UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Jefferson do Nascimento Amará

Critical Event Prediction using Ontology-based Sensor Data Integration and Deep Learning

Juiz de Fora
2024

Jefferson do Nascimento Amará

Critical Event Prediction using Ontology-based Sensor Data Integration and Deep Learning

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação. Área de concentração: Ciência da Computação.

Orientador: Prof. Dr. Victor Ströele de Andrade Menezes

Coorientadora: Prof. D.Sc. Regina Maria Maciel Braga

Juiz de Fora

2024

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Amará, Jefferson do Nascimento.
Critical Event Prediction using Ontology-based Sensor Data Integration and Deep Learning / Jefferson do Nascimento Amará. -- 2024.
97 f. : il.

Orientador: Victor Ströele de Andrade Menezes
Coorientadora: Regina Maria Maciel Braga
Dissertação (mestrado acadêmico) - Universidade Federal de Juiz de Fora, ICE/Engenharia. Programa de Pós-Graduação em Ciência da Computação, 2024.

1. Data Integration. 2. Ontology. 3. Deep Learning. 4. Sensors. 5. Event Prediction. I. Menezes, Victor Ströele de Andrade, orient. II. Braga, Regina Maria Maciel, coorient. III. Título.

Jefferson do Nascimento Amará

Critical Event Prediction using Ontology-based Sensor Data Integration and Deep Learning

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação. Área de concentração: Ciência da Computação.

Aprovada em 27 de agosto de 2024.

Juiz de Fora, 14/08/2024.

AGRADECIMENTOS

Agradeço a Deus, a explicação mais lógica para tudo o que existe e cuja presença percebo em cada um dos meus dias. À minha família, em especial à minha mãe, a quem devo a força e os valores que carrego comigo até aqui. À minha noiva, amiga e companheira, que me motiva e apoia em todos os momentos, e à sua família, que por extensão também se torna a minha. Aos meus pets, um presente de Deus nos meus dias. Ao meu amigo e orientador, um exemplo de profissional, professor, pesquisador e ser humano, que me apoia, ensina, questiona, critica, aponta caminhos e me orienta em como percorrê-los. À minha co-orientadora, por sempre provocar reflexões, tirar-me da zona de conforto e me fazer ser hoje alguém melhor do que fui ontem. Aos meus amigos, pelo companheirismo e momentos compartilhados. Ao meu trabalho, essa entidade abstrata que se personifica nos meus líderes e pares, ampliando meus horizontes e me provendo o pão de cada dia. À UFJF, minha segunda casa já há alguns anos, que expande meus horizontes, cria oportunidades e proporciona, com excelência, uma educação e formação técnica, profissional, criativa, social e humana. Agradeço aos professores, meus amigos, mestres e doutores que, com muita humildade e competência, desde a graduação, compartilham o conhecimento a fim de estender os limites da ciência e transformar a sociedade. A todos os demais profissionais da UFJF que me acompanham a cada dia, dividindo o dia a dia e dedicando seu tempo e esforço para me apoiar na realização deste sonho. Obrigado.

“Não amo a espada brilhante por sua agudeza, nem a flecha por sua rapidez, nem o guerreiro por sua glória. Só amo aquilo que eles defendem.” (TOLKIEN, 1954, p. 649). *O Senhor dos Anéis: As Duas Torres*

RESUMO

A detecção e predição de eventos desempenham um papel crucial em diversos domínios, especialmente em sistemas de alerta precoce, onde a identificação de eventos críticos com antecedência pode prevenir desastres e mitigar riscos. Esta pesquisa aborda o problema da integração de dados de sensores heterogêneos, comum em áreas como gestão de desastres e cidades inteligentes, onde dados brutos podem não fornecer contexto suficiente para uma tomada de decisão eficaz. Para resolver esse desafio, propomos um framework que combina uma abordagem baseada em ontologia para integrar e contextualizar os dados de sensores com técnicas de aprendizado profundo para a predição de eventos. O framework utiliza um modelo de Ontologia de Rede de Sensores Semânticos (SSN) estendido para estruturar e enriquecer os dados com informações semânticas e de contexto. Em seguida, aplicamos modelos de redes neurais recorrentes de memória de longo prazo (LSTM) para realizar a predição de eventos com base nos dados integrados. Um estudo de caso foi realizado no domínio hidrológico, utilizando dados reais de sensores hidrométricos e hidrológicos para prever eventos críticos como enchentes. Os resultados demonstraram que a combinação da integração semântica com técnicas de IA melhora a acurácia das predições e permite a identificação antecipada de eventos críticos que poderiam ser perdidos com dados isolados.

ABSTRACT

Event detection and prediction play a crucial role in various domains, particularly in early warning systems where identifying critical events in advance can prevent disasters and mitigate risks. This research addresses the problem of integrating heterogeneous sensor data, which is common in fields such as disaster management and smart cities, where raw data may not provide sufficient context for effective decision-making. To address this challenge, we propose a framework that combines an ontology-based approach to integrate and contextualize sensor data with deep learning techniques for event prediction. The framework utilizes an extended Semantic Sensor Network (SSN) ontology to structure and enrich the data with semantic and contextual information. Then, Long Short-Term Memory (LSTM) neural networks are applied to predict events based on the integrated data. A case study was conducted in the hydrological domain, using real-world hydrometric and hydrological sensor data to predict critical events such as floods. The results demonstrate that combining semantic integration with AI techniques improves prediction accuracy and enables the early detection of critical events that might be missed when using isolated data.

Keywords: Data Integration; Sensors; Ontology; Deep Learning; Event Prediction.

List of Figures

List of Tables

ACRONYMS

IoT	Internet of Things
AI	Artificial Intelligence
LCS	Local Concept Schema
W3C	World Wide Web Consortium
OGC	Open Geospatial Consortium
SSN	Semantic Sensor Network
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
SVM	Support Vector Machine
SSIO	Synchronized Sequential Input and Output
SISO	Sequential Input and Single Output
SWRL	Semantic Web Rule Language
WFV	Walk Forward Validation
RQ	Research Question
SRQ	Secondary Research Question
SRQ1	First Secondary Research Question
SRQ2	Second Secondary Research Question
GQM	Goal Question Metric
M1	Metric 1
M2	Metric 2
M3	Metric 3
GFDS	Global Flood Detection System
ARIMA	Autoregressive Integrated Moving Average
SARIMA	Seasonal Autoregressive Integrated Moving Average
REST	Representational State Transfer
API	Application programming Interface
GPU	Graphics Processing Unit
CQ	Competency Question

LIST OF SYMBOLS

x_t	Input at time step t
h_t	Hidden state at time step t
c_t	Cell state at time step t
i_t	Input gate at time step t
f_t	Forget gate at time step t
o_t	Output gate at time step t
g_t	Candidate cell state at time step t
W	Weight matrix
b	Bias vector
σ	Sigmoid function
$\tanh$	Hyperbolic tangent function
$\odot$	Element-wise multiplication
$+$	plus
$=$	is equal to
$*$	multiplication
S	The set of Sensors in the domain of interest
O_{s_i}	The set of Observations performed by each sensor s_i
A_{o_k}	The set of Attributes associated with each observation o_k
ϵ_{s_i}	The set of Critical Events associated with a sensor s_i
o_k	An Observation
ϵ_r	A Critical Event
M	A subset of o_k
$\forall$	for all
$\exists$	there exists
$\subset$	is a subset of
$\leq$	less than or equal to
$\geq$	greater than or equal to
$\wedge$	logical AND
$\rightarrow$	implies
$?p1, ?p2$	Variables representing platforms
$?$	Variable representing sensors
$?o$	Variable representing observations
r	Variable representing results
vop	Variable representing the value of precipitation
tp	Variable representing the threshold precipitation value
$vowl$	Variable representing the value of water level
twl	Variable representing the threshold water level value

Contents

1 Introduction

For event detection and prediction, contextualized data is a crucial element (26). In the Internet of Things (IoT) and big data era, multiple and heterogeneous data sources are common, and integrating them is an important task for establishing data context (73). Concerning sensor data sources, the integration of data for event prediction has garnered the attention of numerous researchers, who have proposed solutions and applications across various fields, including digital health, smart cities, logistics and supply chain, environmental risk, and disaster prevention (40, 50, 75, 14).

Event prediction involves detecting events of interest based on data patterns and contextual information. It involves leveraging advanced inference algorithms and Artificial Intelligence (AI) techniques to analyze historical data, identify trends, and predict future occurrences. In the context of sensor data sources, event prediction aims to foresee specific events or anomalies, allowing for proactive decision-making and timely responses. Numerous approaches to address this concern can be found in the literature (39, 74, 70, 48, 46, 65).

Zhao (2021) presents a comprehensive survey that reviews the most common methods for handling event prediction using sensor data. The study provides detailed insights into various approaches and highlights their respective strengths and weaknesses. The limitation and challenge observed in many existing studies is the absence of solutions that effectively consider data from diverse, integrated, and contextualized sources before applying intelligent methods (81). As a result, event predictions are often confined to a single data source, leading to incomplete insights and hindering the holistic view of predictions (53).

Furthermore, event prediction systems can enhance their ability to detect events by considering a network of integrated sensor data (4). As highlighted by (47, 67, 20, 19, 14, 51), the predicted events may exhibit correlations and mutual influences. In Zhao(2021), authors further assert the existence of intricate dependencies among events.

1.1 Proposal

Drawing upon the key insights, limitations, challenges, and suggested avenues identified in a Secondary Study conducted to comprehend the state-of-the-art and identify gaps and paths in semantic data integration, event detection, and prediction the following research question was formulated:

RQ – Can the framework semantically integrate data from multiple data sources to detect and predict events of interest from these data?

In order to answer this research question, this work proposes a framework to process, integrate, contextualize data, and also to detect, correlate, and predict events from multiple data sources. Ultimately, we aim to identify events of interest based on correlated events detected in integrated data, which might remain undetected or experience delayed detection when using a single data source.

To accomplish this, we utilized a Local Concept Schema (LCS) in conjunction with a Global Conceptual Schema (GCS), structured as an ontology, and integrated AI models. This methodology enabled us to consolidate data from diverse sources. Our system was designed to detect individual events based on data from specific sources and identify events by analyzing context across multiple data sources. By employing ontological terms and rules, we were able to extract semantic meanings and propose a unified context for the detected events.

