Data Quality Based Intelligent Instrument Selection with Security Integration

SERGEI CHUPROV, Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, United States
RAMAN ZATSARENKO, Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, United States and University of Texas Rio Grande Valley, Edinburg, USA
LEON REZNIK, Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, United States
IGOR KHOKHLOV, Jack Welch College of Business & Technology, Sacred Heart University, Fairfield, United States

We propose a novel Data Quality with Security (DQS) integrated instrumentation selection approach that facilitates aggregation of multi-modal data from heterogeneous sources. As our major contribution, we develop a framework that incorporates multiple levels of integration in finding the best DQS-based instrument selection: data fusion from multi-modal sensors embedded into heterogeneous platforms, using multiple quality and security metrics and knowledge integration. Our design addresses the security aspect in the instrumentation design, which is commonly overlooked in real applications, by aggregating it with other metrics into an integral DQS calculus. We develop DQS calculus that formalizes the problem of finding the optimal DQS value. We then propose a Genetic Algorithm–based solution to find an optimal set of sensors in terms of the DQS they provide, while maintaining the level of platform security desirable by the user. We show that our proposed algorithm demonstrates optimal real-time performance in multi-platform instrument selection. To facilitate the framework application by the instrumentation designers and users, we develop and make available multiple Android applications.

CCS Concepts: • **Security and privacy**; • **Information systems** → **Data management systems**;

Additional Key Words and Phrases: Data quality, data security, genetic algorithms

ACM Reference Format:
Sergei Chuprov, Raman Zatsarenko, Leon Reznik, and Igor Khokhlov. 2024. Data Quality Based Intelligent Instrument Selection with Security Integration. *ACM J. Data Inform. Quality* 16, 3, Article 15 (October 2024), 24 pages. https://doi.org/10.1145/3695770

This work was partially supported by the U.S. National Science Foundation under Grant No. 2321652 and Grant No. 2415299.
Authors' Contact Information: Sergei Chuprov, Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, New York, United States; e-mail: sergei.chuprov@utrgv.edu; Raman Zatsarenko, Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, New York, United States and University of Texas Rio Grande Valley, Edinburg, Texas, USA; e-mail: rz4983@rit.edu; Leon Reznik, Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, New York, United States; e-mail: lrvcs@rit.edu; Igor Khokhlov, Jack Welch College of Business & Technology, Sacred Heart University, Fairfield, Connecticut, United States; e-mail: khokhlovi@sacredheart.edu.

ACM 1936-1955/2024/10-ART15
https://doi.org/10.1145/3695770

1 Introduction

Modern smartphones and other mobile devices, whose market is predicted to grow up by 7.3% per year in 2022–2029 [43], incorporate numerous instrumentation gadgets, which may generate various sensor-originated data available for community use in science and technology. Diverse crowd-sourcing applications [13, 31], mobile crowdsensing missions [12, 23, 24, 26, 44, 48], and the **Internet of Things (IoT)** have a huge potential to employ these devices to produce unprecedented volumes of data that could be shared and utilized in science and industry. Data collected from these devices can be fused in third-party applications, too. However, this resource remains hugely underused due to the lack of methodological and organizational support and the shortage of automation tools that should provide assistance in instruments selection and their system integration, on one side, and the trust deficit from the scientists and practitioners who might be reluctant to utilize data coming from unverified and uncalibrated sources with no information about their quality on the other.

Diverse instrument gadgets can provide data of various quality that might not satisfy the user's and application's demands. To design mobile instrumentation systems that meet their requirements, users aim at selecting sensors that produce the best quality data that might be hard to accomplish in the case of numerous sensors commonly embedded in miscellaneous platforms. Regular users usually do not possess an expert knowledge of the instruments to make their selection. The development of instrument selection automation tools will help to design more effective and efficient instrumentation that can supply users with the **Data Quality (DQ)** they require.

A common issue that is overlooked when instrument selection is considered is that the security of the platform that embeds sensors is often not evaluated properly and neglected when evaluating the DQ produced by instruments. To address this issue, we develop our framework that integrates DQ metrics with security metrics when evaluating the overall quality of the data supplied by a platform. In this article, we integrate sensor selection methods and tools into a unified **Data Quality with Security– (DQS)** based intelligent framework that allows for the automation of sensor selection and composing mobile instrumentation systems aimed at providing users with the required functionality and the DQS that satisfy their requirements. As instruments and sensors frequently operate in real time, the quality of data they produce may vary due to changing operational conditions. Our solution is aimed to work at both instrumentation design and operation stages, which allows dynamic restructuring of the system in near real time to adjust to the current conditions. This operation poses additional requirements on the optimization techniques that lead to the application of intelligent methods.

In our previous research [8, 19], we collected data, gathered knowledge, and implemented several methods and tools that we now integrate into our framework described in this article. In Reference [19], we developed DQS integration calculus, which incorporated accuracy, security, and other metrics to evaluate a smartphone sensor system. In Reference [8], we expanded the developed DQS calculus from smartphones to other mobile devices and included measurements of various modalities. In this article, we employ our previously developed knowledge base on sensor-embedded mobile device characteristics to develop real-time sensor selection methods and tools based on **Genetic Algorithms (GA)** techniques.

Generalizing from our previous research [8, 19], here we concentrate on the integration of the developed sensor selection methods and tools into a unified framework, which itself can be optimized in real time to satisfy user requirements toward DQS and the system's functionality. Our framework subsumes several integration levels that are described in Section 2. This article's novelty and major contributions include the following:

(1) the integration of the developed methods and tools into a unified instrumentation selection framework that can be employed in both sensor selection system's design and operation stages;
(2) multi-level measurements and the instrumentation integration procedures (see Figure 1), which incorporate the following:
   (a) DQS evaluation with both accuracy and security,
   (b) multi-modal data fusion,
   (c) multi-platform system realization,
   (d) and knowledge utilization;
(3) demonstrations of how the developed methods and tools can be realized and integrated to select sensors on various instrumentation platforms;
(4) expansion of previously developed calculus and integration into a unified framework. In our previous research [8, 19], the DQS calculus was developed specifically for mobile platforms. Here we present a novel theoretical framework, described by our generalized DQS calculus, which can be used on any instrumentation platform equipped with real-time sensing instruments;
(5) demonstration of practical applications of the developed theoretical framework. In Section 6, we provide a comprehensive overview of our practical contributions to the field. We developed multiple Android applications and an instrument-selection knowledge-base aimed at automating the novel DQS-informed instrument selection process on Android platforms.

Our research presented in this article follows the design science methodology. In our work, we design and present various artifacts, such as the DQS evaluation calculus, the metrics defined and used in this calculus, and, finally, the GA-based sensor selection algorithm, to facilitate the DQS-informed instrument selection process. We then demonstrate in practice that our approach is competent at finding a selection of instruments that provides the best possible DQS under constraints that exist in real-time sensing applications. The article proceeds as follows: Section 2 describes the problem relevance and our generic approach to tackle the problem of instrument and sensor selection. Sections 3 and 4 describe artifacts designed for this study. Following the common patterns for a design-based study [14], we evaluate our designed approach analytically and experimentally in Section 5. At last, we describe the practical contributions of our design in Section 6.

2 Integration Framework for Instrument Selection

2.1 Related Work on Sensor Selection and DQ Evaluation

Sensor selection is a well-known problem that is commonly formalized as an optimization task. Several different approaches exist in literature that propose different optimizations for sensor selection in dynamic systems. Debouk et al. [9] proposed to view the sensor selection task as a test-strategy optimization that is subject to minimizing the cost. Joshi et al. [17] proposed a convex optimization heuristic for selecting a subset of sensors out of a given set. Yao et al. [47] considered using GA for optimal sensor placement on large space structures. In a more recent study, Liu et al. [27] proposed a sensitivity-based optimization approach for sensor selection. In general, it has been shown previously that sensor selection is an NP-hard problem [5]. Nevertheless, GA present an effective heuristic to approximately solve the sensor selection problem for a given application. Works such as [16, 34, 36, 37] demonstrate applications of GA for sensor-related optimization tasks. However, the issues of quality of the data supplied by the sensors and the security of sensing platforms traditionally are not considered during sensor selection optimization. In this work we aim to bridge the existing gap by proposing metrics and calculus to evaluate and integrate DQS, which we then use to perform sensor selection based on a GA approach.

