



Article

Semantic-Driven Approach for Validation of IoT Streaming Data in Trustable Smart City Decision-Making and Monitoring Systems

Oluwaseun Bamgboye ^{1,*} , Xiaodong Liu ^{1,*} , Peter Cruickshank ¹ and Qi Liu ²

¹ School of Computing, Engineering and Built Environment, Edinburgh Napier University, Edinburgh EH10 5DT, UK; p.cruickshank@napier.ac.uk

² School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China; qi.liu@nuist.edu.cn

* Correspondence: o.bamgboye@napier.ac.uk (O.B.); x.liu@napier.ac.uk (X.L.)

Abstract: Ensuring the trustworthiness of data used in real-time analytics remains a critical challenge in smart city monitoring and decision-making. This is because the traditional data validation methods are insufficient for handling the dynamic and heterogeneous nature of Internet of Things (IoT) data streams. This paper describes a semantic IoT streaming data validation approach to provide a semantic IoT data model and process IoT streaming data with the semantic stream processing systems to check the quality requirements of IoT streams. The proposed approach enhances the understanding of smart city data while supporting real-time, data-driven decision-making and monitoring processes. A publicly available sensor dataset collected from a busy road in Milan city is constructed, annotated and semantically processed by the proposed approach and its architecture. The architecture, built on a robust semantic-based system, incorporates a reasoning technique based on forward rules, which is integrated within the semantic stream query processing system. It employs serialized Resource Description Framework (RDF) data formats to enhance stream expressiveness and enables the real-time validation of missing and inconsistent data streams within continuous sliding-window operations. The effectiveness of the approach is validated by deploying multiple RDF stream instances to the architecture before evaluating its accuracy and performance (in terms of reasoning time). The approach underscores the capability of semantic technology in sustaining the validation of IoT streaming data by accurately identifying up to 99% of inconsistent and incomplete streams in each streaming window. Also, it can maintain the performance of the semantic reasoning process in near real time. The approach provides an enhancement to data quality and credibility, capable of providing near-real-time decision support mechanisms for critical smart city applications, and facilitates accurate situational awareness across both the application and operational levels of the smart city.

Keywords: IoT streaming data; internet of things; stream quality validation; semantic technology; smart city model; RDF



Academic Editor: Giuseppe Maria Luigi Sarnè

Received: 27 February 2025

Revised: 10 April 2025

Accepted: 17 April 2025

Published: 21 April 2025

Citation: Bamgboye, O.; Liu, X.; Cruickshank, P.; Liu, Q. Semantic-Driven Approach for Validation of IoT Streaming Data in Trustable Smart City Decision-Making and Monitoring Systems. *Big Data Cogn. Comput.* **2025**, *9*, 108. <https://doi.org/10.3390/bdcc9040108>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The increase in the availability of publicly generated sensor streams from the connected Internet of Things (IoT) has become the major driver for most smart city innovations. The IoT Streaming data are usually accessed through the Application Programming Interfaces (APIs) and remains an essential artefact for driving automation and services even at the

lowest granularity level of IoT applications [1]. In the present IoT era, most city planners will rely on these data to make operational decisions or require such at the application layer of the smart city model. The concept of a trustable smart city is the term we used to describe a smart city ecosystem that ensures the reliability and quality of IoT-generated streaming data to improve service delivery and city infrastructures.

It is evident that IoT streaming data may be impacted by inherent quality problems such as stream inconsistency (redundancy or outlier) and missing (or incomplete) data [2], which impact the effectiveness and trustworthiness of smart city systems. Similarly, it is now becoming evident that the notion of trust can be considered a measure of the IoT data quality [3,4], apart from its broader use in the context of data privacy and security. These IoT quality problems can compromise the quality of outputs or accuracy of IoT-based data-driven decisions and related city services subscribing to them even in real time. For example, the World Health Organization (WHO) has stressed the importance of real-time air quality monitoring and data-driven decisions to reduce health-related impacts.

Meanwhile, current approaches to improve the accuracy of sensor readings majorly involve the use of statistical or probabilistic models [5] and point calibrations [6]. A probabilistic model such as Machine Learning can extract meaningful insights from streams but will require initial data storage for model training/learning before such a model can be applied. It does not support data interoperability requirements for data-driven smart systems and is unable to provide runtime reasoning as well as simultaneously analyse data produced from heterogeneous sources [7]. Similarly, the method of point calibrations can sometimes lead to uncertainties in the accuracy of measurements and the theoretical understanding of these measurement uncertainties remains unclear [8]. The need for stronger quality checks with enhanced capabilities for real-time data validation in IoT systems and to support decision-making processes is now in demand [9].

The aim of this research is to provide a semantic IoT data validation approach that supports real-time monitoring, data-driven actuation and decision-making processes. This will enable the deployed smart city monitoring systems, domain expert and city planner to exploit quality enriched publicly available sensor data produced by IoT networks. The approach uses semantic technologies such as ontology for knowledge graph as well as modelling and annotating the IoT data streams. C-SPAQL [10] is used to achieve the semantic stream querying over aggregated stream windows, while the Jena rule language is used to develop the semantic forward rules used to achieve a continuous semantic reasoning process. Currently, the approach is limited in its ability to provide semantic reasoning support to other serialised RDF data formats, such as the Json_LD, TriG, HDT and TriX formats.

The main contribution of this work is developing a semantic-driven approach for identifying poor-quality sensor streams and proposing a software architecture to support the near-real-time requirements of streams. Other contributions include the following:

1. It introduces a method for the exploitation of semantic technology capabilities to define the IoT streaming data validation framework.
2. It develops a continuous semantic reasoning technique that layers the semantic stream processing system with forward rules and serialised RDF data formats for the validation of missing and inconsistent data streams.

The rest of this paper is structured as follows: Section 2 provides the related work on IoT data quality and assessment, including the application of semantic technologies in a smart city. Section 3 describes the semantic approach and the architecture. Section 4 provides the details of the use case in air quality monitoring with the associated experiments involving data collected from IoT sensors. The analysis of the results and relevant

discussions from the evaluation are presented in Section 5. Section 6 concludes this paper with the direction for future work.

2. Related Work

The importance of the IoT and the relevance of data quality have become major drivers of big data initiatives and the decision-making process in smart cities. Data collection, data sharing and integrated systems are part of the major characteristics of such cities. Gaining access to shared data across various integrated IoT platforms and heterogeneous sensors becomes even more problematic and increases the chance of erroneous and inaccurate data [4]. Stakeholders can base their trust on city services using the quality of the IoT streams and analysis, quality of decisions and discovery of more actionable insights that can improve the quality of life of its citizens.

2.1. IoT Data Quality Assessment

The notion of data quality has often been considered as a subjective measurement that depends on the specific IoT domain and relates to the assessment of confidence in the data provided by the IoT nodes. The continuous increase in the number of connected IoT devices has made it hard to assess the IoT data quality with common assessment methods [11]. The assessment of data quality in air quality monitoring by attempting to improve the efficiency and decisions in smart cities was studied in [12]. By contextualising the work in the assessment of air quality monitoring project in the Colombian city, they only provided analytical interpretations of the quality indicators. The main data quality indicator is associated with uncertainty in measurement. This indicator also conforms to the one identified by [13]. Some of the dimensions to the indicators include timeliness, completeness, redundancy, accuracy, etc. Semantically enriched IoT data quality validation was proposed by [3]. However, the validation employed is limited to an RDF graph based on SHACL (<https://www.w3.org/TR/shacl/> (accessed on 10 April 2025)) and does not specifically relate to missing or incomplete IoT streams. A more recent approach [14] focused on applying the semantic framework for enriching the IoT data and using this with SHACL to perform IoT data quality assessment. The author further advocated for better IoT data validation with emphasis on missing or incomplete data and duplicated data, which may differ in another context. Other recent approaches to IoT data quality assessments mainly focused on the application of blockchain technology [15], artificial intelligence (AI) [16] and analytical approaches [11].

2.2. IoT Data Quality in the Context of Trustworthy Cities

The IoT has long been regarded as the primary enabler for many technologies implemented in smart cities. Significant emphasis has been placed on the need for error detection and ensuring the trustworthiness of IoT data streams to support effective decision-making [17]. Generally, the concept of trust is considered to be broad. In describing the notion of trustworthy cities, emphasis is placed on extending the definition of trust to include the degree of confidence in IoT data based on data quality requirements in addition to privacy and security issues. Furthermore, the definition of trustworthy cities has previously been attributed to the concept of a safe city [18] and evaluation of trust in AI application to smart cities [19]. A related study [20] also evaluated the trustworthiness of the IoT noisy data based on policy rules and the reporting history. The main objective of trustworthy cities is to seek ways of improving the quality of life while exploiting the analysis of good quality data collected through connected IoT technologies to make a city more efficient, safer and reliable in terms of service delivery. This notion of trustworthiness

should now be viewed and align with the quality and reliability of the vast amount of data that are produced and consumed by the city services.

2.3. Application of Semantic Technologies in Smart City Innovations

Semantic technologies, usually based on the formalism of the Resource Description Framework (RDF) and ontology, enable smart objects and IoT nodes to interact with each other in an intelligent manner. These technologies also allow IoT systems to automate data/information acquisition and enhance the decision-making process [21]. Motivated by the current challenges relating to data quality issues and supporting decision-making in smart cities, a recent study [22] considered the use of ontology to facilitate data interoperability while relying on a third-party tool for maintaining data quality to develop a semantic-based framework for data sharing. However, their approach to assessing the IoT data quality requirements was performed manually and the details of the specific quality dimensions considered as part of the study were not discussed. Furthermore, the outstanding success of the application of semantic technology in the context of smart city innovations confirms its suitability for dealing with interoperability issues [23–25] and the modelling/annotation [26,27] of sensor readings.

