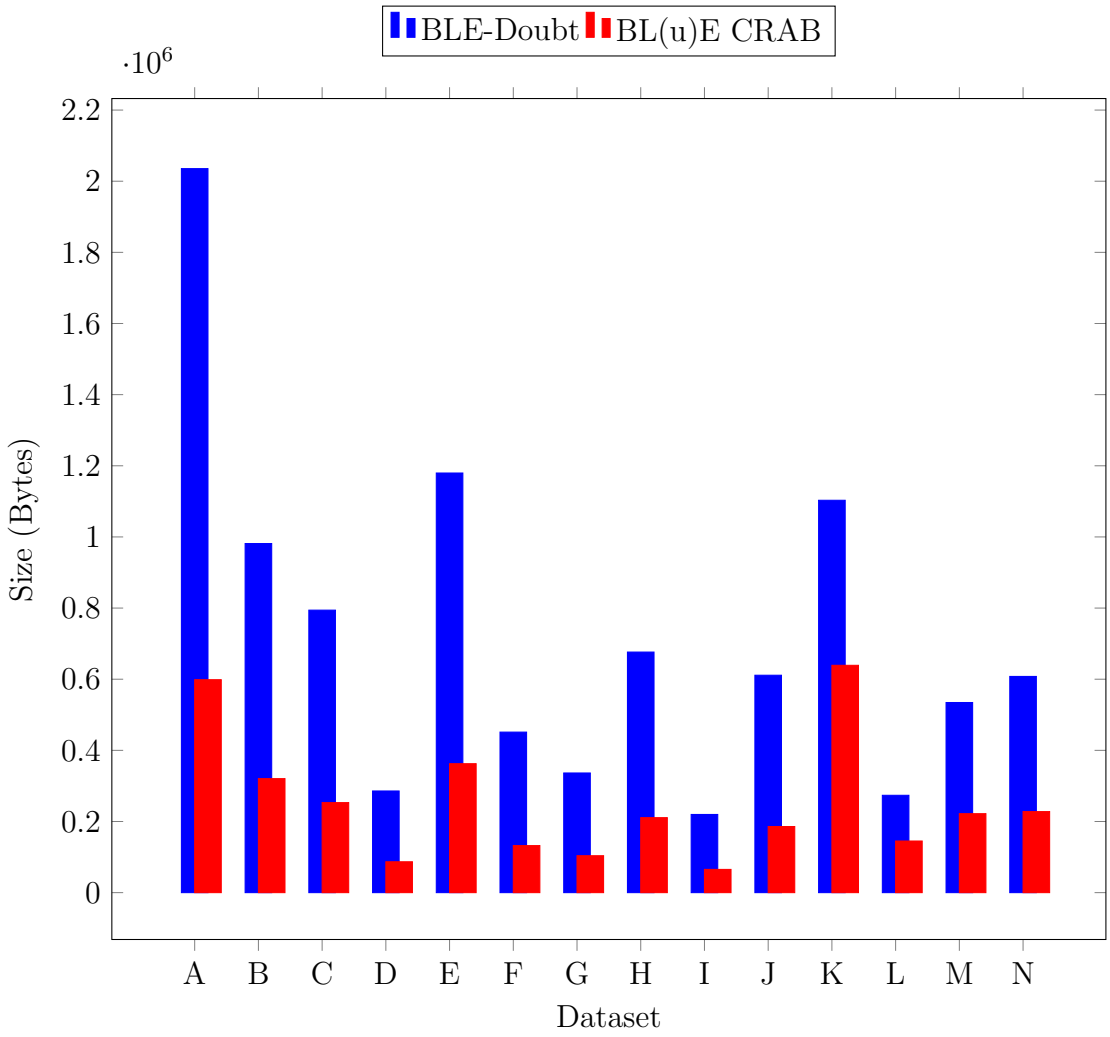

Figure 3.6: Comparison of datasets sizes in BLE-Doubt format vs *BL(u)E CRAB* format.

3.9 Share Datasets

In developer mode and data collection mode, users can share their collected datasets with others. This feature is intended for researchers who want to export collected

datasets to use for data analysis. Datasets are exported as a JSON file using a compact storage format (Section 3.8), and sharing is supported using any method available on the user's device.

3.10 Report Generation

BL(u)E CRAB can generate detailed reports about the collected datasets when in developer mode. The reports are generated in CSV format and include various statistics about the dataset, device risk factors, and classifier performance. The reports are generated on the user's device and can be exported via any supported method on the user's device. Researchers can use these reports to analyze the performance of the classifiers and the characteristics of the collected datasets.

Statistic	**Unit**
Time since initialization	Integer (in seconds)
Number of devices detected	Integer
Number of data points collected	Integer
Number of suspicious devices detected	Integer

Table 3.2: Dataset statistic information reported at each timestamp

Statistic	**Unit**
Time since initialization	Integer (in minutes)
Distance traveled	Double (in meters)
Device count	Integer
Data point count	Integer
Average time with user	Double (in seconds)
Average distance traveled with user	Double (in meters)
Average run-in count	Double
Average area count	Double

Table 3.3: Device risk factor information reported at each timestamp

Statistic	**Unit**
Time since starting scan	Integer (in minutes)
True Positives (Suspicious Devices Classified Correctly)	Integer
False Positives (Non-Suspicious Devices Classified Incorrectly)	Integer
True Negatives (Non-Suspicious Devices Classified Correctly)	Integer
False Negatives (Suspicious Devices Classified Incorrectly)	Integer
Precision	Double (0.0 - 1.0)
Recall	Double (0.0 - 1.0)
F1 score	Double (0.0 - 1.0)

Table 3.4: Classifier performance information reported at each timestamp

4 Risk Factors

This chapter describes the risk factors that *BL(u)E CRAB* use to determine whether a device is suspicious. The risk factors are divided into two categories: aggregate risk factors and trend risk factors. Aggregate risk factors are singular values calculated based on a device's data collected throughout the entire dataset, while trend risk factors are evaluated based on how a value changes throughout the dataset.

4.1 Aggregate Risk Factors

Aggregate risk factors are a single value derived from all data associated with a device. These risk factors provide an atomic measure of the potential risk posed by a device on some behavior the device exhibits over time. Aggregate risk factors do not take into account the trajectory or convergence of values but are useful to compare devices to each other.

4.1.1 Aggregate Risk Factor Prefix

The aggregate risk factors quantify some aspect of the behavior of a device with respect to the behavior of the user traveling. To do this, *BL(u)E CRAB* recreates the path traveled by the user to help quantify the aggregate risk factors for each

device. The user's path is reconstructed by sorting the device's data points [1] by their timestamps, making 2-tuples with neighboring data points, removing tuples with durations between their timestamps that exceed the threshold, and grouping tuples with matching data points together.

For example, let us say that we have data points A through I. There are five seconds between each data point, except for data points C and D, which have ten seconds between them. The time threshold is set to seven seconds. First, we sort the data points by their timestamps (row 1 in figure 4.1). Next, we make 2-tuples with neighboring data points (row 2 in figure 4.1). Then, we remove tuples with durations between their timestamps above the threshold (row 3 in figure 4.1). In this case, the tuple (C, D) is removed because there are ten seconds between these timestamps, which is above the seven-second threshold. Only 2-tuples with durations below the threshold remain (row 4 in figure 4.1). Finally, we group the tuples with matching data points together (row 5 in figure 4.1). The resulting groups of tuples represent the paths the user traveled with the device.

[1] Each data point associated with a device contains a timestamp, location (latitude and longitude) and an RSSI value.

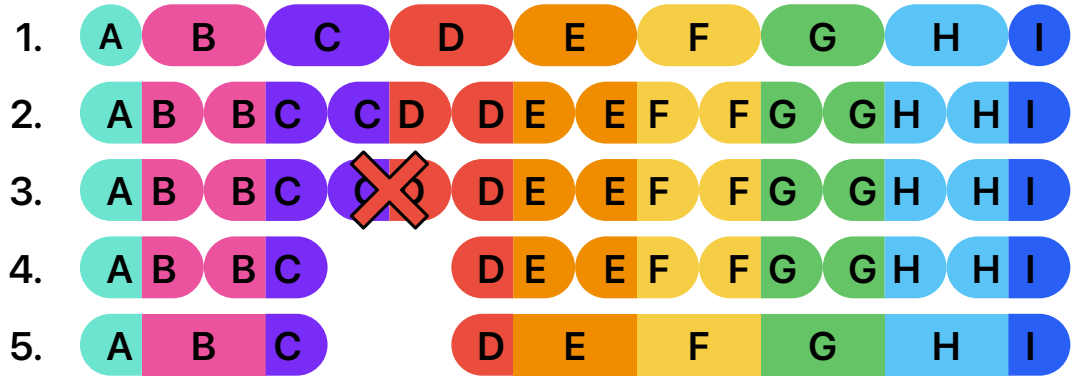

Figure 4.1: The aggregate risk factor prefix involves sorting data points by timestamps, creating 2-tuples of neighboring data points, removing long gaps and grouping neighboring tuples with shared elements.

4.1.2 Distance Traveled with the User

The distance traveled with the user risk factor quantifies the distance a device has traveled near the user. The distance traveled with the user risk factor is useful for detecting suspicious devices using static thresholds or outlier detection for devices that are constantly near the user.

The distance a device traveled with the user is calculated using the aggregate risk factor prefix. Each array of tuples in the prefix is mapped to the sum of distances between the locations of the tuple pairs. The resulting distances for each portion of the path are added together to get the total distance the device traveled with the user.

The distance traveled with the user risk factor is useful to define risk with static thresholds and outlier detection. This value is great for quickly identifying suspicious devices because it increases as the user travels, but does not provide any context awareness to the risk model.

4.1.3 Duration of Time Traveled with User

The time risk factor quantifies the duration of time that a device has appeared near the user. The time risk factor is useful for detecting suspicious devices using static thresholds or outlier detection for devices that are constantly near the user.
The time a device traveled with the user is calculated using the aggregate risk factor prefix. Each array of tuples in the prefix is mapped to the sum of durations between the timestamps of the tuple pairs. The resulting durations for each portion of the path are added together to get the total duration of time the device traveled with the user.
The time traveled with the user risk factor is useful to define risk with static thresholds and outlier detection. This value is great for quickly identifying suspicious devices because it increases while the app scans devices, but it does not provide any context awareness to the risk model.

4.1.4 Run-In Count

The run-in count quantifies the count of time-separated encounters a device has had with the user. The run-in count risk factor is useful for detecting suspicious

devices that frequently appear or disappear over time. It can be used in conjunction with the area count (Section 4.1.5).

The run-in count of a device is calculated by counting the number of arrays in the aggregate risk factor prefix. Each array of tuples in the prefix represents the user's travel history near the device, and each break in the travel history is a gap between encounters.

The run-in count risk factor is best suited for long-term tracking because it can be used to detect if a device disappears and reappears over time. The run-in count risk factor provides context when used in conjunction with other risk factors.

4.1.5 Area Count

The area count quantifies the count of distance-separated clusters of locations where a device has appeared near the user. The area count risk factor is useful for detecting suspicious devices that frequently appear or disappear over large areas of land without detection between location clusters. It can be used in conjunction with the run-in count (Section 4.1.4).

The area count is calculated using a distance threshold. First, we take the set of all locations where the device was detected and pair each location with each other exactly once. Next, we calculate the distance between each pair of locations and remove the pairs that exceed the distance threshold. Then, we initialize a count to track the number of areas in the dataset. Then, we recursively remove locations that are close to each other until no more locations can be removed. This process

is repeated, increasing the area count each time, until there are no more locations that can be removed. Finally, we return the count of areas.

Each location is a node, each pair of locations that are close to each other are connected by an edge, the area count is the number of disconnected node groups. The area count is best suited for long-term tracking because it can be used to detect if a device moves over time. Although the area count does not provide much context on its own, it can provide better context awareness in conjunction with the run-in count. There is no limit on the potential size of each area, which could limit its usefulness when assessing risk.

4.2 Trend-Based Risk Factors

Trend-based risk factors are derived from the way a value associated with a device changes over time. These risk factors provide context to the potential risk posed by a device in some behavior that the device exhibits over time. Trend-based risk factors can provide context to the behavior of a device and are useful for evaluating devices independently of each other.

4.2.1 RSSI Proximity

The RSSI proximity risk factor identifies trends that indicate when a device is close to the user. This risk factor is useful for detecting suspicious behavior by identifying a pattern of elevated RSSI values using static thresholds. A device that exhibits a pattern of elevated RSSI values is only classified as suspicious if

it provides near-values for an extended period of time. RSSI values above the threshold indicate a near value, and RSSI values below the threshold indicate a far value [1].

The proximity of the user is analyzed using static thresholds for the RSSI value and a time tolerance for the device to be close to the user. For the purposes of this project, the RSSI threshold is set at -70 and the acceptable duration of time near the user is 30 seconds. First, device data points with consecutive RSSI values above the threshold are grouped together and data points with RSSI values below the threshold are removed. Then, the duration of time for each device is calculated by subtracting the timestamp of the first data point from the timestamp of the last data point. The device is deemed to have exhibited suspicious behavior if the duration of time with elevated RSSI values exceeds the acceptable duration of time threshold.

The proximity risk factor depends on the RSSI values that we receive from the devices, which are notoriously unreliable values due to their sensitivity to various environmental factors [23]. RSSI does not account for the user context, and our analysis has observed that RSSI values tend to be higher in indoor environments and lower in outdoor environments even when devices have some proximity to the user in both contexts. To keep the approach context-agnostic, *BL(u)E CRAB* does not relate RSSI values to a given distance and simplifies this risk metric by categorizing RSSI values within a time window as near or far.

4.2.2 RSSI Stability

The RSSI stability risk factor identifies trends that indicate when a device is traveling with the user. This risk factor is useful for detecting suspicious behavior by identifying a pattern of stable RSSI values using static thresholds. A device that exhibits a pattern of stable RSSI values is only classified as suspicious if it provides stable values for a period of time.

The idea behind this risk factor is that a device traveling with the user will have a more stable signal because the proximity between the user and the device does not change significantly. A device that is moving independently of the user will have less stable RSSI values as the user moves closer to and farther from the device.

Signal stability is analyzed using static thresholds for RSSI variance, window duration, and phase duration. For the purposes of this project, the RSSI variance threshold is set at 20, the window duration is 30 seconds, and the phase duration is 5 seconds. First, *BL(u)E CRAB* generates a series of time stamps that represent the windows of time to evaluate. This is done by starting with the first timestamp in the device's data points and incrementing by the phase duration until the last timestamp is reached, pairing each timestamp with a corresponding ending timestamp. Next, *BL(u)E CRAB* iterates through each window, collecting all data points within the window duration. Then, the variance of the RSSI values for the data points in the window is calculated using the standard deviation. The device is deemed to have exhibited suspicious behavior if the standard deviation

is below the RSSI variance threshold.

The stability of the signal depends on the RSSI values that we receive from the devices, which are notoriously unreliable values. *BL(u)E CRAB* has to compensate for this by using a moving average to make the RSSI values useful for data analysis. This risk factor relies on static thresholds that may or may not always be correct. Signal stability does not account for user context, as RSSI values tend to have higher variance in crowded environments and changing environmental conditions.

5 Classifiers

This chapter describes the various classifiers implemented in BL(u)E CRAB to identify suspicious devices based on the risk factors described in the chapter 4. For each classifier, we provide a description of the method, its strengths and weaknesses, and the algorithmic details. For classifiers that are more advanced variants of simpler classifiers, we explain the similarities and differences between the approaches and how the advanced variant improves upon the simpler method.

5.1 IQR

The Inter-quartile Range (IQR) classifier flags devices as suspicious based on a risk score, which is calculated using the IQR method and the sum of Z-Scores for each risk factor. The IQR classifier identifies outlier devices with elevated risk scores, indicating potential anomalies in device behavior.

The intuition behind this classifier is to calculate a single risk score for each device to represent its overall risk level. This value can then be used to identify outliers using the IQR method. Devices that are the most suspicious are those with risk scores that exceed the upper bound threshold defined by the formula, which is calculated from values derived from the dataset.

Z-Scores are a statistical measure that describes a value's relationship to the mean of a group of values. They are expressed in terms of standard deviations from the mean. A Z-Score indicates how many standard deviations an element

is from the mean. A positive Z-score indicates that the value is above the mean, while a negative Z-score indicates that the value is below the mean. The formula to calculate the Z-Score for a data point x is:

$$Z = \frac{x - \mu}{\sigma}$$

where x is the value, μ is the mean of the dataset and σ is the standard deviation.

Inter-quartile Range is a statistical method that can be used to identify outliers in a dataset. It is based on the concept of quartiles, which divide the data into four equal parts. The IQR is calculated as the difference between the third quartile (Q3) and the first quartile (Q1):

$$IQR = Q3 - Q1$$

$$\text{Upper Bound} = Q3 + C \times IQR$$

Any device with a risk score that exceeds the upper bound is classified outlier. The IQR classifier is simple to implement and effective at identifying outliers in a dataset. IQR classifier algorithm is described in Algorithm 4.

The IQR classifier may not perform well with small datasets where quartiles are not well-defined because quartiles can be unreliable. It may not perform well with skewed data distributions, particularly with heavily-skewed distributions. Many of the datasets have heavily tailed distributions, leading to poorly selected Q1 and Q3 values, reducing the accuracy of the classifier. The accuracy of the classifier

also depends on the choice of multiplier to determine the upper bound, which is often arbitrary and may need adjustment depending on specific applications.

Algorithm 4: IQR_CLASSIFIER

Input : D (All devices in dataset)

Output : A list of suspicious devices

$x \leftarrow [] ;$

for $d \in D$ **do**

$d_{score} \leftarrow Zscore(d_{distance}) + Zscore(d_{time}) ;$

$x \leftarrow x + [d_{score}] ;$

end

$IQR \leftarrow x_{q3} - x_{q1} ;$

$limit \leftarrow x_{q3} + 1.5 \times IQR ;$

$result \leftarrow [] ;$

for $d \in D$ **do**

if $d_{score} > limit$ **then**

$result \leftarrow result + [d] ;$

end

end

return $result$;

5.2 K-Means

The K-Means classifier flags devices as suspicious based on their clustering behavior in relation to the origin. This classifier flags a class of devices that are clustered together and are distant from the origin, indicating potential anomalies in device behavior.

The intuition behind the K-Means classifier is that devices exhibiting similar behavior will be clustered together based on the enabled risk factors. This is useful for evaluating suspicious device behavior as a group rather than on an individual basis. In theory, suspicious devices would behave in a similar manner to each other, which would lead to them being grouped into the same cluster. The class of devices that are farthest away from the origin have the highest risk factor values as a group, so every device in that cluster should be flagged as suspicious.

The biggest challenge with this classifier is selecting an appropriate k value, or the number of clusters to use. If k is too small, devices that don't exhibit similar behavior may be merged into a single cluster. If k is too large, devices that do exhibit similar behavior may be split into multiple clusters. Both scenarios can lead to inaccurate classification of suspicious devices.

K-Means is a clustering algorithm that partitions data points into k clusters based on their values. The algorithm works by initializing k centroids randomly from the data points, forming k clusters by assigning each data point to the nearest centroid recalculating the centroids as the mean of all points in each cluster and repeating steps 2 and 3 until cluster assignments no longer change (convergence).