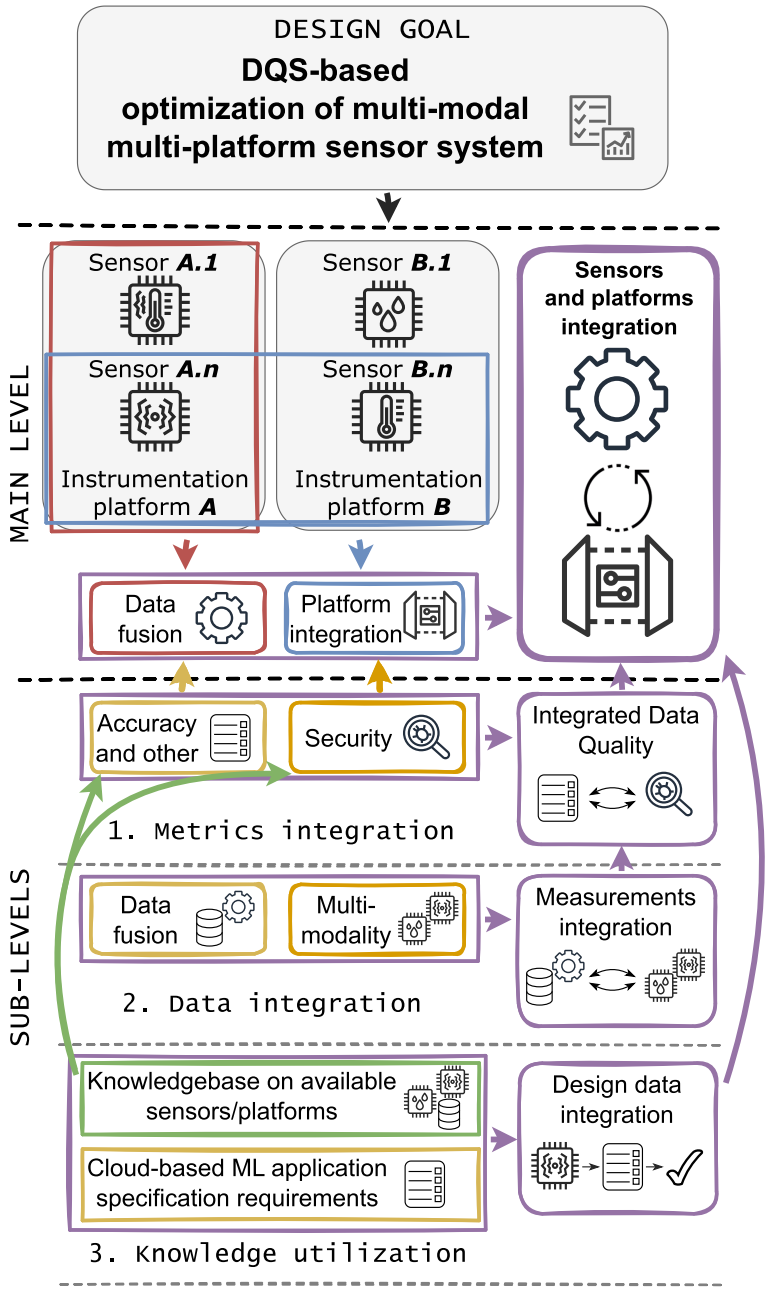

Fig. 1. Multi-level integration procedures in framework design and operation. Sub-level 1 represents metrics integration, where DQ metrics are integrated with security metrics. Here the initial DQS score is produced. Sub-level 2 represents data integration, where we fuse the data available across multiple sensor platforms and produce a fused DQS score for these data. Sub-level 3 stands for knowledge utilization, where we use the knowledge about the sensors or instrumentation platforms (for example, from our developed knowledge base in Reference [22]) along with user expert knowledge to manage the sensor selection.

Evaluating DQ is a research problem that attracts significant efforts, especially now in the era of IoT and Big Data. Previously, researchers have identified and classified DQ metrics into multiple categories [45]: (1) intrinsic DQ metrics that do not depend on the application, (2) contextual DQ metrics that rely on the application, (3) representational DQ metrics that describe representation formats, and (4) accessibility DQ metrics. Similarly, Bar-Noy et al. [3] also divide DQ metrics into intrinsic and contextual. In terms of our study, the metrics proposed in Section 4.2, except for the device security, can be considered intrinsic. The device (instrumentation platform) security in our case is both an intrinsic and a contextual metric. The context of the application domain partially dictates device security. The contextual nature of security is accounted for in our four-way evaluation of security, which considers app security, device features, sensor security, and cloud security.

2.2 Our Integral Framework

We develop a novel instrument (sensor) measurements quality and security integration framework that aims to optimize the instrument selection process through DQS multi-modal and multi-platform instrumentation composition. Our framework takes functionality and DQS specifications as the input and automates selecting sensors whose integration satisfies specific requirements set by the data consumers. It incorporates multi-metric DQS evaluation procedures, which integrate diverse metrics ranging from measurements accuracy to instrumentation security. Commonly, the aspect of security is paid insufficient attention in sensor systems' instrumentation design [11]. However, the security violation might lead to the DQ degradation collected from sensors, especially when measurements from various platforms are fused together and result in the total malfunctioning [8]. To address these issues, we consider the instrumentation platform security characteristics as a DQS component, which is incorporated into the DQS evaluation alongside the measurements accuracy and other quality metrics. In our framework, security characteristics depend not only on the sensor itself, but on the instrumentation platform into which this sensor is incorporated. This approach allows constructing a comprehensive security evaluation process. Hence, our framework can be adapted to each individual sensor instrumentation platform realization and allows evaluating its unique security characteristics.

Measurements from various sensors can be fused to increase their accuracy [32]. To facilitate sensor selection, users usually rely only on the accuracy of the produced data. Security metrics incorporation into the sensor selection may allow the users to enhance the overall DQ and make system operations more secure. The proposed unified sensor selection end-to-end framework integrates methods, techniques, and knowledge to supply sensor system users with the data that best satisfies their requirements.

In Figure 1, we schematically represent the framework structure with the operational levels that demonstrate how multi-layer integration is performed. Our framework goal is to facilitate the design optimization of multi-modal and multi-platform sensor systems. Below the major design goal, we present the main level of the proposed integration. This level demonstrates the major operations proposed by the framework design: merging sensor fusion and sensor platform integration to achieve the optimization goal. Under the main level, we depict three sub-levels that facilitate the operations performed at the main level. Sub-level 1 is metrics integration, in which such data characteristics as accuracy, which depend on the sensor itself, are integrated with security, which depends on the platform the sensor is embedded into. In this sub-level, the DQS characteristic is produced. Sub-level 2 is responsible for the data integration. In this sub-level, sensor measurements are integrated among multiple sensor modalities and platforms. The result of this stage are data obtained from various sensors and platforms. Then, metrics are aggregated to calculate the integral DQS for this data. Sub-level 3 is knowledge utilization, in which the available knowledge on sensors and instrumentation platform characteristics are employed to

manage the sensors/platform integration at the main level. This sub-level also incorporates user and application specifications that are employed to select sensors that meet these requirements.

3 Data Fusion Effect on Accuracy and Security

3.1 Security Metrics Integration into DQS Calculus

Fusing data from various sensors is a well-known approach employed by designers to improve measurement data accuracy and trustworthiness [18]. Multiple fusion techniques have been implemented in sensor systems. For example, fusing measurements from local sensors (e.g., camera, LiDAR, radar) with global sensors (e.g., global navigation satellite systems) [33] or between the inertial and vision sensors [10, 15] allowed to improve the quality and robustness of the obtained data in real time. Alongside the aim to improve data accuracy and trustworthiness, data security characteristics should be paid substantial attention, which was demonstrated by various attacks conducted against sensor systems. For instance, Cao et al. [7] presented a novel adversarial attack against multi-sensor fusion system, which allowed to affect both three-dimensional LiDAR point cloud and camera pixels representation. Despite the benefits achieved with the data fusion, data and sensor platform security ignorance may decrease its overall effectiveness and the quality of the resulting data.

3.1.1 Security Metrics Employed. In this article, we integrate instrumentation platform security with other sensor characteristics into a unified DQS indicator. We consider instrumentation security as an inseparable component of DQS calculus, which allows evaluating of how security conditions affect the overall quality of data. We develop calculus tools, which are described in more detail in Section 4.2 and demonstrate their utilization. We also provide a practical example of how these security metrics might be calculated, and demonstrate their extended definitions and descriptions in Section A.1. Those metrics are introduced in Section 4.3 on a practical example for the reader's convenience.

3.2 Accuracy and Security Evaluation Pipeline

Below we present the steps incorporated into the data accuracy and security evaluation procedure in our framework.

3.2.1 Device Measurement Data Quality Evaluation. In this step, the quality of all sensors embedded into an instrumentation platform is evaluated. To facilitate this evaluation in practice, we develop an Android application that we describe in detail in Section 6.2. We employ this application to collect sensor quality characteristics and include them in the knowledge base, described in Section 6.4.

3.2.2 Measurement Fusion Accuracy Evaluation. In this step, the resulting accuracy after fusing the data from the sensors embedded on a single platform is evaluated. We propose a way to fuse this data below; however, the fusion operation can be fine-tuned based on the data consumer's needs.

3.2.3 Instrumentation Platform Security Evaluation. In this step, the instrumentation security is evaluated. As in the previous step, the security evaluation calculus may vary and depend on the platform realization. To facilitate the platform security evaluation in our use case, we develop an Android application described in Section 6.1.

3.2.4 Multi-platform Fusion Security Evaluation. In this step, the data from multiple platforms are fused together, and the resulting multi-platform security score is calculated. This step can be performed optionally depending on the needs of the end user. In our use case, we demonstrate how the security characteristics of multiple platforms can be fused in practice in Section 5.3.