The methodological process encompassed five main steps:

- A Secondary Study was conducted to comprehend the state-of-the-art and identify gaps and paths in semantic data integration, event detection, and prediction.
- An ontology model was developed for data integration, context building, and event detection.
- An artificial intelligence model was implemented for event prediction.
- An evaluation phase was conducted to assess the effectiveness of the proposed framework in integrating, detecting, and predicting events of interest.

The framework comprising the ontology and AI models was applied and assessed using real data from hydrometric and hydrological sensor stations. We define the ontology as a standardized representation for integrating data from diverse sources. The ontology also established context among these sources and detected events of interest. An AI model based on the integrated data predict events on these sources. As a result, we initially contextualized and integrated data from these sensors, subsequently detecting and predicting flood events. The contexts were constructed based on the ontology model, while the predictions relied on the AI model. Shared contexts were identified based on location and flow direction among these stations, illustrating that events occurring at one station can influence events at another station.

Developing a framework is justified in our proposal as, in computer science, a framework is defined as a structured environment that provides reusable components and standardized guidelines for software development(35). It serves as a foundational structure, enabling developers to build applications more efficiently by abstracting common functionalities, promoting code reuse, and enforcing architectural patterns that seek consistency and scalability. By offering these predefined solutions, a framework simplifies the development process, allowing developers to focus on specific aspects of the application while relying on the framework for core functionalities.

Figure 1 highlights the challenges of developing our work along with the proposed analytics process. These challenges are addressed during the present study. First, we model each reading from the sensor as a unique data instance. Syntactic information is derived from a variety of data collected by hydrometric and hydrological sensors, which are then structured into a common representation using ontology. Ontologies, as logical models, explicitly define the meaning (semantics) and relationships between concepts. Our objective is to identify each data source and represent the associated data using a canonical ontology model, facilitating the seamless integration of new and diverse sensors.

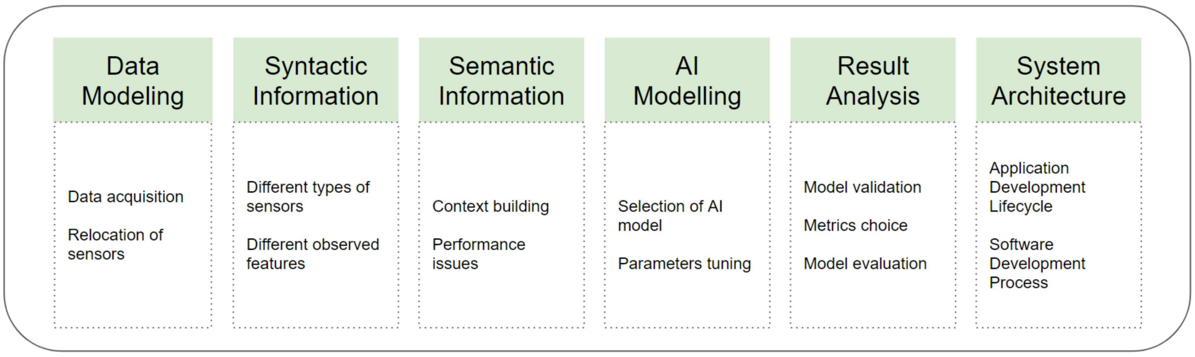


Figure 1 – Research challenges.

Semantic information enables the extraction of meaning from data. When combined with syntactic information, semantic-based methodologies empower the framework to explore the context of analysis comprehensively, extracting insights from semantic structures. To achieve this, we have enhanced the ontology model with contextual rules to define the relationships among stations and sensor data, categorizing each data instance based on predefined threshold values. This allows the detection of events of interest within the observed domain, such as flood events.

AI modeling entails selecting the most suitable model based on the characteristics of the analyzed data. Factors such as the nature and quantity of the data and statistical characteristics like seasonality are considered. Once the optimal model is identified, adjustments to its parameters are necessary to mitigate biases, enhance computational performance, and improve the accuracy of predictions.

Following the implementation and fine-tuning of the model, efforts are directed towards result analysis to validate the obtained results. This involves utilizing metrics that best align with the model and the problem being studied, ensuring the robustness of the conclusions drawn.

Finally, throughout the entire implementation life cycle, we address questions regarding the software development process and software implementation cycle. This is aimed at selecting the better choice system and architectural design to effectively handle the data and achieve the desired solution behavior once we have a high-throughput data flow.

In summary, the solution makes the following contributions: (i) allows the integration of data from different sensors, (ii) provides the abstraction of sensor data structure, allowing add new data sources, (iii) relates events from different sensors, (iv) detect events of interest based on ontology definitions, (v) predict events of interest based on context and AI patterns recognition.

This dissertation is organized as follows: Chapter 2 presents concepts about data integration, event prediction, and related work. Chapter 3 introduces the architecture of the proposed solution, including explaining each step involved in the approach, and describes the data, ontology, and AI models used in this study. Chapter 4 delves into the pre-processing layer, result analysis, AI model definition, implementation, and experimental results. Chapter 5 presents the result analysis, including semantic integration performance, AI model performance, and the impact of context rules. Finally, Chapter 6 presents conclusions and future works.

The development of this work resulted in two published papers (4)(5) and ongoing submissions.

All code and data used in this work can be accessed at Bitbucket project[1] and Bitbucket Project[2].

[1] https://bitbucket.org/JeffersonAmara/obiwan/src/main/
[2] https://bitbucket.org/JeffersonAmara/ontology-based-integration-system/src/master/

2 Background and Related Works

Considering the research area of this dissertation, a secondary study was carried out, to assess how the literature discusses the topic. We conducted a search considering scientific articles that discuss processes for semantic sensor data integration, event detection, and event prediction. We combined searches in digital libraries and then processed snowballing forward (SF). For this, the searches were carried out in the Scopus digital library and Google Scholar[1].

We searched also for related papers dealing with flood event prediction once our feasibility study relied on data from hydrometric and hydrological stations. Therefore, the secondary study aimed to identify specific studies to help understand the state-of-art (37).

2.1 Data Integration and Event Prediction

Data integration is critical in managing the diversity and abundance of available information. It offers a strategic approach to unify disparate sources, improve data quality, enrich context, and facilitate Big Data analysis. This integration not only supports informed decision-making but also enhances the effectiveness of AI systems (42). Data integration contributes to data accuracy by providing a holistic view, empowering organizations and individuals to extract meaningful insights from the extensive data (73).

However, this process of data integration is not without its challenges. The data landscape presents obstacles such as managing source diversity (databases, applications, cloud, social networks, and IoT), handling massive data volumes, adapting to various data formats, and ensuring real-time integration for decision-making (53).

In this complex data landscape, ontologies emerge as a potential solution (6). In computer science, an ontology is defined as a formal representation of a set of concepts within a domain and the relationships between those concepts (28). Additionally, an ontology is considered a logical theory that describes the intended meaning of a formal vocabulary. This description forms the ontology's commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary are constrained by this ontological commitment, which means that an ontology indirectly reflects this commitment by approximating these intended models.

Practically speaking, an ontology includes a vocabulary, which is a set of terms describing the domain, concepts, which are entities or classes within the domain, relationships, which are the associations between the concepts, and axioms, which are rules that define the usage boundaries of the terms and the logic governing the relationships between them (29).

[1] https://scholar.google.com/

This structure allows the creation of ontology-based information systems, where systems are designed and operate according to the concepts and relationships defined in the ontology, facilitating information integration, knowledge reuse, and interoperability between different systems (58).

Ontologies also aid in defining semantic relationships between disparate datasets, contributing to standardization and semantic clarity. They address the challenge of heterogeneity by harmonizing data with diverse structures and meanings (50).

A highlighted example of data heterogeneity is data produced by sensors. A sensor is a device that detects and responds to some type of input from the physical environment. The specific input could be light, heat, motion, moisture, pressure, or any other environmental phenomenon. The output is generally a signal that is converted to a readable display or transmitted for further processing (59, 52). In sensor data, which includes information captured by IoT devices and environmental sensors, ontologies play a crucial role in overcoming the diversity in formats, structures, and contexts.

Sensor data integration goes beyond physical unification; it involves semantic harmonization to ensure a consistent understanding of the data (40). Applying ontologies in sensor data integration allows for creating semantic structures defining and relating underlying concepts (75, 6). The World Wide Web Consortium (W3C) and Open Geospatial Consortium (OGC) propose the Semantic Sensor Network (SSN) as an ontology standardization for sensor data, formalizing the representation of terms and concepts (15).

The SSN ontology is used to represent and describe sensors, their observables, the observations they make, and the entities involved in a sensor network environment. It is part of a group of ontologies aimed at the Semantic Web. It is designed to be modular and flexible, allowing detailed representation of various aspects of sensors and their operations. It includes several main components: the sensor device, which encompasses information about the sensor's capabilities, functionality, and physical characteristics; the observation, which represents the act of measuring or observing a phenomenon, including details about what is being observed, when, and how the observation was made; the entity, which refers to the object or event being observed by the sensor; the property, which is the characteristic or attribute of the observed phenomenon, such as temperature, humidity, or pressure; and the measurement system, which includes the methods and processes used to carry out the measurements(15).

The SSN ontology utilizes standards from the Semantic Web, such as RDF (Resource Description Framework) and OWL (Web Ontology Language), to define a common vocabulary that can be used to describe sensors and their observations in an interoperable manner. The main components of the ontology include the sensor, which is a device performing observations (e.g., a digital thermometer); the observation, which is an instance of a sensor collecting data (e.g., a temperature reading at a specific time); the observable property, which is the characteristic of the phenomenon being measured (e.g., air temperature); the result, which is the value or set of values obtained from the observation (e.g., 25°C). Figure 2 shows the overview of the SSN classes and properties based on the Observation perspective.