The K-means classifier can handle large datasets with many risk factors effectively, because K-means excels at clustering multi-dimensional data. K-means does not have a mechanism to identify or adjust for outliers, but this classifier is being used specifically to look for outliers. This can be advantageous when there are multiple outlier devices that should be flagged as suspicious, but it can be a weakness if there are only a few outlier devices that should be flagged as suspicious.

The K-means classifier requires specifying the number of clusters k in advance, which may be arbitrary. It is also sensitive to initial centroid placement, as different initializations can lead to vastly different results, particularly if the value k is not ideal. Cluster and centroid assignments can be affected by outliers, leading to inaccurate clustering. This classifier may miss suspicious devices that are not part of the group of devices that have been flagged as suspicious. The class of devices that are flagged as suspicious may have a similar behavior to each other, but devices that do not exhibit a similar behavior to that group may fly under the radar.

Algorithm 5: K_MEANS_CLASSIFIER

Input : k (The number of clusters)

Output : A list of suspicious devices

Constants : D (All devices in dataset)

: $RF_{enabled}$ (List of enabled risk factors)

$i \leftarrow []$;

for $d \in D$ **do**

  $d_{loc} \leftarrow []$;

  for $rf \in RF_{enabled}$ **do**

    $d_{loc} \leftarrow d_{loc} + [d_{rf}]$;

  end

  $i \leftarrow i + [d_{loc}]$;

end

$C \leftarrow kmeans(i)$;

$result \leftarrow []$;

for $c \in C$ **do**

  if *distanceFromOrigin(c)* $\geq$ *distanceFromOrigin(result)* **then**

    $result \leftarrow c$;

  end

end

return $result$;

5.3 Smallest K-Cluster

The smallest K-Cluster classifier is a variant of the K-Means classifier that, given some range of K values to evaluate (e.g., 3 - 10), focuses on identifying the "correct" k value. It does this by finding the smallest cluster of devices produced by the K-Means classifier for every value in the range. This classifier flags devices that are part of the smallest cluster, which is assumed to be the most accurate value for k.

The intuition behind this classifier is to select the k value that produces the smallest amount of suspicious devices, under the assumption that this cluster is more likely to represent suspicious devices while minimizing the number of false positives. By evaluating multiple k values and selecting the smallest cluster, this method attempts to mitigate the issue of classifying too many device as suspicious caused by arbitrary k selection in K-Means clustering.

We believe that this method is a more robust variant of the K-Means classifier, as we have found that in most circumstances, our classifier tends to classify a fair number of devices as suspicious without inflating the number of false negatives (Table 5.1). Table 5.2 shows that the average F1 score is consistent across all k values dispelling any notion that a higher F1 score is tethered to any particular k value.

The Smallest K-Cluster classifier builds upon the strengths of the K-Means classifier by attempting to automatically identify an appropriate k value. By evaluating multiple k values and selecting the k that yields the smallest cluster of suspicious

devices, this method attempts to mitigate the issue of arbitrary k selection in K-Means.

The Smallest K-Cluster classifier inherits the weaknesses of the K-Means classifier, including sensitivity to initial centroid placement and potential impact from outliers. Our assumption that the smallest cluster yielded is correlated with the ideal value k for the detection of outliers turned out to be not always true (see Table 5.2). This classifier is a more computationally intensive variant of the standard K-Means classifier, as it requires running the K-Means algorithm multiple times for different values k.

Algorithm 6: SMALLEST_K_CLUSTER_CLASSIFIER

Input : k_{min} (The minimum number of clusters)
: k_{max} (The maximum number of clusters)

Output : A list of suspicious devices

$result \leftarrow$ `NULL` ;

for $k \in range(k_{min}, k_{max})$ **do**
    $c \leftarrow k\_means\_classifier(k)$;
    if $result =$ `NULL` $\vee\ c_{count} \leq result_{count}$ **then**
        $result \leftarrow c$;
    end
end

return $result$;

Dataset	**Selected Devices (Avg)**	**False Negatives (Avg)**	**False Negatives (Std Dev)**
A	3.466	1.363	0.82
B	2.854	1.473	0.87
C	3.511	2.105	0.52
D	6.198	0.404	0.96
E	5.074	1.044	0.44
F	6.229	0.364	1.02
G	5.611	0.243	0.79
H	5.028	1.137	0.64
I	7.103	0.117	0.37
J	7.994	0.351	1.09
K	7.428	0.877	0.98
L	3.898	0.991	0.84
M	6.415	0.800	0.74
N	2.025	0.020	0.19
Total	5.177	1.072	0.959

Table 5.1: Correlation of the number of devices selected as suspicious by the K-Means classifier and the number of false negatives across all datasets.

k	**F1 Score (Avg)**	**F1 Score (Std Dev)**
2	0.557	0.341
3	0.568	0.364
4	0.550	0.390
5	0.545	0.395
6	0.543	0.396
7	0.542	0.397
8	0.542	0.397
9	0.542	0.398
10	0.542	0.398
Total	0.548	0.387

Table 5.2: Correlation of the k value and the F1 Score across all datasets.

5.4 IQR / K-Means Hybrid

The IQR / K-Means Hybrid classifier uses the adaptability provided by K-Means clustering to improve the quality of the IQR classifier. It first uses K-Means to cluster similar device together, then determines alternative values for finding the IQR value to filter out outliers. This approach helps in achieving a more accurate upper bound by reducing the influence of skewed data distributions within datasets.

The intuition behind the IQR / K-Means Hybrid classifier is to leverage the power of K-means clustering with IQR outlier detection. This classifier attempts to improve upon the IQR classifier by selecting better Q1 and Q3 values in skewed

datasets. By grouping similar devices together, the classifier can better identify the quartiles needed for IQR calculation, leading to a more reliable detection of outliers. In this classifier, the Q1 and Q3 values are not the real Q1 and Q3 values from the dataset, but rather the data points that appear to be reasonable values for finding an acceptable range of values for calculating the IQR.

The IQR / K-Means Hybrid classifier enhances the robustness of IQR with K-Means by reducing the negative impact of skewed data distributions when using IQR by providing a more accurate upper bound for outlier detection.

This classifier requires specifying the number of k clusters in advance. Similarly to the previous approach, the K-Means algorithm is sensitive to the initial placement of the centroid, which can lead to different results. The choice of multiplier to determine the upper bound is arbitrary and may need adjustment depending on specific applications.

Algorithm 7: IQR_K_MEANS_HYBRID_CLASSIFIER

Input : D (All devices in dataset)

: k (The number of clusters)

Output : A list of suspicious devices

$C \leftarrow kmeans(k)$;

$C \leftarrow sortedBy(\Sigma(c_{loc} \in C))$;

$d \leftarrow C[0][-1]$;

$q1 \leftarrow Zscore(d_{distance}) + Zscore(d_{time})$;

$d \leftarrow C[-1][0]$;

$q3 \leftarrow Zscore(d_{distance}) + Zscore(d_{time})$;

$IQR \leftarrow q3 - q1$;

$limit \leftarrow q3 + 1.5 \times IQR$;

$result \leftarrow []$;

for $d \in D$ **do**

  $d_{score} \leftarrow Zscore(d_{distance}) + Zscore(d_{time})$;

  if $d_{score} > limit$ **then**

    $result \leftarrow result + [d]$;

  end

end

return $result$;

5.5 Single-Dimensional Classifier with Optional RSSI Evaluation (SCORE)

This classifier uses the Jenks natural breaks algorithm [24] to determine individual thresholds based on devices' time traveled with user risk factor values, then on devices' distance traveled with user risk factor values. Devices with risk factor values that exceed each threshold are filtered into the optional proximity and stability analyzer. The classifier will flag devices that have have been nearby the user for a significant amount of time and have stable RSSI values during the time frames the device has been near the user.

The intuition behind this classifier is to filter out devices by leveraging Jenks natural breaks to determine appropriate thresholds for individual risk factors. This approach entails evaluating devices based on individual risk factors rather than upon an accumulation of risk factors or the group behavior of similar devices. Classifiers that rely on K-means clustering intentionally select devices that exhibit similar behavior, but this classifier attempts to filter out devices that do not exhibit suspicious behavior.

Once devices have been selected based on their individual risk factor values, the proximity analyzer flags devices with consecutive RSSI values indicating that the device has been nearby the user for a significant amount of time, and the stability analyzer flags devices with stable RSSI values indicating that the device has maintained a constant proximity from the user for a significant amount of time. This allows the classifier to identify suspicious devices based on how the data changes over time, which makes a compelling case for further investigation.

The SCORE classifier uses Jenks natural breaks to automatically determine thresholds based on natural breaks in the data. It determines separate time and distance traveled with user thresholds to better filter out devices without selecting devices that exhibit similar behavior to each other. It leverages aggregate and trend-based risk factors to identify suspicious devices based on their behavior over time, rather than relying on a single value to represent elements of risk.

The SCORE classifier relies on static thresholds for RSSI proximity and duration while evaluating whether devices exhibit suspicious behavior, which may not adapt well to different environments, and must be set by the user ahead of time. This requires the user to have prior knowledge about how the classifier evaluates risk and the environment in which the classifier will be used.

Algorithm 8: SCORE_CLASSIFIER

Input : D (All devices in dataset)

Output : A list of suspicious devices

Constants : $threshold_{close\_duration}$ (Close duration threshold)

: $threshold_{stability}$ (RSSI stability threshold)

$result \leftarrow []$;

$threshold_{time} \leftarrow jenks(D_{time})$;

$threshold_{distance} \leftarrow jenks(D_{distance})$;

for $d \in D$ **do**

  $S \leftarrow get\_segments(d)$;

  $a \leftarrow d_{time} \geq threshold_{time}$;

  $b \leftarrow d_{distance} \geq threshold_{distance}$;

  $c \leftarrow \Gamma$;

  $d \leftarrow \Gamma$;

  for $s \in S$ **do**

    if $durationOf(s) \geq threshold_{close\_duration}$ **then**

      $c \leftarrow \Delta$;

    end

    if $avg(s_{rssi}) \leq threshold_{stability}$ **then**

      $d \leftarrow \Delta$;

    end

  end

  if $a \wedge b \wedge c \wedge d$ **then**

    $result \leftarrow result + [d]$;

  end

end

return $result$;

Algorithm 9: GET_SEGMENTS

Input : d (A single device)

Output : A list of data point segments where the device is considered "close"

Constants : $threshold_{rssi}$ (RSSI value threshold)

$segments \leftarrow []$;

for $dp \in d_{datapoints}$ **do**

  if $dp_{rssi} \geq threshold_{rssi}$ **then**

    if $segments = [] \vee segments[-1][-1]_{time} \neq dp_{time}$ **then**

      $segments \leftarrow segments + [dp]$;

    else

      $segments[-1] \leftarrow segments[-1] + [dp]$;

    end

  end

end

return $segments$;

5.6 Risk Factors Used by Classifiers

Traditional Aggregate	-	Distance Traveled with User, Duration of Time Traveled with User
Extended Aggregate	-	Run-in Count, Area Count
Trend-based	-	RSSI Stability, RSSI Proximity

	Traditional Aggregate	**Extended Aggregate**	**Trend based**
AirGuard RSVC (Baseline)	✓		
BLE-Doubt (Baseline)	✓		
IQR	✓*	✓*	
K-Means	✓*	✓*	
Smallest K Cluster	✓*	✓*	
IQR / K-Means Hybrid	✓*	✓*	
SCORE	✓		✓*

** - optional*

Table 5.3: Comparison of supported risk factors in *BL(u)E CRAB* classifiers

6 Experiments

6.1 Experimental Setup

The *BL(u)E CRAB* app includes a built-in evaluation framework that allows researchers to run experiments on various datasets to evaluate the performance of different classifiers and configurations. For our experiments, we used this framework to evaluate each classifier described in Chapter 5. *BL(u)E CRAB* processes the dataset by generating timestamps for every one minute in the dataset, generating a new dataset using only data from before the specified timestamp, and running each classifier multiple times. After running the classifiers, the evaluation framework calculates the average value for true positives, false positives, true negatives, and false negatives to calculate the precision, recall, and F1 score for the classifier at that timestamp. This method of generating new datasets based on timestamps simulates the user's movement through the environment and gives insight into how the classifiers perform over time.

The evaluation yields reports that summarize the content of the dataset and the performance of the classifier. The reports include statistics on the number of devices detected over time, risk factor values for each device, RSSI values for devices over time, aggregate RSSI values across all devices, and various accuracy metrics to quantify the performance of the classifiers. For the sake of consistency with other studies on BLE tracking, we quantify the performance of the classifiers

using F1-scores.

For researchers to conduct an experiment, they should add valid datasets to an asset directory in the codebase, add a ground truth file to specify suspicious device identifiers, and select a classifier to be evaluated. The parameters specified on the settings page or the settings file will be used for the simulation. The *BL(u)E CRAB* codebase [25] is open source and available to use on GitHub [26].

6.2 Datasets

6.2.1 BLE-Doubt Dataset

The BLE-Doubt dataset is a publicly available dataset provided by BLE-Doubt [14] that contains user location data, as well as BLE device identifiers and their associated signal data. The dataset includes the ground truth for each file in the dataset, making it useful for evaluating device classifiers.

6.2.2 Collected Data

In addition to the BLE-Doubt dataset, we collected our own dataset using *BL(u)E CRAB*, which records the user's location, as well as device identifiers and their associated RSSI values. Data were collected in various environments, including urban areas near the Portland State University campus, suburban neighborhoods, and indoor environments. It also includes various modes of transportation, including public transportation systems, automobiles, and walking. We believe that this

represents a diverse range of realistic scenarios in which the user might encounter BLE devices. The collected dataset also includes ground truth information to help evaluate the performance of the classifiers.

6.3 Metrics

To evaluate the performance of the classifiers, we prioritized the efficacy of dynamic thresholding and clustering methods rather than the utility of the novel risk factors we proposed (Sections 4.1.4 and 4.1.5). We evaluated the SCORE classifiers separately from the other classifiers to isolate the effect that trend-based risk factors have on accuracy, but the variant with proximity and signal stability features disabled is comparable to the other classifiers evaluated.

To evaluate the efficacy of each classifier over time, we utilized the risk factors used in the baseline classifiers, including the duration of time traveled with the user and the distance traveled with the user (Sections 4.1.2 and 4.1.3). We use F1 scores to evaluate the efficacy of each classifier over time, because it takes into account precision and recall, which is important when evaluating false positives and false negatives. We included a summary of two results in Section 6.4, but a detailed analysis of each dataset is available in Appendix A.

6.4 Summary of Results

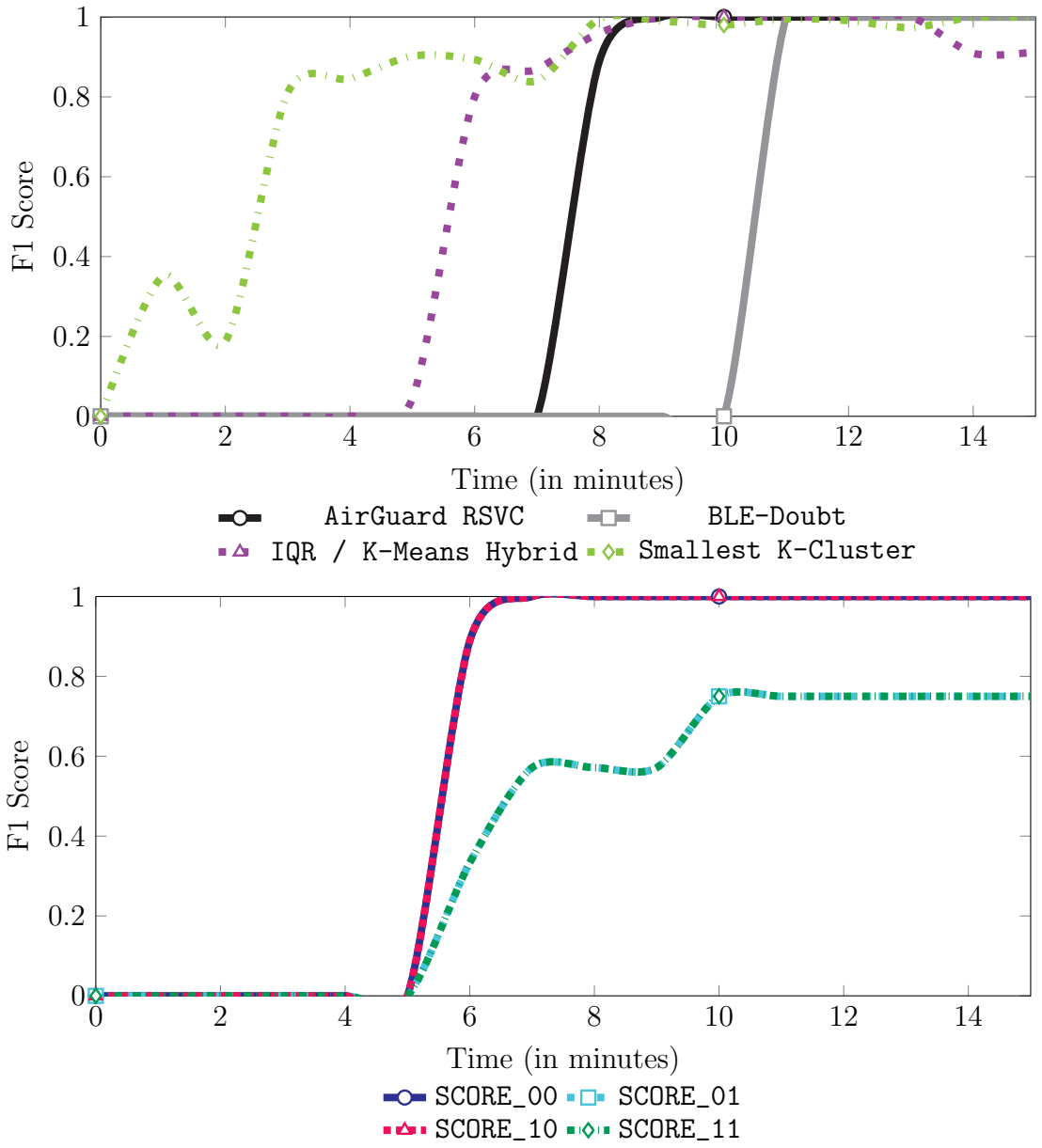

Figure 6.1: Classifier F1 Scores for BLE-Doubt Dataset D

For dataset D, the Smallest K-Cluster, IQR / K-Means Hybrid, and SCORE (signal stability disabled) classifiers correctly identify the suspicious devices before the baseline classifiers. The static classifiers maintain perfect accuracy for the remainder of the dataset, while the dynamic classifiers have some misclassifications at the beginning of the dataset, but reach near-perfect accuracy as they collect more data. The Smallest K-Cluster and IQR / K-Means Hybrid classifiers consistently perform as well as the baseline classifiers, but have slight variations due to the randomized initialization of their clustering methods. The SCORE classifiers with the signal stability feature enabled have lower accuracy in this dataset, as they tend to classify devices as suspicious less often than their counterparts with the signal stability feature disabled. The dynamic classifiers have a clear advantage over the static classifiers in terms of speed, as they are able to quickly identify suspicious devices and maintain high accuracy for the remainder of the dataset. This dataset demonstrates how the classifiers typically perform when the user is in a stable environment with a consistent rate of movement, near-unchanging average time and distance traveled with the user across all devices, and a consistent rate of device movement in the environment (i.e., no large surges or drop offs in the number of devices near the user).