Semantic Technologies in Air Quality Monitoring

The use of semantic technology in air quality monitoring is increasingly becoming popular, with many approaches applying this to support knowledge representation and inference [28], semantic reasoning [29] and the integration of real-time data [30]. Recently, an innovative framework that integrates Complex Event Processing (CEP) and SPARQL queries for monitoring air quality within smart cities was proposed by [31]. The framework adopts decision trees to generate rules based on certain air quality parameters, which facilitates the classification of air quality levels by filtering complex patterns from pollutant data streams. This was able to produce a more optimised query processing and a decision support system that can alert stakeholders to air quality conditions. However, consideration for missing or noisy data streams from sensors was not included as part of the requirement for the proposed framework. Overall, most of the innovations and approaches continue to exploit the benefits of the semantic technologies in smart city and air quality monitoring without adequate consideration for quality validation of the data streams used for maintaining the air quality index.

3. Our Approach

In context of the layered smart city model shown in Figure 1, we considered the emphasis on city operations such as air quality monitoring (mainly data driven) to be very essential for exploiting valuable insights from sensed data and effective smart city deployment.

Figure 2 provides an overall view of the approach and its application to the smart city domain. Heterogeneous sources of raw IoT streaming data are labeled numbers 1 to 5 within the IoT streams ecosystem.

The goal of the approach and its architecture is presented in two main parts: to provide a semantic data model to facilitate IoT streaming data annotation and RDF data representation and the near-real-time quality validation of IoT streaming data relating to inconsistency and missing/incomplete streams against a known quality index.

In more specific detail, heterogeneous raw IoT streaming data providing certain context measurements such as temperature, carbon monoxide and so on are generated by various sensors deployed within the city. The generated IoT streams are processed and validated for quality requirements and in accordance with the real-time requirements. This will be made available for subsequent stream subscribers or smart monitoring systems to

support further actionable decisions. The use of such quality-enriched IoT streams can change or influence the type of events or actions that will be triggered by other connected smart systems. In cases where a data anomaly has been detected, actionable decisions are made and a newly identified data pattern can be used to reconfigure the sensor network to process a more reliable reading.

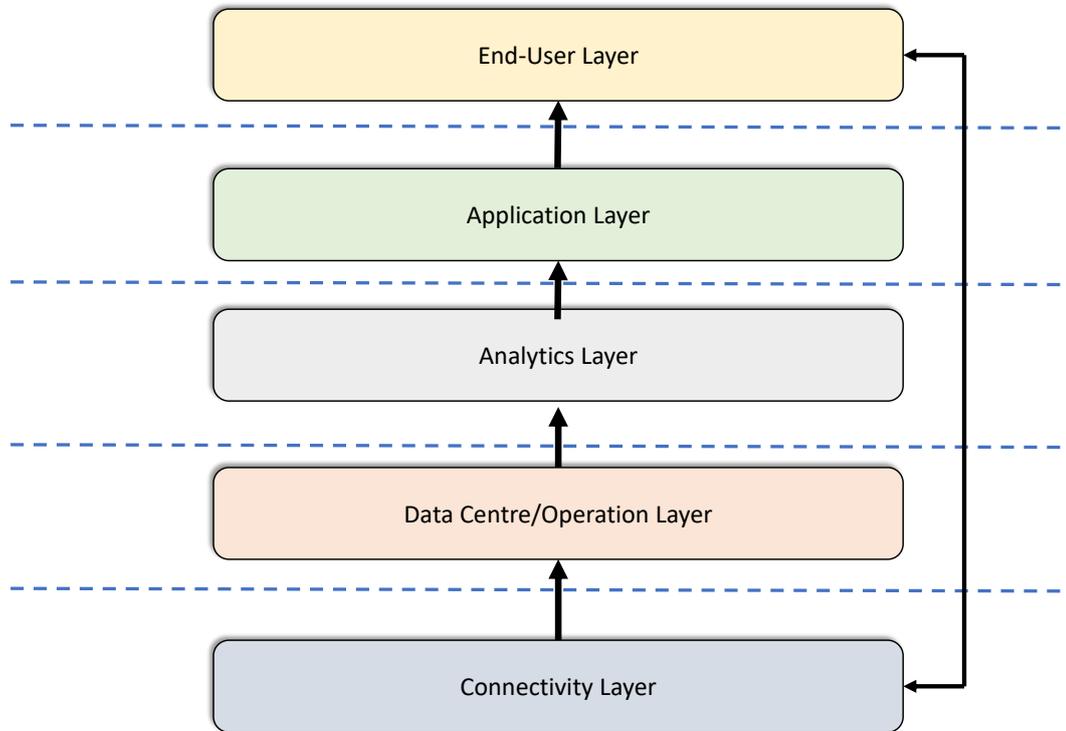


Figure 1. Smart city layered Model (Adapted from [32]).

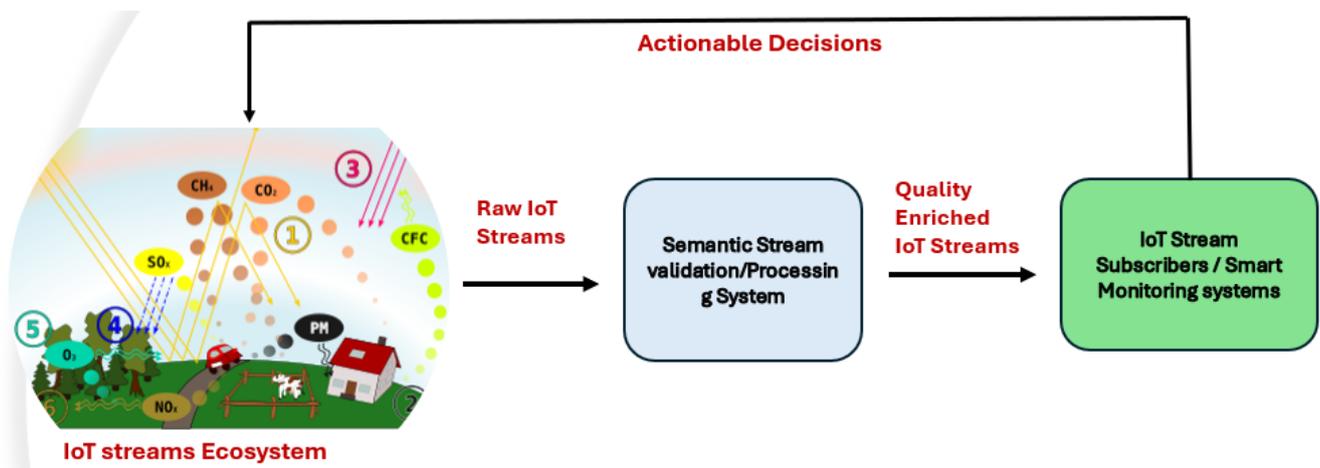


Figure 2. Overall approach in smart city context.

The purpose of this research is to provide the semantic stream quality validation architecture for IoT streaming data. The architecture is primarily based on two different technologies: the RDF stream processing system and the window-based semantic stream reasoning system. These two technologies have been carefully outlined and integrated into a three-layer architecture presented in Figure 3. It implements the second stage in the overall approach in the smart city context: the semantic stream validation/processing system.

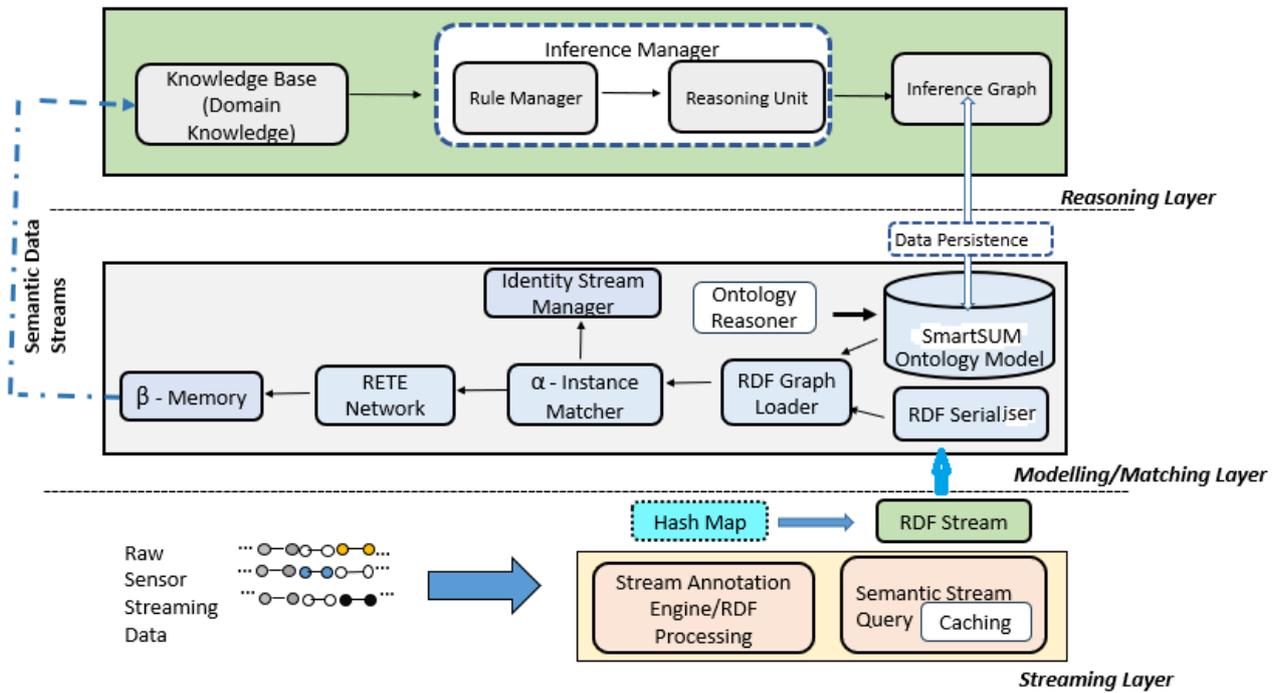


Figure 3. Proposed semantic IoT streaming data validation architecture.