Table 1. Sensor Platform Accuracy and Security after Data Fusion

	Platform A+B	Platform A+C	Platform B+C
Combined Security	3.49	10	3.49
Accuracy	95%	95%	80%
Overall DQS Score	6	9.8	5.45

3.2.5 Sensor Selection. In this step, the combination of instruments that best satisfies the user's accuracy and security requirements is selected from the evaluated ones. For sensor selection, various techniques may be employed. We propose a novel GA-based approach, the realization of which is available on Android platforms and described in Section 6.3.

3.3 Security Role in Measurement Fusion

In fusion design, one has to consider security and privacy protection and evaluation from various perspectives. For example, some platforms might store or produce private or confidential data. The integration of these data with publicly available information may result in decreasing the confidentiality level to the lowest in the system. As one can observe from Table 1, the integration of "Platform A" and "Platform B" leads to a higher accuracy but lower overall score. The flaws in "Platform B" security result in the lower overall DQS score in case of A+B platforms integration. There might be security conflicts in case of various mandatory access control policies employment in the instrumentation system. For instance, if the Bell-LaPadula model is employed for the access control evaluation [6], then the integration of data with various confidentiality characteristics is unacceptable in terms of this model.

In case when the sensor measurements are integrated from the platforms with various clearance levels, the clearance level of all integrated platforms is decreased according to the lowest one. This leads to potential data leakage. One can see that relying only on measurements obtained from a platform with a higher clearance level will not decrease the overall one. However, this strategy will decrease the overall data accuracy and robustness and may not be compatible with integrity-enforcing access control models. One of the examples is Biba access control model [29], employment of which may lead to improper decisions in the case of multi-platform measurements fusion. For instance, fusing the data from platforms with various veracity leads to decisions made based on data with a lower veracity level. This multi-platform sensor fusion strategy is not acceptable by the integrity-enforcing access control policies.

As Table 1 demonstrates, while the measurements fusion from "Platform A" and "Platform B" allow improving an overall data accuracy, it results in lowering the overall data security level according to "Platform B." The same situation occurs in the case of platforms "B" and "C" measurements integration. Fusing the data obtained from "A" and "C" platforms not only allows us to achieve higher accuracy but also results in a higher overall DQS, as both "A" and "C" possess high data security level that complements a higher overall DQS score.

4 Intelligent Sensor Selection Use Case

GA represent a subclass of evolutionary algorithms, which have been widely employed to optimize search and selection processes in various applications, such as routing optimization in IoT [46] and network Quality of Service improvement [30]. In our framework, we employ GA to select sensors that best satisfy data consumers' requirements. In comparison to other search techniques, GA are known for their ability to handle high-dimensional problems even in real time—the feature that motivated us to select this technique in our implementation.

In References [20, 35], as the GA fitness function, we used the integral DQS indicator composed from accuracy and security metrics. However, those studies were highly limited in terms of sensor

devices population and their diversity. In this article, we address those limitations by significantly extending the metrics employed in DQS calculus and the employed sensor device population size. We collect knowledge on multi-modal sensors embedded into heterogeneous platforms and employ this knowledge to expand and diversify our population, from which the best instrument combination is selected. In addition to these extensions, we also remove the limitation on the number of devices included in a single generation. DQS calculus, developed in Reference [20], could not handle data of various modalities, so in this article we sufficiently extend it by supplementing it with the deducible hierarchical levels. These levels incorporate multiple metrics calculated for various sensors and instrumentation platform properties, which are integrated on a higher level to produce the overall DQS score, expressed by Equation (8). We implement the developed calculus in our Android application [40], which is described in more detail in Section 6.2. To verify the effectiveness of our GA-based instrument system optimization approach, we conduct an empirical study. We compare GA with the brute-force sensor selection in terms of elapsed wall-time and the achieved DQS score. In Section 5, we provide more detailed analysis of the results achieved.

4.1 Formalization of Instrument Selection Problem for Multi-Modal Data Fusion

In instrument selection, we evaluate the DQS of two major instrumentation components: sensor devices themselves and instrumentation platforms, into which these sensors are incorporated. In our use case, we integrate sensors embedded into various Android mobile devices. We use various types of sensors embedded into a single platform employed for data collection. Below we formalize the sensor selection and data fusion problem. *Given:*

- a set of N instrumentation platforms, which include the sensors P_i, $i \in \{1, \ldots, N\}$;
- a set of quality indicators PQ_{iq}, $q \in \{1, \ldots, M\}$, where M represents the number of quality indicators defined by each platform's technical characteristics;
- each platform is composed of K sensors S_{ij}, $j \in \{1, \ldots, K\}$;
- each sensor's quality indicator can also be defined with PS_{ijr}, $r \in \{1, \ldots, L\}$, where L is a number of quality indicators determined for the sensor.

Goal: to find such instrumentation system configuration $\overline{S}$ that will provide the required level of the overall DQS indicator.

The configuration includes the integration of (1) sensor devices and (2) instrumentation platforms, whose data are fused to achieve the required DQS. The data collected and fused by the instrumentation can be defined as $D = FZ(data_{ij})$, where $FZ(\cdot)$ is the fusion operation over the data collected from S_{ij} sensor combination. Then FZQ corresponds to the DQS integration operator, which aggregates various DQS metrics, such as sensor accuracy, instrumentation platform security, and others. FZQ operator depends on particular DQS metrics and may vary to adjust the data fusion function toward multi-modal data. The integration of the DQS indicators obtained from various platforms is performed by A operator to calculate the overall DQS indicator. Then, the overall DQS after the multi-modal and multi-platform data integration is defined as $DQS_o = A(FZQ(PQ_{iq}))$. We shall define the goal as the DQS maximizing $FZQ(PS_{ij}) \rightarrow max$ or achieving at least the required level β, established by the data consumer.

4.2 Data Quality and Security Evaluation Calculus

In this section, we present an example of the DQS calculus that facilitates DQS evaluation for the employed set of sensors described in Section 4.5. The characteristics of sensor devices and instrumentation platforms may vary, so the calculus is flexible enough and can be adjusted to the properties of sensors and platforms. We design our calculus based on our previous developments [20] intended for fusing the data of a single modality derived from an instrumentation platform.

We substantially extend those developments by incorporating the calculus operations under the multi-modal data, and the multi-platform integration functions. Below we present our calculus developed and adjusted toward the set of sensors and platforms employed in our case study.

4.2.1 Instrumentation Platform and Sensor Object. An instrumentation platform S is composed of a number of sensor devices $s_{i_j} \in S,\ j \in \{1, \ldots, m\},\ m \geq 1$, which are divided into groups $t_i \in T,\ i \in \{1, \ldots, n\},\ t \geq 1$, by the data type these sensors produce. In our GA implementation, described in Section 4.4, we refer to each instrumentation platform as the *sensor object*. The metrics for evaluating the DQS indicator of the employed sensors are formalized below.

4.2.2 Sensor Accuracy and Total Sensor Accuracy. The **Sensor Accuracy (*SA*)** metric refers to the precision and is calculated based on the sensor's resolution characteristics. Here resolution generally refers to the sensor's ability to capture detail or clarity in a received signal. Specific measurements of resolution depend on the sensor type and were previously studied in Reference [21]. In Appendix A.2, we provide some descriptive statistics, accumulated from our previous research, which can aid in this metric computation. **Total Sensor Accuracy (*TSA*)** is a metric calculated based on SA over all the sensors in a platform. SA and TSA can be formalized according to (1) and (2), respectively,

$$SA = 1 - \frac{resolution(s_{i_j})}{max(resolution)}, \tag{1}$$

$$TSA = \sqrt{\frac{\sum_{i=1}^{m} SA^2}{m}}. \tag{2}$$

4.2.3 Sensor Latency. This metric is calculated based on the average of *min* and *max* delays the sensor device demonstrates between its measurements. As delays of the sensor devices vary in practice, we normalize the delay values by dividing them by *max* for each sensor. In this case, the resulting delay value for each sensor ranges in the $\{0, 1\}$ interval. To ensure that higher delay results in lower **Sensor Latency (*SL*)** values, we define the latency value l, which is calculated according to Equation (3). The smallest l value across all the sensors in the combination is then used to define SL, which corresponds to Equation (4),

$$l_{i_j} = 1 - \frac{delay(s_{i_j})}{max(delay(t_i))}, \tag{3}$$

$$SL = min(l_{i_j}). \tag{4}$$

4.2.4 Instrumentation Power Consumption. This metric is based on the power that instruments are expected to consume over their operation in both measuring and idling conditions. The power consumption characteristics are taken from the sensor device specifications or other publicly available documentation. We assume that all the sensors work simultaneously while performing the measurements, so **Instrumentation Power Consumption (*IPC*)** can be formalized according to Equation (5),

$$IPC = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} power(s_{i_j})}{n \times m}. \tag{5}$$