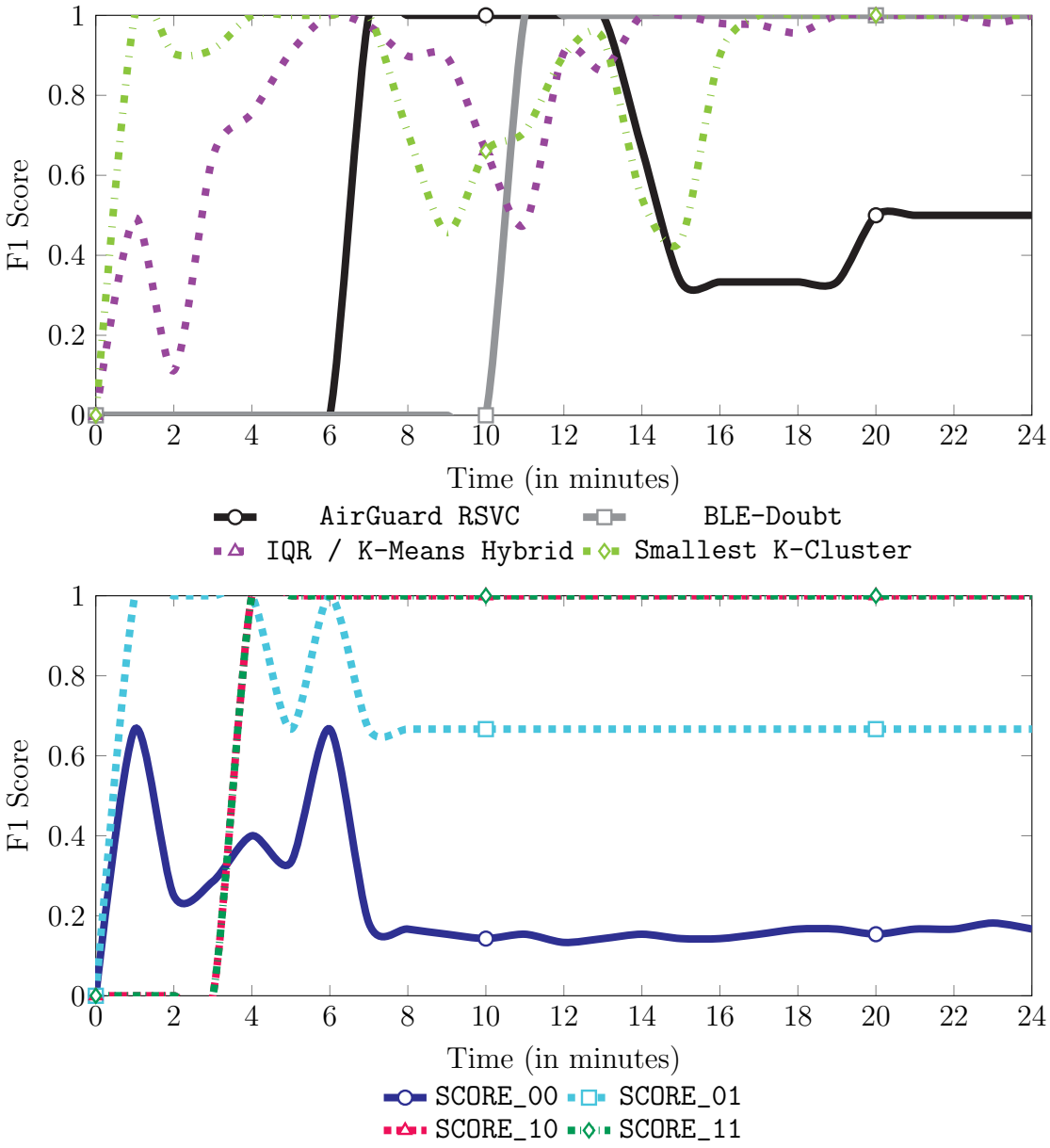

Figure 6.2: Classifier F1 Scores for Walking Dataset 2

For the second walking dataset, the dynamic classifiers exhibit their more er-

ratic behavior, with occasional dips in accuracy due to the user's limited travel distance and pattern of circular movement. The static baseline classifiers exhibit more consistent performance, but the baseline with a lower time threshold (AirGuard RSVC) also experiences a sustained dip in accuracy for reasons similar to those of dynamic classifiers. This dataset also demonstrates how the SCORE classifiers with the proximity feature enabled prevent those variants from classifying too many devices as suspicious, thus lowering the false positive rate and increasing the overall accuracy of those classifiers in this scenario. The SCORE classifier with the proximity and signal stability features enabled performs just as well as the SCORE variant with only the proximity feature enabled, indicating that signal stability may be a factor to consider in scenarios where there are many suspicious devices in the environment and identifying the most suspicious devices may be important.

7 Conclusions

We presented a novel approach to device classifiers based on risk factors derived from data from the Bluetooth device. This research introduces three new ways to identify suspicious devices from time, location, and signal strength data. Our approaches include a composite risk score based on dynamic thresholds calculated using the inter-quartile range, a multi-dimensional clustering approach with a method to determine an ideal value k, and a single-dimensional clustering approach with optional signal strength trend evaluation. We also implemented versions of AirGuard's [19] reverse-engineered single-visit classifier (RSVC), modeled on a reverse-engineered prototype of the Find My classifier, and BLE-Doubt [18] classifiers as baselines.

Our findings show that our classifiers generally outperform the other classifiers in terms of speed in identifying suspicious devices and datasets where many devices are close to the user for long periods of time. The baseline classifiers performed better in scenarios involving dramatic speed changes, as dynamic classifiers were susceptible to false positives in these situations. However, our classifiers were able to identify suspicious devices more quickly in scenarios involving consistent movement patterns. These results suggest that clustering-based approaches, particularly those that consider multiple dimensions of data, are promising for quickly identifying suspicious devices, rather than traditional methods based on static thresholds.

Each classifier has its strengths and weaknesses, indicating that a hybrid approach combining multiple methods may yield the best results in real-world applications. Based on our benchmarking results, we recommend traveling at a consistent speed in public areas away from home to increase the chance that the user will be alerted to suspicious devices. This recommendation does not guarantee success in finding a tracker if one is present, but our observations suggest that this increases the likelihood of detection.

7.1 Future Work

This thesis provides several novel contributions to the domain of object tracking via Bluetooth trackers, but there are several facets of this project that could be improved and explored in future research ventures, many of which could build upon the findings of this thesis. Some potential improvements for future work may include the following.

- **Identify Additional Risk Factors** - Identify additional risk factors to provide a more comprehensive understanding of BLE tracker behavior.
- **Identify Environmental Changes for Context-Awareness** - Identify context-awareness to help researchers develop classifiers that can respond to environmental changes in the user context. Changes in context can include increasing or decreasing speed, mode of travel, crowd density, and indoor and outdoor placement.
- **Experiment with More Diverse Datasets** - Explore the limits of de-

veloped methods via more diverse datasets to help validate their robustness and expose shortcomings. Further refinement of the algorithms used could lead to improved efficiency and accuracy. Emulating real-world scenarios in experimental setups could provide more insightful results.

- **Integration with Existing Systems** - Integration of developed methods with existing systems could provide insight to the pragmatism of context-aware risk factors in large-scale networks.

- **Inter vs. Intra Device Analysis** - Combine inter-device (networks of devices) and intra-device (single device) data analysis methods could provide insights into the capabilities of context-aware risk factor analysis across different devices and manufacturers.

- **Support Additional Device Types** - Support identifying devices that are not BLE-based devices, so 4G LTE and GPS devices can be identified as trackers as well.

- **Improve and Optimize the BL(u)E CRAB app** - Improving features such as the scanning mechanism to support background scanning, power consumption, and user-friendliness in our implementation of this framework. *BL(u)E CRAB* provides great features in its current state, but could always be improved to become a commercial-grade app.

7.2 Discussion

User privacy and security should be a top priority for device manufacturers. The existing measures taken by some manufacturers to protect user data are commendable, but there is still room for improvement. After working with various object location networks, we recommend the following strategies to enhance user privacy and security.

- **Implement Platform Agnostic Security Features** - Manufacturers should implement a universal standard for inter-device communication to provide security features that are effective across all platforms akin to the proposal between Google and Apple [10], not just specific mobile devices and trackers. This includes a robust mechanism for reliable device detection, so that tracking applications can quickly identify compliant tracking devices, independently evaluate the risk they pose to users, and help the user find and disable suspicious tracking devices [31].

- **Enforce Stronger Privacy Protections** - Devices should be designed with secure privacy features in mind. This includes limiting the amount of data shared with first- and third-party services and ensuring that users are informed about what data is being collected and how it is used. User data retention should also be minimized to prevent bad actors from being able to re-construct sensitive user information based on retained data.

- **Provide User Education** - Manufacturers should provide users with clear

and accessible information about the security features of their devices, how to use these features effectively, and how tracker devices can be misused to compromise the user's privacy. This material should include educating users on proactive measures to avoid being tracked by BLE-enabled devices, practice scenarios to respond to detected trackers, and guidance on how to appropriately report suspicious devices.

Bibliography

[1] Covid-19 and your smartphone: Ble-based smart contact tracing. Ng PC, Spachos P, Plataniotis KN. COVID-19 and Your Smartphone: BLE-Based Smart Contact Tracing. IEEE Syst J. 2021 Mar 9;15(4):5367-5378. doi: 10.1109/JSYST.2021.3055675. PMID: 35582390; PMCID: PMC8843047.

[2] https://www.apple.com/airtag/, 2024.

[3] https://www.samsung.com/us/mobile/mobile-accessories/phones/galaxy-smarttag2-black-ei-t5600bbegus/#benefits, 2024.

[4] Find your devices with tile: Laptops, phones, electronics, and more. https://www.tile.com/blog/how-to-find-my-devices-using-tile-apple-ios-android, 2024.

[5] Landairsea 54 gps tracker. https://landairsea.com/products/landairsea-54, 2026.

[6] Life 360. Tile. https://www.life360.com/tile-trackers. Accessed 2025.

[7] Matthias Hollick Alexander Heinrich, Leon Würsching. Please unstalk me: Understanding stalking with bluetooth trackers and democratizing anti-stalking protection. *Proceedings on Privacy Enhancing Technologies*, pages 353–371, 2024.

[8] Inc. Apple. Find my. https://www.apple.com/icloud/find-my/. Accessed 2025.

[9] Inc. Apple. Use find my to locate your lost apple device or airtag. https://support.apple.com/en-us/104978. Accessed 2025.

[10] Inc. Apple. Apple and google lead initiative for an industry specification to address unwanted tracking. https://www.apple.com/newsroom/2023/05/apple-google-partner-on-an-industry-specification-to-address-unwanted-track 2023.

[11] Inc. Apple. Apple platform security. https://help.apple.com/pdf/security/en_US/apple-platform-security-guide.pdf, 2026.

[12] Inc. Apple. Cbperipheral. https://developer.apple.com/documentation/corebluetooth/cbperipheral, 2026.

[13] Inc. Bluetooth SIG. Bluetooth technology overview. https://www.bluetooth.com/learn-about-bluetooth/tech-overview/, 2021.

[14] Jimmy Briggs and Christine Geeng. Ble-doubt: Smartphone-based detection of malicious bluetooth trackers. In *SafeThings 2022*. IEEE, 2022.

[15] Chipolo. Chipolo. https://chipolo.net/en-us/. Accessed 2025.

[16] Chipolo. Does my chipolo work with google's find my device app? https://chipolo.net/en-us/blogs/does-my-chipolo-work-with-googles-find-my-device-app. Accessed 2025.

[17] Tess Despres, Noelle Davis, Prabal Dutta, and David Wagner. Detagtive: Linking macs to protect against malicious ble trackers. In *Proceedings of the Second Workshop on Situating Network Infrastructure with People, Practices, and Beyond*, SNIP2+ '23, pages 1–7, New York, NY, USA, 2023. Association for Computing Machinery.

[18] Briggs et al. Ble-doubt: Smartphone-based detection of malicious bluetooth trackers. In *SafeThings 2022*. IEEE, 2022.

[19] Heinrich et al. Airguard - protecting android users from stalking attacks by apple find my devices. WiSec '22. Association for Computing Machinery, 2022.

[20] Samsung Group. Smartthings find. https://support.smartthings.com/hc/en-us/articles/10863369660052-SmartThings-Find. Accessed 2025.

[21] Samsung Group. Smartthings find. https://smartthingsfind.samsung.com/login. Accessed 2025.

[22] Alexander Heinrich, Niklas Bittner, and Matthias Hollick. Airguard - protecting android users from stalking attacks by apple find my devices. WiSec '22, pages 26–38, New York, NY, USA, 2022. Association for Computing Machinery.

[23] Karel Heurtefeux and Fabrice Valois. Is rssi a good choice for localization in wireless sensor network? In *2012 IEEE 26th International Conference on Advanced Information Networking and Applications*, pages 732–739, 2012.

[24] George F. Jenks. The data model concept in statistical mapping. 1967.

[25] DIPr Lab. Bl(u)e crab on github. https://github.com/DIPrLab/BLuE-CRAB, 2026.

[26] DIPr Lab. Database and internet privacy (dipr) lab. https://github.com/DIPrLab, 2026.

[27] Google LLC. Find hub. https://www.google.com/android/find. Accessed 2025.

[28] Google LLC. Bluetoothdevice. https://developer.android.com/reference/android/bluetooth/BluetoothDevice, 2026.

[29] Inc. MathWorks. https://www.mathworks.com/help/bluetooth/ug/bluetooth-packet-structure.html#mw_d580926e-5e4b-4e45-9884-e5e7476e4948, 2026.

[30] Travis Mayberry, Ellis Fenske, Dane Brown, Jeremy Martin, Christine Fossaceca, Erik C. Rye, Sam Teplov, and Lucas Foppe. Who tracks the trackers? circumventing apple's anti-tracking alerts in the find my network. In *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, WPES '21, pages 181–186, New York, NY, USA, 2021. Association for Computing Machinery.

[31] Narmeen Shafqat, Nicole Gerzon, Maggie Van Nortwick, Victor Sun, Alan Mislove, and Aanjhan Ranganathan. Track you: A deep dive into safety alerts for apple airtags. *Proceedings on Privacy Enhancing Technologies*, pages 132–148, 2023.

[32] Inc. Silicon Laboratories. https://docs.silabs.com/bluetooth/2.13/bluetooth-fundamentals-advertising-scanning/bluetooth-adv-data-basics, 2026.

[33] Tingfeng Yu, James Henderson, Alwen Tiu, and Thomas Haines. Security and privacy analysis of samsung's Crowd-Sourced bluetooth location tracking system. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 5449–5466, Philadelphia, PA, August 2024. USENIX Association.

APPENDICES

A Results

ID	Dataset	Duration (H:MM)	Movement	Tracker Location
A	BLE-Doubt	1:15	Walking	Backpack
B	BLE-Doubt	1:35	Walking	Backpack
C	BLE-Doubt	1:15	Walking	Pockets
D	BLE-Doubt	0:14	Walking	Pockets
E	BLE-Doubt	1:24	Car	Vehicle
F	BLE-Doubt	0:25	Jogging	Backpack
G	BLE-Doubt	0:21	Walking	Backpack
H	BLE-Doubt	0:35	Walking	Backpack
I	BLE-Doubt	0:14	Train	Backpack
J	BLE-Doubt	0:28	Train	Backpack
K	BLE-Doubt	1:30	Walking	Unknown
L	BLE-Doubt	0:45	Walking	Unknown
M	BLE-Doubt	0:50	Train	Unknown
N	BLE-Doubt	1:04	Car	Unknown
O	*BL(u)E CRAB*	0:31	Walking	Pocket
P	*BL(u)E CRAB*	0:23	Walking	Backpack

Table A.1: Dataset Information

A.1 BLE-Doubt Dataset A

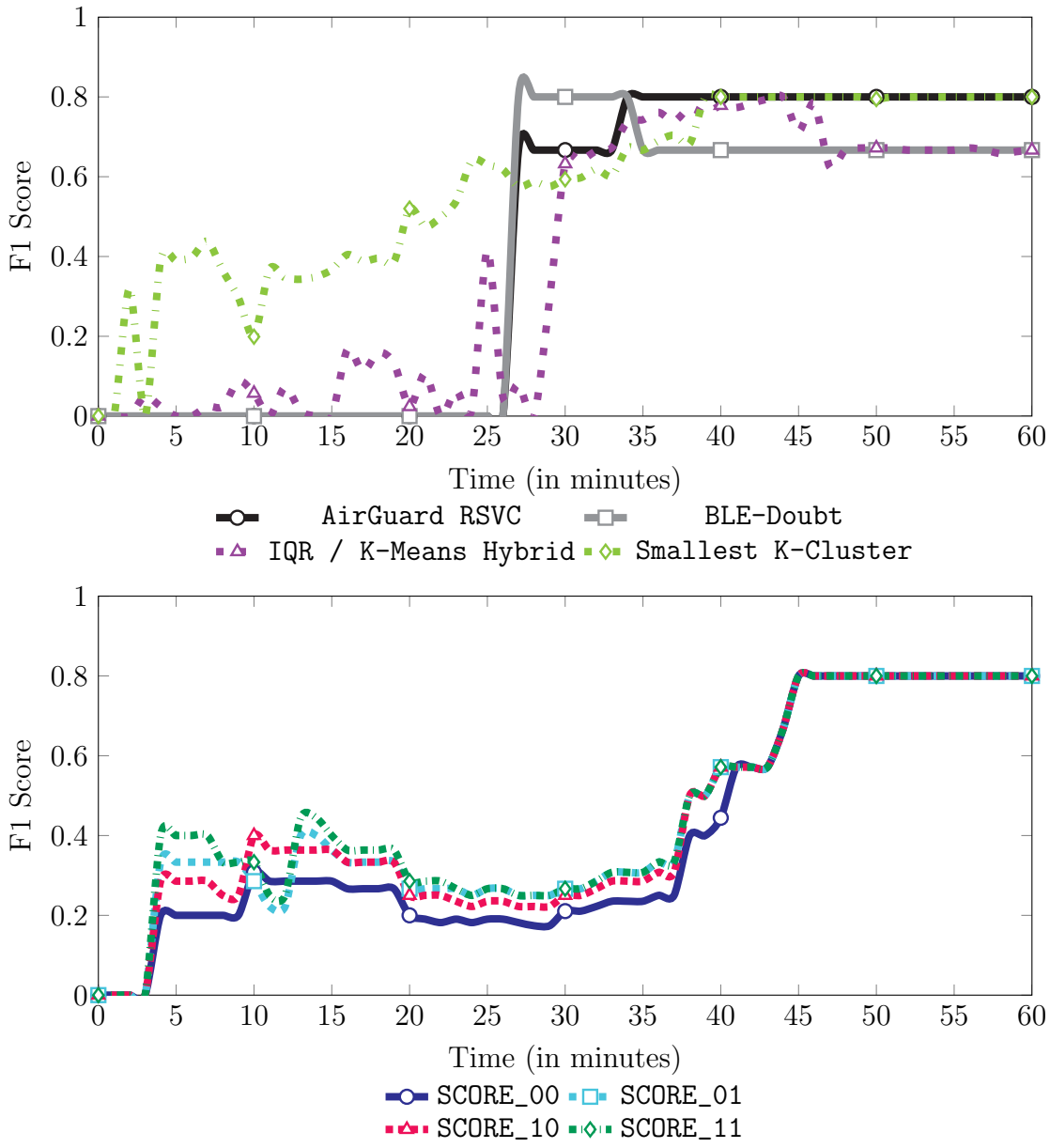

Figure A.1: Classifier F1 Scores for BLE-Doubt Dataset A

Note: This dataset was collected with a few sporadic scans by the user with long pauses between scans, followed by a period of more consistent scanning. We adjusted the timestamps to start when the more consistent scanning began but left in the data before that point as well.