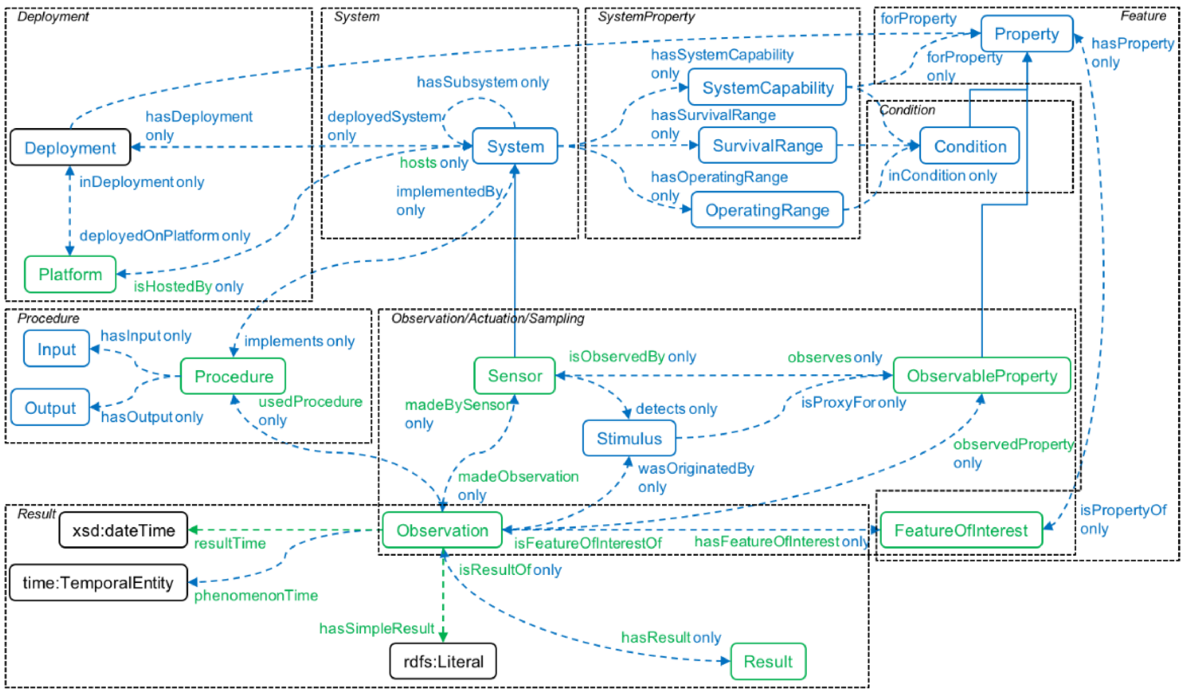


Figure 2 – SSN classes and properties. Source:(15)

Examples of the SSN ontology's application include environmental monitoring, where temperature and humidity sensors can be described using SSN to monitor environmental conditions in a forest; smart cities, where sensors in urban infrastructure (such as traffic cameras and pollution sensors) can be described and integrated using the SSN ontology to improve city management; and healthcare, where medical devices like glucose monitors can be represented using SSN to track and analyze patient health data (57, 30).

This standardization offers several advantages, including interoperability, which allows the integration of sensor data from different manufacturers and systems, facilitating the exchange and use of information; reusability, where ontology components can be reused in different applications and domains; and semantic richness, providing detailed and semantic descriptions of sensors and their observations, enhancing the understanding and analysis of the collected data(62).

To achieve an ontology representation, getting data from various sensor types and data sources, an effective approach is to use a Local Conceptual Schema (LCS). The LCS is a set of rules that define the mapping from data sources to a final representation and is useful for handling the lexical and syntactic aspects of data integration, providing an intermediate layer that facilitates the translation between raw sensor data representation and the ontological representation (60).

Using an LCS helps to deal with data heterogeneity, source autonomy, and the need to reconcile different data schemas, offering a common framework that standardizes data semantics regardless of their origin (17, 43). This approach is particularly useful in environments where data is collected from multiple sensors with varied formats and communication protocols.

Figure 3 schematically illustrates the process. Multiple data sources provide input to a wrapper component. The wrapper consults the mapping rules defined in each corresponding LCS and transforms the data into a Global Conceptual Schema (GCS). This GCS can be implemented using an ontological representation, seeking a unified and coherent view of the integrated data.

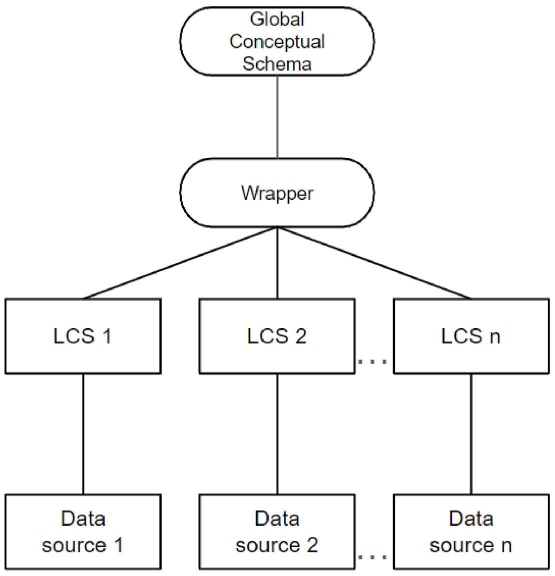


Figure 3 – LCS mapping schema.

By using an LCS, the system can manage the complexity of integrating heterogeneous data sources, ensuring that the data is both accurate and semantically meaningful when represented in the GCS. This facilitates better data interoperability and consistency across the system, addressing the core challenges of data integration(60).

A natural aspect of sensor data integration is the incorporation of Artificial Intelligence, particularly predictive capabilities. When combined with ontologies that semantically structure sensor data, advanced machine learning algorithms enable meaningful predictions(14). Recurrent Neural Networks (RNNs), in particular, are designed to process sequential data such as time series, where current data depends on previous information(78).

RNNs are a class of neural networks suitable for sequential data as they have internal loops that allow information persistence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs, making them ideal for tasks such as time series prediction, automatic translation, and speech recognition (49).

The basic architecture of an RNN consists of layers of neurons where the output of one layer is fed back as input to the same layer at the next time step. Mathematically, for a sequence of input $x = (x_1, x_2, ..., x_T)$, the RNN updates its hidden state h_t using the formula:

$$h_t = \tanh(W_{hx}x_t + W_{hh}h_{t-1} + b_h)$$

where W_{hx} and W_{hh} are weight matrices, b_h is the bias, and tanh is the hyperbolic tangent activation function (49).

LSTM networks, a variant of RNNs, were designed to overcome the limitations of traditional RNNs, such as the vanishing and exploding gradient problems. These problems occur during the training of deep neural networks when gradients used for updating weights either become too small (vanishing) or too large (exploding), leading to inefficient learning or numerical instability. LSTMs introduce memory cells that can maintain information for long periods and are controlled by three types of gates: input gate, forget gate, and output gate. The input gate regulates the amount of new information added to the cell state, the forget gate controls the retention of existing information, and the output gate determines the extent to which the cell state influences the output (31). These mechanisms enable LSTMs to preserve and utilize relevant information over extended sequences, effectively mitigating the vanishing and exploding gradient issues.

Figure 4 illustrates the internal architecture of an LSTM cell. It has three main gates: the forget gate (f_t), the input gate (i_t), and the output gate (o_t), all controlled by sigmoid functions. The forget gate determines how much of the previous cell state (c_{t-1}) should be retained. The input gate regulates the extent to which new information (x_t) will be stored in the cell state (c_t), processed through a tanh function to generate candidate values (g_t). The output gate controls how much of the internal cell state (c_t) is used to compute the cell output (h_t), also passing through a tanh function to normalize the values. These operations allow the LSTM cell to retain relevant information over long periods, addressing the limitations of traditional RNNs (31). The updates are given by the equations next, where σ is the sigmoid function and $\odot$ denotes element-wise multiplication:

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o)$$

$$g_t = \tanh(W_{gx}x_t + W_{gh}h_{t-1} + b_g)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$

$$h_t = o_t \odot \tanh(c_t)$$

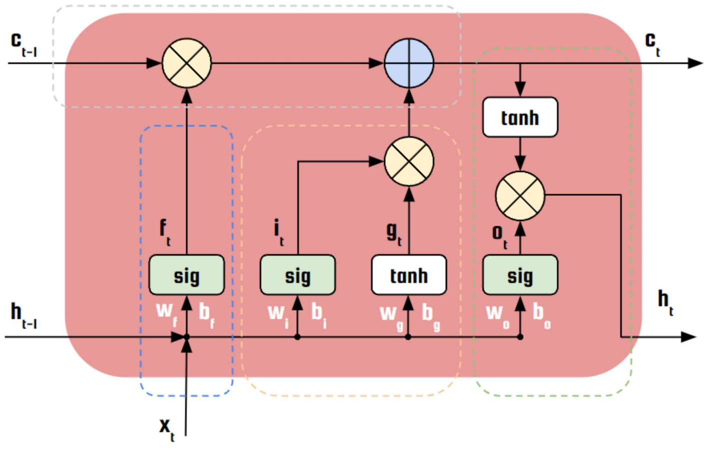


Figure 4 – LSTM cell model.

The main hyperparameters of an LSTM network include the number of LSTM layers, the number of units in each layer, the learning rate, the batch size, and the number of training epochs. Fine-tuning these hyperparameters is crucial to optimize the model's performance. Techniques such as grid search and random search are commonly used to explore different combinations of hyperparameters (8).

To prevent overfitting, a condition where a machine learning model learns the training data too well, capturing noise and specific patterns rather than generalizing to new data, regularization techniques such as dropout are often employed. Dropout works by randomly deactivating a fraction of neurons during training, preventing the model from becoming overly fitted to the training data. Additionally, L2 regularization, which adds a penalty proportional to the square of the magnitude of the weight coefficients to the loss function, is a common practice. This helps to keep the weights small and the model simpler, thereby promoting better generalization (71).

This predictive capability enhances operational efficiency and allows for implementing preventive measures in response to anticipated events. The synergy between semantically integrated data sources, facilitated by ontologies, and the predictive capabilities of AI, exemplified using LSTM networks for sensor data, presents a powerful approach to navigating the challenges of the current Data Era. This combination promotes a unified understanding of diverse data, empowering organizations and individuals to make informed decisions and proactively respond to future events.

Several compelling reasons drive the decision to evaluate the framework using data from hydrometric and hydrological stations. Hydrological data provide a rich and complex dataset characterized by time-series observations, environmental variables, and spatial dependencies, making it ideal for testing the robustness and versatility of our framework.

Hydrological data encompass various attributes such as water levels, flow rates, and precipitation measurements. This complexity requires data integration and predictive capabilities, which our framework aims to address. According to research, data-driven hydrological models, especially those incorporating machine learning techniques, are increasingly being utilized to handle these complexities and provide accurate predictions (42, 55).

Accurate forecasting and event detection related to water levels and precipitation can significantly contribute to disaster prevention and management, such as flood warnings and water resource management(13, 10). This underscores the practical utility of our framework.

2.2 Related Work

We selected five studies relying on these topics. Two of them bring directions, patterns, and ongoing strategies on event prediction methods (26, 81). One focuses on edge technologies for disaster management (1). Finally, another two of them propose solutions for sensor network AI-based disaster detection (3, 46).