The proposed bottom-up layered architecture consists of the streaming layer, matching layer and reasoning layer. It is necessary to note that the architecture is positioned between the lowest connectivity layer (comprising various sensors and IoT devices, leveraging connectivity infrastructure such as Bluetooth, LTE, 5G, or other communication technologies, which are owned and operated by public or private entities) and the upper application layer (which implements various industry-specific and horizontal applications) of the smart city model. The following sections describe the details of each layer of the proposed architecture.

3.1. Streaming Layer

The focus of this layer is to semantically pre-process the heterogeneous IoT streaming data and their temporal aspects while enriching the streams with the appropriate metadata. The layer typically consists of the semantic stream query and Semantic Annotation Engine. The Semantic Annotation Engine contained in Figure 4 uses a lightweight ontology model (called SmartSUM ontology), which we developed from the ontology re-engineering process that consists of two base ontology models (ssn/sosa ontology [33] and Time ontology [34] enhanced with other relevant entities associated with stream quality dimensions and smart space entities. SmartSUM ontology represents an ontology that provides the semantic description and relationships between IoT data and related concepts within the IoT network. It is also required for maintaining the interoperability of the heterogeneous IoT data streams and smart devices. Both ssn/sosa ontology (a modular framework using SOSA at its core that describes sensors, actuators, measurement capabilities, observations, related procedures, observed features, features of interest and deployments) and Time ontology are separate ontology models previously developed by other researchers to, respectively, model the semantic sensor network and time measurements. These models are especially required to form the foundation for the development of SmartSUM ontology to achieve a consistent and reusable ontology model. The description of the taxonomy including the various concepts considered during SmartSUM development are discussed in the next section: Ontology Reuse and Re-Engineering for SmartSUM Construction.

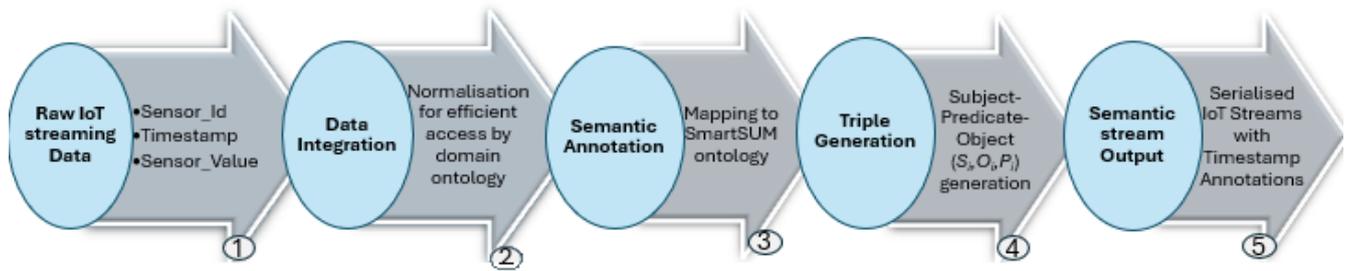


Figure 4. Semantic IoT streaming data generation.

Raw IoT streaming data are admitted into the architecture after it has been successfully transformed by the semantic stream generation (see Figure 4) for the appropriate indexing of streams. This process is enhanced by the hash table within the Hash Map module. The Hash Map module mimics a data structure that is based on key–value pairs automatically generated to maintain the distinct identification of each annotated IoT streaming datum. This provides the initial elimination of repeated or redundant IoT stream instances caused by overlapping streaming windows of raw streaming data. It is achieved by defining a unique key for each semantic stream at this stage of the process.

The semantic stream generation process starts by initially combining all the related sensor data from different sensor nodes into a single unified view through the data integration. The data streams will then be semantically annotated and converted to unique RDF stream data using the SmartSUM ontology. The resulting RDF streams represent a triple-encoded representation describing an unbounded, ordered sequence of evolving IoT data. The equivalent RDF stream is achieved with the support of the Semantic Annotation Engine by further annotating serialised RDF data with individual timestamps. The engine relies mainly on the vocabularies from SmartSUM ontology to define the semantic streaming data. The semantic streaming data (also called quadruple statements) as defined by [10] are triple representations with timestamps representing the raw streaming data:

$$\cdots (S_k, O_k, P_k, T_k), (S_{k+1}, O_{k+1}, P_{k+1}, T_{k+1}) \cdots \quad (1)$$

The semantic representation of the streaming data in Equation (1) is necessary to facilitate the semantic query of multiple quadruple statements.

The semantic stream query registers a continuous stream query that can process serialised IoT streaming data as they are produced by the Stream Annotation/Generation Engine. The stream query is based on C-SPARQL [10], used for runtime multiple selection and the ordering of semantic IoT streaming data over a continuous sliding window. C-SPARQL, previously built as an extension of the SPARQL query language, was developed to simplify the continuous querying of RDF streams and executed over a continuous stream window. To speed up the query processing, a caching method is introduced within the query processing module. This approach to querying semantic streaming data supports multiple and parallel selections of RDF representations of the IoT streaming data. The output of the query in the form of semantic streams is received in the matching layer of the architecture for further processing.

Ontology Reuse and Re-Engineering for SmartSUM Construction

The combined ssn/sosa ontology with the Time ontology formed the basis for the ontology re-engineering and reuse process [35,36] during the construction of the SmartSUM ontology. As a lightweight ontology, SmartSUM facilitates the semantic enrichment of raw sensor data streams by defining a structured vocabulary for annotating quadruple RDF statements—each representing a sensor observation with a subject, predicate, object

and timestamp. The ontology provides the semantic backbone for the Stream Annotation Engine, which transforms heterogeneous sensor readings into standardised RDF streams that are interoperable and machine-understandable. The SmartSUM ontology currently contains an additional 144 new concepts with 59 object properties and 42 data properties, which are missing from the base models. It is used for the semantic annotation of IoT streaming data and facilitating the RDF data serialisation. The ontology re-engineering process is used to facilitate the enhancement by identifying missing domain features and representing them as new concepts (subclass with properties) to the base ontology model. During the ontology reuse process, all the related base ontological models are linked together through a method of ontology alignment. The approach to ontology alignment is considered a classification issue that can be resolved through the ontology concept mapping [37]. For specific measurement concepts M_x and M_y defined within the base ontology X and Y , the alignment between the two concepts can be defined as follows:

$$\mathcal{R} = \{(M_x, M_y) \in M_x \times M_y \mid M_x \equiv M_y\} \quad (2)$$

Assuming M_x and M_y are defined within the same ontology in any of the ones chosen for reuse purpose, then $M_x \equiv M_y$ as defined by Equation (3) is not satisfied. Therefore, M_x from ontology X can be aligned with the closest concept M_y in Y ontology. For instance, both concepts *SSN:Output* in *SSN* ontology are the same as *SOSA:Result* in the *SOSA* ontology.

The hierarchical concept diagram in Figure 5 represents the taxonomy of concepts that describes the major concepts involved in the construction of the SmartSUM ontology. The ontology is mainly developed from concepts that are specific to IoT streaming data and smart environment/spaces. The hierarchical relationships allow for concepts to be described as either superclass or subclass. The major concepts from the upper ontological model are used directly and extended where applicable to avoid unnecessary ambiguity among the concepts and to achieve a reusable lightweight model. New domain-related concepts not found within the upper ontology models are explicitly created as either a superclass or subclass in the concept hierarchy.

The namespace adopted by the SmartSUM ontology model is known as *SMARTSPACE*. For instance, to re-engineer and reuse the upper ontology for the ontology construction, SmartSUM extends the *SOSA* ontology by organising the concept *SMARTSPACE:Hub* as a subclass of *SOSA:Platform*. This means *SMARTSPACE:Hub* is a kind of platform for hosting other sensors, actuators, controllers and other smart space systems. The *SMARTSPACE:DoorActuator*, *SMARTSPACE:VoiceRecognition* and *SMARTSPACE:Controller* are all subclasses of *SSN:System*, enabling the proper identification of what qualifies as a system and differentiating it from other concepts. Furthermore, the *SOSA:Sensor* enables the definition of any object that can respond to stimulus. A sensor detects inputs and produces certain outputs. Therefore, both *SMARTSPACE:PhysicalSensor* and *SMARTSPACE:VirtualSensor* are a subclass of *SOSA:Sensor* in the concept taxonomy. The *SMARTSPACE:PhysicalSensor* represents the hardware-based sensors (such as temperature sensor, camera, pressure sensor, etc.), while the *SMARTSPACE:VirtualSensor* concept is for non-hardware-based or software-based sensors (such as a crowd and feelings) captured within the space. Finally, *SOSA:FeatureOfInterest* is used as a superclass for concepts such as *SMARTSPACE:Temperature*, *SMARTSPACE:CO₂*, *SMARTSPACE:NO₂*. The *SMARTSPACE:SensorReading* \subseteq *SOSA:ObservableProperty* is a concept that describes various sensor readings from various properties, such as temperature, pressure, humidity, Benzene, etc. Each of the readings is associated with *SMARTSPACE:Timestamp*, which happens to be a subclass of *TIME:TemporalEntity*.