This dataset starts with inconsistent results from many of the classifiers, but they balance out over time and ultimately have high accuracy by the end of the dataset. The user has a consistent travel pattern after the adjusted start time, leading to gradual improvement of classifier performance.

00:02 - 00:04	The Smallest K-Cluster classifier starts to correctly classify one suspicious device, but also incorrectly classifies some non-suspicious devices as suspicious.
00:04 - 00:10	The SCORE classifiers start to identify suspicious devices, but also incorrectly classify some non-suspicious devices as suspicious. The variant with the proximity and stability features disabled has worst the performance at this time due its tendency to classify more devices as suspicious, while the variant with those features enabled has the best performance due to its tendency to classify fewer devices as suspicious. All variants of the SCORE classifier continue to incorrectly classify many non-suspicious devices as suspicious, hurting their F1 scores.

00:23 - 00:24	The Smallest K-Cluster classifier starts to correctly classify the other suspicious device, but continues to incorrectly classify some non-suspicious devices as suspicious.
00:27	The AirGuard and BLE-Doubt classifiers start to correctly classify both suspicious devices, but the AirGuard classifier also incorrectly classifies two non-suspicious devices as suspicious, while the BLE-Doubt classifier incorrectly classifies two non-suspicious devices.
00:30	The IQR / K-Means classifier starts to correctly classify both suspicious devices, but also incorrectly classifies two non-suspicious devices as suspicious. It maintains this performance for the rest of the dataset.
00:34 - 00:35	The AirGuard and BLE-Doubt classifiers continue to correctly classify both suspicious devices, but BLE-Doubt starts to incorrectly classify two non-suspicious devices as suspicious due to the user's prolonged contact with the the device, while AirGuard improves its F1 score by reducing the number of non-suspicious devices it incorrectly classified previously.

00:38 - 00:45	The SCORE classifiers continue to correctly classify both suspicious devices, but reduce the number of non-suspicious devices they incorrectly classify to one device, boosting their F1 scores. All variants of the SCORE classifier maintain the same level of performance for the rest of the dataset.
00:39	The Smallest K-Cluster classifier reduces the number of non-suspicious devices it incorrectly classifies as suspicious to one device, increasing its F1 score. It maintains this performance for the rest of the dataset.

A.2 BLE-Doubt Dataset B

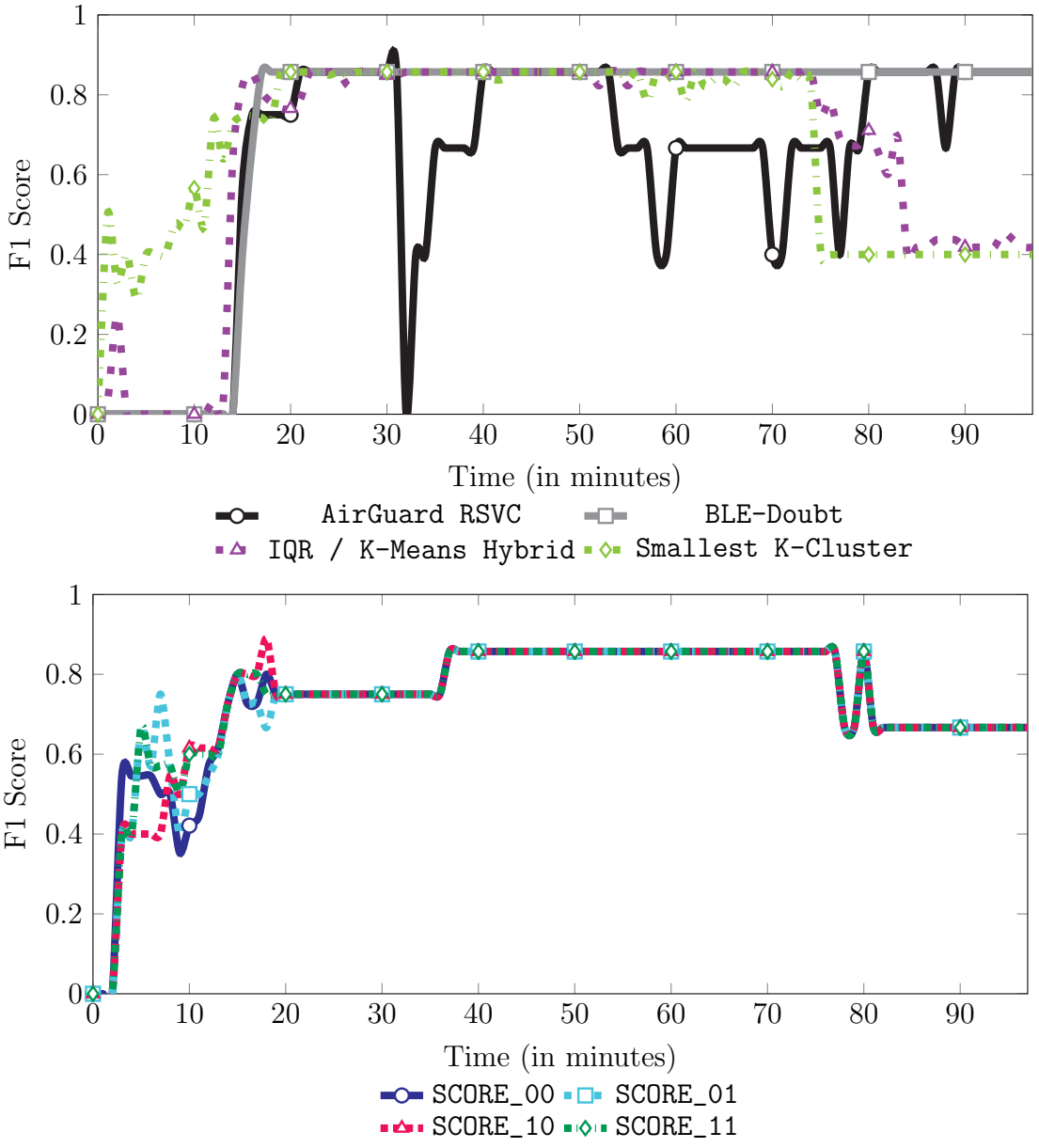

Figure A.2: Classifier F1 Scores for BLE-Doubt Dataset B

This dataset demonstrates how recent device behavior and contact patterns can affect performance. The dynamic classifiers respond well to changes in recent device behavior, while the static classifier with a "recently seen" feature has less consistent accuracy.

00:01 - 00:02	The IQR / K-Means Hybrid classifier correctly classifies zero to one suspicious devices because the suspicious devices remain near the user's device during this time, but because the dataset is small, it does not have enough data to consistently cluster the devices accurately. It's F1 score falls back to zero as other devices exhibit similar behavior.
00:01 - 00:12	The Smallest K-Cluster classifier immediately classifies one suspicious device correctly because it always classifies at least one device as suspicious. Its F1 score gradually increases as it collects more data and is able to more reliably classify a second suspicious devices correctly. It also incorrectly classifies a devices as suspicious during this time.
00:03	The SCORE classifiers quickly identify three out four suspicious devices, but they also incorrectly classify several non-suspicious devices as suspicious.

00:03 - 00:14	The SCORE classifier's F1 score steadily increases as it correctly stops classifying non-suspicious devices as suspicious while maintaining correct classifications of suspicious devices. The variant of SCORE with proximity and stability features disabled performs has a drop in precision as it is more prone to incorrectly classifying non-suspicious devices as suspicious. By the 14-minute mark, all variants of SCORE have similar F1 scores throughout the remainder of the dataset.
00:12 - 00:19	The Smallest K-Cluster classifier's F1 score increases to it's peak as it correctly identifies a third suspicious device and stops classifying a non-suspicious device as suspicious.
00:15	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 14 and 15-minute mark. Every other classifier (other than Smallest K-Cluster) starts flagging suspicious devices because they have enough data to accurately classify suspicious devices.
00:25	The user appears to stop collecting data for several minutes.

00:32	AirGuard's F1 score plummets because the user lost contact with all suspicious devices several minutes prior, and now the suspicious devices have not been "seen recently" by AirGuard. The other classifiers are unaffected. The user appears to start collecting data again.
00:33 - 00:40	AirGuard's F1 score gradually recovers as the suspicious devices are "seen recently" again, moving them from false negatives to true positives.
00:50	AirGuard's F1 score gradually recovers as the suspicious devices are "seen recently" again, moving them from false negatives to true positives.
00:54 - 01:19	AirGuard's F1 score fluctuates due to the "seen recently" status of the suspicious devices changing as the user moves. One suspicious device is correctly classified for the remainder of the dataset, while one other suspicious device is intermittently classified as a false negative when it is not "seen recently". This devices had sporadic contact with the user with long breaks in between contacts. The remaining two suspicious devices are never classified correctly, due to being out of range for most of the remainder of the dataset.

01:14 - 01:16	The Smallest K-Cluster classifier's F1 score drops as two of the three suspicious devices it correctly identified are no longer clustered with the remaining device correctly classified as suspicious. This is due to lost or sporadic contact with these devices for an extended period of time while other devices continue to have increasing risk factor values.
01:14 - 01:26	The IQR / K-Means Hybrid classifier's F1 score decreases as risk score for two of the three suspicious devices it identified no longer exceed the threshold. This is likely due to lost or sporadic contact with these devices for an extended period of time while other devices continue to have increasing risk factor values.
01:17 - 01:21	The SCORE classifiers' F1 scores fluctuate one, then drop as two of the three suspicious devices they identified are no longer selected by the classifier due to these devices not exceeding the threshold set by the Jenks natural breaks optimization method. This is likely due to lost or sporadic contact with these devices for an extended period of time while other devices continue to have increasing risk factor values.
01:20	AirGuard starts to classify three out of four suspicious devices correctly as they come into range again, improving its F1 score for the remainder of the dataset.

A.3 BLE-Doubt Dataset C

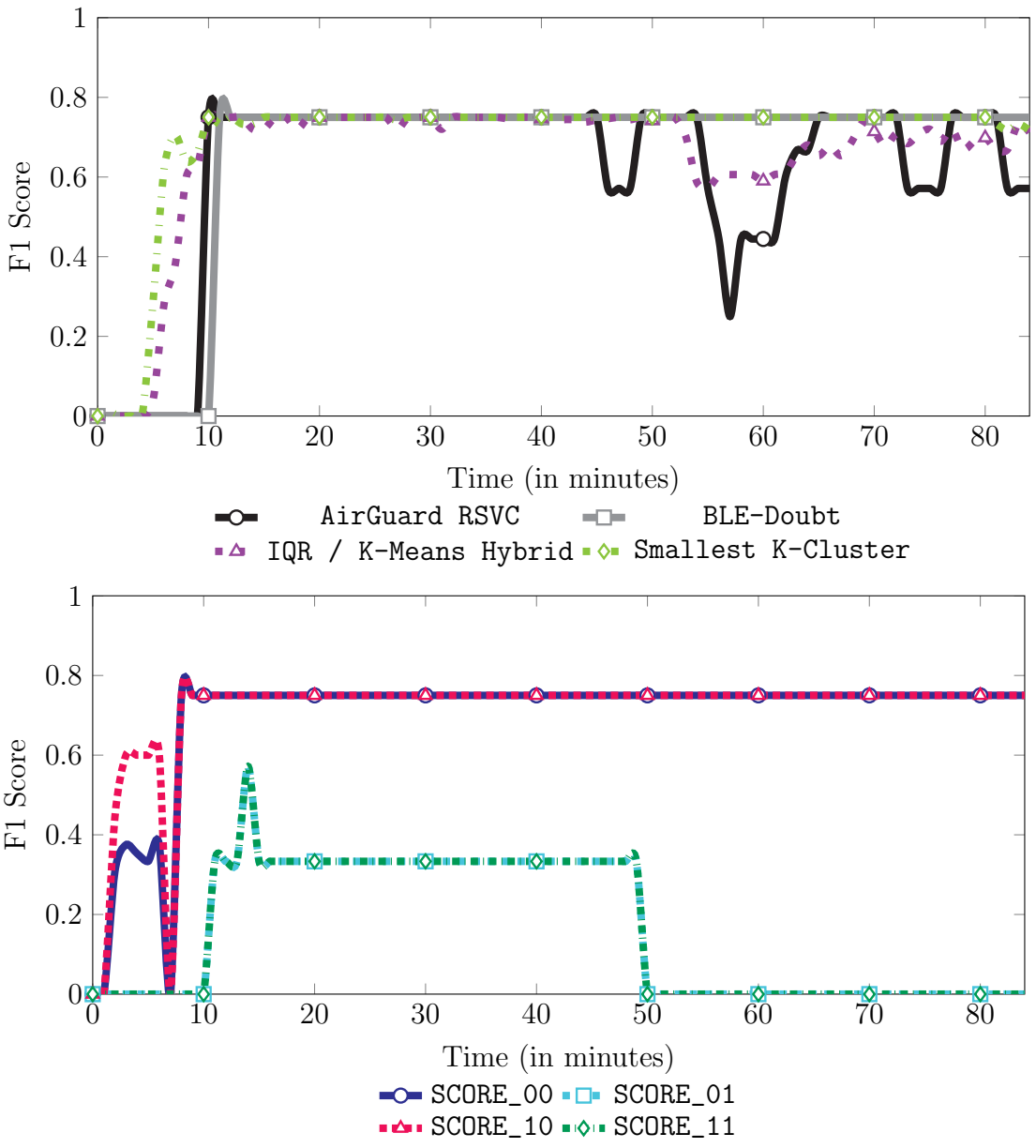

Figure A.3: Classifier F1 Scores for BLE-Doubt Dataset C

Note: Two of the five suspicious devices for this dataset were never classified correctly by any classifier due to sporadic short contact with the user.

This dataset demonstrates how the classifiers can quickly identify suspicious devices when the user has a consistent travel pattern, but some classifiers lose accuracy when the context (in this case, the speed of travel) changes.

00:02 - 00:06 The SCORE classifiers with the stability feature disabled start to identify three out of five suspicious devices correctly, but misclassify a high amount of non-suspicious devices as suspicious due to the small size of the dataset at this point. The SCORE classifiers with the stability feature enabled also misclassify a low amount of non-suspicious devices, but do not classify any suspicious devices correctly during this time.

00:05 - 00:09 The Smallest K-Cluster classifier has enough data to start identifying suspicious devices. It quickly classifies three out of five suspicious devices correctly, and continues to do so throughout the remainder of the dataset.

00:06 - 00:08	The average time with the user drops significantly among devices across the dataset, causing the SCORE classifiers with the stability feature disabled to temporarily misclassify all suspicious devices as non-suspicious at the 7-minute mark, but to also stop misclassifying non-suspicious devices as suspicious. By the 8-minute mark, the data has stabilized enough for the SCORE classifiers with the stability feature disabled to correctly classify three out of five suspicious devices, which they maintain for the remainder of the dataset. The SCORE classifiers with the stability feature enabled stop classifying non-suspicious devices as suspicious, but they do not classify any suspicious devices correctly during this time due to the lack of stable signal from the suspicious devices.
00:06 - 00:10	The IQR / K-Means Hybrid classifier has enough data to start identifying suspicious devices. It quickly classifies three out of five suspicious devices correctly.
00:10	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 9 and 10-minute mark.

00:11	The BLE-Doubt classifier starts flagging suspicious devices as the suspicious devices surpass its time near the user threshold. It correctly classifies three out of five suspicious devices correctly, but misses the remaining two suspicious devices throughout the rest of the dataset.
00:11 - 00:14	The SCORE classifiers with the stability feature enabled start to correctly classify one out of five suspicious devices as that device exhibits slightly more stable signal strength while near the user. It correctly classifies another suspicious devices at the 14-minute mark, but misclassifies it by the 15-minute mark due to unstable signal strength.
00:24 - 00:33	The user appears to stop collecting data for several minutes.
00:46 - 00:48	The AirGuard classifier temporarily misclassifies one suspicious device as non-suspicious due to extended contact without contact with the device, but it recovers a few moments after the user re-gains contact with the device.
00:50	The SCORE classifier with the stability feature enables starts to misclassify the one suspicious device it had previously classified correctly due to the device exhibiting unstable signal strength while near the user.

00:53 - 01:02	The IQR / K-Means Hybrid classifier starts to misclassify two non-suspicious devices as suspicious, causing its F1 score to drop significantly.
00:56 - 01:04	The AirGuard classifier starts to misclassify two non-suspicious devices as suspicious due to extended contact with the device while moving, causing its F1 score to drop.
01:03 - 01:23	The IQR / K-Means Hybrid classifier somewhat recovers, but continues to misclassify one non-suspicious device as suspicious, causing its F1 score to remain low.
01:13 - 01:16	The AirGuard classifier temporarily misclassifies one suspicious device as non-suspicious due to extended contact without contact with the device, but it recovers a few moments after the user re-gains contact with the device.
01:21 - 01:23	The Smallest K-Cluster classifier has a slight drop in F1 score, as it occasionally falters in correctly classifying one of the three suspicious devices that had been correctly classified up to this point.