Hydrological stations typically have extensive historical records, providing a reliable basis for evaluating the performance and accuracy of predictive models. Studies have shown that the availability of long-term data enhances the development and validation of predictive models, ensuring they are robust and reliable (42). Also, hydrometric and hydrological data often involve challenges such as missing values, noise, and temporal correlations. Addressing these issues is crucial for accurate modeling and prediction. Research indicates that advanced machine learning techniques, such as LSTM networks and hybrid models, are effective in managing these complexities, thereby validating the need for robust frameworks like ours (42, 55).

By focusing on hydrometric and hydrological data, we aim to showcase the framework's ability to deliver meaningful insights and actionable predictions, highlighting its potential applications in various environmental and engineering fields. This choice underscores the framework's relevance and adaptability to domains where precise data integration and forecasting are crucial.

In Gmati et al. (2019), the study identifies the main mature and classical approaches to event prediction, presenting a comprehensive taxonomy that transcends specific application domains. This interdisciplinary perspective enables a deeper understanding of event prediction challenges and facilitates the development of new techniques. Our proposal aligns with the category of quantitative inferential approaches described by Gmati et al., utilizing LSTM networks to analyze sensor data and predict events.

Quantitative inferential approaches, as discussed by Gmati et al. (2019), are effective in extracting patterns from historical data using machine learning algorithms. However, they face challenges such as considering complex data interactions, addressing temporal dynamics, and depending on high-quality historical data. Our work addresses these issues by integrating data from multiple sources via an extended ontology, providing a richer and more contextualized analysis. This approach overcomes the truncated data view and improves prediction accuracy by capturing temporal dynamics and long-term dependencies with LSTM.

In Zhao (2021), the author systematically reviews technologies, applications, and evaluations in event prediction in the big data era. They emphasize that predicting multiple aspects of events often requires joint prediction of heterogeneous yet correlated outputs and highlight the importance of addressing complex dependencies among predictions. Unlike isolated tasks in machine learning, event prediction involves scenarios where events can be interrelated and influence one another and need methods to capture these interdependencies.

Our research effectively applies established strategies to manage these complexities, as highlighted by Zhao (2021). The use of an extended ontology is useful for integrating data from various sources, handling correlations, and understanding the interplay between different events. For example, combining hydrometric and hydrological data can predict floods, where rainfall in one region affects river levels elsewhere, triggering subsequent events.

In Li et al. (2021), the author presents a study on flood prediction modeling using LSTM. The article suggests that the synchronized sequential input and output (SSIO) architecture is more adept at capturing long-term dependencies than the sequential input and single output (SISO) architecture. This advantage is particularly noted in scenarios requiring fine temporal resolution and precise modeling of extended rainfall-runoff processes. However, the study primarily focuses on the architectural comparison and optimization of LSTM networks for a specific hydrological dataset, with limited exploration of the broader applicability of their proposed solution to alternative historical datasets and data integration.

In Al Qundus et al. (2022), the authors address the detection of flood-induced disasters by building a wireless sensor network and a decision model based on a Support Vector Machine (SVM). The network-based decision model observes changes in weather conditions compared to historical information at a specific location. The main limitations of the solution include the reliance on relative values to determine disaster detection thresholds, which may not be applicable in different locations and may require the collection of new data for each specific location. The authors suggest that future research could concentrate on devising methods to establish disaster detection thresholds that are more widely applicable and less reliant on specific historical data from a particular location.

Nearing et al. (2024) explores the use of LSTM networks for predicting extreme floods in ungauged watersheds on a global scale. Their research demonstrates the potential of artificial intelligence to enhance the reliability of flood forecasts, extend lead times, and provide accurate warnings across more than 80 countries. This global approach leverages vast public datasets, allowing for widespread applicability and a comprehensive understanding of hydrological patterns across diverse geographic regions.

In contrast, our project also utilizes LSTM networks but is designed with a specific focus on localized prediction. We aim to develop a generic framework adaptable to various application domains by modifying ontology definitions and context rules to fit different types of sensor data. This framework is not limited to flood predictions; it is intended for the detection and prediction of generic events across multiple domains. While our study includes a feasibility evaluation within the hydrological and hydrometric domain, this does not confine the framework's applicability to these areas alone. The framework's flexibility allows it to be adapted for various applications, providing tailored predictions that are highly accurate and context-specific, catering to the unique characteristics and requirements of different areas.

While Nearing et al. (2024) emphasizes broad, global predictions using extensive datasets, our project focuses on integrating local data sources to refine predictions and provide more precise and actionable insights at the local level. This localization enables us to address specific challenges and nuances that may not be captured by a global model, offering a more detailed and relevant forecast for the targeted area. Thus, while both studies employ LSTM networks, our work distinguishes itself through its emphasis on adaptability and precision in localized contexts, as opposed to the wide-reaching scope of global flood prediction. Moreover, the versatile nature of our framework inted to ensure that it can be applied beyond the hydrological domain, supporting a wide range of applications in various fields.

Table 1 provides a comprehensive summary of the limitations identified in these studies related to event prediction.

Table 1 – Identified limitations of related works

Study	Limitations of related works
Gmati et al.(2019)	**Identified Limitations:** Difficulty in holistically considering complex interactions within the data. Challenges in addressing temporal dynamics. Dependence on high-quality historical data. Generalization across different domains can be difficult.
Zhao(2021)	**Identified Limitations:** Need for joint prediction of heterogeneous yet correlated outputs. Complexity in managing interdependencies among prediction outputs. Difficulty in continuously updating models to reflect real-time data changes.
Li et al.(2021)	**Identified Limitations:** Limited focus on specific LSTM architectures and their optimization for a particular hydrological dataset. Lack of exploration of broader applicability beyond specific dataset limitations. Emphasis on architectural comparison rather than practical deployment.
Al Qundus et al.(2022)	**Identified Limitations:** Dependence on relative values for disaster detection thresholds, making generalization across locations difficult. Requirement for specific historical data collection for new locations. Potential limitations in scalability and adaptability.
Nearing et al.(2024)	**Identified Limitations:** Global focus may overlook local nuances and precision in specific areas. Relies on extensive public datasets, which might not capture local specifics adequately. Challenges in providing actionable insights for localized flood predictions.

The forward snowballing technique was applied with the aim of identifying relevant publications that cited the key studies previously identified in the initial search. This approach is effective for discovering subsequent research that has contributed to the development of the field of study and ensuring that the literature review is comprehensive and up-to-date (80). Table 2 shows the seminal portfolio, containing the base and starting point to the forward snowballing.

Table 2 – Seminal Portfolio: Starting point

Year	Authors	Title	Journal
2019	Gmati et al. (2019)	A taxonomy of event prediction methods	Advances and Trends in Artificial Intelligence
2021	Zhao (2021)	Event prediction in the big data era: A systematic survey	ACM Computing Surveys (CSUR)
2021	Li et al. (2021)	Exploring the best sequence LSTM modeling architecture for flood prediction	Neural Computing and Applications
2022	Al Qundus et al. (2022)	Wireless sensor network for AI-based flood disaster detection	Annals of Operations Research
2024	Nearing et al. (2024)	Global prediction of extreme floods in ungauged watersheds	Nature

The criteria for selecting key studies, derived from the base research, included:

- **Filter 1 (F1) - Terms in Title or Abstract**: "sensor", "data integration", "ontology", "event prediction", "artificial intelligence". These terms were combined using OR logical operator.

```
"sensor" OR "integração de dados" OR "ontologia" OR
"previsão de eventos" OR "predição de eventos" OR
"inteligência artificial" OR "data integration" OR
```

```
"ontology" OR "event prediction" OR "artificial intelligence"
```

- **Filter 2 (F2) - Languages**: Portuguese or English.
- **Filter 3 (F3) - Scientific relevance assessment**: Journals classified in Quartile 1 (Q1) - CiteScore[2].
- **Filter 4 (F4) - Content**: Sensor, data integration, ontology, event prediction, and artificial intelligence. These terms were combined using the AND logical operator.

```
("sensor" AND "integração de dados" AND "ontologia" AND
"previsão de eventos" AND "predição de eventos" AND
"inteligência artificial") OR
("sensor" AND "data integration" AND "ontology" AND
"event prediction" AND "artificial intelligence")
```

To enrich the searches for relevant related works, it was observed that the term "ontology" overly restricted the returned results. In cases where Filter 4 returns zero relevant works, a new filter (-ontology) was applied, which consists of removing the term "ontology" from the search string of Filter 4. The Google Scholar[3] and Scopus[4] tools were used to identify articles that cited the key studies. For each key study, a search was conducted to identify all publications that cited it. The results were filtered to include only peer-reviewed articles published in scientific journals. The included articles were read and analyzed in detail to identify the main contributions, methodologies employed, and the results found. Special focus was given to how these subsequent studies expanded, challenged, or confirmed the findings of the key studies.

The quantitative synthesis of the results was organized in Table 3 to facilitate the visualization and comparison of information and summarize the results obtained in the application of snowballing for each base after the application of each filter in each iteration.

Table 3 – Forward snowballing: Quantitative synthesis

Seminal Portfolio	Cited By	Iteration 1					Iteration 2						Iteration 3					
		F1	F2	F3	F4	-ontology	Cited By	F1	F2	F3	F4	-ontology	Cited By	F1	F2	F3	F4	-ontology
Gmati et al. (2019)	9	8	8	4	1	-	-	-	-	-	-	-	-	-	-	-	-	-
Zhao (2021)	127	85	83	46	7	-	18	11	11	7	3	-	6	4	4	3	3	-
Li et al. (2021)	51	24	24	17	-	12	180	118	116	66	2	-	19	18	18	9	-	-
Al Qundus et al. (2022)	72	44	43	27	-	19	287	202	202	92	6	-	31	31	31	21	-	-
Nearing et al. (2024)	22	12	12	7	-	7	14	3	3	1	-	2	-	-	-	-	-	-

[2] https://www.elsevier.com/products/scopus/metrics/citescore
[3] https://scholar.google.com/
[4] https://www.scopus.com/home.uri

After completing the snowballing iterations, a total of 62 articles were identified. Duplicates were removed, resulting in a refined list. From this list, the 12 most relevant and closely related articles to our research theme were selected. The selection criteria prioritized articles that addressed data integration and event prediction, specifically validated within the fields of hydrological and disaster predictions. Additionally, the articles with the highest CiteScore among them were chosen. A summary of these selected works is provided in Table 14 in the Appendix.