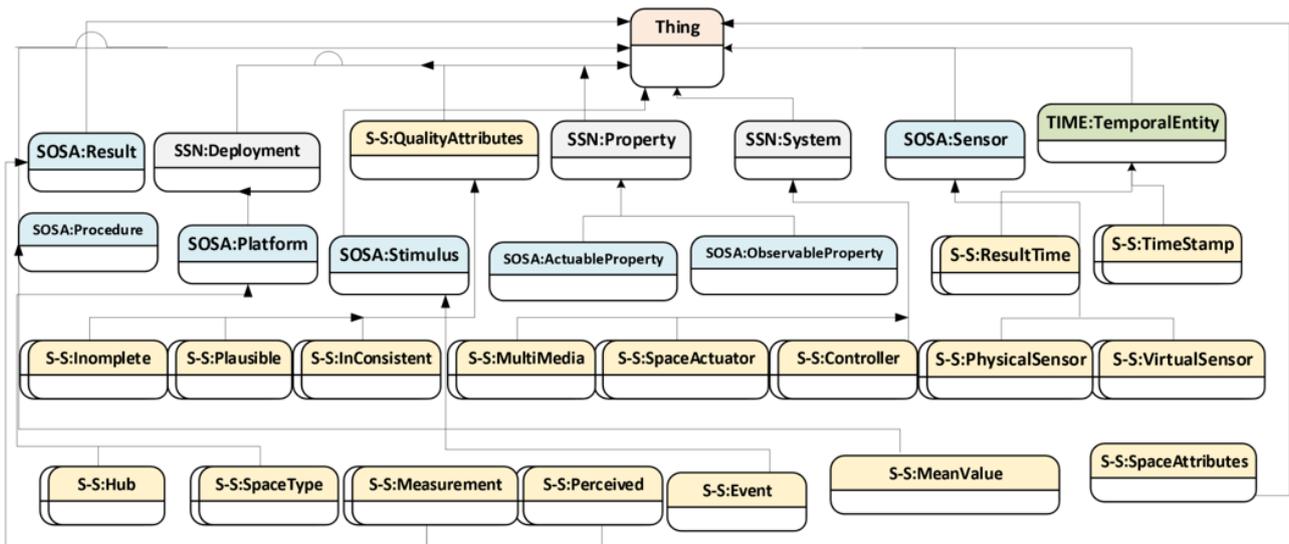


Figure 5. Abstract concept description of SmartSUM model.

3.2. Matching Layer

This layer consists of modules that support the semantic processes in the streaming layer and reasoning layer of the architecture. The entry point to the matching layer is the RDF serialiser, which is responsible for converting the RDF stream into alternative RDF data. The choice of serialised data formats is determined by its compatibility with C-SPARQL. Alternative RDF formats supported by the architecture include the Turtle (*.ttl*), Notation3 (*.n3*), RDF/XML (*.rdf*) and N-Triple (*.nt*) formats. Other serialised formats such as JSON_LD, NQuads, RDF/JSON and HDT are outside the scope of this work and may not be suitable for the proposed domain ontology. The RDF serialiser module use the RDF schema derived from the SmartSUM ontology to convert the streams into RDF serialised data formats. The importance of the RDF serialiser module is to provide an encoding that will enhance the expressiveness of the semantic validation of the IoT streaming data. The output of the module is transferred to the RETE Network module, which contains the RETE network as specified by [38], α -Instance Matcher and β -memory. The RETE module maintains a static directed graph pattern that maintains the conditions for incoming semantic streams to ensure only completed triples are allowed to progress to the next phase of the semantic process.

The RETE network is known for its ability to support real-time consistency validation and incremental pattern matching. The RETE network is required to maintain the state of the current semantic process and compute partial results for future use without the need to re-process the request when new semantic streams are received by this module.

The (α -)Instance Matcher forms part of the RETE network and relies on the pseudo-code for the stream Instance Matcher defined by Algorithm 1 to check whether each semantic stream (serialised RDF statement) conforms to the predefined RDF schema for all the semantic streams. Any of the semantic streams that violate the matching process (e.g., a missing *object* node from each quadruple statement) are classified as an identity stream and are immediately isolated from the current window by pushing them to the identity stream manager that contains static methods for excluding such streams from the sliding window. Otherwise, the output is received by the β -memory of the RETE network, which forms an intermediate memory that interfaces with the upper reasoning layer for further semantic reasoning and inference process.

Algorithm 1 Stream Instance Matcher

```

1: Inputs:
2:  $K$ : a fixed literal value
3:  $t', t''$ : are initial and current timestamps respectively
4:  $LS$ : a set of triples with individual timestamp
5:  $M$ : previous triple in  $n - 1^{th}$  position
6:  $N$ : a current triple in  $n^{th}$  position
7:  $\omega$ : a current streaming window with set of triples
8:  $W$ : Aggregate streaming windows
9:  $\alpha$ -memory,  $\beta$ -memory: are separate temporary memory
10: procedure READ( $M, N$ )
11:   for each  $M$  and  $N$  in  $\alpha$ -memory do
12:     if  $N = LS \in \omega$  and  $\omega \in W$  then
13:       Load  $M$  and  $N$  into  $\alpha$ -memory
14:       Decompose  $M$  and  $N$  into triple pattern
15:        $M = (s_M, p_M, o_M, t_M)$ 
16:        $N = (s_N, p_N, o_N, t_N)$ 
17:     end if
18:   end for
19: end procedure
20: procedure MATCH-PRUNE( $N$ )
21:   do
22:     Check match conditions:
23:      $s_M = s_N, o_M = o_N, t_M = t_N$ 
24:     if
25:        $N = M$  and  $N(t') = M(t')$ 
26:        $N = \emptyset$ 
27:        $N = K$  then
28:         Remove  $N$  from  $\alpha$ -memory
29:     else
30:       Set  $\beta$ -memory to  $N$ 
31:     end if
32:   while  $t' < t''$  and  $N \in \omega$ 
33:   Proceed to the next  $N$ 
34: end procedure

```

Given n represents the number of triples in a sliding window ω , the worst-case time complexity of Algorithm 1 is estimated as $\mathcal{O}(n)$. This linear complexity ensures that instance matching can scale efficiently with increasing stream sizes, provided that the number of triples per window remains bounded (e.g., 10 min slices). In terms of applicability to the IoT context, the algorithm is highly suitable for fog-level deployments in smart cities, where moderate computing power is available and responsiveness is critical. This is due to its low computational overhead and reliance on basic memory structures (e.g., hash tables and β -memory). Furthermore, its stateless design also supports the parallel processing of windows, enabling real-time anomaly detection in high-throughput environments (e.g., traffic congestion sensors or air pollution monitors).

3.3. Reasoning Layer

The reasoning layer consider a reasoning approach that can support the definition of domain rules (with forward rules), as this is currently missing in the existing semantic-based reasoners. It consists of the rule base, inference manager and inference graph modules. The layer access-serialised RDF data formats from the matching layer through the β -memory. The rule base stores a set of stream quality validation rules defined by domain experts, which are defined using some set of known data quality indicators (e.g., using a government-approved scale for the air quality index). The rule is used to filter inconsistent

IoT streaming data from streaming windows. The rules can be modified and apply to suit specific patterns of data distribution for the purpose of checking stream anomalies. The rule manager in the inference manager module facilitates the implementation of the rule pseudo-code described in Algorithm 2 by the domain experts for validating semantic streams for consistency checks. The rule can combine several relevant and related conditions to arrive at a conclusion and for semantic inference. It enforces the rule and constraints on domain-specific conditions defined by the domain experts. Similarly, given m to be the number of rule conditions and n the number of semantic statements per window, the computational complexity of Algorithm 2 is defined as $\mathcal{O}(m \cdot n)$. This is considered valid and overall remains computationally manageable since the number of rules m is typically small and domain-specific (e.g., 3–5 rules per pollutant type), and each stream window is temporally bounded. Algorithm 2 will be most appropriate for cloud-level or high-capacity fog deployments, where slightly higher reasoning latency (e.g., sub-second) is acceptable in exchange for richer semantic inference.