A.4 BLE-Doubt Dataset D

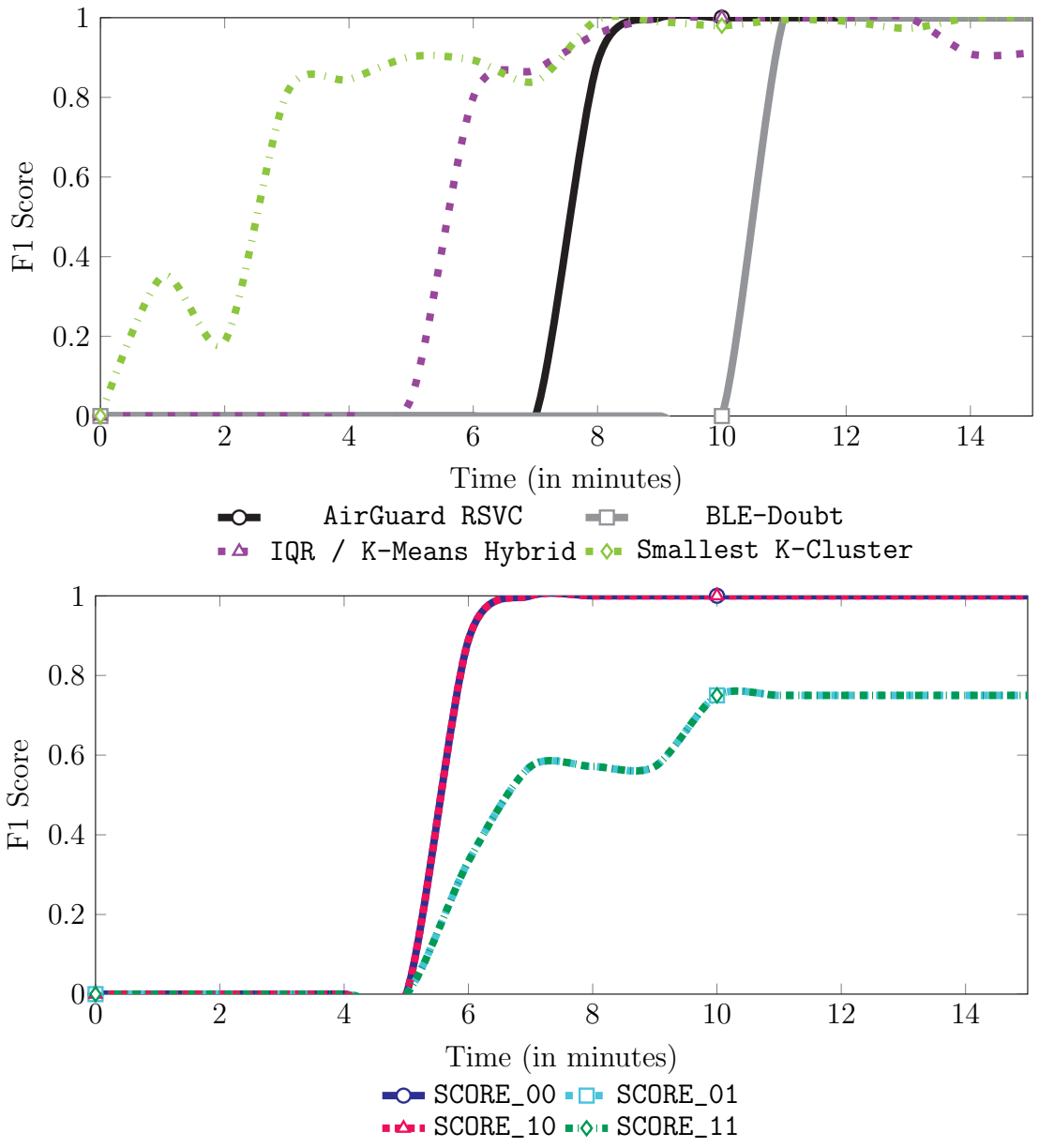

Figure A.4: Classifier F1 Scores for BLE-Doubt Dataset D

This dataset demonstrates how classifiers can quickly identify suspicious devices with limited data.

00:00 - 00:03	The Smallest K-Cluster classifier immediately identifies one suspicious device correctly, but incorrectly classifies two other devices as suspicious. Its F1 score gradually jumps up at the 3-minute mark as it collects collects enough data to more reliably classify a other suspicious devices correctly.
00:03 - 00:06	The Smallest K-Cluster classifier correctly classifies four out of five suspicious devices accurately, but sometimes misclassifies one non-suspicious device as suspicious.
00:05 - 00:08	The IQR / K-Means Hybrid classifier has enough data to start classifying devices. It is unreliable at the 5-minute mark, but by the 6-minute mark, it reliably classifies devices with an F1 score above 0.9.
00:06 - 00:07	The SCORE classifiers start classifying devices. The variants with the signal stability feature disabled quickly classify every device correctly, which they maintain for the remainder of the dataset. The variants with the signal stability feature enabled incorrectly classify three devices as non-suspicious at first based on their unstable signal strength.

00:06 - 00:08	The Smallest K-Cluster classifier's F1 score starts to classify all five suspicious devices correctly, and no longer misclassifies the one non-suspicious device as suspicious.
00:08	AirGuard starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 7 and 8-minute mark.
00:10	An additional suspicious device is correctly classified by the SCORE classifiers with the signal stability feature enabled, bringing its F1 score to 0.75.
00:11	BLE-Doubt starts flagging suspicious devices as the first devices it will classify as suspicious surpass its time threshold.
00:14	The IQR / K-Means Hybrid classifier slightly falls due to a device being misclassified as suspicious.

A.5 BLE-Doubt Dataset E

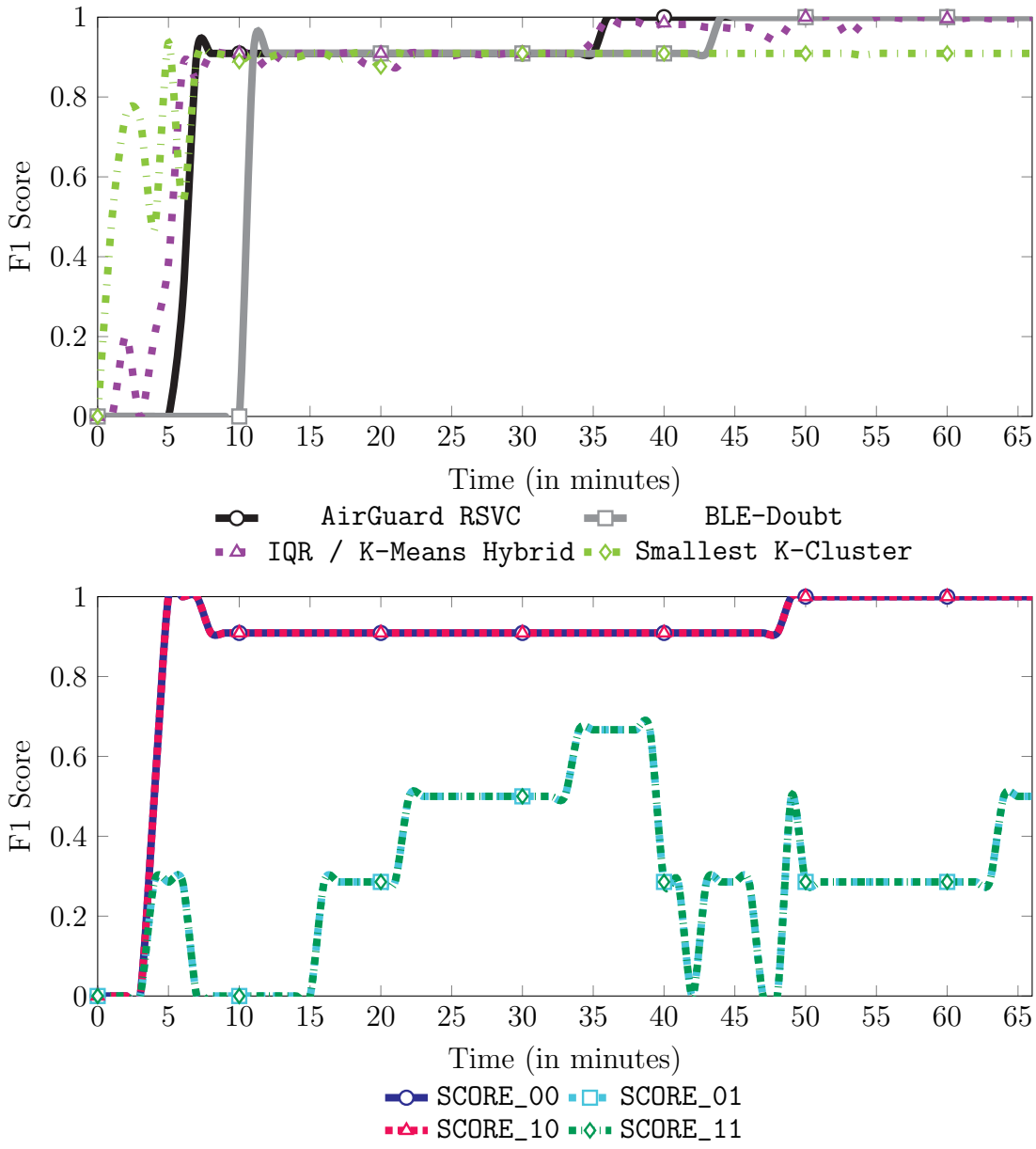

Figure A.5: Classifier F1 Scores for BLE-Doubt Dataset E

This dataset demonstrates how classifiers can quickly identify suspicious devices with consistent travel patterns and how they can identify new suspicious devices when the user is moving faster than the initial speed in a new context.

00:01 - 00:07 The Smallest K-Means classifier starts to identify suspicious devices. It always classifies at least one device as suspicious, and the dataset is small at this time, so it's F1 score fluctuates as it classifies two to five devices correctly out of six suspicious devices. At the 7-minute mark, it converges on correctly classifying five out of six suspicious devices, and it maintains this accuracy for the remainder of the dataset.

00:04 - 00:06	The IQR / K-Means Hybrid classifier starts to classify devices. It quickly identifies two to three suspicious devices and correctly classifies five out of six suspicious devices by the 6-minute mark. The SCORE classifiers with the signal stability feature disabled start to classify suspicious devices. They identify five out of six suspicious devices. The SCORE classifiers with the signal stability feature enabled also start to classify suspicious devices. However, their accuracy fluctuates wildly due to the stability or instability of each device's signal strength while near the user. All changes in their F1 scores are due to changes in recall, as they never misclassify any non-suspicious devices as suspicious during throughout this dataset.
00:06 - 00:07	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 5 and 6-minute mark. It starts to correctly classify five out of six suspicious devices by the 7-minute mark. It misses the remaining suspicious device due to lost contact with the user after a short period of time and short travel distance.
00:11	The BLE-Doubt classifier starts to correctly classify devices five out of six devices. It misses the remaining suspicious device due to lost contact with the user after a short period of time and short travel distance.

00:35 - 00:36	One suspicious device that had been incorrectly classified by all the classifier to this point in the dataset comes back into contact with the user as their travel speed increases, elevating its distance traveled with the user risk factor value. The AirGuard and IQR / K-Means Hybrid classifiers start to correctly classify this device as suspicious. They both consistently classify all six suspicious devices correctly for the remainder of the dataset.
00:44	The BLE-Doubt classifier starts to correctly classify the remaining suspicious device, and it continues to do so for the remainder of the dataset.
00:49	The SCORE classifiers with the signal stability feature disabled start to correctly classify the remaining suspicious device, and they continue to do so for the remainder of the dataset. The remaining device's distance traveled with the user risk factor value finally crosses the threshold set by the Jenks natural breaks optimization method into the higher echelon of values, so it is classified as suspicious.

A.6 BLE-Doubt Dataset F

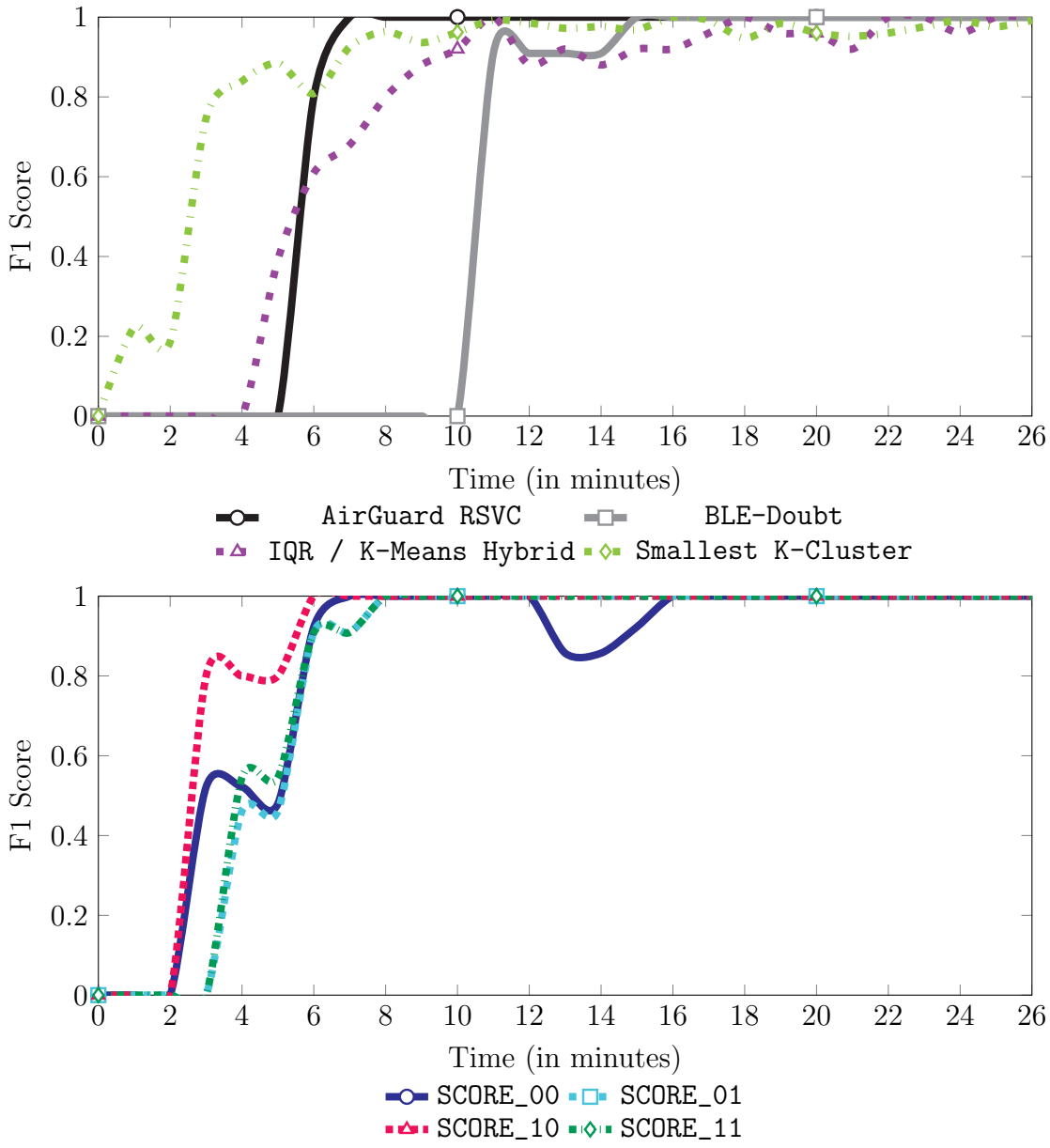

Figure A.6: Classifier F1 Scores for BLE-Doubt Dataset F

This dataset shows how classifiers can quickly identify suspicious devices with limited data.

00:01 - 00:03 The Smallest K-Cluster classifier starts to identify suspicious devices. It always classifies at least one device as suspicious, so it incorrectly classifies one to three non-suspicious devices as suspicious due to the small size of the dataset at this point.

00:03 - 00:05 The SCORE classifiers start to classify suspicious devices. The variants with the signal stability feature disabled do this slightly before the variants with the signal stability feature enabled. They all misclassify at least two non-suspicious devices as suspicious due to the small size of the dataset at this point.

The variant with only the proximity feature enabled performs the best, identifying all the suspicious devices correctly while only misclassifying three non-suspicious devices as suspicious.

The variant with both proximity and stability features disabled exhibits an unusually high number of false positives, likely because it is prone to classifying more devices as suspicious.

00:05 - 00:09 The IQR / K-Means Hybrid classifier has enough data to start classifying devices as suspicious. It gradually identifies more suspicious devices correctly, and it never incorrectly classifies devices as suspicious.

00:06 - 00:07	The AirGuard classifier starts to correctly classify devices as suspicious. At the 6-minute mark, it correctly classifies four suspicious trackers, and by the 7-minute mark, it correctly classifies the remaining two. It maintains this accuracy throughout the dataset.
00:07	The Smallest K-Cluster classifier's starts to correctly identify five to six suspicious devices correctly, but it fluctuates between misclassifying one suspicious device as non-suspicious and classifying all suspicious devices correctly.
00:10	The IQR / K-Means Hybrid classifier starts to consistently identify every suspicious device correctly. There are some instances where it received perfect scores, but wavers between 0.88 and 1.0 depending on how the clustering in initiated.
00:11	The BLE-Doubt classifier starts to correctly classify devices five devices as suspicious because they exceed its time near the user threshold.
00:13 - 00:15	The SCORE classifier with both proximity and stability features disabled temporarily misclassifies two non-suspicious devices as suspicious due to the lack of trend-based features that prevented this misclassification in the other SCORE variants.

00:15	The BLE-Doubt classifier starts to correctly classify the remaining suspicious devices. It maintains this accuracy throughout the dataset.

A.7 BLE-Doubt Dataset G

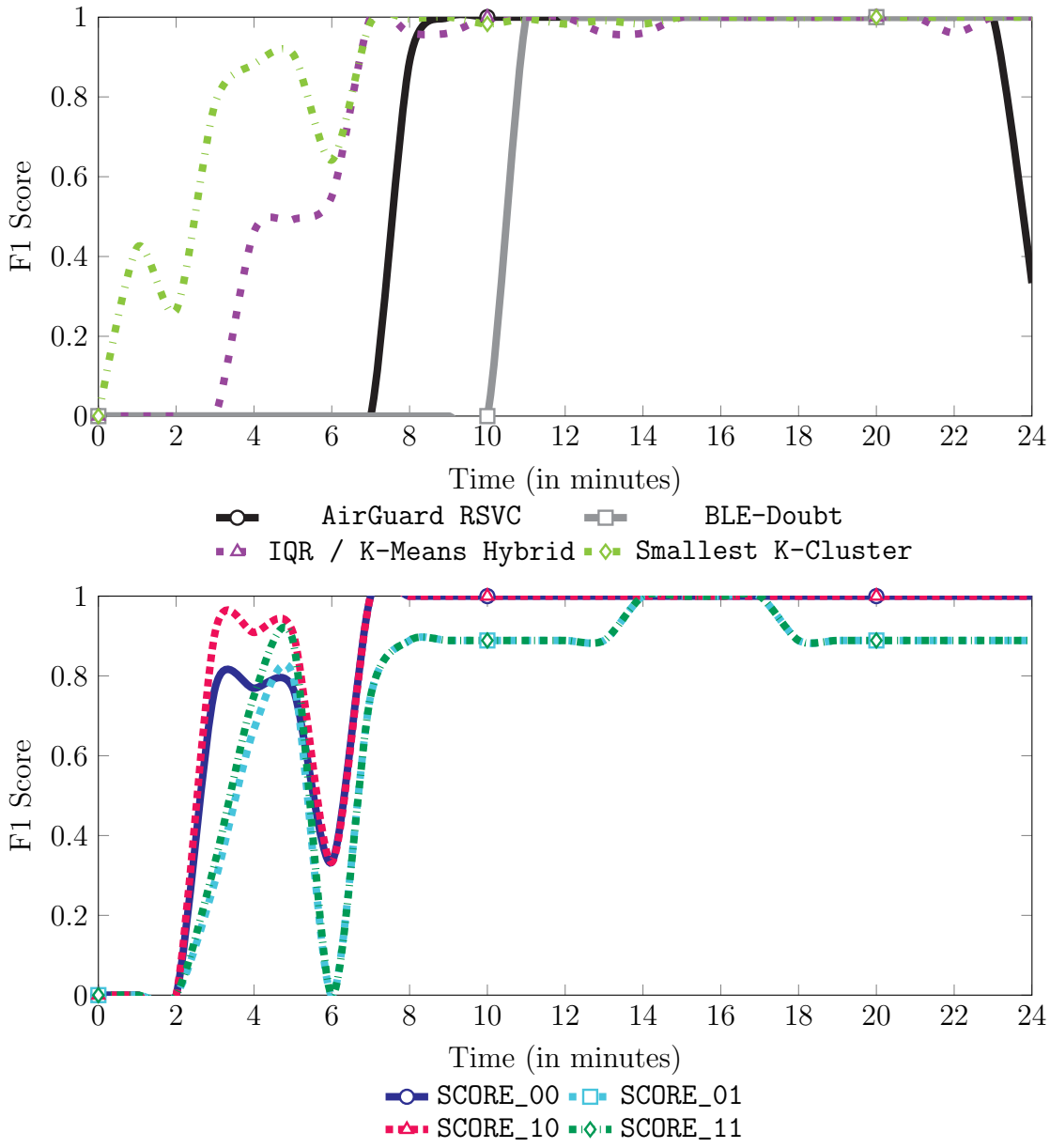

Figure A.7: Classifier F1 Scores for BLE-Doubt Dataset G

This dataset demonstrates how classifiers can quickly identify suspicious devices with limited data, but their performance can degrade by changes in user behavior, limited activity such as lack of movement and device availability.