4.2.5 Instrumentation Platform Security. This metric is based on the platform security characteristics. In our case, as we employ Android OS mobile devices, we use such security metrics as "screen lock activation status," "Android OS basic integrity test," and others. After calculating the metrics for each sensor device, the *min* value across all the sensors is used to initialize **Instrumentation Platform Security (*IPS*)**. The overall IPS metric for a particular instrumentation platform

can be calculated as Equation (6),

$$IPS_s = sl(s) + dm(s) + bit(s) + act(s) + \sqrt{(us_{max} - us(s)) \times |us(s)|} + (pha_{max} - pha(s)) \times |pha(s)|, \tag{6}$$

where IPS_s is an instrumentation platform s security evaluation, $sl(s)$ is a value based on the platform's screen lock parameter, $dm(s)$ is a value based on the platform's s developer menu parameter, $bit(s)$ is a value based on the platform's s basic integrity test parameter, $act(s)$ is a value based on the platform's s Android OS compatibility test parameter, $us(s)$ corresponds to the platform's s unknown source value, us_{max} is a maximum value over all the evaluated instrumentation platforms, $|us(s)|$ corresponds to the number of unknown applications installed on the instrumentation platform s, $pha(s)$ is a value for potentially harmful applications installed on the instrumentation platform s, pha_{max} corresponds to the pha's maximum value, and $|pha(s)|$ is a number of potentially harmful applications installed on the platform S. The overall IPS value can be calculated as Equation (7). We refer our reader to Appendix A.1 where we explain in detail the metrics acquisition and evaluation process. This process was also implemented in the application in Section 6.1,

$$IPS = min(IPS_s). \tag{7}$$

In this article, we present a specific example of the DQS calculus implementation for the employed set of sensors and instrumentation platforms. However, the instrumentation designer is able to adjust it. To achieve this, on the higher DQS integration level, we define the overall DQS fitness function that incorporates weights whose values can be modified and adjusted. An example of this overall DQS function defined for our established metrics is demonstrated in Equation (8),

$$DQS = \frac{w_1 TSA + w_2 SL + w_3 \frac{1}{IPC} + w_4 IPS}{\sum_{i=1}^{W} w_i}, \tag{8}$$

where w_1, w_2, w_3, and w_4 represent the weight coefficients and W is the number of weights incorporated in the DQS calculation. In our use case, we calculate the DQS value with equal weights, however, they might be adjusted according to the data consumers' priorities and needs.

4.3 DQS Calculus Use Case

Here we present a scenario of how the proposed DQS calculus can be employed to evaluate the accuracy of measurements and security of a particular platform.

The research company MedResML (not a real name) is conducting medical research that investigates the feasibility of movement system impairment syndrome diagnostics based on an individual's motion patterns analysis. To facilitate the research, the company collects a comprehensive set of measurement data that are processed to extract diagnosis patterns. Such sensors as a gyroscope, accelerometer, pedometer, and magnetometer are employed to collect measurements. As modern smartphones are equipped with these devices, MedResML decides to use personal smartphones as measurement collection instruments. This approach is cost-efficient, as there is no need to distribute other devices to the research subjects, and convenient, as the majority of research subjects keep their smartphones with them during their walking activities. To facilitate data collection, MedResML develops an Android OS application.

The agency decides to fuse the data obtained from various users and multiple instruments to improve its quality. Users' smartphones are equipped with diverse embedded devices of varying characteristics, which can influence the overall DQS. A higher accuracy score can be achieved if security is neglected. However, the overall DQS of the collected data is affected by a low-security level. Conventional methods of multi-modal and multi-platform data fusion rely only on accuracy characteristics while ignoring the security of the platform from which the data are acquired. These

Table 2. Instrumentation Platforms Accuracy and Security Evaluation Example

		Platform A	Platform B	Platform C	Platform D
App security	blacklisted apps	0%	20%	0%	0%
	potentiallyDangerous	0%	10%	0%	0%
	unknown sources	0%	50%	0%	20%
	app permission	1%	60%	1%	1%
		10.00	**4.78**	**10.00**	**6.80**
Device feature	OS version	26 [API 26]	24 [API 24]	26 [API 26]	26 [API 26]
	security patches	2 [1-Jun-18]	8 [1-Dec-17]	2 [1-Jun-18]	2 [1-Jun-18]
	device model	5.00	9.00	5.00	5.00
		10.00	**5.00**	**10.00**	**5.43**
Sensor security	bootLoader	locked	unlocked	locked	unlocked
	rootAccess	disabled	enabled	disabled	disabled
	developer's menu	disabled	enabled	disabled	enabled
	device lock	locked	unlocked	locked	locked
		10.00	**0.00**	**10.00**	**0.00**
Cloud Security	historic trend	1 [increasing]	(-)0.5 [decreasing]	1 [increasing]	0.1 [increasing]
	same device comparison	0.95 [top 5%]	0.2 [bottom 20%]	0.95 [top 5%]	0.75[top 25%]
		9.81	**2.74**	**9.81**	**5.16**
Device security		**10.00**	**3.49**	**10.00**	**5.44**
Sensor Accuracy	accelerometer	90%	5%	40%	70%
	gyroscope	90%	10%	10%	40%
	proximity sensor	90%	12%	60%	50%
Total Sensor Accuracy		**90%**	**9.47%**	**42.03%**	**61.64%**
Overall DQS Score		**9.76**	**0.39**	**5.36**	**5.21**

Accuracy values are scaled and represented as a percentage for convenience.

accuracy-oriented approaches might lead to prioritizing low-security platforms for data fusion that may jeopardize the research subjects' privacy and the research results trustworthiness.

To address this issue, MedResML decides to integrate instrumentation platform security evaluation into their data collection application. From Table 2, one can observe the results of four Android smartphones security evaluations, which are integrated with the accuracy produced by the sensor devices embedded in these smartphones to obtain the "Overall DQS Score," which was computed based on the calculus described previously. Appendix A provides a more detailed explanation on which calculus was used to compute the values represented in Table 2. Here we represent a high level explanation how device security is evaluated. In general, the devices (instrumentation platforms) have the following characteristics:

— Application security—presence of any suspicious applications, application permissions, sources of applications installed.
— Device feature security—what OS software is running, were security patched installed.
— Sensor security—is the bootloader locked, root access disabled, developer tools disabled, a pin-lock enabled.
— Cloud security—what is the historic security trend for this device; how does this device compare to similar devices.

Several metrics are used to numerically describe the above security characteristics of a particular device with concrete definitions detailed in Appendices A and A.1. Those metrics are then fed to an expert system that finally performs the security evaluation in relation to application security, device feature security, sensor security, and cloud security.

4.4 Genetic Algorithms for Sensor Selection Description

GA are a family of derivative-free heuristics that are empirically good at finding an optimal solution under certain constraints. The motivation for employing GA in the sensor selection scenario is based on the fact that the quality of data provided by each particular set of sensors can be defined in terms of discrete characteristics of the sensors, as shown in Section 4.2 and, as we demonstrate later in Section 5, GA are capable of finding an optimal solution over a discrete dataset under given time constraints, which is an important factor in real-time sensor selection applications. Researchers and developers already tried to employ GA in sensor networks optimization [4, 16, 34, 36, 37]. However, these publications mainly focus on reducing sensor energy consumption by improving a sensor network topology. In this research, we employ a GA to facilitate data consumers' decisions on what data source to choose for the aggregation to achieve optimal DQS. With GA, the decision time could be adjusted to the external conditions changes, for example, to the time of the attack against sensor networks. Further, we present and discuss in more detail how we implement them in our article.

GA is a type of optimization algorithm that finds either minimum or maximum value for a fitness function. The value of the fitness function is called a *fitness* value. As defined previously, we can formulate the goal for our GA as $FZQ(DQS) \rightarrow max$, where the DQS for an individual platform is defined by Equation (8), and FZQ corresponds to the DQS integration operator. Below, we provide a clear definition of the classic terminology used in GA in the context of our sensor selection problem.

- *Population* is a group or list of solutions, each of which can solve the problem at hand. This would be represented by a list of all *sensor objects*.
- *Chromosome* is a single value in the population, i.e., a single solution to the problem. In our case, it is a combination of sensors represented by a *sensor object*, as previously described in Section 4.2.1.
- *Gene* is a single element in the solution/chromosome, i.e., a single sensor.
- *Fitness function* is a measure of the optimality of a solution (*sensor object*). The formula to evaluate the fitness of a particular solution is mathematically presented by Equation (8).