Algorithm 2 Semantic Rule for Consistency Check

```

1: Definitions:
2:  $C_1, C_2, C_3 \dots C_n$ : are set of rule conditions.
3:  $X$ : Specific measurement value of the target triple with associated timestamp.
4:  $\phi$ : Represents the  $n^{th}$  triple value with associated timestamp.
5:  $\wedge$ : 'AND' logical operator connecting statements and rule conditions.
6:  $\Delta$ : Set of comparison operators for IoT measurement timestamps
7: procedure CONSISTENCYCHECK( $C_1, C_2, C_3, \dots C_n$ )
8:   Rule Mode := "forward"
9:   if
10:     Conditions: (
11:       Condition 1:
12:        $C_1 = \{(?x \text{ namespace:p } ?K) \wedge (?x \text{ namespace:q } ?x\text{TimeStamp}) \wedge (\text{lowerBound} < x$ 
< upperBound)}  $\wedge$ 
13:       Condition 2:
14:        $C_2 = \{(?y \text{ namespace:p } ?K) \wedge (?y \text{ namespace:q } ?y\text{TimeStamp}) \wedge (\text{lowerBound} < x$ 
< upperBound)}  $\wedge$ 
15:       Condition 3:
16:        $C_3 = \{(?z \text{ namespace:p } ?K) \wedge (?z \text{ namespace:q } ?z\text{TimeStamp}) \wedge (\text{lowerBound} < x$ 
< upperBound)}  $\wedge$ 
17:       ...
18:       Condition N:
19:        $C_N = \{(?\phi \text{ namespace:p } ?K) \wedge (? \phi \text{ namespace:q } \phi \text{ TimeStamp}) (\text{lowerBound} < \phi$ 
< upperBound)} )
20:        $\wedge$ 
21:       Comparison of corresponding Timestamp:
22:        $\Delta ( ?x\text{TimeStamp}, z\text{TimeStamp}) \wedge$ 
23:        $\Delta ( ?x\text{TimeStamp}, z\text{TimeStamp}) \wedge$ 
24:       ...
25:        $\Delta ( ?x\text{TimeStamp}, \phi\text{TimeStamp})$  then
26:       Annotate value  $X$  as literal string: ( $?x \text{ namespace:p } 'X\text{-literal string}'$ )
27:     end if
28: end procedure

```

The reasoning unit adopts a continuous reasoning approach by layering each sliding-window session of the stream query with rules defined by Algorithm 2 and stored in the rule base. In practice, Algorithm 2 runs over a sliding window by executing the consistency rule over each semantic stream query window that contains the current snapshots of the semantic streams. The reasoning process is further enhanced with BIND or SCHEMABIND calls on the sliding windows to ensure there is no information loss during the continuous

reasoning process. The purpose of the reasoning is to produce new knowledge about streams by a method of semantic inference that is facilitated by the inference graph module while also taking into consideration the time component of the data.

The continuous validated output and semantically inferred knowledge produced by the inference graph component of the architecture persisted as RDF statements on the knowledge graph (KG) to support new semantic inferences. Finally, the IoT stream subscribers (which could be in the form of software agents or smart applications) and other smart monitoring systems as indicated in Figure 2 can continue to consume the quality-enriched data streams from the reasoning layer to support actionable decisions at the smart city application level.

4. Use Case and Experiment

The architecture proposed in the previous section has been prototyped as a software implementation applicable to a smart city scenario. We have demonstrated the feasibility of the proposed approach in attempt to evaluate the effectiveness in terms of accuracy and its efficiency in terms of the performance and response to real-time requirements. It is anticipated that the full implementation of the proposed architecture will cover edge, fog and cloud computing layers to support the semantic reasoning and real-time validation of IoT streaming data.

4.1. Use Case and Dataset

The proposed use case has been contextualised in the estimation of an urban pollution monitoring scenario in the city of Milan. The dataset [39] was collected from a monitoring campaign in the urban centre of the Italian city, specifically at a busy main road in Milan. This was performed as a collaborative effort between Pirelli Labs and the Lombardy Regional Environmental Protection Agency (ARPA), as part of a year-long study to evaluate sensor accuracy under urban conditions and in response to variable factors, like seasonal shifts and decisions on traffic levels. The use case considered processing the quality requirements of heterogeneous sensor streaming data to effectively enhance data-driven decision support systems in the context of smart city innovations. In particular, it addressed the operational decision-making that focuses on knowing when to reduce the carbon monoxide concentration in the air to sustain a cleaner environment in an attempt to realise one of the pillars of the IBM smart city model [40].

The dataset has been made available to support research on air quality and pollution management in dense urban areas. It contains 9358 chemical sensor instances that include measurements of benzene (C₆H₆) concentrations and other pollutants, like carbon monoxide (CO), Nitrogen Oxides (NO_x) and Nitrogen Dioxide (NO₂). The sensors were deployed in significantly polluted areas marked on the map in Figure 6. Each sensor is programmed to record data at hourly intervals. The data reflect real-world environmental variability, including missing values, outliers and duplicated timestamps—conditions commonly encountered in large-scale IoT deployments. Missing data points are assigned a placeholder value of −200 to indicate their absence within the affected data points. Instances of inconsistent values are recorded as duplicated values with the same timestamp as the previous interval of sensor reading. Furthermore, values found outside the defined range of the air quality index (AQI) are also tagged as an erroneous reading.



Figure 6. Map of sensor deployment at busy main road in Milan.

4.2. Data Stream Processing and Technologies

We like to emphasise that the data used for the experiments are constructed based on real-world data from the one described in Section 4.1. At the data preprocessing stage, the data are grouped and identified based on the relevance of the measured phenomena and their categorisation as an air pollutant. Similarly, all the data affected by cross-sensitivities and sensor drifts are excluded from the choice of candidate data points. Each value of the sensor readings has been simulated and aligned with the specific timestamps using the C-SPARQL Engine (<https://github.com/streamreasoning/CSPARQL-engine> (accessed on 10 April 2025)), a notable java-based stream processing system. In order for each stream to be annotated with a semantic description, the Jena library (<https://jena.apache.org/download/#apache-jena-binary-distributions> (accessed on 10 April 2025)) has been used to access the SmartSUM ontology, which also represents the base ontology model for transforming each sensor reading into an equivalent RDF stream (quadruple statements).

From the implementation perspective, the streaming layer represents the edge layer that captures fine-grained IoT streaming data relating to air quality measurements. Communication at this layer will be established through the MQTT (<https://activemq.apache.org/components/classic/documentation/mqtt> (accessed on 10 April 2025)) and CoAP protocols (<https://github.com/open-coap/java-coap> (accessed on 10 April 2025)), and the data are converted to RDF streams. To understand the effect of the semantic process, each RDF stream is reconstructed. This is achieved via the C-SPARQL Engine for the construction of instances of the semantic streams. Each stream selection is achieved with the equivalent semantic query. Figure 7 shows a sample C-SPARQL query for the stream selection over a continuous streaming window for a duration of 10 min with a maximum sleep time of 2 min between successive query executions. This approach to stream management adheres to the requirement for continuous data stream processing and real-time query execution across the streaming windows. Flexible data delivery between the layers of the

framework is managed by Apache camel (Available: <http://camel.apache.org/> (accessed on 10 April 2025)), which is a lightweight data transfer framework.

```

REGISTER QUERY sensorValueOf AS
PREFIX smartSpace:<http://localhost:8080/smartSpace#>
SELECT *
FROM STREAM <http://localhost:8080/smartSpace/streamCO> [RANGE 10m STEP 5m]
FROM STREAM <http://localhost:8080/smartSpace/streamNOX> [RANGE 10m STEP 5m]
FROM STREAM <http://localhost:8080/smartSpace/streamNO2> [RANGE 10m STEP 5m]
FROM STREAM <http://localhost:8080/smartSpace/streamBenzene> [RANGE 10m STEP 5m]
WHERE
{
    ?COReadings smartSpace:hasCOValue ?COVal.
    ?COReadings smartSpace:COHasTimestamp ?COTime.
    ?NOXReadings smartSpace:hasNOxValue ?NOXVal.
    ?NOXReadings smartSpace:NOxHasTimestamp ?NOXTime.
    ?NO2Readings smartSpace:hasNO2Value ?NO2Val.
    ?NO2Readings smartSpace:NO2HasTimestamp ?NO2Time.
    ?benzeneReadings smartSpace:hasBenzeneValue ?benzeneVal.
    ?benzeneReadings smartSpace:benzeneHasTimestamp ?benzeneTime.
}
ORDER BY ASC(?COTime)

```

Figure 7. Sample stream construction with timestamps.

The matching layer implements the native Jena RIOT API (<https://jena.apache.org/documentation/io/rdf-output.html> (accessed on 10 April 2025)) while relying on the background SmartSUM model to serialise the RDF streams received from the streaming layer into the four identified RDF serialisation formats. The implementation of this layer represents a fog layer deployed with the capability to validate and enrich each instance of a stream to produce the quadruple statement before pushing to the cloud layer. A Java programming class was implemented to achieve the instance matching in Algorithm 1 with emphasis on the quadruple statement pattern stored in the embedded triple store (RDF graph loader as static RDF graph). The algorithm uses a pattern-matching logic to flag incomplete streams in near real time. Each quadruple statement is discriminated once there is evidence of an incomplete data value or non-conformity with the defined schema, and the *object* node of the quadruple statement is labelled as a literal string.