00:01 - 00:06	The Smallest K-Cluster classifier starts to identify suspicious devices. It always classifies at least one device as suspicious, so it incorrectly classifies one to two non-suspicious devices as suspicious due to the small size of the dataset at this point.
00:03 - 00:05	The SCORE classifiers start to classify suspicious devices. The variants with the signal stability feature disabled do this slightly before the variants with the signal stability feature enabled. All SCORE variants, with the exception of the variant with both proximity and stability features enabled, misclassify one to three non-suspicious device as suspicious due to the small size of the dataset at this point.
00:04 - 00:06	The IQR / K-Means Hybrid classifier has enough data to start classifying devices. It incorrectly classifies two to three non-suspicious devices as suspicious.
00:06	The user appears to not be moving for a short period of time, causing the average distance with the user across all devices to decrease. This causes the SCORE classifiers to temporarily misclassify some or all suspicious devices as non-suspicious based on the distance thresholds.

00:07	Two devices that were near the user early in the dataset are no longer near the user, so the IQR / K-Means Hybrid and Smallest K-Cluster classifiers stop misclassifying non-suspicious devices as suspicious. They both have enough data to correctly classify all suspicious devices consistently. The SCORE classifiers start to consistently classify all suspicious devices correctly. The variants with the signal stability feature disabled maintain perfect F1 scores for the remainder of the dataset, while the variants with the signal stability feature enabled consistently misclassify one suspicious device as non-suspicious due the device's unstable signal strength.
00:08	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 7 and 8-minute mark.
00:11	The BLE-Doubt classifier starts flagging suspicious devices as the suspicious devices surpass its time near the user threshold. It immediately classifies all suspicious devices correctly, and continues to do so throughout the dataset.
00:16 - 00:23	The user appears to stop collecting data for several minutes.
00:23	The AirGuard classifier loses contact with all suspicious devices, causing it to misclassify four out of five suspicious devices after not receiving any recent data from them.

A.8 BLE-Doubt Dataset H

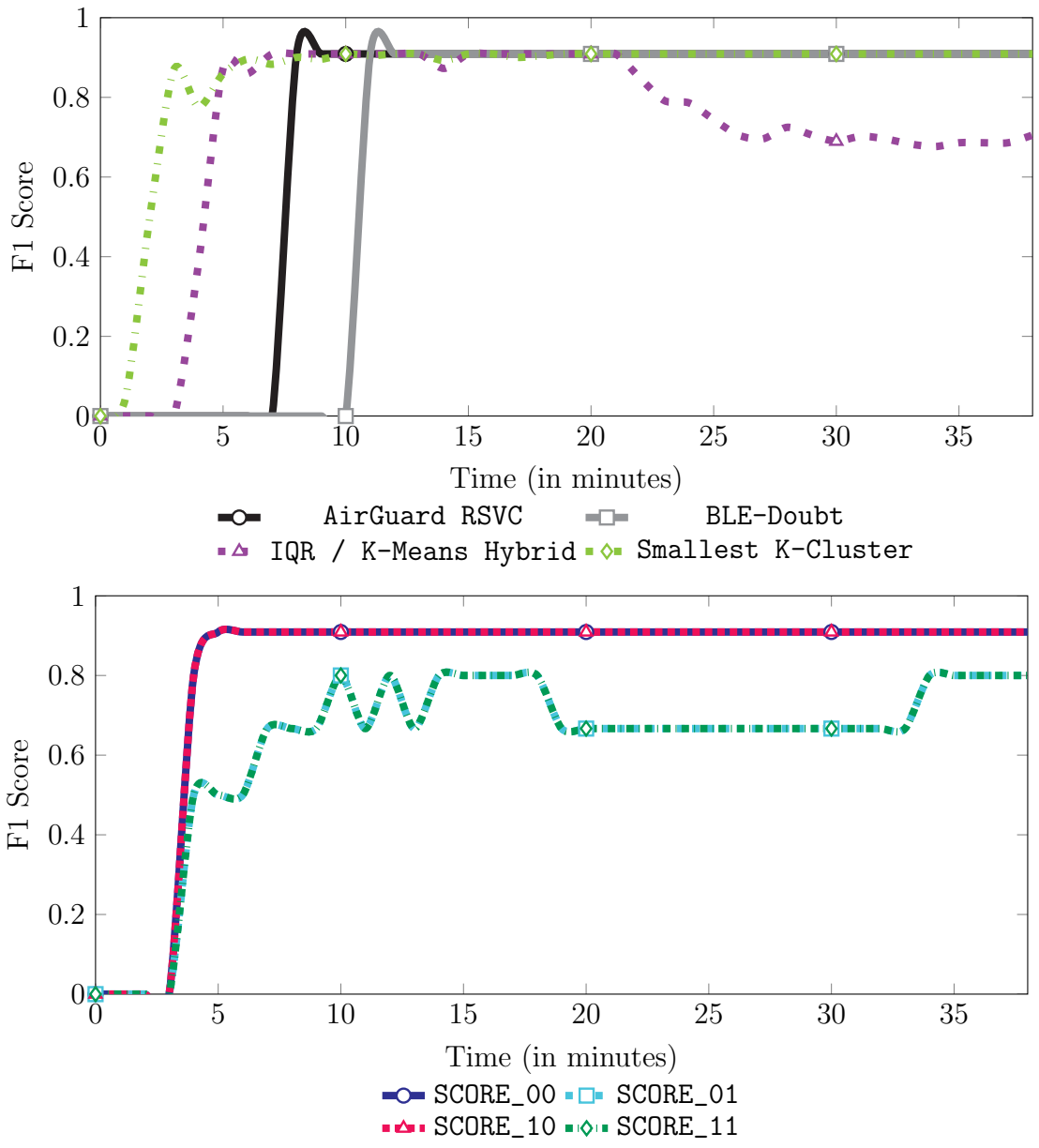

Figure A.8: Classifier F1 Scores for BLE-Doubt Dataset H

This dataset demonstrates how classifiers can quickly identify suspicious devices, but composite scoring can degrade accuracy due to not recognizing similar behavior among suspicious devices.

00:02 - 00:05	The Smallest K-Cluster classifier starts to identify suspicious devices. It occasionally classifies one or two non-suspicious device as suspicious due to the small size of the dataset at this point, but the number of incorrectly classified non-suspicious devices decreases as the dataset size increases. It correctly identifies five out of six suspicious devices for the remainder of the dataset.
00:04 - 00:05	The IQR / K-Means Hybrid classifier start to correctly identify five out of six suspicious devices.
00:06	The SCORE classifiers start to classify suspicious devices. The variants with the signal stability feature disabled consistently classify five out of six suspicious devices correctly, which they maintain for the remainder of the dataset. The variants with the signal stability feature enabled incorrectly classify two to three suspicious device as non-suspicious due to their unstable signal strength. Their F1 scores fluctuate as these devices exhibit varying levels of signal stability throughout the dataset.

00:08	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 7 and 8-minute mark. It correctly identifies five out of six suspicious devices for the remainder of the dataset.
00:11	The BLE-Doubt classifier starts flagging suspicious devices because the user has traveled long enough for devices to surpass its time threshold. It correctly identifies five out of six suspicious devices for the remainder of the dataset.
00:22 - 00:38	The average distance with the user increases among devices across the dataset, causing the IQR / K-Means Hybrid classifier to lose accuracy as it misclassifies several non-suspicious devices as suspicious. It consistently classifies five out of six suspicious devices correctly, but the increase of false positives causes its F1 score to drop significantly.

A.9 BLE-Doubt Dataset I

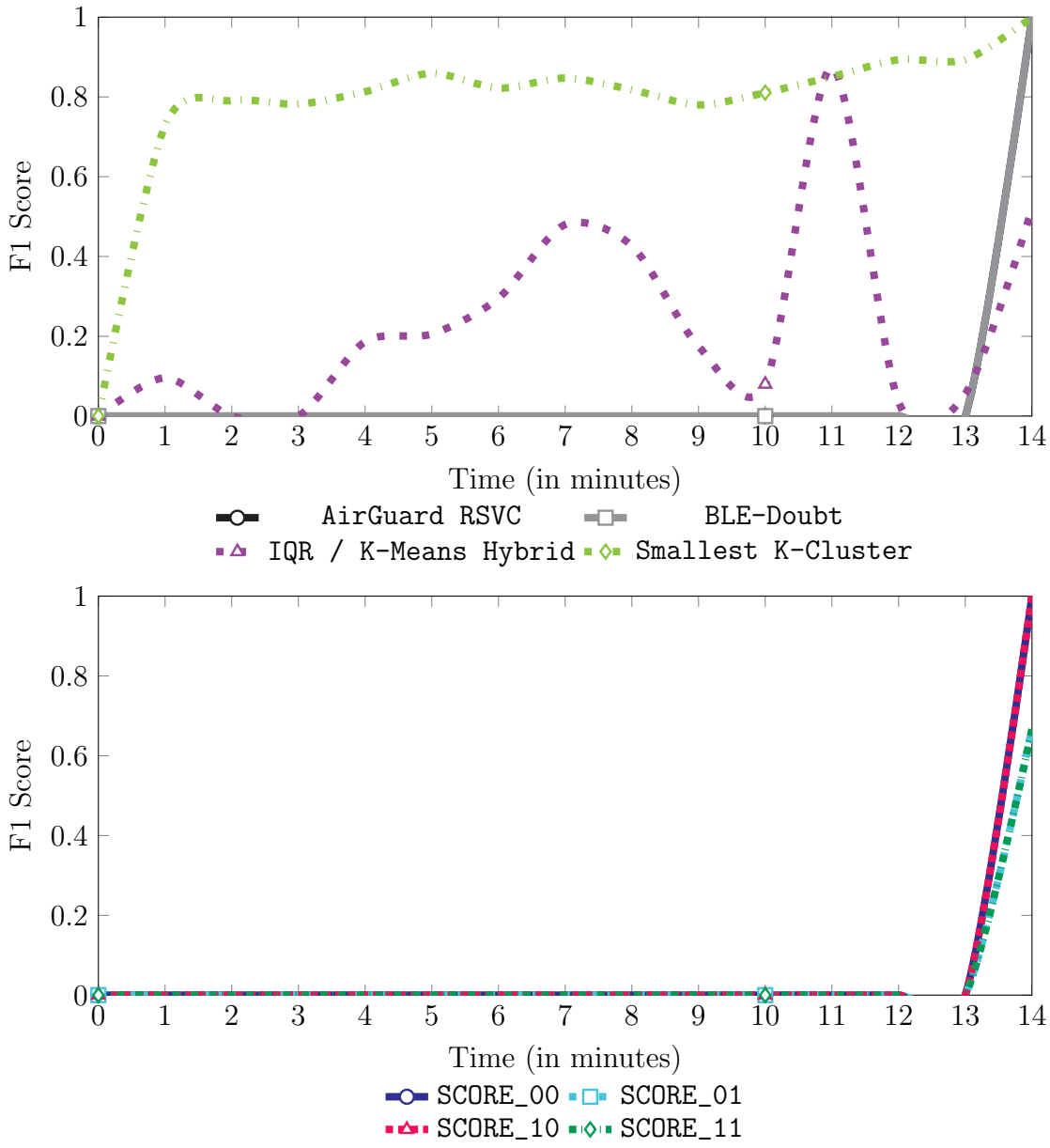

Figure A.9: Classifier F1 Scores for BLE-Doubt Dataset I

This dataset demonstrates how multi-dimensional classifiers can outperform static threshold and single-dimensional classifiers when there is a lack of variation in the data on some dimensions.

00:01 - 00:02	The Smallest K-Cluster classifier quickly identifies three of four suspicious device correctly.
00:04 - 00:10	The IQR / K-Means Hybrid classifier is only classifying devices based on duration of time with the user due to the user not moving during most of the dataset. As a result, it fluctuates between correctly classifying one to three suspicious devices while simultaneously misclassifying one to three non-suspicious devices as suspicious in a similar pattern.
00:06 - 00:09	The Smallest K-Cluster classifier's F1 score fluctuates slightly as it gradually identifies the fourth suspicious device correctly, but increasingly also incorrectly classifies two non-suspicious device as suspicious at times.
00:11	The IQR / K-Means Hybrid classifier correctly classifies three out of four devices as suspicious, but this falls back to zero because the user is not moving. These devices are being classified this way based on duration of time with the user alone, and is being heavily influenced by the behavior of other non-suspicious devices.

00:14 Because the user starts moving with less than a minute left in the dataset, the AirGuard, BLE-Doubt and SCORE classifiers don't flag any suspicious devices until the very end.

The AirGuard and BLE-Doubt classifiers correctly classify all devices.

The SCORE classifiers with signal stability disabled correctly classify all devices, but the variants with signal stability enabled incorrectly classify two out of four suspicious device as non-suspicious due to their unstable signal strength.

The Smallest K-Cluster classifier stops misclassifying a non-suspicious device as suspicious, improving its F1 score.

The IQR / K-Means Hybrid classifier correctly classifies two out of four devices as suspicious, but does not have enough variation in data with other devices to correctly classify the remaining two suspicious devices.

A.10 BLE-Doubt Dataset J

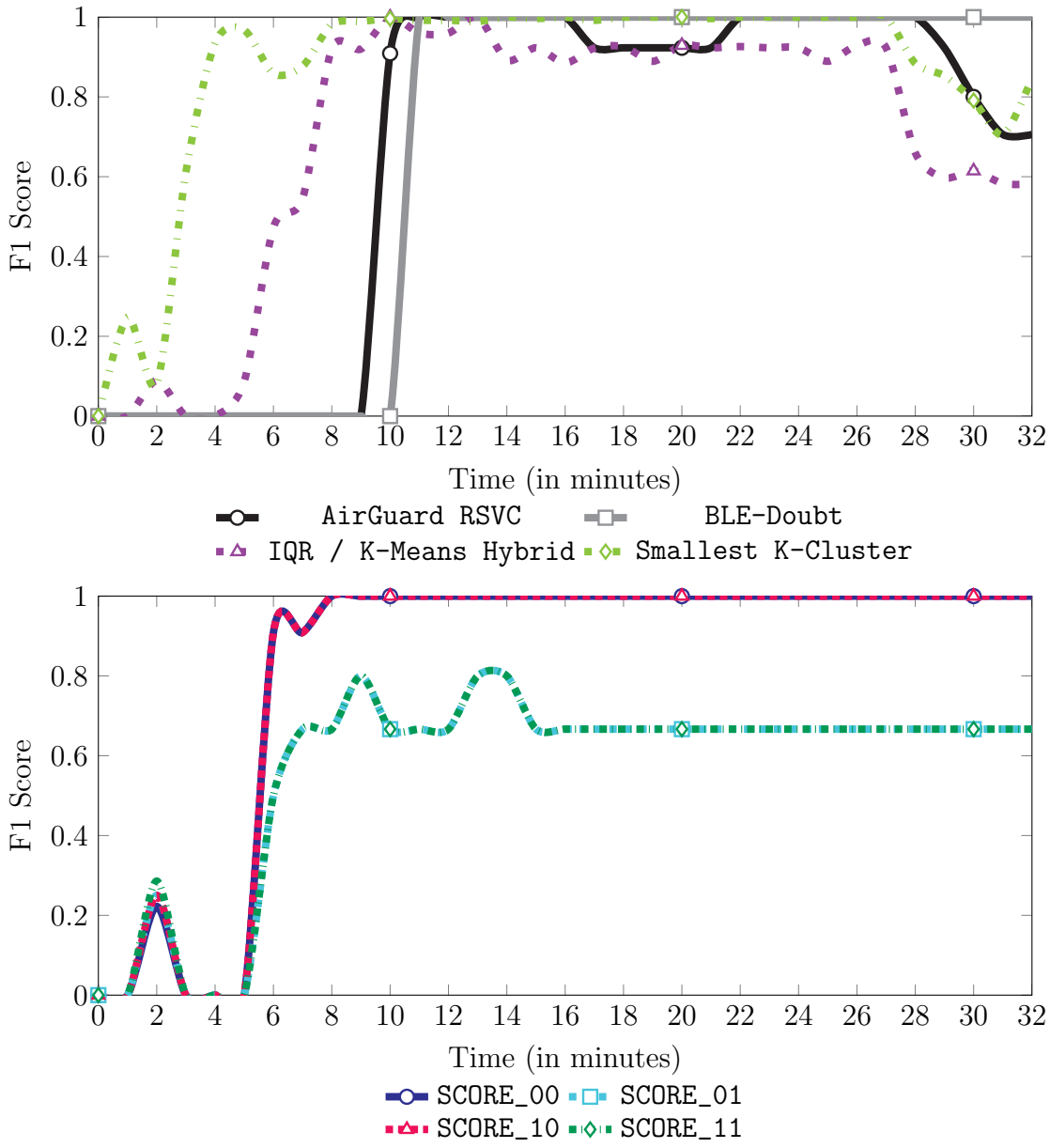

Figure A.10: Classifier F1 Scores for BLE-Doubt Dataset J

This dataset demonstrates the performance of the classifiers in a scenario where the user travels at high speed after a period of slow movement. The dynamic classifiers are more vulnerable to this context change.

00:01 - 00:04	The Smallest K-Cluster classifier starts to identify suspicious devices. It always classifies at least one device as suspicious, so it incorrectly classifies one to two non-suspicious device as suspicious due to the small size of the dataset at this point. The number of incorrectly classified non-suspicious devices decreases as the dataset size increases.
00:02	The IQR / K-Means Hybrid classifier and SCORE classifiers may inadvertently classify devices as suspicious or not due to the small size of the dataset at this point. Over several simulations, the IQR / K-Means Hybrid classifier occasionally classifies one suspicious device correctly, but more often classifies one non-suspicious devices incorrectly.
00:05 - 00:08	The IQR / K-Means Hybrid classifier has enough data to start classifying devices. It gradually identifies more suspicious devices correctly.

00:06 - 00:08	The SCORE classifiers have enough data about the user's travel distance to start classifying suspicious devices. The variants with the signal stability feature disabled classify all suspicious devices correctly, while the variants with the signal stability feature enabled consistently misclassify two to three devices as non-suspicious due to their unstable signal strength.
00:10 - 00:11	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 9 and 10-minute mark. By the 11-minute mark, it classifies all suspicious devices correctly and maintains this accuracy for most of the remainder of the dataset.
00:11	The BLE-Doubt classifier starts flagging suspicious devices because the user has traveled long enough for devices to surpass its time threshold. It maintains a perfect F1 score for the remainder of the dataset.
00:14 - 00:27	The IQR / K-Means Hybrid classifier correctly classifies all suspicious devices consistently, but also misclassifies one non-suspicious device as suspicious.
00:17 - 00:21	The AirGuard classifier temporarily misclassifies one non-suspicious device as suspicious due to extended contact with the device while moving, but recovers a few moments after the user loses contact with the device.