These articles discuss sensor data integration, event detection, and prediction methodologies. The works collectively highlight the importance of integrating heterogeneous data sources and utilizing advanced algorithms to enhance predictive accuracy.

The research in (7) emphasizes the need for a unified approach to event prediction across various domains, underscoring the challenges of handling diverse datasets and the necessity for cross-disciplinary methodologies. Similarly, (2) demonstrates the application of various AI models for predicting hydrological events, showcasing the effectiveness of different techniques in managing complex data environments.

In (34), the authors explore the optimization of LSTM parameters for flood forecasting using genetic algorithms. This study provides insights into enhancing the performance of predictive models through parameter optimization. Another significant contribution, (41), offers a review of recent advances in flood modeling approaches, categorizing and evaluating various techniques, and discussing their strengths and weaknesses.

The study (56) compares four machine learning models for flood prediction, highlighting the potential of ensemble and deep learning techniques in improving prediction accuracy. Similarly, (66) develops an integrated LSTM model for daily flow discharge prediction, demonstrating the applicability of deep learning in water management.

In (76), the authors provide an overview of deep learning applications in hydrology and water resources, emphasizing the potential of these techniques. The real-time flood forecasting capabilities explored in (77) and (45) further illustrate the practical implications of advanced neural network frameworks in critical environmental monitoring.

The work (24) discusses the role of explainable AI in disaster risk management, highlighting the importance of transparency and interpretability in AI applications. Meanwhile, (64) develops a multivariate LSTM network for water level forecasting, showcasing the model's superior accuracy compared to traditional methods.

Lastly, (68) provides an in-depth discussion on hydrological forecasting techniques, including traditional methods and modern technologies like remote sensing and GIS(Geographic Information System). This study underscores the integration of these methods to enhance forecast accuracy and reliability.

Our solution proposes a framework that builds upon these foundational works by integrating an ontology-based approach with AI for sensor data integration and event prediction. Unlike the reviewed studies, which often focus on specific applications or algorithms, the proposed framework generalizes across multiple domains through an extended ontology, allowing for more comprehensive data contextualization and integration. This methodology not only overcomes the limitations of isolated data sources but also facilitates the inclusion of new sensor types and datasets, thereby enhancing the flexibility and adaptability of the system.

Common among these studies and our proposal is the use of artificial intelligence for predictive analytics. However, a key distinction is the our proposal's emphasis on semantic integration via ontologies, which provides a structured and standardized representation of data, enabling more accurate and contextually relevant predictions. This semantic layer is less prominent in the other works, which tend to focus more on the technical optimization of prediction algorithms.

The works reviewed share certain limitations, such as the dependency on high-quality historical data and the challenges associated with ensuring data consistency and accuracy. Our solution addresses these issues through the use of an ontology, which standardizes data representation and supports interoperability across different systems and datasets. This approach not only improves data quality but also facilitates the seamless integration of heterogeneous data sources.

Therefore, the proposed framework complements existing research by offering a more holistic and versatile solution for sensor data integration and event prediction. By combining ontological models with AI, this proposal advances the current state-of-the-art and provides an option for future research across various application domains. This integrated approach addresses the gaps identified in the literature and presents a scalable solution capable of adapting to diverse predictive needs.

The application of the forward snowballing technique allowed the identification of a comprehensive set of publications that significantly contributed to the understanding of the domain. This approach intend to ensure that the literature review was not only exhaustive but also incorporated the most recent and relevant developments.

As a result, we introduce a framework that extends the reviewed approaches, intending to advance abstraction in data integration and contextualized AI prediction. The objective is to enhance the detection and prediction of events of interest. In this regard, we propose an ontology extended from SSN to facilitate the abstraction of the data integration step. The presented ontology serves to standardize and integrate data by employing a canonical semantic representation for sensor data.

2.2.1 Final Considerations of the Chapter

In this chapter, we reviewed the key concepts and related work on semantic data integration and event prediction based on sensor data, with a focus on the use of ontologies and artificial intelligence. We discussed the challenges and opportunities in handling heterogeneous data and building AI-based prediction systems. Furthermore, we presented an overview of relevant works in the field and the main methodological approaches currently available.

This theoretical background provided an essential foundation for the development of our proposed solution, which aims to integrate sensor data using an ontology-based framework. In the next chapter, we will describe the proposed solution in detail, discussing its architecture and the methods employed for integrating data and predicting critical events.

3 Proposed Solution

This chapter presents the proposed framework and the solution model for sensor data integration and critical event detection and prediction. In this scope, we consider a ‘critical event‘, all the observations made by a specific sensor in a hydrometric station that is classified as a flood. The framework input is data from multiple sensors from the same domain, in this case, hydrological and hydrometric stations and sensors. The framework integrates data from multiple sensors, correlates context, and predicts future events based on the integrated data. The output is the events detected. The following section outlines the key aspects of our approach.

The Figure 5 represents the structured layers of the proposed framework, each addressing specific challenges related to sensor data integration, preprocessing, event detection, and prediction. The decision to structure the framework in these distinct layers was driven by the need to meet certain critical quality attributes.These quality attributes are essential for ensuring that the framework can handle the complexities and requirements of real-time sensor data environments. Revisiting these attributes aids in understanding how each layer of the framework contributes to meeting these needs. The most relevant attributes for our solution, as discussed in detail in subsection **3.1.6**, include:

- **Scalability**: The ability to efficiently handle an increasing number of data sources without significant degradation in performance.
- **Real-Time Processing**: Ensuring timely analysis of sensor data for immediate event detection and response.
- **Data Quality and Integrity**: Implementing mechanisms to handle errors, noise, and missing data from heterogeneous sensor sources.
- **Flexibility and Adaptability**: Allowing the framework to easily accommodate new types of sensors and data sources.
- **Heterogeneity of Data Sources**: The need to standardize and integrate data from multiple sensors with varied formats and protocols.

With these attributes in mind, Figure 5 was developed to illustrate how the framework meets these challenges through a modular architecture, with specific layers dedicated to tasks such as data acquisition, cleansing, ontology-based contextualization, and AI-based prediction.

The framework is structured and implemented through distinct layers, providing an insight into the data flow, layer composition, and component interactions. We first describe each layer and the development process in this section. Section 4 evaluates the framework in real-world sensor data from hydrometric and hydrological stations.

In this work, the data preprocessing and enrichment were carried out, as stated before, with an ontology model extended from SSN, which can extract syntactic, semantic, and context knowledge from sensor data. This model considers pre-defined context to define sensor relationships and helps detect events based on the sensor network.

Long Short-Term Memory (LSTM) is applied for event prediction. LSTM is a Recurrent Neural Network architecture widely used in Deep Learning. It captures long-term dependencies, making it ideal for sequence prediction tasks.

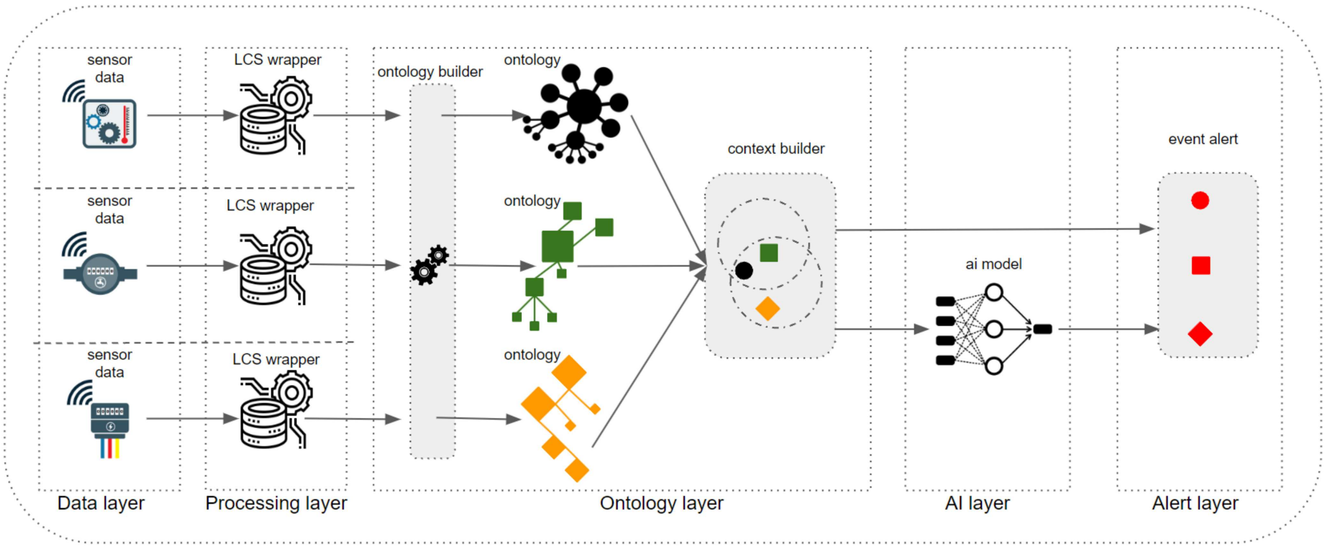


Figure 5 – Framework Overview.

3.1 Data Layer

The Data Layer is a key component in systems that integrate sensor data for event detection and prediction. This layer is designed to be generic and adaptable, allowing it to be utilized across various sensor data domains. It is responsible for the acquisition, storage, initial processing, and management of data collected by sensors in real time. The effectiveness of this layer is critical to the overall performance and reliability of the system. Typical challenges and problems in this scope include handling heterogeneous data sources, ensuring data quality, and providing timely processing and integration. Next subsections explain in details Data Layer functioning and the challenges that this layer has to deal with.

3.1.1 Data Acquisition

Data acquisition is the entry point for any sensor data integration system. Sensors of various types and manufacturers capture raw data and transmit it to the central system.

Examples include temperature, humidity, pressure, water flow, and air quality sensors. In the context of a smart city, sensors monitor traffic, air quality, and energy consumption, while in an industrial environment, sensors monitor machinery for predictive maintenance to prevent failures and optimize productivity.

3.1.2 Data Cleansing

Data cleansing is a decisive step to ensure that the data used in analyses is accurate and reliable. This process involves removing noise by identifying and eliminating data that does not represent valid information, such as readings outside the physical limits of the sensors. Error correction adjusts incorrect or anomalous values based on predefined rules or statistical methods. Handling missing data involves techniques such as imputing values or removing incomplete records. Consistency checks verify and correct discrepancies and inconsistencies in data from different sources. For instance, in an air quality monitoring system, anomalous readings due to temporary sensor failures can be identified and corrected or removed.