The implementation of the reasoning layer allows for semantic stream management pipelines to be maintained via cloud processing, enabling flexible querying and integration with other smart city platforms. Stream ingestion has been achieved by referencing the C-SPARQL query engine and facilitated via the ActiveMQ broker and Java Message Service (JMS). The knowledge base integration allows for the serialised RDF streams to be matched with static ontology (such as SSN/SOSA) to enrich the context and semantics of the streams. A consistency check (Algorithm 2) is bound with a C-SPARQL query and allowed to execute against the sliding-window intervals to collect snapshots of matched streaming data and compare each statement with the defined rule. Figure 8 shows a sample rule for checking the availability of inconsistent streams within the streaming windows. The serialised RDF format of the semantic IoT streaming data in each current sliding window is accessed with the support of the RDF Application Programming Interface (API). This allows the reasoning unit to evaluate the snapshot of each semantic stream against the predefined rule in the rule base, thereby enabling continuous inference. The entire reasoning approach has been facilitated with the use of the Jena library (<http://jena.sourceforge.net/> (accessed on 10 April 2025)) and its corresponding subsystems.

```

@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
@prefix owl: http://www.w3.org/2002/07/owl#
@prefix rdfs: http://www.w3.org/2000/01/rdf-schema#
@prefix xsd: http://www.w3.org/2001/XMLSchema#
@prefix smartSpace: http://localhost:8080/smartSpace#

[inconsistentCheck:
  (?COReadings smartSpace:hasCOValue ?COValue)
  (?COReadings smartSpace:COHasTimestamp ?COTime)
  greaterThan(?COValue,11.9)
  (?NOXReadings smartSpace:hasNOxValue ?NOxValue)
  (?NOXReadings smartSpace:NOxHasTimestamp ?NOxTime)
  greaterThan(?NOxValue,322)
  lessThan(?NOxValue,2683)
  (?benzeneReadings smartSpace:hasBenzeneValue ?benzeneValue)
  (?benzeneReadings smartSpace:benzeneHasTimestamp ?benzeneTime)
  notEqual(?COValue,?benzeneValue)
  equal(?COTime,?NOxTime)
  equal(?COTime,?benzeneTime)
  ->
  (?COReadings smartSpace:isInconsistent 'Inconsistent Check')
]

```

Figure 8. Sample reasoning rule for inconsistent carbon monoxide readings.

4.3. Experiments

The experimentation process was designed to evaluate the efficiency and accuracy of the proposed semantic-based IoT streaming data validation architecture under realistic smart city conditions. It focused on reconstructing the raw streaming data produced from the physical sensor nodes described in Section 4.1 at different experimental runs.

4.3.1. Simulation Setup

The experimental setup was simulated on a single node running on a multiple processor computer (Pentium Core (TM) i7-4770 CPU @ 3.40 GHz–16 GB RAM). The Java Virtual Machine (JVM) was configured with an initial heap size of 1024 MB and a maximum of 2048 MB. The entire software stack, including semantic processing, rule-based validation and stream management, was implemented using the Java programming language. The libraries and tools used during the experiments include Apache Jena (for ontology access, RDF serialisation and semantic reasoning), Apache camel for stream orchestration and inter-layer communication, ActiveMQ with JMs for real-time stream delivery and C-SPARQL to simulate the real-time querying.

4.3.2. Data Reconstruction and Preprocessing

Sensor observations based on raw sensor data from the Milan air quality dataset were pre-processed to simulate real-time streaming behaviour. The data preprocessing stage starts by filtering irrelevant sensor readings before using the Java program to reconstruct and align the sensor values with the corresponding timestamps to reflect the true hourly observations. Data quality anomalies in the form of missing values (–200), inconsistent duplicates (repeated readings with identical timestamps) and out-of-range values (outside the acceptable AQI range) were injected under controlled simulated conditions. The observations were categorised by pollutant type (e.g., CO, NO_x and NO₂) and converted into RDF quadruple statements using the SmartSUM ontology. Each RDF stream instance was annotated with temporal metadata for compatibility with semantic stream processing.

4.3.3. Experiment Design

Two main experimental runs were conducted, which differ in duration and stream query configurations. The first experimental run was based on a 72 h simulation with a 1 min streaming interval and 10 min query window. The second experimental run was varied to consider a 120 h simulation with a 4 min streaming interval and 20 min query window. In both cases, semantic stream queries were executed at regular intervals with a 2 min sleep duration between executions to emulate real-time continuous processing. Each run involved the validation of the semantic streams using all four RDF serialisation formats (N3, RDF/XML, Turtle and N-Triple). Poor-quality streams were introduced at consistent intervals to evaluate detection sensitivity. The choice of the durations for the query executions caters for an extreme case of a reduced time interval for sensors and IoT streaming data generation within the smart city context. During the reasoning process of the experiments, we based our domain expert knowledge on verified values of sensor readings and the established relationships between carbon monoxide (CO) and Nitrogen Oxides (NO_x)/Nitrogen Dioxides (NO_2), which is defined as inversely proportional, while the relationship between (NO_x) and (NO_2) is directly proportional [41]. The reasoning rule specifically compares the current CO reading with a possible range of consistent values with timestamps, which was determined by the city planner as a certified safe level in quality index. The semantic validation process from the experiments produced a total of 7,657,016 inferred quadruples in the two experimental runs.

5. Results and Discussion

The performance evaluation of the semantic stream validation architecture focused on two core metrics, accuracy in detecting incorrect RDF streams and the reasoning time required for stream validation, across multiple RDF serialisation formats. Four widely used RDF formats—N-Triple (NT), Notation3 (N3), RDF/XML and Turtle—were examined under two streaming configurations involving 1 min and 4 min intervals, each with differing query window durations. Figures 9 and 10 present the experimental outcomes across 350 validation cycles. Each validation cycle contains an average of 36 quadruples resulting from several streaming windows, and they were processed by the semantic stream quality validation approach with the output expressed as a percentage value.

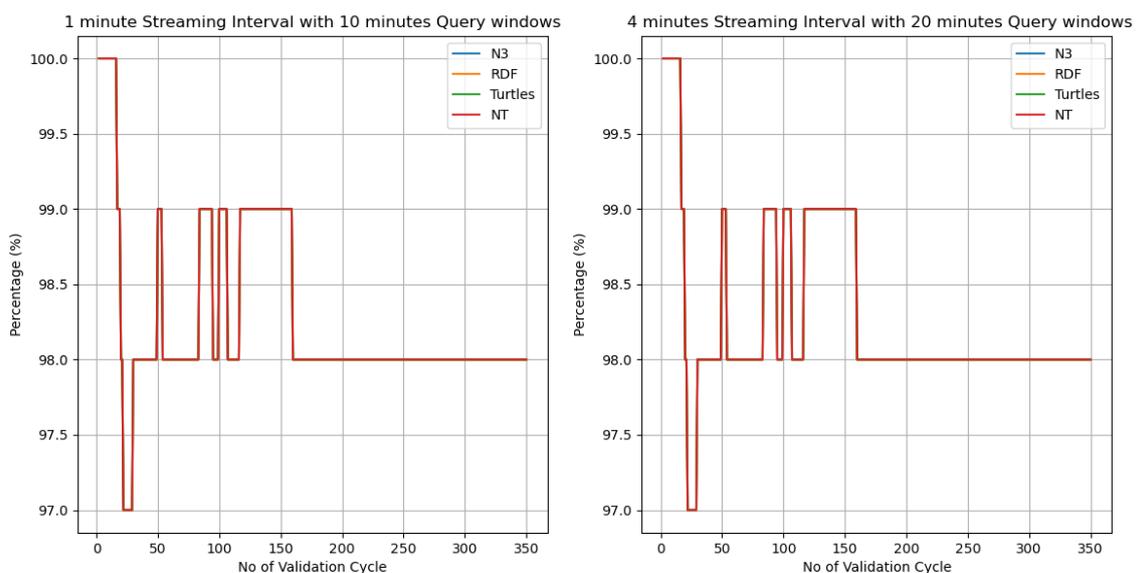


Figure 9. Accuracy of semantic approach over different stream processing intervals.

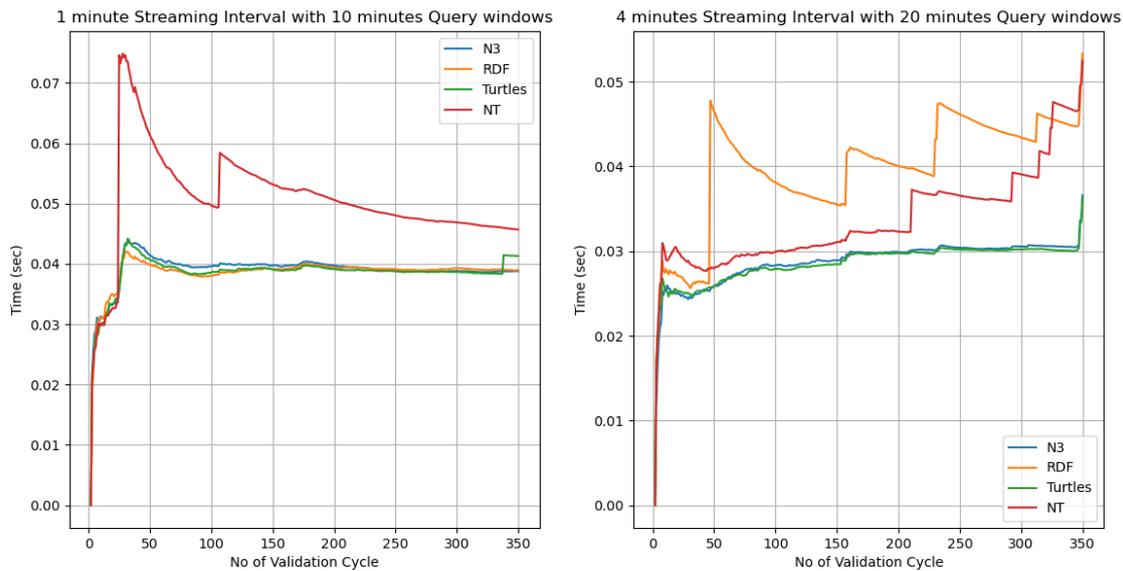


Figure 10. Execution times of semantic reasoning approach.