00:28 - 00:31	The user experiences a dramatic change in speed in a short period of time, causing the IQR / K-Means Hybrid classifier and Smallest K-Cluster classifiers to misclassify several non-suspicious device as suspicious. The devices in the dataset that are present for this context change have elevated distance traveled with the user values compared to the other devices in the dataset, causing the devices with elevated value to be misclassified as suspicious.

A.11 BLE-Doubt Dataset K

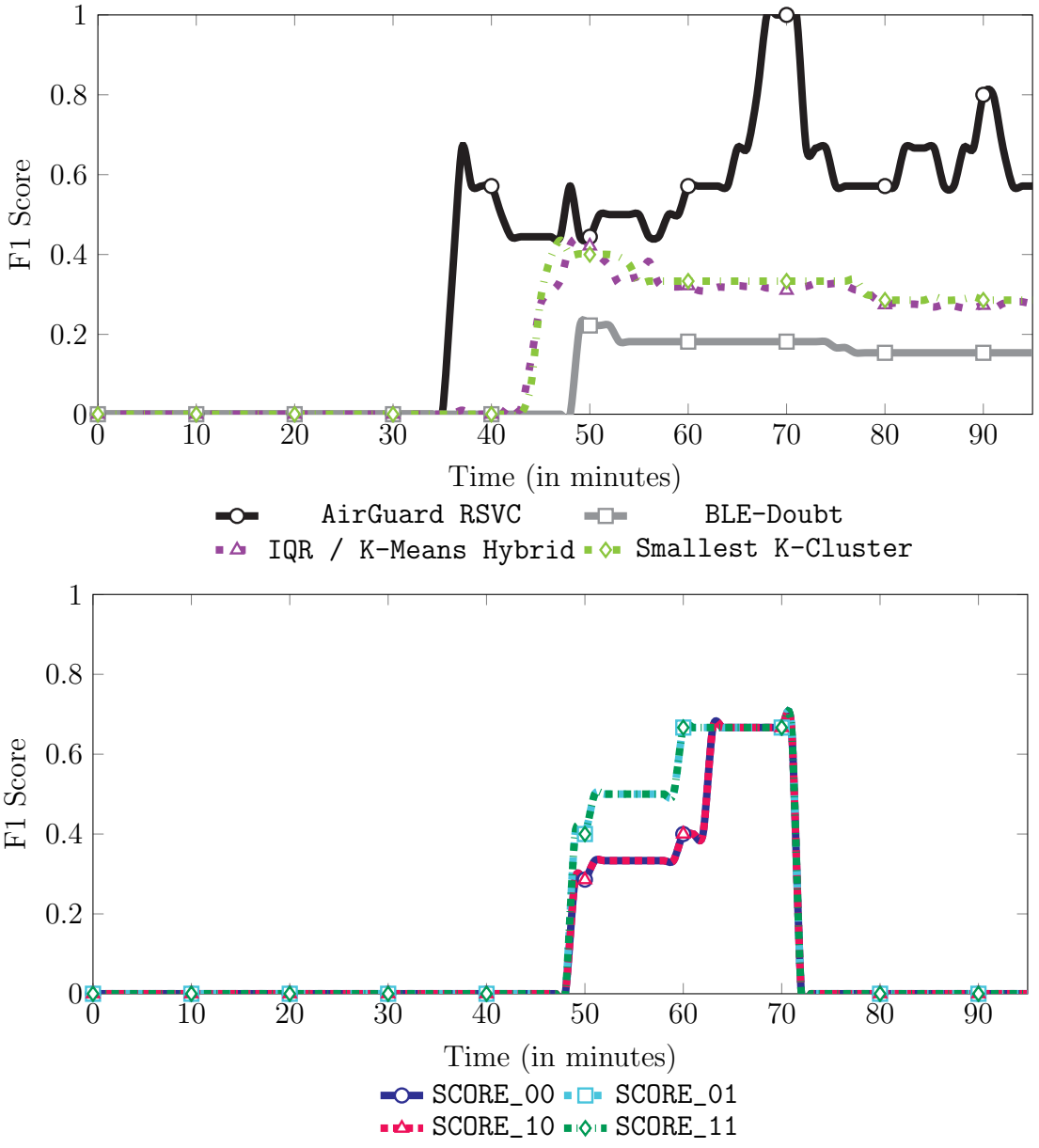

Figure A.11: Classifier F1 Scores for BLE-Doubt Dataset K

This dataset demonstrates the performance of the classifiers when contact with suspicious devices is inconsistent compared to other nearby devices. All the classifiers struggle to maintain accuracy in this scenario.

00:01 - 00:04	All dynamic classifiers start to misclassify devices as suspicious.
00:01 - 00:04	The AirGuard and BLE-Doubt classifiers starts to misclassify devices as suspicious.
00:36 - 00:37	The AirGuard classifier starts to correctly classify both suspicious devices, but continues to misclassify many non-suspicious devices as suspicious. It continues to correctly identify both suspicious devices for the remainder of the dataset, and continues to misclassify three to five non-suspicious devices as suspicious.
00:45 - 00:46	The IQR / K-Means and Smallest K-Cluster classifiers start to consistently classify both suspicious devices correctly, but continue to misclassify many non-suspicious devices as suspicious. They continue the correctly identify both suspicious devices for the remainder of the dataset, and continue to misclassify up to ten or eleven non-suspicious devices as suspicious for the remainder of the dataset.

00:45 - 01:03	The SCORE classifiers start to consistently classify one suspicious device correctly. During this time, the variants with the signal stability feature disabled misclassify more non-suspicious devices as suspicious than the variants with the signal stability feature enabled. All variants experience at least one or two non-suspicious devices that maintain a strong and stable signal strength.
00:49	The BLE-Doubt classifier starts to correctly classify one suspicious device, but continues to misclassify many non-suspicious devices as suspicious, and will continue to incorrectly classify more non-suspicious devices as suspicious for the remainder of the dataset.
01:08 - 01:11	The AirGuard classifier temporarily ceases to classify non-suspicious devices as suspicious, and continues to correctly classify both suspicious devices. After this time, the AirGuard classifier starts to misclassify one to three non-suspicious devices as suspicious again and will do so for the remainder of the dataset.

01:12	The suspicious devices have sporadic contain with the user, so all variants of the SCORE classifiers start to misclassify both suspicious devices as non-suspicious. The variants with the signal stability feature disabled continue to misclassify many non-suspicious devices as suspicious, but the variants with the signal stability feature enabled no longer misclassify any non-suspicious devices as suspicious.

A.12 BLE-Doubt Dataset L

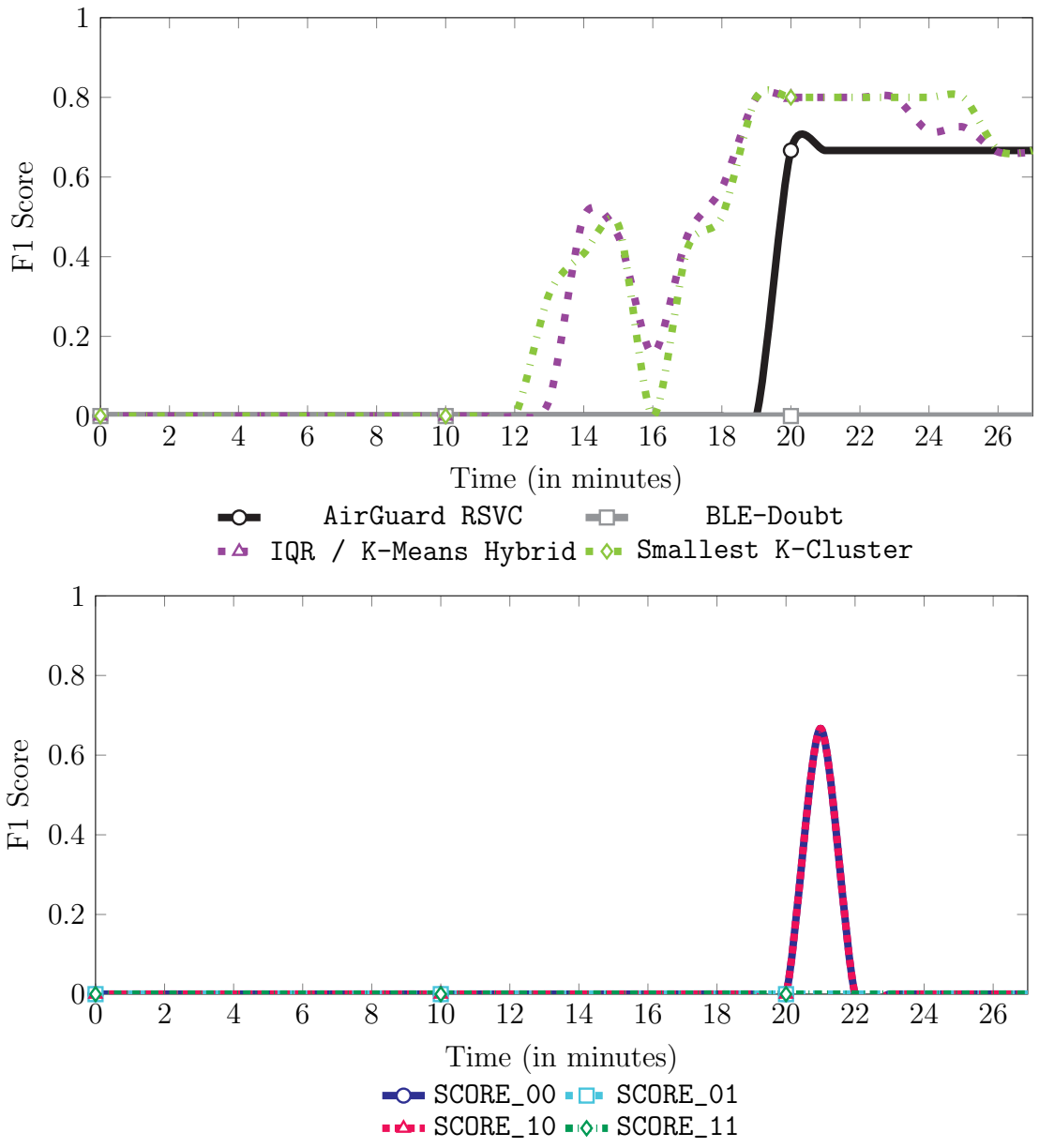

Figure A.12: Classifier F1 Scores for BLE-Doubt Dataset L

This dataset demonstrates the performance of the classifiers when the scanned data is inconsistent but eventually stabilizes.

00:02 - 00:08	The user appears to stop collecting data for several minutes.
00:10 - 00:11	The user appears to again stop collecting data.
00:13 - 00:14	The Smallest K-Cluster and IQR / K-Means Hybrid classifiers gets enough data on one suspicious device to start classifying it correctly. The user loses contact with this device at about this time.
00:16	The Smallest K-Cluster and IQR / K-Means Hybrid classifiers incorrectly classify the one suspicious device due to losing contact with it, but they quickly recover as the user regains contact with the device. The IQR / K-Means Hybrid classifier more consistently classifies the device as suspicious during this period than the Smallest K-Cluster classifier.
00:19	The IQR / K-Means Hybrid classifier starts correctly classifying both suspicious devices as the user retains consistent contact with both devices.
00:20	AirGuard starts to correctly classify one device as devices because the device exceeds AirGuard's duration of time traveled.

00:21	The SCORE classifiers with signal stability disabled correctly classify one suspicious device. This is the only time when it classifies any suspicious devices correctly during the dataset. The SCORE classifiers with signal stability enabled do not classify any suspicious devices correctly at any point during the dataset.
00:26	BLE-Doubt never classified either suspicious device correctly during the dataset due to neither device exceeding its duration of time traveled with the user threshold.

A.13 BLE-Doubt Dataset M

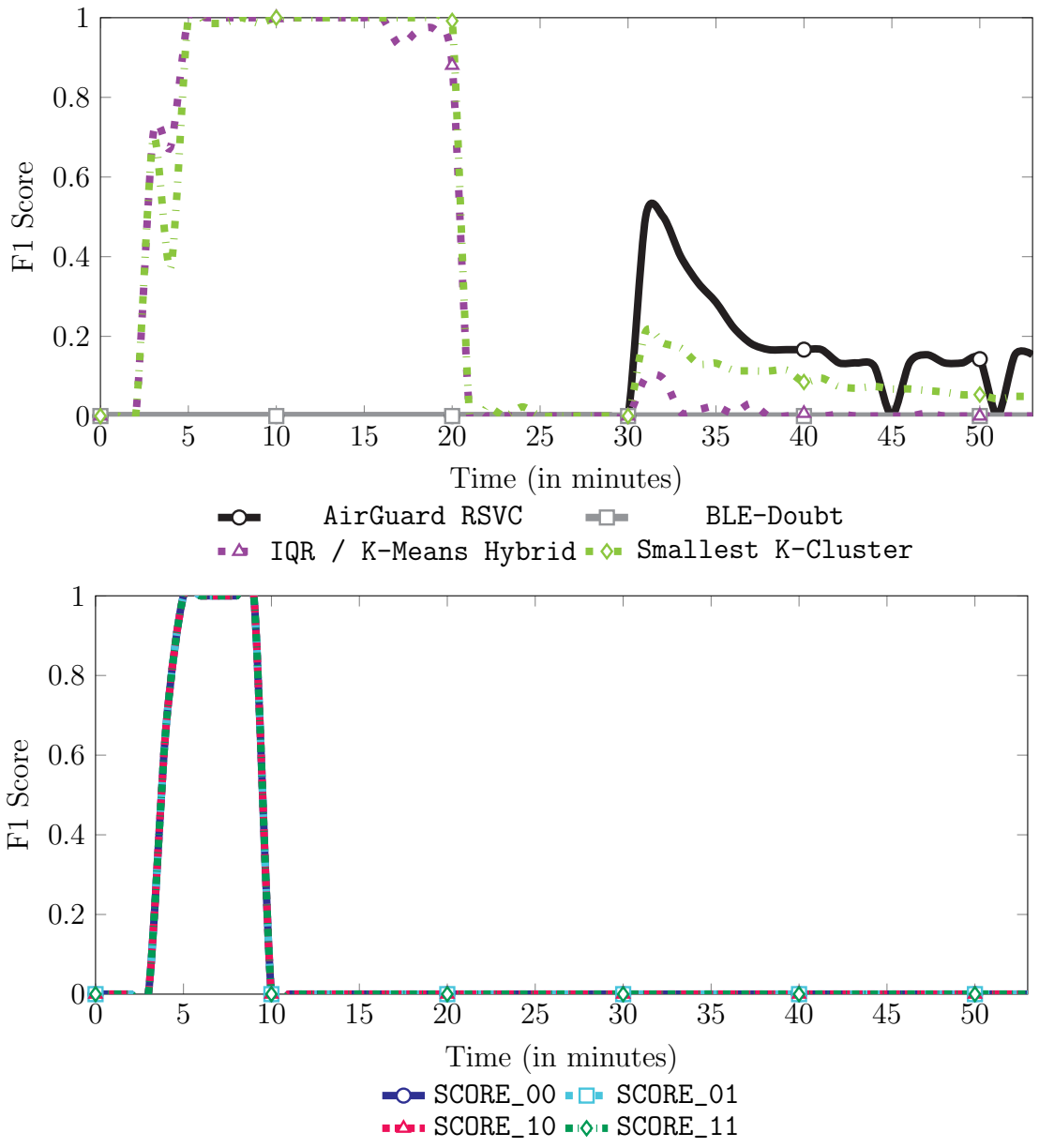

Figure A.13: Classifier F1 Scores for BLE-Doubt Dataset M

Note: AirGuard, IQR / K-Means Hybrid, and the Smallest K-Cluster classifiers exhibited an abnormally high number of false positives during this dataset.

This dataset demonstrates the performance of the classifiers when the user's speed is inconsistent across contexts. This dataset also covers when a device disappears before a context change, which greatly affects the performance across all classifiers.

00:00	The BLE-Doubt never correctly classified either suspicious device due to neither device exceeding its duration of time traveled with the user threshold, and one device never exceeding its distance traveled with the user threshold.
00:03 - 00:05	The Smallest K-Cluster, IQR / K-Means Hybrid and SCORE classifiers collect enough data to start identifying both suspicious devices correctly.
00:07	One of the suspicious devices disappears from the dataset, and does not reappear at any point throughout the remainder of the dataset.

00:10	One suspicious device disappears from the dataset at the 7-minute mark, and the other disappears at the 6-minute mark. The risk factors for these devices no longer exceed the SCORE classifier's dynamically-set threshold for the remainder of the dataset, even after one of the suspicious devices reappears later in the dataset.
00:31 - 00:37	The AirGuard classifier starts to consistently classify one of the suspicious devices correctly as it reappears in the dataset when the user moves at a high speed. The Smallest K-Cluster and IQR / K-Means Hybrid classifier temporarily classifies one suspicious device correctly as the suspicious device reappears in the dataset when the user is moving at a high speed, but it quickly starts to misclassify the device again due to sporadic contact with the device. The user is on a train with other devices, causing the AirGuard, Smallest K-Cluster and IQR / K-Means Hybrid classifiers to misclassify several non-suspicious devices as suspicious due to extended contact while moving.