First, the algorithm encodes information of sensors within a *sensor object*. Then it randomly generates a list of *sensor objects*. The number of sensors within each *sensor object* is defined by the *chromosome length* parameter. The number of *sensor objects* in the population is defined by the *population limit* parameter. To *initialize the population*, we need an initial list of possible solutions that can be improved in the next steps. Hence, we randomly generate the initial population with a size equal to the *population limit* that is a hyper parameter. This parameter was set as a rounded value of 1/10th of all available platforms with embedded sensors. To create a population, we go through every platform and randomly decide whether to select it or not. After selecting the platforms, we go through sensors associated with these platforms and again randomly decide whether to select them. In this way we form one *sensor object* and continue in this manner until we have as many *sensor objects* as the population limit value. Once the initial population is produced, the evolution process starts as follows:

(1) The fitness value, as defined by Equation (8), for each *sensor object* in the population is calculated;
(2) The population of *sensor objects* is then sorted in descending order of their fitness values, such that the sensor with the best fitness value appears first;
(3) Based on the parameter *retention limit*, a percentage of *sensor objects* are selected, and a list of the best sensor objects in this population (*rank-based selection*) is generated. To avoid sorting altogether, the *roulette-based selection* is used, wherein a selection probability based

Table 3. Employed Sensor Types and Their Characteristics

Sensor Type	Characteristics
Accelerometer (A)	Sensitivity, Non-linearity, Noise Density
Gyroscope (G)	Sensitivity, Noise Density, Cross-axis Sensitivity, Non-linearity
Proximity (P)	Resolution, Range, Absolute Response

on the relative fitness of the sensor object is assigned. A sensor object with a higher fitness value has a higher chance of being retained;

(4) A list of *sensor objects* that are ready for mutation and crossover is generated;
(5) To ensure that the sensor selection process is not stuck in a local maximum, based on the parameter mutation probability, each of the *sensor objects* is altered, wherein one of the randomly chosen sensors within the *sensor objects* is replaced by another sensor from the list of all available sensors. Then, two *sensor objects* are randomly chosen for the crossover operation, where a new child *sensor object* is generated by combining half of the sensors from each of the selected parent *sensor objects*. As such, during the crossover operation, the sensors (*genes*), along with their characteristics, that belong to a particular *sensor object* are being crossed over.
(6) A newly generated *sensor object* is added to the new population. This crossover process repeats until the population limit is reached;
(7) The average fitness value of the new population is evaluated;
(8) Once there is no sufficient change in the average fitness value, the selection process stops and returns the sensors contained within the best population. These sensors are expected to have high DQS if data from these sensors are fused.

4.5 Sensor Devices and Platform Characteristics with Their Data Quality Evaluation Knowledge Base

To verify our framework and test it with real-life data, we employ our knowledge base, described in References [22, 25] and in Section 6.4, which contains characteristics of thousands of diverse sensor devices. We evaluate our intelligent sensor selection on the three most popular sensor types (accelerometer, gyroscope, and proximity sensor) incorporated in the devices from our collection. Our knowledge base includes information on 52 accelerometers manufacturer brands, 20 proximity sensors brands, and 14 gyroscope brands. In Table 3 one can see the characteristics presented in our knowledge base for these sensor types.

5 Framework Evaluation Results

5.1 Brute-force Algorithm Analysis

To provide a baseline for the evaluation of our proposed GA, we developed a brute-force algorithm that exhaustively selects devices with the best DQS score while generating all possible combinations of sensors for each type. Let us consider a list of available sensors n as well as a number of sensor types t for a particular device. We can represent a selection of sensors for a particular sensor type as a binary string. As an example, let us consider $n = 3$ and $t = 1$, which means that we have three sensors (s_1, s_2, and s_3) of a single type to select from. We can represent each possible selection as a binary string, e.g., 111 for a selection of $[s_1, s_2, s_3]$, or 101 for a selection of $[s_1, s_3]$. For a single device or instrumentation platform, the number of possible selections to generate can be evaluated as 2^{nt}. Given d platforms, the number of possible platforms to select from can also be represented as a binary string equal to at most 2^d when all platforms are selected. As a result, a brute-force algorithm would have to go through all the possible sensor selections of each type of

sensors for each platform, with the number of possible selections equal to $2^{nt} \cdot 2^{d}$, which results in the overall time complexity of the algorithm being exponential: $O(2^{nt+d})$.

5.2 Genetic Algorithm Analysis

The time complexity of a generic GA can be defined in terms of the population size N, number of generations G, fitness evaluation time $T_{fitness}$, and the complexity of *selection, crossover,* and *mutation*,

$$O(G \cdot (N \cdot T_{fitness} + N \cdot O(Selection) + N \cdot O(Crossover) + N \cdot O(Mutation))). \tag{9}$$

In our implementation the crossover operation takes $O(1)$ time as we are simply recombining the data encoded into the parent *sensor objects*. The mutation operation is linear in terms of time complexity, $O(d)$, where d is the number of all available devices (i.e., *sensor objects* in step (5) in Section 4.4). The evaluation time of a fitness of a particular solution $T_{fitness}$ takes $O(1)$ time as it is a numerical computation. In Section 4.4, we describe two selection strategies: *rank-based* and *roulette-based*. If the *rank-based* selection is used, then Equation (9) is dominated by $O(Selection)$ and the overall complexity of the algorithm largely depends on the sorting method used. Assuming a sorting method similar in time complexity to merge sort is used for selection, and also assuming the population size of $N = d$, where d is the number of platforms, Equation (9) can be simplified to Equation (10),

$$O(Gd \log d). \tag{10}$$

If the *roulette-based* selection method is used, then the selection step takes $O(d)$ time and Equation (9) can be transformed into Equation (11),

$$O(Gd). \tag{11}$$

In practice, we run the *roulette-based* algorithm until the desired level of DQS is reached or a user-imposed time limit is exceeded instead of running the algorithm for G generations.

5.3 Practical Evaluation

In Reference [22], we developed a knowledge base of sensors incorporated in various IoT devices. In our practical evaluation of the proposed sensor selection algorithm, we employ the data from the developed knowledge base. Our knowledge base has data on 19 sensor types with information about 9,443 sensor incorporating devices. Here we concentrate on the most widespread types of sensors—accelerometer, gyroscope, and proximity sensors. For the purposes of our practical evaluation and with the sensor types narrowed down to three, we possess data about *2,886 sensors* embedded into *107 devices*, which were employed for the evaluation of our GA. We provide descriptive statistics regarding the characteristics of each particular type of sensor used in this evaluation in Appendix A.2.

We evaluate our intelligent instrument selection framework by measuring the achieved overall DQS and its computational performance on actual sensor characteristics' data from our knowledge base [22, 25]. As the instrument selection should be able to work in real time, to measure computational performance we rely on the wall-time elapsed between the moment of starting the selection process and the moment of obtaining the instrument combination with the highest DQS value. To facilitate our framework application in practice, we implement the developed GA-based instrument selection procedures as a software application and compare its performance against the brute-force-based selection.

In Figure 2, we present instrument selection techniques comparison results. Figure 2(a) compares the results of the achieved overall *DQS* value of the instrument combination selected by the

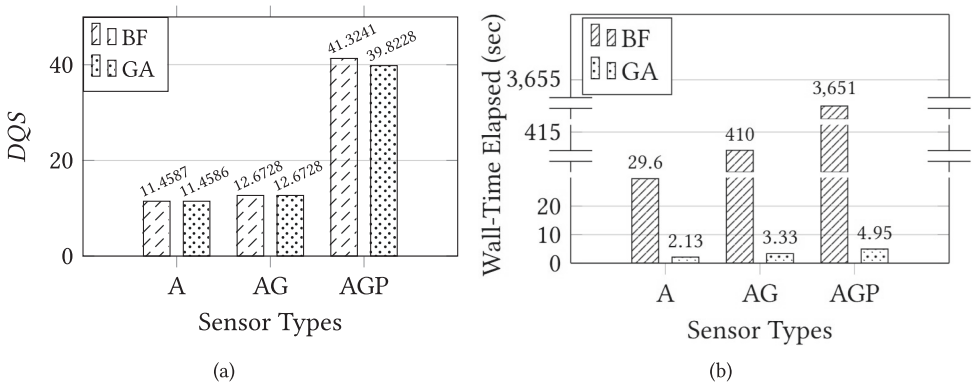

Fig. 2. Evaluation of GA vs. brute-force search: (a) in terms of DQS; (b) in terms of computational performance; BF refers to brute-force; GA to Genetic Algorithms, A means accelerometer, AG means accelerometer and gyroscope (search across two sensor types), and AGP means accelerometer, gyroscope, and proximity sensor (search across three sensor types). In (b) we represent the case with the maximum wall-time taken by the brute force to return the result for each sensor type over all experiments.

GA-based tool against the brute-force procedures. In this case, we do not stop the tools performing the search and give them enough time to converge. The GA-based tool demonstrates results commensurable to the brute-force technique. However, the brute force performs slightly better as it directly iterates through each sensor combination and finally outputs the best result. More interesting is to verify how fast the GA-based tool performs against the brute-force one.

In Figure 2(b), we compare the elapsed wall-time taken by each technique. The wall-time values are rounded up for the representation convenience. In this case, we decided to limit the time for the solution output, as security in IoT applications is a dynamic characteristic that changes very rapidly, which was previously demonstrated in some IoT malware spread studies [2, 28]. As shown in Reference [28], malware can spread at a rate of 0.75 devices per second or even faster. Taking this into account, in most experiments we pick 500 seconds as a time limit in our practical evaluation, which is a realistic limit given the rates of contemporary malware infections spread. As can be seen from the comparison, the GA-based technique requires sufficiently less time to output the result with a nearly similar performance for all sensor types. Notably, the GA-based approach finishes execution well within the 500 seconds time limit in every case. At the same time, the brute-force approach comes very close to the threshold in case of sensor selection from two categories (AG, A: accelerometer, G: gyroscope). Moreover, in the case of selection from three categories (AGP, P: proximity sensor) the brute-force algorithm runs well over the imposed limit of 500 seconds, taking a total of 3,651 seconds to finish executing.