5.1. Accuracy Analysis

The definition of accuracy in our context conforms to the definition provided in [42], which means identifying the correct and unambiguous sensed values that agree with the actual value of measurement. Therefore, for each validation cycle,

$$Accuracy = 1 - \frac{T_E}{Q_W} \quad (3)$$

where T_E is the number of quadruple statements with one or more incorrect values within the validation cycles, and Q_W represents the total quadruple statements produced within each validation cycle.

The accuracy for both the separate intervals of streaming windows and stream selection duration was computed using Equation (3).

As shown in Figure 9, the non-visible variations for the N3, RDF and Turtle serialised formats may be attributed to consistency by close margins in terms of their percentage of accuracy across the validation cycles. N-Triple consistently demonstrated the highest validation accuracy, peaking at 100% and stabilising between 98 and 99% across both stream configurations. This superior performance is largely attributed to the syntactic simplicity and line-based structure of the N-Triple format, which facilitates efficient parsing and minimises ambiguity during semantic annotation and validation. Each triple is written on a single line, which simplifies the stream processing and aligns well with the sliding-window execution logic of C-SPARQL.

In contrast, the N3, RDF/XML and Turtle formats exhibited comparable accuracy levels but with less visible fluctuation in the plots due to their close margins. Among these, N3 slightly outperformed RDF/XML and Turtle in both configurations. The N3 format, with its concise notation and extended logical expressiveness (e.g., support for formulas and rules), allows for more compact stream representations, which aids in intermediate memory management and efficient query execution under constrained computational settings.

RDF/XML and Turtle, while syntactically richer and more human-readable, introduced slight inconsistencies during high-frequency streaming intervals (notably in the 1 min configuration), likely due to their increased syntactic overhead and multi-line structure, which require more complex parsing operations during stream instantiation.

5.2. Analysis of Reasoning Time

In terms of estimating the reasoning time of the four serialised formats, Figure 10 presents the reasoning times across the serialisation formats and configurations. Clearly, N-Triple initially incurred the highest reasoning time (up to approximately 0.07 s) in the early cycles of the 1 min configuration, attributed to overhead from processing a larger number of individual triples (as each line represents a distinct triple without syntactic compression). However, as the cycles progressed and the system's caching and memory structures stabilised, the reasoning time improved and stabilised around 0.05 s. N3 consistently showed the lowest and most stable reasoning time across both configurations. This suggests that N3 offers a balanced trade-off between expressiveness and parsing efficiency, aided by its support for logic-based constructs that align with the forward-rule inference mechanism used in the reasoning layer. RDF/XML and Turtle exhibited gradual increases in the reasoning time, especially beyond 200 validation cycles in the 4 min streaming configuration. This can be attributed to the increasing complexity of XML-based parsing (in RDF/XML) and the nested structure in Turtle, which tend to scale poorly with higher volumes of streaming data and larger semantic windows. Additionally, these formats require more memory-intensive parsing operations, which, under longer validation cycles, may introduce noticeable latency in reasoning tasks.

The graph also indicates varying the query time with interval sleep time plays a significant role in the semantic reasoning process. The observed variations in accuracy and reasoning time can be theoretically explained by the computational and syntactic characteristics of the RDF formats. N-Triple (NT) is the most machine-friendly due to its minimalistic, line-delimited syntax. Its design prioritises processing speed over compactness or readability, making it ideal for stream-based systems where each statement is treated independently and parsed in a stateless manner. However, its lack of syntactic shortcuts leads to redundancy, increasing payload size and processing time under certain conditions. Notation3 (N3) extends Turtle by supporting logical formulas and inference rules, which complements the forward-chaining reasoning strategy adopted in the architecture. Its concise syntax and logic-native structure reduce the parsing time and enhance performance in rule-based validation contexts. Furthermore, RDF/XML, being a verbose and hierarchical representation, introduces overhead due to the XML parsing complexity and namespace handling. It is less suited for rapid stream ingestion unless XML-optimised parsers or parallel XML processing engines are utilised. Finally, the Turtle serialised format is more compact than RDF/XML and supports readable RDF expression, but its reliance on prefixes and multi-line expressions can introduce additional parsing effort, particularly under large and continuous data streams.

5.3. Implications for Real-World Applications

The results underscore the importance of serialisation format selection based on application-specific constraints, such as high-throughput, latency-sensitive smart city applications. The reasoning approach resonates with a similar semantic stream processing approach based on the use of using an existing reasoner [21] where performance is under limited computational resources. However, the work did not consider the granularity of their approach to specific low-level issues in individual IoT streams. On the other hand, our results show the approach can achieve finer granularity while focusing on individual IoT streaming data and providing actionable insights related to specific stream quality issues. It also suggests a better approach to sensor stream reasoning. It further establishes that the semantic stream processing can still be contained within the interval of IoT streaming data generation of sensor reads in smart city application scenarios, such as air quality monitoring and smart transportation [7,21]. In addition, the approach reinforces its suitability for

timely decision-making in smart safety critical applications especially, as a new evolving era of intelligent-driven smart cities is currently within sight.

6. Conclusions and Future Work

The continuous deployments of sensors and IoT technologies to support the data-driven decisions and actuation in smart city services has identified the need for integrating quality-enriched data from heterogeneous sources and managing the quality requirements of the IoT/sensor nodes. In this paper, we proposed and evaluated a semantic-driven approach for validating IoT streaming data, aimed at enhancing trustworthiness and decision-making in smart city monitoring systems. The architecture integrates semantic stream processing with forward-rule reasoning to detect and isolate inconsistent and incomplete sensor data in near real time. Through software implementation and simulation using a real-world dataset from a busy road in Milan, Italy, we demonstrated the practical feasibility and performance of the approach under urban conditions.

The effectiveness of the proposed approach was validated through a set of controlled experiments involving RDF stream instances derived from real-world air quality and pollution sensor data. The semantic validation process successfully identified up to 99% of incorrect streams across 350 validation cycles, with consistent performance across different RDF serialisation formats. In particular, N-Triple and N3 formats demonstrated high accuracy and stability, while reasoning time remained within sub-second ranges, meeting the low-latency requirements for real-time urban monitoring systems.

Furthermore, the semantic reasoning process exhibited robustness in aligning with known domain rules, such as inverse and direct relationships between pollutants (e.g., CO, NO_x and NO₂), enabling domain-aware validation and inferencing. This capability enhances the quality of situational awareness and enables data-driven actions, such as triggering alerts or recommending traffic or environmental interventions. For example, the detection of hazardous pollutant levels can support decisions to reroute traffic or activate pollution control mechanisms in heavily congested areas.

In conclusion, the experimental outcomes substantiate the viability of the semantic stream validation framework in real-world smart city applications, particularly those requiring high data fidelity, such as air quality monitoring, traffic management and public health alert systems. The architecture's modularity and compatibility with multiple RDF serialisations ensure interoperability across heterogeneous platforms and ease of integration with existing urban infrastructure.

Future developments include enhancement of semantic reasoning to support a more distributed processing and exploring optimisation techniques for NT formats to support high-frequency queries. This will considerably improve the scalability of the semantic streaming data validation process, especially in a high computing environment where constraints on computing resources are highly predominant and inevitable.