A.14 BLE-Doubt Dataset N

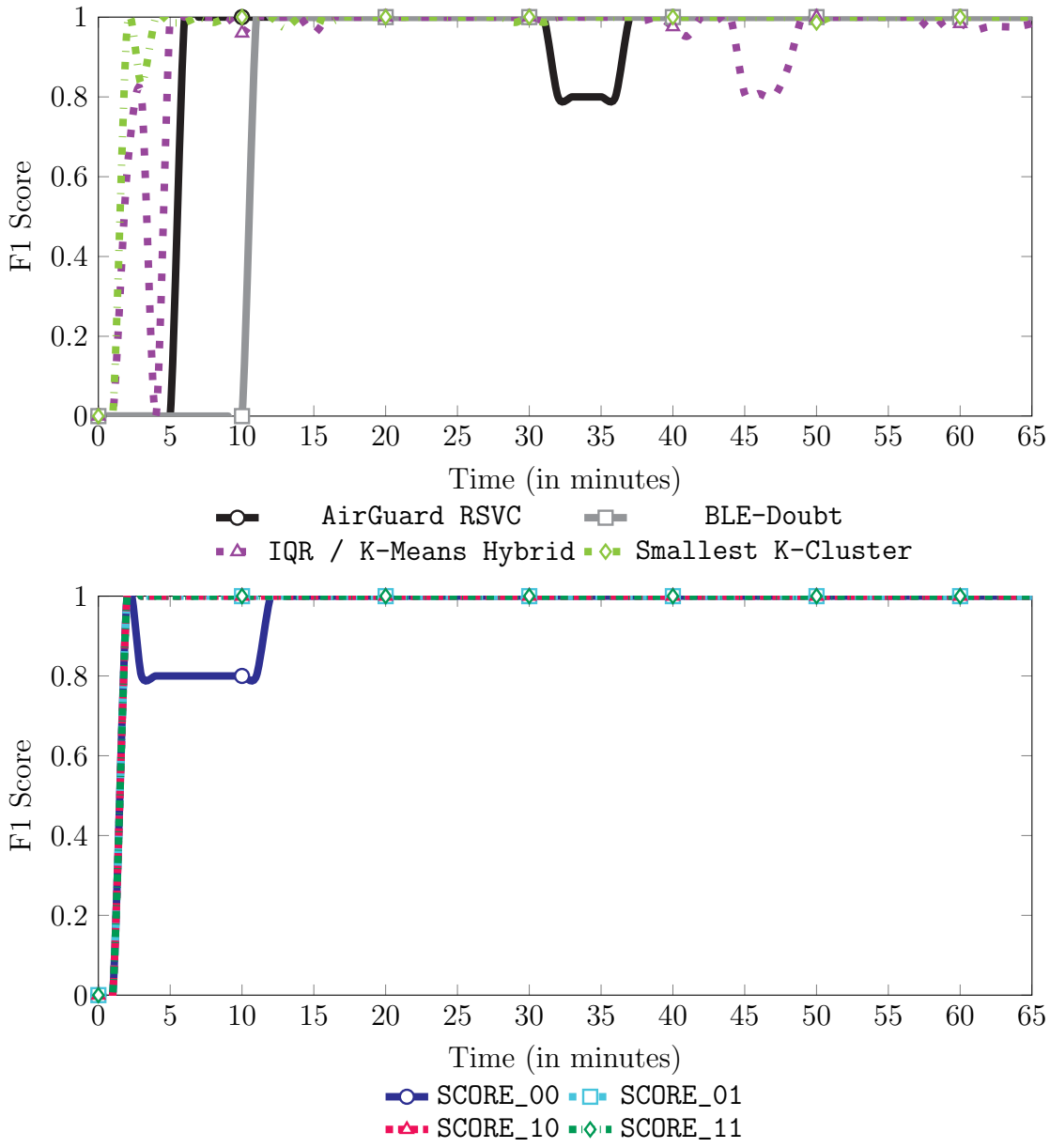

Figure A.14: Classifier F1 Scores for BLE-Doubt Dataset N

This dataset demonstrates the performance of the classifiers in a long-standing dataset with a consistent user context and close proximity to the suspicious devices.

00:01 - 00:03	The Smallest K-Cluster classifier starts to consistently classify all suspicious devices correctly, but also misclassifies one non-suspicious device as suspicious during this time, likely due to the small size of the dataset at this point.
00:02	The SCORE classifiers all identified both suspicious devices correctly. All but the variant with both proximity and stability features disabled maintain perfect F1 scores for the remainder of the dataset.
00:03 - 00:12	The SCORE classifier with both proximity and stability features disabled incorrectly classify one non-suspicious device as suspicious. This occurred because the misclassified device had elevated values for time and distance traveled with the user, lower RSSI values and an unstable signal strength. The other variants of the SCORE classifier correctly classified this device based on the trend-based risk factors. This variant of the SCORE classifier returns to correctly classifying all devices by the 12-minute mark.

00:04	The Smallest K-Cluster classifier has enough data to correctly classify the suspicious devices consistently without misclassifying any non-suspicious devices as suspicious. It maintains a near-perfect F1 score for the remainder of the dataset. The IQR / K-Means Hybrid classifier exhibits unexpected behavior by incorrectly classifying both suspicious devices as non-suspicious. Although the exact reason is unclear, this is likely due to the clustering portion small size of the dataset at this point
00:06	The AirGuard classifier starts flagging suspicious devices because the user hits AirGuard's distance traveled threshold between the 1 and 2-minute mark and its duration of time traveled with the user threshold at the 6-minute mark.
00:11	The BLE-Doubt classifier starts flagging suspicious devices because the user has traveled long enough for devices to surpass its time threshold. It maintains a perfect F1 score for the remainder of the dataset.
00:32 - 00:37	The AirGuard classifier's F1 score temporarily drops as one device is incorrectly classified as suspicious. This is due to the device being near the user for an extended period of time while the user's train was moving. It recovers shortly after the device moves away from the user.

00:45 - 00:48	The IQR / K-Means Hybrid classifier temporarily misclassifies one device as suspicious due to the user moving at a higher speed, causing the misclassified device to have an elevated value for distance traveled with the user.

A.15 *BL(u)E CRAB* Dataset O

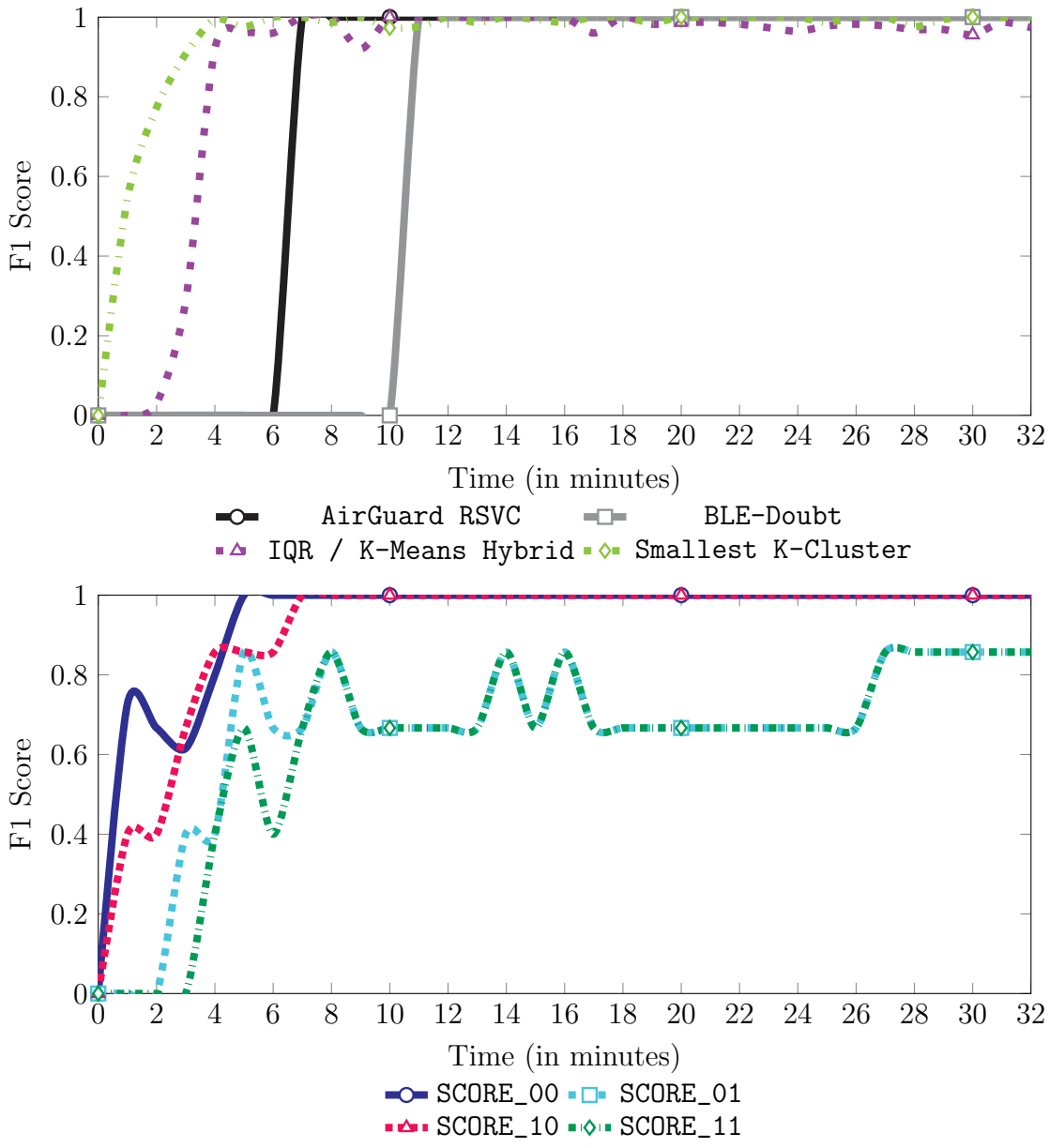

Figure A.15: Classifier F1 Scores for *BL(u)E CRAB* Dataset O

This dataset demonstrates how classifiers can quickly identify suspicious devices on a medium-sized dataset with consistent user context and close proximity to the suspicious devices.

00:01 - 00:04	The Smallest K-Cluster classifier starts to correctly classify all four suspicious devices. It maintains an F1 score over 0.97 for the remainder of the dataset.

00:01 - 00:07	The SCORE classifiers with the signal stability feature disabled start to correctly classify all four suspicious devices. They maintain a perfect F1 score for the remainder of the dataset after the 7-minute mark. The SCORE classifier with both proximity and stability features disabled also starts to correctly classify all four suspicious devices, but it temporarily misclassifies one to two non-suspicious devices as suspicious due to the lack of trend-based features that prevented this misclassification in the other variants. It corrects this issue by the 7-minute mark. From the 1 to 4-minute mark, the variant with the proximity feature and signal stability feature disabled incorrectly classified non-suspicious devices as suspicious, but it corrected the issue by the 5-minute mark. This is the only variant to incorrectly classify any non-suspicious devices as suspicious during this dataset, likely because it is more prone to classifying devices as suspicious due to the lack of trend-based features that prevented this misclassification in the other variants.
00:03 - 00:04	The IQR / K-Means Hybrid classifier starts to correctly classify all four suspicious devices. It maintains an F1 score over 0.92 for the remainder of the dataset.

00:03 - 00:08	The SCORE classifiers with the signal stability feature enabled start to classify suspicious devices. It fluctuates between correctly classifying two to three suspicious devices due to the stability or instability of each device's signal strength while near the user.
00:07	The AirGuard classier starts to correctly classify all four suspicious devices because the user hits AirGuard's distance traveled threshold between the 6 and 7-minute mark. It maintains this accuracy for the remainder of the dataset.
00:11	The BLE-Doubt classifier starts to correctly classify all four suspicious devices because the user has traveled long enough for devices to surpass its time threshold. It maintains this accuracy for the remainder of the dataset.

A.16 *BL(u)E CRAB* Dataset P

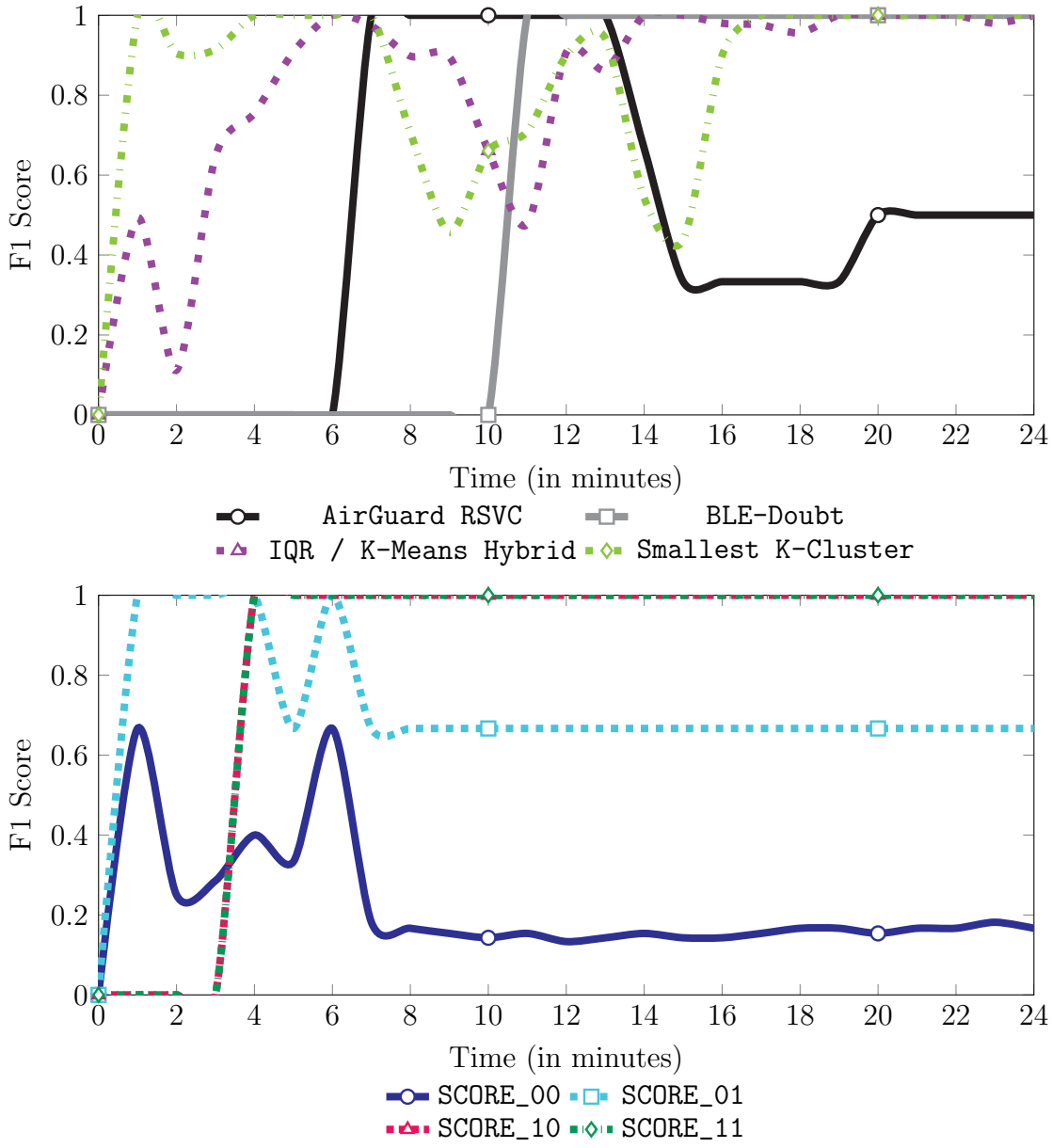

Figure A.16: Classifier F1 Scores for *BL(u)E CRAB* Dataset P

This dataset demonstrates how classifiers identify suspicious devices in a dataset with abnormal user movements in a small area, which causes elevated risk factor values in some nearby devices.

00:01 The Smallest K-Cluster classifier quickly identifies the suspicious device.

The IQR / K-Means Hybrid classifier starts to correctly classify the suspicious device. It initially misclassifies one non-suspicious device as suspicious, but it corrects this issue by the 6-minute mark.

The SCORE classifiers with the signal stability feature disabled start to correctly classify the suspicious device. The variant with the proximity and signal stability features disabled start to misclassify many non-suspicious device as suspicious due to the user moving in a small area and this variant of SCORE's tendency to classify more devices as suspicious than the other variants.

00:04 The SCORE classifiers with the signal stability feature enabled start to correctly classify the suspicious device. The trackers remains on the user's person and maintains a stable signal for the remainder of the dataset, so these variants of the SCORE classifiers maintain a perfect F1 score for the remainder of the dataset.

00:05 - 00:07	The SCORE classifier with the proximity feature enabled and signal stability feature disabled starts to misclassify one non-suspicious device as suspicious due to the user moving in a small area and the misclassified device maintaining a strong signal strength with the user's device.
00:06 - 00:07	The AirGuard classier starts to correctly classify the suspicious device because the user hits AirGuard's distance traveled threshold between the 6 and 7-minute mark. The SCORE classifier with the proximity and signal stability features disabled starts to misclassify more non-suspicious devices as suspicious due to the user moving in a small area, causing those devices to have elevated risk factor values. It misclassifies over nine non-suspicious devices as suspicious for the remainder of the dataset.
00:08 - 00:16	The Smallest K-Cluster classifier temporarily misclassifies three to four non-suspicious device as suspicious due to the user moving in a small area, causing the classifier's F1 score to dramatically fall. It temporarily recovers at the 12 to 13-minute mark, but it continues to misclassify multiple non-suspicious device as suspicious while the user remains in the area.

00:10 - 00:11	The IQR / K-Means Hybrid classifier temporarily misclassifies one to two non-suspicious device as suspicious due to the user moving in a small area, causing the misclassified devices to have elevated risk factor values. It quickly recovers as the user moves away from those devices.
00:11	The BLE-Doubt classifier starts to correctly classify the suspicious device because the user has traveled long enough for devices to surpass its time threshold. It maintains a perfect F1 score for the remainder of the dataset.
00:14 - 00:15	The AirGuard classifier starts to misclassify four non-suspicious device as suspicious due to the user moving in a small area, causing the misclassified devices to have elevated risk factor values.
00:20	The AirGuard classifier starts to misclassify only two non-suspicious devices, down from the four non-suspicious device it misclassified previously. The user is still moving in a small area, but the classifier's F1 score improves because the user has moved away from two of the previously misclassified devices, causing those devices to not have been seen recently.

Dataset	AirGuard RSVC	BLE-Doubt	IQR / K-Means Hybrid	Smallest K-Cluster
A	0.800	0.667	0.667	0.800
B	0.857	0.857	0.418	0.400
C	0.571	0.750	0.712	0.717
D	1.000	1.000	0.913	1.000
E	1.000	1.000	1.000	0.909
F	1.000	1.000	1.000	0.992
G	0.333	1.000	1.000	1.000
H	0.909	0.909	0.705	0.909
I	1.000	1.000	0.520	1.000
J	0.706	1.000	0.582	0.847
K	0.571	0.154	0.276	0.286
L	0.667	0.000	0.663	0.667
M	0.154	0.000	0.000	0.048
N	1.000	1.000	0.984	1.000
Dataset	SCORE 00	SCORE 01	SCORE 10	SCORE 11
A	0.800	0.800	0.800	0.800
B	0.667	0.667	0.667	0.667
C	0.750	0.000	0.750	0.000
D	1.000	0.750	1.000	0.750
E	1.000	0.500	1.000	0.500
F	1.000	1.000	1.000	1.000
G	1.000	0.889	1.000	0.889
H	0.909	0.800	0.909	0.800
I	1.000	0.667	1.000	0.667
J	1.000	0.667	1.000	0.667
K	0.000	0.000	0.000	0.000
L	0.000	0.000	0.000	0.000
M	0.000	0.000	0.000	0.000
N	1.000	1.000	1.000	1.000

Table A.2: Comparison of classifier's F1 scores at the end of each dataset

B *BL(u)E CRAB* Dataset Sample


```
  {
"devices": {
  "Device A": {
    "$1": "Greg's Bluetooth Tracker",
    "$2": "Audio OS",
    "$3": [1660],
    "$4": {
      "2025-12-09T12:10:00.000": [-92],
      "2025-12-09T12:30:00.000": [-86]
      }
  },
  "Device B": {
    "$1": "Dylan's Smartphone",
    "$2": "Smartphone OS",
    "$3": [76],
    "$4": {
      "2025-12-09T12:10:00.000": [-98],
      "2025-12-09T12:30:00.000": [-80],
      "2025-12-09T12:50:00.000": [-76]
```

```
        }
      }
    },
    "locationHistory": {
      "2025-12-09T12:00:00.000": {"$1": 45.509035, "$2": -122.681059},
      "2025-12-09T12:20:00.000": {"$1": 45.511620, "$2": -122.683165},
      "2025-12-09T12:40:00.000": {"$1": 45.511598, "$2": -122.686134}
    }
}
```