As Figure 2 demonstrates, there exists a tradeoff between finding the best possible DQS score and executing the search in a reasonable time. The benefit of our GA-based tool is that, while the DQS score obtained by it might be marginally lower, the execution time is substantially shorter when compared to the brute-force search. We note that in applications where the pure DQS score is of utmost importance and the execution time is not important, our GA-based tool might be irrelevant as it does not always achieve the best overall performance given unlimited execution time. However, in situations where both the DQS values and the execution time are important, our heuristic approach stands superior.

In Table 4, we present a different example of the calculated metrics and the overall DQS value for three sensor types combinations. As one can see from this example, fusing multi-modal data from

Table 4. Results for Sensors DQS Evaluation

Sensor Type	*TSA*	*SL*	*SPC*	*SPS*	*DQS*
Accelerometer (A)	84.918	0.9831	9.304	3	22.2521
Accelerometer (A) , Gyroscope (G)	82.9613	0.9831	52.1095	4	21.9909
Accelerometer (A) , Gyroscope (G), Proximity (P)	85.9504	0.9830	47.9222	4	22.7385

Table 5. Instrument Selection Framework Integration Aspects, Methods, and Tools

Multi-modal and multi-platform data fusion	**Our contributions**	**Our prototypes and products**
Metrics integration	Data Quality and Security Evaluation calculus	Instrument Quality Assessment Android OS application; Instrument Platform Security Evaluation Android OS application
Data fusion and multi-platform integration	GA-based instrument selection technique	Instrument Selector Android OS application
Knowledge utilization	Autonomous instrument selection tools implementation on the collected database	Collected knowledge base

various platforms may influence the overall DQS value. Low deviations in the DQS value across various sensor combinations relate to equal weights we employ in Equation (8) for our calculations. However, more sensitivity can be added into the DQS calculus by adjusting the coefficients in Equation (8).

6 Prototypes Implementation

To facilitate the usability of our design in practice, we develop software tools integrated into multiple Android applications, publicly available in the Google Play store. In addition, we make available our knowledge base on measurement devices and instrumentation platform characteristics, which can be employed by our applications as well as beneficially utilized by the community for further research. In Table 5, we summarize various integration aspects, methods and tools, and describe how our developed methods and tools facilitate the instrument selection and how they were realized in practice. Below, we list and provide an overview of the developed and publicly available products and prototypes mentioned in Table 5 in more detail.

6.1 Instrumentation Platform Security Evaluation Tool

The developed application is a tool for assessing the security level of instrumentation platforms, such as smartphones and tablets, that are used for collecting and processing data. The application computes a security score for each device based on a set of criteria that reflects the potential risks and vulnerabilities of the platform. The security score is calculated by integrating the following platform parameters with the quantitative definitions detailed in Appendix A.1:

— *Screen lock activation status*: This parameter indicates whether the device has a screen lock mechanism enabled, such as a password, a PIN, a pattern, a fingerprint, or face recognition. A screen lock can prevent unauthorized access to the device and its data in case of loss or theft.
— *OS version*: This parameter reflects the operating system version of the device, which determines the availability of security patches and updates. A newer OS version usually means a more secure device, as it may fix some known bugs and exploits. In addition, it has less *known vulnerabilities*.
— *Applications installed from unverified app sources*: This parameter shows whether the device has any applications that are not downloaded from the official app store, such as Google Play. Unverified app sources may contain malicious or compromised applications that can harm the device or its data.
— *Presence of the potentially harmful software*: This parameter detects whether the device has any software that is classified as potentially harmful by Google's SafetyNet service [38]. Potentially harmful software includes malware, spyware, ransomware, phishing apps, and other types of unwanted or harmful applications.
— *Enabled developer's menu*: This parameter checks whether the device has the developer's menu enabled, which is a hidden menu that provides access to some advanced settings and features. Enabling the developer's menu may expose the device to some security risks, such as allowing USB debugging, installing apps from unknown sources, or modifying system settings.
— *Android OS basic integrity test*: This parameter verifies whether the device meets a basic level of integrity, which means that it has not been tampered with or modified in a way that compromises its security. The basic integrity test is performed by Google's SafetyNet service [1, 38] and includes various parameters of the device, such as the bootloader state (locked or unlocked) and root access. A locked bootloader prevents unauthorized modifications to the device's firmware, while root access grants full control over the device's system and data.
— *Android OS compatibility test*: This parameter confirms whether the device is compatible with the Android OS, which means that it conforms to the Android compatibility definition document. The compatibility test is also performed by Google's SafetyNet service [1, 38] and has stricter rules than the basic integrity test. A compatible device ensures that the device's software and hardware work properly with the Android OS and its applications. SafetyNet is a part of the Android OS and does not require downloading additional libraries.

The prototype of the application is publicly available and can be downloaded from Google Play [41]. The application can be used by researchers, developers, and users who want to evaluate the security of their instrumentation platforms and compare them with other devices. The application can also provide suggestions and recommendations for improving the security score of the device.

6.2 Instrument Quality Assessment Tool

This tool evaluates various instruments embedded in mobile platforms like tablets, wearable devices, and smartphones. The quality evaluation tool checks which instruments are embedded in the given platform, gathers further information about the instrument's quality from our supporting database [39], and then rates each instrument as "good," "bad," or "average." In addition, this application may be used to educate users about instruments available in their mobile devices, their possible use, and their limitations. The developed tool is publicly available in Google Play [42].

6.3 Instrument Selector Tool

We release this tool as an Android OS application, which allows selecting instruments and the mobile platforms they are embedded in to satisfy the given DQS specifications. The application employs the developed GA-based instrument selection techniques to find the best combination of instruments for data fusion. To calculate the overall DQS value, we integrate data accuracy, platform security, and other metrics. To improve the application's usability and its adaptation to various user requirements, we allow users to manually select and exclude metrics to be employed in the instrument's selection. The application of the GA-based technique allows producing results in real time. Due to user's security and privacy considerations, the public version of the application employs pre-uploaded instrument devices and platforms characteristics dataset [39] produced in References [22, 25]. However, the data for instrument selection can be provided by the user based on their mobile device instrument accuracy and security characteristics. The application is publicly available in Google Play [40].

6.4 Knowledge Base on Sensor Devices and Platforms Quality Characteristics

In References [22, 25], we described how we collected knowledge on the characteristics of thousands of mobile instrument-incorporating devices. We obtained their technical characteristics alongside with evaluating their DQS metrics. The database includes various instrument-incorporating devices' technical characteristics, such as measurements type, dimensions, resolution, camera, instrument performance, hardware characteristics, and so on. At present, our database contains the characteristics of 9,443 instrument-embedded platforms, such as mobile devices, tablets, wearable devices, and so on. The knowledge base contains 58 various attributes for the omnipresent instrument-incorporating devices manufactured by over 114 brands. There are 19 types of sensor devices presented in the database currently [39], such as a barometer, pedometer, gyroscope, accelerometer, and others. In our knowledge base, the data representation has the following structure.

6.4.1 Generic Information on the Instrument-incorporating Devices. contains data on the manufactured devices' technical characteristics. The device list includes major well-known manufacturers that allow their devices to run Android OS, such as Samsung, OnePlus, Xiaomi, Motorola, and so on. Technical characteristics might include device's form-factor, dimensions, camera characteristics, and so on.

6.4.2 Information on Instrument Characteristics. This represents data on the available instruments and their association with the device they are incorporated in. Depending on the instrument type, the characteristics might include delay (minimal and maximal), resolution, range, power consumption, and others.

6.4.3 Information on Instrument Platform Security. This represents data on instrument-incorporating device security, including the screen lock activation status, Android OS basic integrity test results, number of potentially harmful applications, the status of installing the applications from unknown sources, and others.

7 Conclusion

In this article, we developed and presented the unified intelligent DQS-based instrumentation selection framework, which integrates methods and tools to automatically select instruments that best satisfy the user's specifications. The framework supports integration on multiple levels: from fusing multi-modality data to incorporating diverse metrics for DQS evaluation. The developed solutions allow instrumentation designers and users to integrate multi-modal data obtained from

heterogeneous platforms to achieve the required DQS level. The integral DQS indicator application as an optimization goal enhanced the conventional sensor-originated DQS evaluation by incorporating security metrics into it. This, in contrast to using accuracy oriented metrics only, allowed to consider security characteristics too in configuring the best instrumentation. The empirical evaluation of the developed integrated DQS calculus and GA-based intelligent selection tools demonstrated the advantage of the proposed solutions against conventional techniques in instrumentation design. Our software and data prototypes and products are available for community use. Their application will facilitate the instrumentation design automation.