Author Contributions: Conceptualization, O.B. and X.L.; methodology, O.B.; software, O.B.; validation, O.B., P.C. and Q.L.; resources, O.B.; writing—original draft preparation, O.B.; writing—review and editing, O.B., P.C. and X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available in this article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Tu, D.Q.; Kayes, A.; Rahayu, W.; Nguyen, K. IoT streaming data integration from multiple sources. *Computing* **2020**, *102*, 2299–2329. [[CrossRef](#)]
2. Sta, H.B. Quality and the efficiency of data in “Smart-Cities”. *Future Gener. Comput. Syst.* **2017**, *74*, 409–416. [[CrossRef](#)]
3. Frank, M.T.; Bader, S.; Simko, V.; Zander, S. Lsane: Collaborative validation and enrichment of heterogeneous observation streams. *Procedia Comput. Sci.* **2018**, *137*, 235–241. [[CrossRef](#)]
4. Byabazaire, J.; O’Hare, G.; Delaney, D. Using Trust as a Measure to Derive Data Quality in Data Shared IoT Deployments. In Proceedings of the 29th International Conference on Computer Communications and Networks (ICCCN), Honolulu, HI, USA, 3–6 August 2020; pp. 1–9. [[CrossRef](#)]
5. Alwan, A.A.; Ciupala, M.A.; Brimicombe, A.J.; Ghorashi, S.A.; Baravalle, A.; Falcarin, P. Data quality challenges in large-scale cyber-physical systems: A systematic review. *Inf. Syst.* **2022**, *105*, 101951. [[CrossRef](#)]
6. Zauli-Sajani, S.; Marchesi, S.; Pironi, C.; Barbieri, C.; Poluzzi, V.; Colacci, A. Assessment of air quality sensor system performance after relocation. *Atmos. Pollut. Res.* **2021**, *12*, 282–291. [[CrossRef](#)]
7. Elsaleh, T.; Enshaeifar, S.; Rezvani, R.; Acton, S.T.; Janeiko, V.; Bermudez-Edo, M. IoT-Stream: A lightweight ontology for internet of things data streams and its use with data analytics and event detection services. *Sensors* **2020**, *20*, 953. [[CrossRef](#)] [[PubMed](#)]
8. Samuelsson, O.; Lindblom, E.U.; Björk, A.; Carlsson, B. To calibrate or not to calibrate, that is the question. *Water Res.* **2023**, *229*, 119338. [[CrossRef](#)] [[PubMed](#)]
9. Sirisha, N.; Gopikrishna, M.; Ramadevi, P.; Bokka, R.; Ganesh, K.; Chakravarthi, M.K. IoT-based data quality and data preprocessing of multinational corporations. *J. High Technol. Manag. Res.* **2023**, *34*, 100477. [[CrossRef](#)]
10. Barbieri, D.F.; Braga, D.; Ceri, S.; Valle, E.D.; Grossniklaus, M. Querying rdf streams with c-sparql. *ACM SIGMOD Rec.* **2010**, *39*, 20–26. [[CrossRef](#)]
11. Byabazaire, J.; O’Hare, G.M.; Collier, R.; Delaney, D. Iot data quality assessment framework using adaptive weighted estimation fusion. *Sensors* **2023**, *23*, 5993. [[CrossRef](#)]
12. Buelvas, H.P.J.; Buelvas, H.P.J.; Avila, E.B.F.; Avila, E.B.F.; Gaviria, G.N.; Gaviria, G.N.; Múnera, D.; Munera, A.R.D. Data Quality Estimation in a Smart City’s Air Quality Monitoring IoT Application. In Proceedings of the 2021 2nd Sustainable Cities Latin America Conference (SCLA), Medellin, Colombia, 25–27 August 2021. [[CrossRef](#)]
13. Bamgboye, O.; Liu, X.; Cruickshank, P. Towards Modelling and Reasoning About Uncertain Data of Sensor Measurements for Decision Support in Smart Spaces. In Proceedings of the IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan, 23–27 July 2018; pp. 744–749.
14. Mante, S.; Mante, S.; Hernandez, N.; Hernandez, N.; Hussain, A.; Hussain, A.M.; Chaudhari, S.; Chaudhari, S.; Gangadharan, D.; Gangadharan, D.; et al. 5D-IoT, a semantic web based framework for assessing IoT data quality. In Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, 25–29 April 2022. [[CrossRef](#)]
15. Du, Y.; Wang, Z.; Leung, C.; Victor, L. Blockchain-based Data Quality Assessment to Improve Distributed Machine Learning. In Proceedings of the 2023 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 20–22 February 2023; pp. 170–175.
16. Martín, L.; Sánchez, L.; Lanza, J.; Sotres, P. Development and evaluation of Artificial Intelligence techniques for IoT data quality assessment and curation. *Internet Things* **2023**, *22*, 100779. [[CrossRef](#)]
17. Ding, X.; Wang, H.; Li, G.; Li, H.; Li, Y.; Liu, Y. IoT data cleaning techniques: A survey. *Intell. Converge. Netw.* **2022**, *3*, 325–339. [[CrossRef](#)]
18. Risdiana, D.M.; Susanto, T.D. The Safe City: Conceptual Model Development—A Systematic Literature Review. *Procedia Comput. Sci.* **2019**, *161*, 291–299. [[CrossRef](#)]
19. Luckey, D.; Fritz, H.; Legatiuk, D.; Dragos, K.; Smarsly, K. Artificial intelligence techniques for smart city applications. In Proceedings of the 18th International Conference on Computing in Civil and Building Engineering: ICCCBE 2020, São Paulo, Brazil, 18–20 August 2020; Springer: Berlin/Heidelberg, Germany, 2021; pp. 3–15.
20. Li, W.; Song, H.; Zeng, F. Policy-based secure and trustworthy sensing for internet of things in smart cities. *IEEE Internet Things J.* **2017**, *5*, 716–723. [[CrossRef](#)]
21. D’Aniello, G.; Gaeta, M.; Orciuoli, F.; Simonetti, A. An approach based on semantic stream reasoning to support decision processes in smart cities. *Telemat. Inform.* **2018**, *35*, 1795. [[CrossRef](#)]
22. Chhetri, T.R.; Dehury, C.K.; Varghese, B.; Fensel, A.; Srirama, S.N.; DeLong, R.J. Enabling privacy-aware interoperable and quality IoT data sharing with context. *Future Gener. Comput. Syst.* **2024**, *157*, 164–179. [[CrossRef](#)]
23. Quek, H.Y.; Sielker, F.; Akroyd, J.; Bhave, A.N.; von Richthofen, A.; Herthogs, P.; van der Laag Yamu, C.; Wan, L.; Nocht, T.; Burgess, G.; et al. The conundrum in smart city governance: Interoperability and compatibility in an ever-growing ecosystem of digital twins. *Data Policy* **2023**, *5*, e6. [[CrossRef](#)]
24. Rubí, J.N.S.; de Lira Gondim, P.R. IoT-based platform for environment data sharing in smart cities. *Int. J. Commun. Syst.* **2021**, *34*, e4515. [[CrossRef](#)]

25. Pliatsios, A.; Kotis, K.; Goumopoulos, C. A systematic review on semantic interoperability in the IoE-enabled smart cities. *Internet Things* **2023**, *22*, 100754. [CrossRef]
26. Bianchini, D.; De Antonellis, V.; Garda, M.; Melchiori, M. Smart city data modelling using semantic web technologies. In Proceedings of the 2021 IEEE International Smart Cities Conference (ISC2), Manchester, UK, 7–10 September 2021; pp. 1–7.
27. Paulus, A.; Burgdorf, A.; Langer, T.; Pomp, A.; Meisen, T.; Pol, S. PLASMA: A Semantic Modeling Tool for Domain Experts. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 4946–4950.
28. Ghorbani, A.; Zamanifar, K. Type-2 fuzzy ontology-based semantic knowledge for indoor air quality assessment. *Appl. Soft Comput.* **2022**, *121*, 108658. [CrossRef]
29. Alirezaie, M.; Loutfi, A. Reasoning for sensor data interpretation: An application to air quality monitoring. *J. Ambient. Intell. Smart Environ.* **2015**, *7*, 579–597. [CrossRef]
30. Sejdiu, B.; Ismaili, F.; Ahmed, L. A Real-Time Integration of Semantic Annotations into Air Quality Monitoring Sensor Data. In Proceedings of the Software Technologies; van Sinderen, M., Maciaszek, L.A., Fill, H.G., Eds.; Springer: Cham, Switzerland, 2021; pp. 98–113.
31. Kumar, S.S.; Chandra, R.; Agarwal, S. Rule based complex event processing for an air quality monitoring system in smart city. *Sustain. Cities Soc.* **2024**, *112*, 105609. [CrossRef]
32. Zygiaris, S. Smart city reference model: Assisting planners to conceptualize the building of smart city innovation ecosystems. *J. Knowl. Econ.* **2013**, *4*, 217–231. [CrossRef]
33. Haller, A.; Janowicz, K.; Cox, S.J.; Lefrançois, M.; Taylor, K.; Le Phuoc, D.; Lieberman, J.; García-Castro, R.; Atkinson, R.; Stadler, C. The SOSA/SSN ontology: A joint WeC and OGC standard specifying the semantics of sensors observations actuation and sampling. In *Semantic Web*; IOS Press: Amsterdam, The Netherlands, 2018; Volume 1, pp. 1–19.
34. Zhou, Q.; Fikes, R. A reusable time ontology. In Proceedings of the AAAI Workshop on Ontologies for the Semantic Web, Edmonton, AB, Canada, 28 July–1 August 2002.
35. Silva-López, R.B.; Méndez-Gurrola, I.I.; Pablo-Leyva, H. Comparative Methodologies for Evaluation of Ontology Design. In *Proceedings of the Advances in Computational Intelligence*; Martínez-Villaseñor, L., Herrera-Alcántara, O., Ponce, H., Castro-Espinoza, F.A., Eds.; Springer: Cham, Switzerland, 2020; pp. 92–102.
36. Salamon, J.S.; Barcellos, M.P. Towards a Framework for Continuous Ontology Engineering. In Proceedings of the ONTOBRAS, Victoria, ES, Brazil, 22–25 November 2022; pp. 158–165.
37. Tayur, V.M.; Suchithra, R. Multi-ontology mapping generative adversarial network in internet of things for ontology alignment. *Internet Things* **2022**, *20*, 100616. [CrossRef]
38. Varró, G.; Deckwerth, F. A rete network construction algorithm for incremental pattern matching. In Proceedings of the Theory and Practice of Model Transformations: 6th International Conference, ICMT 2013, Budapest, Hungary, 18–19 June 2013; Proceedings 6; Springer: Cham, Switzerland, 2013; pp. 125–140.
39. Vito, S. Air Quality. UCI Machine Learning Repository. 2008. Available online: <https://archive.ics.uci.edu/dataset/360/air+quality> (accessed on 10 April 2025).
40. Scuto, V.; Ferraris, A.; Bresciani, S. Internet of Things: Applications and challenges in smart cities: A case study of IBM smart city projects. *Bus. Process. Manag. J.* **2016**, *22*, 357–367. [CrossRef]
41. Ravina, M.; Caramitti, G.; Panepinto, D.; Zanetti, M. Air quality and photochemical reactions: Analysis of NO_x and NO₂ concentrations in the urban area of Turin, Italy. *Air Qual. Atmos. Health* **2022**, *15*, 541–558. [CrossRef]
42. Schwabe, D.; Becker, K.; Seyferth, M.; Klaß, A.; Schaeffter, T. The METRIC-framework for assessing data quality for trustworthy AI in medicine: A systematic review. *NPJ Digit. Med.* **2024**, *7*, 203. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.