3.1.3 Data Preprocessing

Data preprocessing transforms raw data into a usable format. This step includes normalization, which adjusts different units of measure to a common format, and aggregation, which consolidates data from multiple sensors into a single dataset. For example, in a network of hydrometric sensors, data on water level and precipitation can be normalized to a common scale and aggregated to provide an overview of the current state of a river. Preprocessing also involves creating a canonical data model to standardize and harmonize data from various sources, facilitating integration and analysis.

3.1.4 Data Storage

Preprocessed data needs to be stored efficiently to allow for quick access and subsequent analysis. Storage solutions must be capable of handling large volumes of data while ensuring fast retrieval and scalability(33). Relational databases are suitable for structured data with well-defined relationships, NoSQL databases for semi-structured or unstructured data such as sensor logs, and data lakes for storing vast amounts of raw data in its original format, allowing for flexible analysis. In an environmental monitoring scenario, data from various sensors can be stored in a data lake for detailed future analyses (16).

3.1.5 Stream Processing

Real-time processing (streaming) is essential for immediate event detection in sensor systems. Stream processing enables the system to capture, process, and analyze data as it is received, ensuring timely responses to critical events. For example, in a traffic monitoring system, stream processing can detect congestion in real time and alert drivers to avoid certain routes.

3.1.6 Quality Attributes

Heterogeneity of Data Sources: Integrating data from diverse sensors with different formats and communication protocols is a significant challenge. A robust data acquisition and preprocessing framework is required to standardize and harmonize this data.

Data Quality and Integrity: Ensuring the accuracy, consistency, and completeness of data is crucial. This includes implementing thorough data cleansing processes to remove noise, correct errors, and handle missing data.

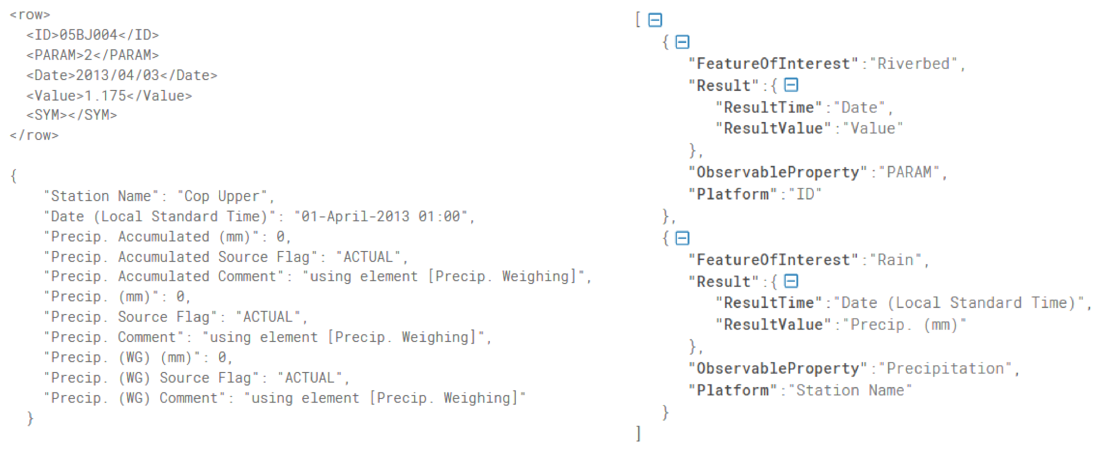
Scalability: The system must be able to scale to handle increasing volumes of data as more sensors are added or as the frequency of data collection increases. Efficient data storage and processing mechanisms are essential to maintain performance.

Real-Time Processing: Timely processing of data is critical, especially for applications requiring immediate response, such as disaster prevention or real-time monitoring systems. Stream processing technologies must be employed to ensure low-latency data handling.

Flexibility and Adaptability: The Data Layer must be designed to be flexible and adaptable to different domains and applications. This includes the ability to integrate new types of sensors and data sources without significant reconfiguration.

3.2 Preprocessing Layer

The **Preprocessing Layer** receives sensor data (Figure 6(a)) from the Data Layer and is tasked with establishing a canonical representation of this information. The *LCS wrapper* components construct this representation based on a *Local Concept Schema* (LCS), defining the transformation rules for converting sensor data into a canonical model, as shown in (Figure 6(b)).



```
<row>
  <ID>05BJ004</ID>
  <PARAM>2</PARAM>
  <Date>2013/04/03</Date>
  <Value>1.175</Value>
  <SYM></SYM>
</row>

{
    "Station Name": "Cop Upper",
    "Date (Local Standard Time)": "01-April-2013 01:00",
    "Precip. Accumulated (mm)": 0,
    "Precip. Accumulated Source Flag": "ACTUAL",
    "Precip. Accumulated Comment": "using element [Precip. Weighing]",
    "Precip. (mm)": 0,
    "Precip. Source Flag": "ACTUAL",
    "Precip. Comment": "using element [Precip. Weighing]",
    "Precip. (WG) (mm)": 0,
    "Precip. (WG) Source Flag": "ACTUAL",
    "Precip. (WG) Comment": "using element [Precip. Weighing]"
  }
```

```
[
   {
      "FeatureOfInterest":"Riverbed",
      "Result":{
         "ResultTime":"Date",
         "ResultValue":"Value"
      },
      "ObservableProperty":"PARAM",
      "Platform":"ID"
   },
   {
      "FeatureOfInterest":"Rain",
      "Result":{
         "ResultTime":"Date (Local Standard Time)",
         "ResultValue":"Precip. (mm)"
      },
      "ObservableProperty":"Precipitation",
      "Platform":"Station Name"
   }
]
```

Figure 6 – Sensor data representation and LCS mapping rules.
(a) Raw Data (b) Equivalent LCS Mapping

The primary purpose of the canonical model is to address data heterogeneity, ensuring that information from different sources is consistently interpreted and used. This is particularly important in environments where data is collected from multiple sensors with varied formats and communication protocols.

The model is defined using Local Conceptual Schemas (LCS) mapped to a Global Conceptual Schema (GCS) (61). An intermediary component called a wrapper carries out this process, applying mapping rules defined in each LCS to transform the data from the sources into a unified format.

The GCS represents a unified ontological schema of the integrated data, ensuring a coherent view of the data. The wrapper is an intermediary component that processes data from various sources according to the rules defined in the LCS. It ensures consistent interpretation and translation of data into the GCS. The LCS consists of specific mapping rules for each data source, facilitating scalability and flexibility by allowing new data sources to be integrated without affecting the existing setup. The data sources are the multiple data providers connected to the system, each linked to the wrapper through its specific LCS.

Figure 6(a) shows an example of sensor data, including information such as ID, PARAM, Date, and Value. Figure 6(b) presents the transformation of sensor data into the canonical format using elements like ‘FeatureOfInterest’, ‘Result’, ‘ObservableProperty’, and ‘Platform’ to describe the data in a standardized manner.

‘FeatureOfInterest’ identifies the object of interest, such as ‘Riverbed’ or ‘Rain’. ‘Result’ describes the measurement result, including the result type and measured value. ‘ObservableProperty’ is the observable property associated with the result, such as ‘Precipitation’ or ‘PARAM’. ‘Platform’ indicates the platform or station where the measurement was taken.

The canonical model facilitates data interoperability by providing a common framework for different sources. It ensures data consistency, enabling consistent interpretation. The model's flexibility allows for the easy integration of new data sources without impacting the existing system. Its modularity enables scalability through the addition of new LCSs.

This canonical model is essential for managing the complexity and diversity of data in integrated sensor systems, allowing for more efficient and accurate analysis and utilization of the collected data.

3.3 Ontology Layer

Based on LCS definitions performed by the Preprocessing Layer, the **Ontology Layer** is responsible for processing those results and modeling them into an ontology representation.

We extended a simplified version of the SSN Ontology using the Methontology (23), an accepted methodology that defines the ontology development process. The methodology includes four phases: (i) Specification, (ii) Conceptualisation, (iii) Formalisation, and (iv) Evaluation. Next, we describe each phase.

3.3.1 Specification

The specification is considered an essential step in system development. Therefore, this step established the Ontology Requirement Specification Document (ORSD). The ORSD allows us to identify the ontology's knowledge and define the requirements the ontology must cover. In this document, we describe the ontology's: (i) purpose, (ii) scope, (iii) implementation language, and (iv) intended End-Users. The ORSD is shown in Table 4.

Table 4 – Ontology Requirements Specification Document for SSN

Ontology Requirements Specification Document	**Description**
Purpose	Identify the correct type of sensor network and provide a standardized way to represent sensor descriptions, measurements, and observations. To facilitate interoperability among sensor networks and applications. To allow defining context to relate one sensor to another. To detect critical events based on pre-defined limit thresholds.
Scope	The ontology focuses on integrating sensor data based on context into a sensor network. The level of granularity is directly related to the competency questions that are defined in Subsection **3.3.4**.
Implementation Language	The ontology is specified in the OWL 2 language (Web Ontology Language) using the Protégé ontology tool and Owlready2, a module that can manage ontologies in Python.

End Users	User 1 – Sensor Network Engineers;
	User 2 – Application Developers;
	User 3 – Decision-makers.

Suárez-Figueroa et al. (2009) present guidelines based on using the Competency Questions (CQ) and the existing methodologies to build ontologies. Competency Questions specification is vital since it allows us to determine its scope and validate the ontology (38). Therefore the verification step is done with the ontology responding correctly to the CQ (9). The CQs designed to aid in the validation activity are shown in Table 5 and are answered in Chapter 5 within the Research Questions.

Table 5 – Main Competency Questions

Competency Questions (CQ)	**Description**
CQ1	What are the platforms in the same location context?
CQ2	What are the platforms in the same flow direction context?
CQ3	What are the critical events detected when observation results trespass the platform threshold?