A Appendix

The sensor data evaluation alone does not provide sufficient information to identify data integrity violations. Thus, the company decides to include a device security evaluation in conventional DQ estimation procedures. In this case, the application that gathers sensor data also acquires information from the framework's module, which is responsible for the device security evaluation.

A smartphone may provide the raw data from sensors such as an accelerometer, a gyroscope, a GPS, a magnetometer, a gravity sensor, an illumination sensor, a proximity sensor, and so on. Each "sensor" entity has technical characteristics of the sensor hardware that participate in DQ evaluation. In this use case, two types of calculus implementation were used: based on expert system and based on a neural network that approximates the expert system input-output surface.

We developed an expert system module for a sensor correctness evaluation that takes as inputs sensor accuracy, noise, consistency, and time resolution. Scaling these parameters between "0" and "100" allows us to use developed calculus for various types of sensors.

Here we focus on platforms "A" and "C", considered in Table 2. As stated in Table 2, both platforms "A" and "C" have excellent device security; both devices have the following characteristics:

— Application security: no suspicious applications, no unknown source applications, and relatively strict permission controls.
— Device feature security: Android OS Oreo (API 26), security patch installed two months ago, and running on Samsung Galaxy S9, which was released on June 1st, 2018, about five months ago (on the moment of table generation).
— Sensor security: bootloader is locked, root access is disabled, developer menu is disabled, and has a pin code locking mechanism.
— Cloud security: the device evaluation history has an upward trend.

Naturally, both platform "A" and platform "C" achieve a high score of 10 on device security evaluation. We now have a secure environment that can foster the production of high-quality data. Since both platforms "A" and "C" have highly secure devices, does that mean the data quality is high for both devices? Looking back at the lower portion of Table 2, a clear distinction in the accuracy of the sensors between the two platforms stands out.

Platform "A" has a relatively good accuracy of 90%. After we combined platform "A" device security evaluation, "10," and the accuracy, "90%," we calculated the overall DQS score of "9.76/10."

However, platform "C" device sensors performed poorly with a bad accuracy of 42.03%. Considering device security alone may lead the researchers to overlook important details that could tamper with the data quality. The integration of both security and accuracy scores results in medium overall DQS score of "5.36/10."

A.1 Initial Security Metrics Acquisition

We have developed an Android OS library and a specialized application based on this library that retrieves security and privacy-related parameters, which are used as inputs to the expert system.

Table 6. System's Parameters That Are Gathered for the Initial Security Evaluation

Metric	Symbol	Values
Screen lock	M_{SL}	1 - Pattern, PIN or password; 0 - otherwise
Android OS version	M_V	2 - The latest version, 1 - previous version; 0 - otherwise
Unknown sources	M_{US}	1 - Unknown sources disabled; 0 - otherwise
Potentially harmful applications	M_{PH}	0 - Installed at least one potentially harmful application; 1 - otherwise
Developer's menu	M_{DO}	1 - Developer option menu disabled; 0 - otherwise
Basic integrity test	M_{BI}	1 - System passed basic integrity test; 0 - otherwise
Android compatibility test	M_{CT}	1 - System passed Android compatibility test; 0 - otherwise

The developed library provides an API that can be used by other researchers and software developers and is available at the Google Play Store. The developed software provides recommendations on how to improve the smartphone security level and gives an explanation of each parameter to a user. In addition, it allows opening a "simulation" screen where users can experiment with these parameters and see how they influence overall security. Table 6 provides the library's API.

Table 7 presents equations of metrics presented in Table 2.

Table 7. Data Security Metric Formulas

Metric and its description	Formula
OS version score	$VerScore = \begin{cases} 0 \quad \text{if } CurVer > VerThreshold \\ MaxVerScore - \\ \quad - MaxVer - CurVer \\ \quad \text{if } CurVer \leq VerThreshold \end{cases}$
Security patch score	$PatchScore = \begin{cases} 0 \quad \text{if } CurVer > VerThreshold \\ MaxPatchScore - \\ \quad - MaxVer - CurVer \\ \quad \text{if } CurVer \leq VerThreshold \end{cases}$
Device model score	$ModelScore = \begin{cases} 0 \quad \text{if } CurVer > VerThreshold \\ MaxVerScore - \\ \quad - MaxVer - CurVer \\ \quad \text{if } CurVer \leq VerThreshold \end{cases}$
Overall firmware score	$FirmwareSecurity = VerScore + \\ + PatchScore + ModelScore$

Continued on next page

Table 7 – *Continued from previous page*

Metric and its description	Formula
App unknown sources score. NumOfUnkn—number of all devices in an organization that allows unknown application sources	$UnknSrcScore = \begin{cases} 0 & \text{if UnknSrc is ON} \\ 1 - P(UnknSrc) & \\ & \text{if UnknSrc is OFF} \end{cases}$ $P(UnknSrc) = \frac{NumOfUnkn}{NumOfAllUsers}$
Black listed app score	$BlkLstScore = \begin{cases} 0 & \text{if Number} > \text{Threshold} \\ MaxScore- & \\ \quad -(Num - Threshold) & \\ & \text{if Number} \leq \text{Threshold} \end{cases}$
Dangerous permission utilization score	$PDanScore = \begin{cases} 0 & \text{if Number} > \text{Threshold} \\ MaxScore - (Number- & \\ \quad - Threshold) & \\ & \text{if Number} \leq \text{Threshold} \end{cases}$
Application security score	$AppSecScore = UnknSrcScore \hat{L} BlkLstScore \hat{L} \hat{L} PDanScore \hat{L} PermScore$

$\hat{L}$ is metrics integration through an expert system.

A.2 Descriptive Statistics of Sensor Characteristics

Here we provide some descriptive statistics of the sensors and their features used in the practical evaluation of our GA. Table 8 provides descriptive statistics for the accelerometer sensor type based on the data accumulated in our developed knowledge base of sensors, while Tables 9 and 10 provide descriptive statistics for the gyroscope and proximity sensor type, respectively.

Table 8. Descriptive Stats for Accelerometer Sensor Type

Stat	Sensitivity	Non-linearity	Noise Density
Min	16	0.10	75
Max	17039	2.00	800
Mean	6033.91	0.61	308.08
SD	7434.01	0.39	183.23

Table 9. Descriptive Stats for Gyroscope Sensor Type

Stat	Sensitivity	Noise Density	Cross-axis sensitivity	Non linearity
Min	33.8	0.0038	1.0	0.10
Max	131.2	0.030	2.0	0.20
Mean	114.55	0.0117	1.66	0.142
SD	24.62	0.008345	0.32025	0.036

Table 10. Descriptive Stats for Proximity Sensor Type

Stat	Resolution	Range	Absolute Response
Min	8.00	50.00	100.00
Max	20.00	100.00	165.00
Mean	12.91	93.75	131.42
SD	3.43	11.023	17.54

References

[1] Protect against security threats with SafetyNet. Retrieved March 5, 2024 from https://developer.android.com/training/safetynet

[2] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. 2017. Understanding the mirai botnet. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security'17)*, 1093–1110. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[3] Amotz Bar-Noy, Greg Cirincione, Ramesh Govindan, S. Krishnamurthy, T. F. LaPorta, Prasant Mohapatra, M. Neely, and Aylin Yener. 2011. Quality-of-information aware networking for tactical military networks. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops'11)*. IEEE, 2–7.

[4] Ataul Bari, Shamsul Wazed, Arunita Jaekel, and Subir Bandyopadhyay. 2009. A genetic algorithm based approach for energy efficient routing in two-tiered sensor networks. *Ad Hoc Netw.* 7, 4 (2009), 665–676.

[5] Fang Bian, David Kempe, and Ramesh Govindan. 2006. Utility based sensor selection. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*. 11–18.

[6] Ebru Celikel Cankaya. 2011. *Bell-LaPadula Confidentiality Model*. Springer US, Boston, MA, 71–74. https://doi.org/10.1007/978-1-4419-5906-5_773

[7] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. 2021. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *Proceedings of the IEEE Symposium on Security and Privacy (SP'21)*. IEEE, 176–194.

[8] Sergei Chuprov, Leon Reznik, Igor Khokhlov, and Karan Manghi. 2022. Multi-modal sensor selection with genetic algorithms. In *Proceedings of the IEEE Conference on Sensors (SENSORS'22)*. IEEE, 1–4.

[9] Rami Debouk, Stéphane Lafortune, and Demosthenis Teneketzis. 2002. On an optimization problem in sensor selection. *Discr. Event Dynam. Syst.* 12 (2002), 417–445.

[10] Arif Tanju Erdem and Ali Özer Ercan. 2014. Fusing inertial sensor data in an extended Kalman filter for 3D camera tracking. *IEEE Trans. Image Process.* 24, 2 (2014), 538–548.

[11] Laura Erhan, M Ndubuaku, Mario Di Mauro, Wei Song, Min Chen, Giancarlo Fortino, Ovidiu Bagdasar, and Antonio Liotta. 2021. Smart anomaly detection in sensor systems: A multi-perspective review. *Inf. Fusion* 67 (2021), 64–79.