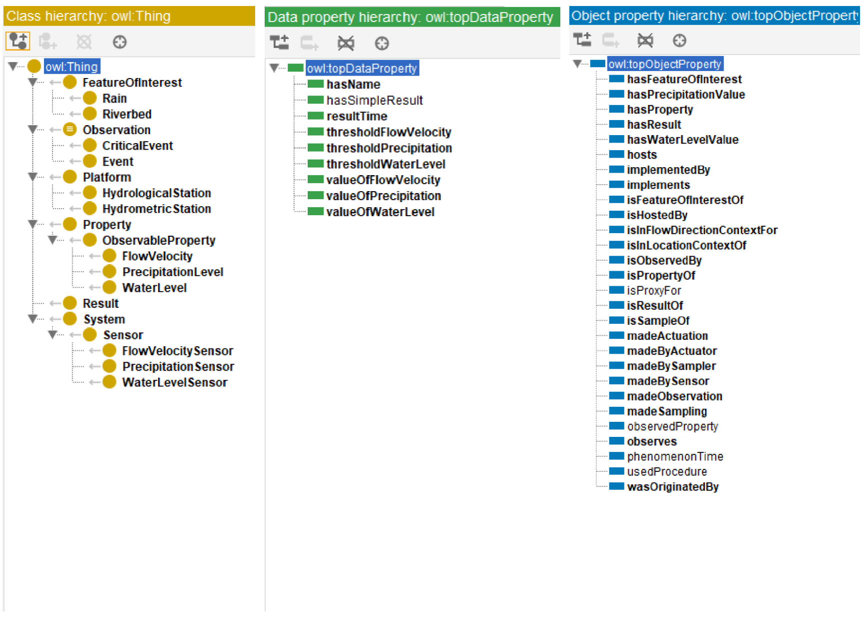
3.3.2 Conceptualization

The conceptualization phase focuses on organizing and structuring the semantic meaning of data. Therefore, we developed a taxonomy[1] to categorize the knowledge in the context addressed. Once our solution uses a feasibility study based on flood detection and prediction, we extend the SSN base ontology to contain a more specific data representation. Table 6 presents the main classes of the simplified version of SSN Ontology, and Figure 7 shows the extended classes, derived data properties, and object properties.

From the FeatureOfInterest class, the Rain and Riverbed classes were extended to represent the two main features of the hydrometric and hydrological domain. The Observation class was extended to include CriticalEvent and Event, to represent the only types of events classified. 'Event' refers to observations not classified as flood events, while 'CriticalEvent' refers to flood events. Additionally, new classes were extended from Platform, with HydrologicalStation and HydrometricStation representing different types of monitoring stations. The ObservableProperty class was expanded to include FlowVelocity, PrecipitationLevel, and WaterLevel, which are specific observable properties. In terms of sensors, specific subclasses were defined under the Sensor class, including FlowVelocitySensor, PrecipitationSensor, and WaterLevelSensor, which are responsible for measuring their respective observable properties.

Finally, in Figure 8, we represent the taxonomy model of the simplified extended ontology with the main classes and their respective relationships.

[1] A Taxonomy is a hierarchical structure representing the formal organization of classes or types of objects within a domain.



Class hierarchy: owl:Thing

- owl:Thing
  - FeatureOfInterest
    - Rain
    - Riverbed
  - Observation
    - CriticalEvent
    - Event
  - Platform
    - HydrologicalStation
    - HydrometricStation
  - Property
    - ObservableProperty
      - FlowVelocity
      - PrecipitationLevel
      - WaterLevel
  - Result
  - System
    - Sensor
      - FlowVelocitySensor
      - PrecipitationSensor
      - WaterLevelSensor

Data property hierarchy: owl:topDataProperty

- owl:topDataProperty
  - hasName
  - hasSimpleResult
  - resultTime
  - thresholdFlowVelocity
  - thresholdPrecipitation
  - thresholdWaterLevel
  - valueOfFlowVelocity
  - valueOfPrecipitation
  - valueOfWaterLevel

Object property hierarchy: owl:topObjectPropert

- owl:topObjectProperty
  - hasFeatureOfInterest
  - hasPrecipitationValue
  - hasProperty
  - hasResult
  - hasWaterLevelValue
  - hosts
  - implementedBy
  - implements
  - isFeatureOfInterestOf
  - isHostedBy
  - isInFlowDirectionContextFor
  - isInLocationContextOf
  - isObservedBy
  - isPropertyOf
  - isProxyFor
  - isResultOf
  - isSampleOf
  - madeActuation
  - madeByActuator
  - madeBySampler
  - madeBySensor
  - madeObservation
  - madeSampling
  - observedProperty
  - observes
  - phenomenonTime
  - usedProcedure
  - wasOriginatedBy

Figure 7 – Ontology classes, data properties and object properties
(a) Class (b) Data Property (c) Object Property

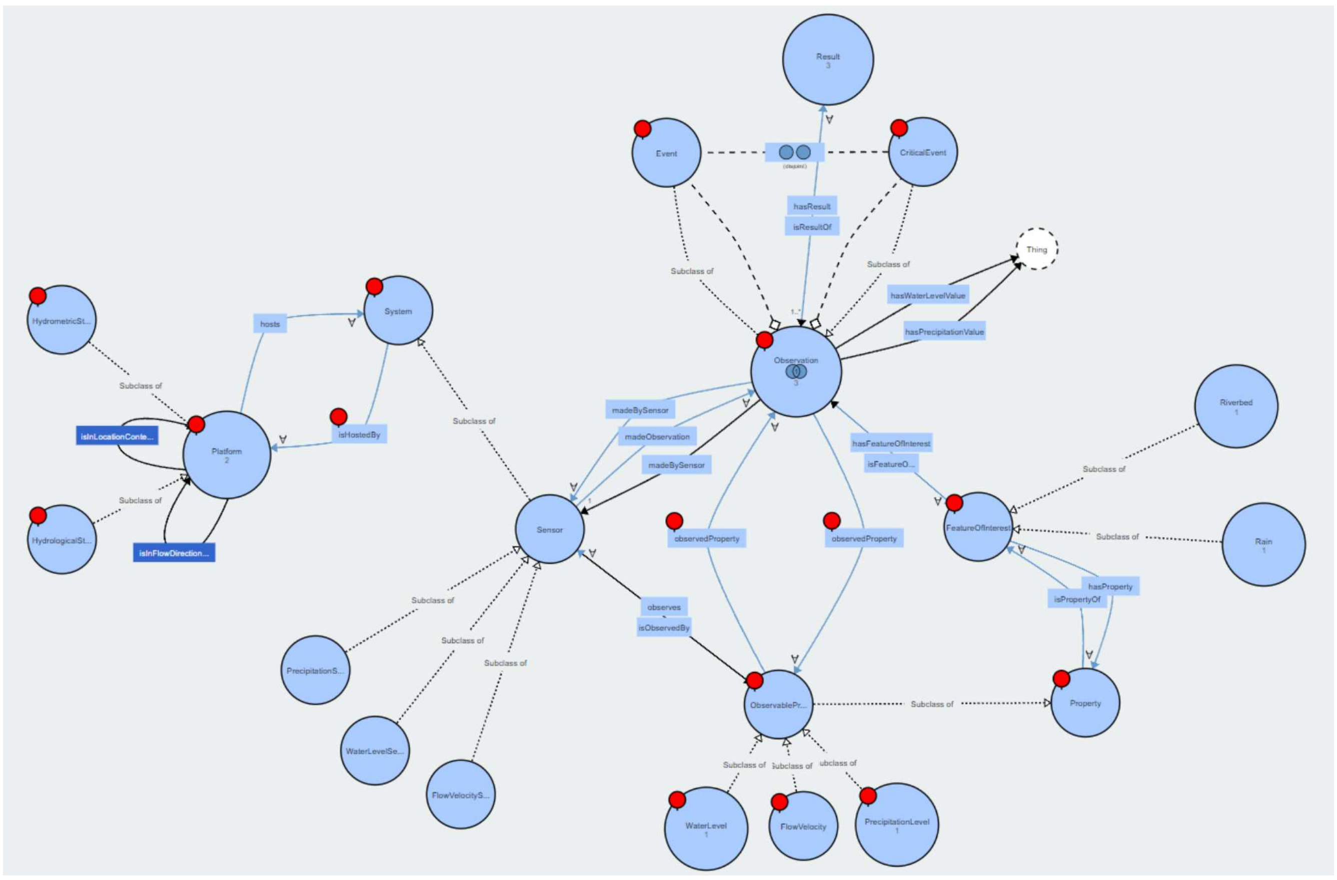


Figure 8 – Hydrologic and Hydrometric domain taxonomy

Table 6 – Main classes of extended SSN Ontology

Class	Description	Instantiated by
CriticalEvent	Represents critical events that can have significant impacts, in this case, floods for hydrometric stations and high precipitation for hydrological stations.	Automatic process extraction
Event	Refers to any non-critical events.	Automatic process extraction
FeatureOfInterest	An entity (real-world object, process, or event) that is being sensed. A feature is an abstraction of real-world phenomena.	Predefined in ontology
FlowVelocity	Represents the velocity of flow, typically in a hydrometric context, such as water flow in a river.	Automatic process extraction
FlowVelocitySensor	Specific sensors used to measure the water flow velocity.	Automatic process extraction
HydrologicalStation	Stations that monitor hydrological parameters, in this case, precipitation.	Automatic process extraction
HydrometricStation	Specifically, stations that measure and record data on the characteristics of water bodies, like water levels and flow.	Automatic process extraction
ObservableProperty	Properties that can be observed or measured, such as temperature, humidity, water level, etc.	Predefined in ontology
Observation	Data collected by sensors about a particular observable property at a specific time.	Automatic process extraction
Platform	Infrastructure or devices that support and house sensors, like buoys, satellites, or ground stations.	Predefined in ontology
PrecipitationLevel	Measures of the amount of precipitation, such as rain or snow, in a given area.	Automatic process extraction
PrecipitationSensor	Sensors used to measure the amount of precipitation.	Automatic process extraction
Property	Any characteristic or attribute that can be observed, such as temperature, pressure, humidity, etc.	Predefined in ontology
Rain	Specifically, observations or events related to rain.	Predefined in ontology
Result	The outcome or data generated from an observation.	Automatic process extraction
Riverbed	Riverbeds are often an object of interest for hydrometric studies and measurements.	Predefined in ontology
Sensor	Devices that detect and measure physical, chemical, or biological properties and send data for analysis.	Predefined in ontology
System	Set of interrelated components, including sensors and platforms, that work together to monitor and record environmental or system data.	Predefined in ontology
WaterLevel	Water level in a body of water, such as a river or reservoir.	Automatic process extraction
WaterLevelSensor	Sensors specifically designed to measure the water level.	Automatic process extraction

3.3.3 Formalization

The formalization phase is when we convert the conceptual model (taxonomy) into a computable model. We used the Protégé [2], a tool that uses the Ontology Web Language (OWL)[3] to define an ontology by specifying its classes, properties, and semantic rules.