[12] Raghu K. Ganti, Fan Ye, and Hui Lei. 2011. Mobile crowdsensing: Current state and future challenges. *IEEE Commun. Mag.* 49, 11 (2011), 32–39. https://doi.org/10.1109/MCOM.2011.6069707

[13] Antonio Ghezzi, Donata Gabelloni, Antonella Martini, and Angelo Natalicchio. 2018. Crowdsourcing: A review and suggestions for future research. *Int. J. Manage. Rev.* 20, 2 (2018), 343–363.

[14] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. 2008. Design science in information systems research. *Manage. Inf. Syst. Quart.* 28, 1 (2008), 6.

[15] Jeroen D. Hol, Thomas B. Schön, Henk Luinge, Per J. Slycke, and Fredrik Gustafsson. 2007. Robust real-time tracking by fusing measurements from inertial and vision sensors. *J. Real-Time Image Process.* 2, 2 (2007), 149–160.

[16] Shiyuan Jin, Ming Zhou, and Annie S. Wu. 2003. Sensor network optimization using a genetic algorithm. In *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics*. 109–116.
[17] Siddharth Joshi and Stephen Boyd. 2008. Sensor selection via convex optimization. *IEEE Trans. Sign. Process.* 57, 2 (2008), 451–462.
[18] Lisa Kessler, Felix Rempe, and Klaus Bogenberger. 2021. Multi-sensor data fusion for accurate traffic speed and travel time reconstruction. *Front. Fut. Transport.* 2 (2021), 766951.
[19] Igor Khokhlov, Sergei Chuprov, and Leon Reznik. 2022. Integrating security with accuracy evaluation in sensors fusion. In *Proceedings of the IEEE Conference on Sensors (SENSORS'22)*. 1–4. https://doi.org/10.1109/SENSORS52175.2022.9967235
[20] Igor Khokhlov, Akshay Pudage, and Leon Reznik. 2019. Sensor selection optimization with genetic algorithms. In *Proceedings of the IEEE Conference on Sensors (SENSORS'19)*. IEEE, 1–4.
[21] Igor Khokhlov, Akshay Pudage, and Leon Reznik. 2019. Sensor selection optimization with genetic algorithms. In *Proceedings of the IEEE Conference on Sensors (SENSORS'19)*. https://doi.org/10.1049/ic.2013.0047
[22] Igor Khokhlov, Leon Reznik, and Sahil Ajmera. 2020. Sensors in mobile devices knowledge base. *IEEE Sens. Lett.* 4, 3 (2020), 1–4.
[23] Igor Khokhlov, Leon Reznik, and Rohit Bhaskar. 2019. The machine learning models for activity recognition applications with wearable sensors. In *Proceedings of the 18th IEEE International Conference On Machine Learning And Applications (ICMLA'19)*. 387–391. https://doi.org/10.1109/ICMLA.2019.00072
[24] Igor Khokhlov, Leon Reznik, Justin Cappos, and Rohit Bhaskar. 2018. Design of activity recognition systems with wearable sensors. In *Proceedings of the IEEE Sensors Applications Symposium (SAS'18)*. 1–6. https://doi.org/10.1109/SAS.2018.8336752
[25] Igor Khokhlov, Leon Reznik, and Sergey Chuprov. 2021. Framework for integral data quality and security evaluation in smartphones. *IEEE Syst. J.* 15, 2 (2021), 2058–2065. https://doi.org/10.1109/JSYST.2020.2985343
[26] Igor Khokhlov, Leon Reznik, Suresh Babu Jothilingam, and Rohit Bhaskar. 2018. What can data analysis recommend on design of wearable sensors? In *Proceedings of the 15th IEEE Annual Consumer Communications and Networking Conference (CCNC'18)*. 1–2. https://doi.org/10.1109/CCNC.2018.8319288
[27] Siyu Liu, Xunyuan Yin, Zhichao Pan, and Jinfeng Liu. 2022. A sensitivity-based approach to optimal sensor selection for process networks. arXiv:2208.00584. Retrieved from https://arxiv.org/abs/2208.00584
[28] Arash Mahboubi, Seyit Camtepe, and Keyvan Ansari. 2020. Stochastic modeling of IoT botnet spread: A short survey on mobile malware spread modeling. *IEEE Access* 8 (2020), 228818–228830. https://doi.org/10.1109/ACCESS.2020.3044277
[29] Jonathan K. Millen. 2011. *Biba Model*. Springer US, Boston, MA, 81–82. https://doi.org/10.1007/978-1-4419-5906-5_812
[30] Mahmoud Moshref, Rizik Al-Sayyed, and Saleh Al-Sharaeh. 2021. An enhanced multi-objective non-dominated sorting genetic routing algorithm for improving the QoS in wireless sensor networks. *IEEE Access* 9 (2021), 149176–149195. https://doi.org/10.1109/ACCESS.2021.3122526
[31] Steve Olenski. 2015. The State Of Crowdsourcing. Retrieved March 5, 2024 from https://www.forbes.com/sites/steveolenski/2015/12/04/the-state-of-crowdsourcing/
[32] Ujjval N. Patel and Imraan A. Faruque. 2021. Sensor fusion to improve state estimate accuracy using multiple inertial measurement units. In *Proceedings of the IEEE International Symposium on Inertial Sensors and Systems (INERTIAL'21)*. IEEE, 1–4.
[33] Tong Qin, Shaozu Cao, Jie Pan, and Shaojie Shen. 2019. A general optimization-based framework for global pose estimation with multiple sensors. arXiv:1901.03642. Retrieved from https://arxiv.org/abs/1901.03642
[34] Qinru Qiu, Qing Wu, Daniel Burns, and Douglas Holzhauer. 2006. Lifetime aware resource management for sensor network using distributed genetic algorithm. In *Proceedings of the International Symposium on Low Power Electronics and Design*. 191–196.
[35] L. Reznik and Elisa Bertino. 2013. POSTER: Data quality evaluation: Integrating security and accuracy. 1367–1370. https://doi.org/10.1145/2508859.2512502
[36] Navrati Saxena, Abhishek Roy, and Jitae Shin. 2009. QuESt: A QoS-based energy efficient sensor routing protocol. *Wireless Commun. Mob. Comput.* 9, 3 (2009), 417–426.
[37] Nidamarthi Srinivas and Kalyanmoy Deb. 1994. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evol. Comput.* 2, 3 (1994), 221–248.
[38] Ailing Tang, Yufan Hu, and Rong Yan. 2023. Enhancing BERT for short text classification with latent information. In *Neural Information Processing*, Mohammad Tanveer, Sonali Agarwal, Seiichi Ozawa, Asif Ekbal, and Adam Jatowt (Eds.). Springer International Publishing, Cham, 122–132.
[39] Data Quality Lab. Data Collected by the Data Quality Lab. Retrieved March 5, 2024 from http://www.dataqualitylabs.com/dataView
[40] Data Quality Lab. Sensor Selector Android OS Application. Retrieved March 5, 2024 from https://play.google.com/store/apps/details?id=edu.rit.dataqualitylab.sensorselector

[41] Data Quality Lab. System Security Evaluation Android OS Application. Retrieved March 5, 2024 from https://play.google.com/store/apps/details?id=com.igorkh.trustcheck.securitycheck&hl=en_US&gl=US
[42] Data Quality Lab. Sensor Quality Assessment Android OS Application. Retrieved March 5, 2024 from https://play.google.com/store/apps/details?id=com.dataqualitylab.sensorquality
[43] Fortune Business Insights. Smartphone Market Size, Share & COVID-19 Impact Analysis. Retrieved March 5, 2024 from https://www.fortunebusinessinsights.com/industry-reports/smartphone-market-100308
[44] Ayush Vora, Leon Reznik, and Igor Khokhlov. 2018. Mobile road pothole classification and reporting with data quality estimates. In *Proceedings of the 4th International Conference on Mobile and Secure Services (MobiSecServ'18)*. 1–6. https://doi.org/10.1109/MOBISECSERV.2018.8311437
[45] Richard Y. Wang and Diane M. Strong. 1996. Beyond accuracy: What data quality means to data consumers. *J. Manage. Inf. Syst.* 12, 4 (1996), 5–33.
[46] Zeli Xue. 2021. Routing optimization of sensor nodes in the Internet of Things based on genetic algorithm. *IEEE Sens. J.* 21, 22 (2021), 25142–25150. https://doi.org/10.1109/JSEN.2021.3068726
[47] Leehter Yao, William A. Sethares, and Daniel C. Kammer. 1993. Sensor placement for on-orbit modal identification via a genetic algorithm. *AIAA J.* 31, 10 (1993), 1922–1928.
[48] Xinglin Zhang, Zheng Yang, Wei Sun, Yunhao Liu, Shaohua Tang, Kai Xing, and Xufei Mao. 2016. Incentives for mobile crowd sensing: A survey. *IEEE Commun. Surv. Tutor.* 18, 1 (2016), 54–67. https://doi.org/10.1109/COMST.2015.2415528

Received 30 November 2023; revised 23 June 2024; accepted 15 August 2024