In OWL, classes are interpreted as a set of individuals or objects. For example, all classes are subclasses of the class Thing, which represents a set of all individuals. Figure 7(a) shows the classes defined in the ontology. Properties in OWL represent relationships between individuals. There are two types of properties: object property and data property. An object property links an object, such as an individual, to another object, while data properties link objects to a data type, such as Integer, String, or Date values. Table 7 shows the properties defined in the ontology, including their type, domain, and range.

Table 7 – Properties in the extended SSN Ontology

Property	Type	Domain	Range
hasFeatureOfInterest	ObjectProperty	Observation	FeatureOfInterest
hasPrecipitationValue	ObjectProperty	Observation	Result
hasProperty	ObjectProperty	FeatureOfInterest	ObservableProperty
hasResult	ObjectProperty	Observation	Result
hasWaterLevelValue	ObjectProperty	Observation	Result
hosts	ObjectProperty	Platform	Sensor
isFeatureOfInterestOf	ObjectProperty	FeatureOfInterest	Observation
isHostedBy	ObjectProperty	Sensor	Platform
isObservedBy	ObjectProperty	ObservableProperty	Sensor
isPropertyOf	ObjectProperty	ObservableProperty	FeatureOfInterest
isResultOf	ObjectProperty	Result	Observation
madeBySensor	ObjectProperty	Observation	Sensor
madeObservation	ObjectProperty	Sensor	Observation
observedProperty	ObjectProperty	Observation	ObservableProperty
observes	ObjectProperty	Sensor	ObservableProperty
isInFlowDirectionContextFor	ObjectProperty	Observation	FeatureOfInterest
isInLocationContextOf	ObjectProperty	Observation	FeatureOfInterest
hasName	DataProperty	System	xsd:string
resultTime	DataProperty	Observation	time:Instant
thresholdFlowVelocity	DataProperty	ObservableProperty	xsd:float
thresholdPrecipitation	DataProperty	ObservableProperty	xsd:float
thresholdWaterLevel	DataProperty	ObservableProperty	xsd:float
valueOfFlowVelocity	DataProperty	Result	xsd:float
valueOfPrecipitation	DataProperty	Result	xsd:float
valueOfWaterLevel	DataProperty	Result	xsd:float

Table 8 shows the Semantic Web Rule Language (SWRL) rules created to infer implicit relationships. SWRL is a language for the Semantic Web used to express semantic

[2] https://protege.stanford.edu/
[3] https://www.w3.org/TR/owl-features/

rules and logic. The rules from 1 to 14 define the context rules based on flow direction and location context. Rules 15 and 16 define the occurrence of critical events in the hydrological domain (high precipitation level) and hydrometric domain (high water level).

In the ontology layer, rule-based event detection is defined by considering predefined thresholds for each data source. This allows the classification of observations as critical events when they surpass these thresholds. The results of this threshold-based event detection are then directly forwarded to the alert layer. So, for example, if an observation from a hydrometric sensor has a measured value result of 10 meters, and the threshold for the station where that sensor is based is 8 meters, that observation is classified as a critical event. Otherwise, it is an event. Defining context is a crucial step, as the AI input layer relies on it.

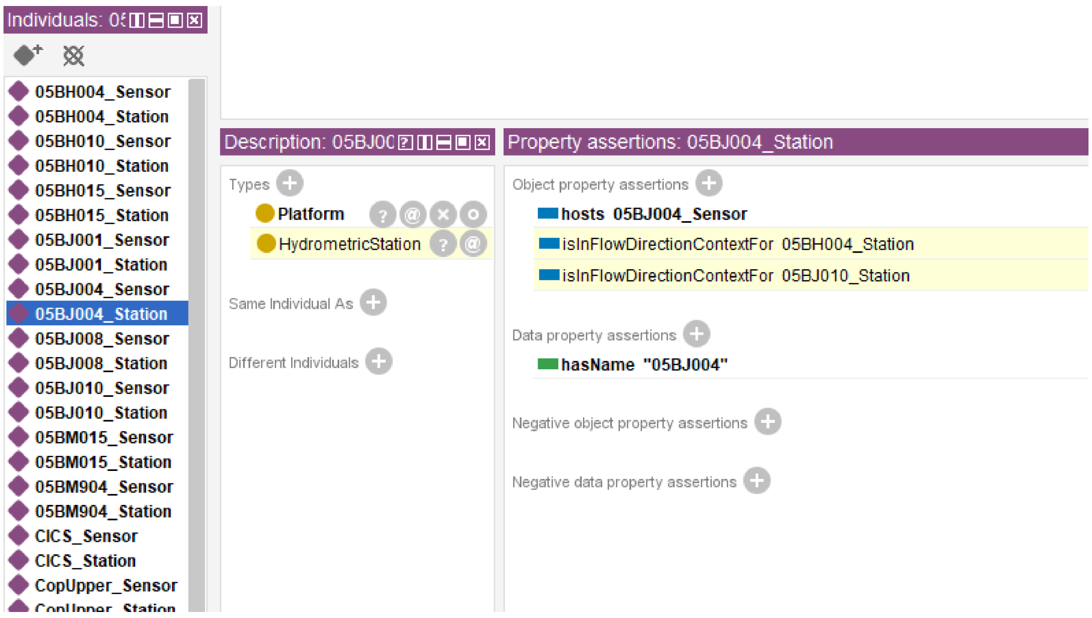
Table 8 – SWRL Rules

	Rules	
	Antecedent	**Consequent**
1	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BH015") ∧ hasName(?p2, "05BH010")	→ isInFlowDirectionContextFor(?p1, ?p2)
2	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BJ004") ∧ hasName(?p2, "05BJ010")	→ isInFlowDirectionContextFor(?p1, ?p2)
3	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BJ010") ∧ hasName(?p2, "05BH010")	→ isInFlowDirectionContextFor(?p1, ?p2)
4	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BJ010") ∧ hasName(?p2, "05BJ008")	→ isInFlowDirectionContextFor(?p1, ?p2)
5	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BJ008") ∧ hasName(?p2, "05BJ001")	→ isInFlowDirectionContextFor(?p1, ?p2)
6	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BJ008") ∧ hasName(?p2, "05BM904")	→ isInFlowDirectionContextFor(?p1, ?p2)
7	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BJ001") ∧ hasName(?p2, "05BH004")	→ isInFlowDirectionContextFor(?p1, ?p2)
8	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "05BM015") ∧ hasName(?p2, "05BH004")	→ isInFlowDirectionContextFor(?p1, ?p2)
9	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "CopUpper") ∧ hasName(?p2, "05BH010")	→ isInLocationContextOf(?p1, ?p2)
10	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "CSBA") ∧ hasName(?p2, "05BH010")	→ isInLocationContextOf(?p1, ?p2)
11	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "CSBA") ∧ hasName(?p2, "05BH015")	→ isInLocationContextOf(?p1, ?p2)
12	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "CICS") ∧ hasName(?p2, "05BH004")	→ isInLocationContextOf(?p1, ?p2)
13	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "ElbowAuto") ∧ hasName(?p2, "05BJ004")	→ isInLocationContextOf(?p1, ?p2)
14	Platform(?p1) ∧ Platform(?p2) ∧ hasName(?p1, "ElbowRanger") ∧ hasName(?p2, "05BJ004")	→ isInLocationContextOf(?p1, ?p2)
15	Observation(?o) ∧ PrecipitationSensor(?s) ∧ Platform(?p) ∧ Result(?r) ∧ hosts(?p,?s) ∧ madeBySensor(?o, ?s) ∧ hasResult(?o, ?r) ∧ valueOfPrecipitation(?r, ?vop) ∧ thresholdPrecipitation(?p, ?tp) ∧ swrlb:greaterThan(?vop, ?tp)	→ CriticalEvent(?o)
16	Observation(?o) ∧ WaterLevelSensor(?s) ∧ Platform(?p) ∧ Result(?r) ∧ hosts(?p,?s) ∧ madeBySensor(?o, ?s) ∧ hasResult(?o, ?r) ∧ valueOfWaterLevel(?r, ?vowl) ∧ thresholdWaterLevel(?p, ?twl) ∧ swrlb:greaterThan(?vowl, ?twl)	→ CriticalEvent(?o)

3.3.4 Evaluation

The evaluation activity is carried out during all phases in the traditional Methontology Framework (23). In our work, we considered this activity as another phase in the proposed methodology, consisting of carrying out the following tasks: verification and validation. According to the ORSD, the verification step consists of carrying out a technical judgment of the ontology by verifying its correctness and validating it. The correctness of the ontology is verified using the Pellet plugin reasoner on Protégé, a piece of software able to infer logical consequences from a set of asserted facts or axioms. The validation is a step to ensure that the ontology fulfills its purpose.

Figure 9 shows a sample of context inference made based on context rules. Object property assertions show that '05BJ004_Station' hosts the sensor '05BJ004_Sensor' and is contextually linked in flow direction with '05BH004_Station' and '05BJ010_Station'. This implies that '05BJ004_Station' may influence '05BJ010_Station'.


Individuals: 05

05BH004_Sensor
05BH004_Station
05BH010_Sensor
05BH010_Station
05BH015_Sensor
05BH015_Station
05BJ001_Sensor
05BJ001_Station
05BJ004_Sensor
05BJ004_Station
05BJ008_Sensor
05BJ008_Station
05BJ010_Sensor
05BJ010_Station
05BM015_Sensor
05BM015_Station
05BM904_Sensor
05BM904_Station
CICS_Sensor
CICS_Station
CopUpper_Sensor
CopUpper_Station

Description: 05BJ00

Types
Platform
HydrometricStation

Same Individual As

Different Individuals

Property assertions: 05BJ004_Station

Object property assertions
hosts 05BJ004_Sensor
isInFlowDirectionContextFor 05BH004_Station
isInFlowDirectionContextFor 05BJ010_Station

Data property assertions
hasName "05BJ004"

Negative object property assertions

Negative data property assertions

Figure 9 – Context inference example

3.4 AI Layer

The **AI Layer** stands out as a central component, employing artificial intelligence techniques for advanced data analysis and event prediction. Leveraging a Long Short-Term Memory neural network, a recurrent neural network type, this layer establishes connections between sensor data in the same context. The primary goal is to enhance the system's predictive capabilities based on the combined information from these sensors.

The LSTM neural network stands out in handling sequential and time-series data, making it an effective choice for analyzing sensor readings over time. Trained on historical sensor data, the network learns intricate patterns and dependencies among various environmental factors.

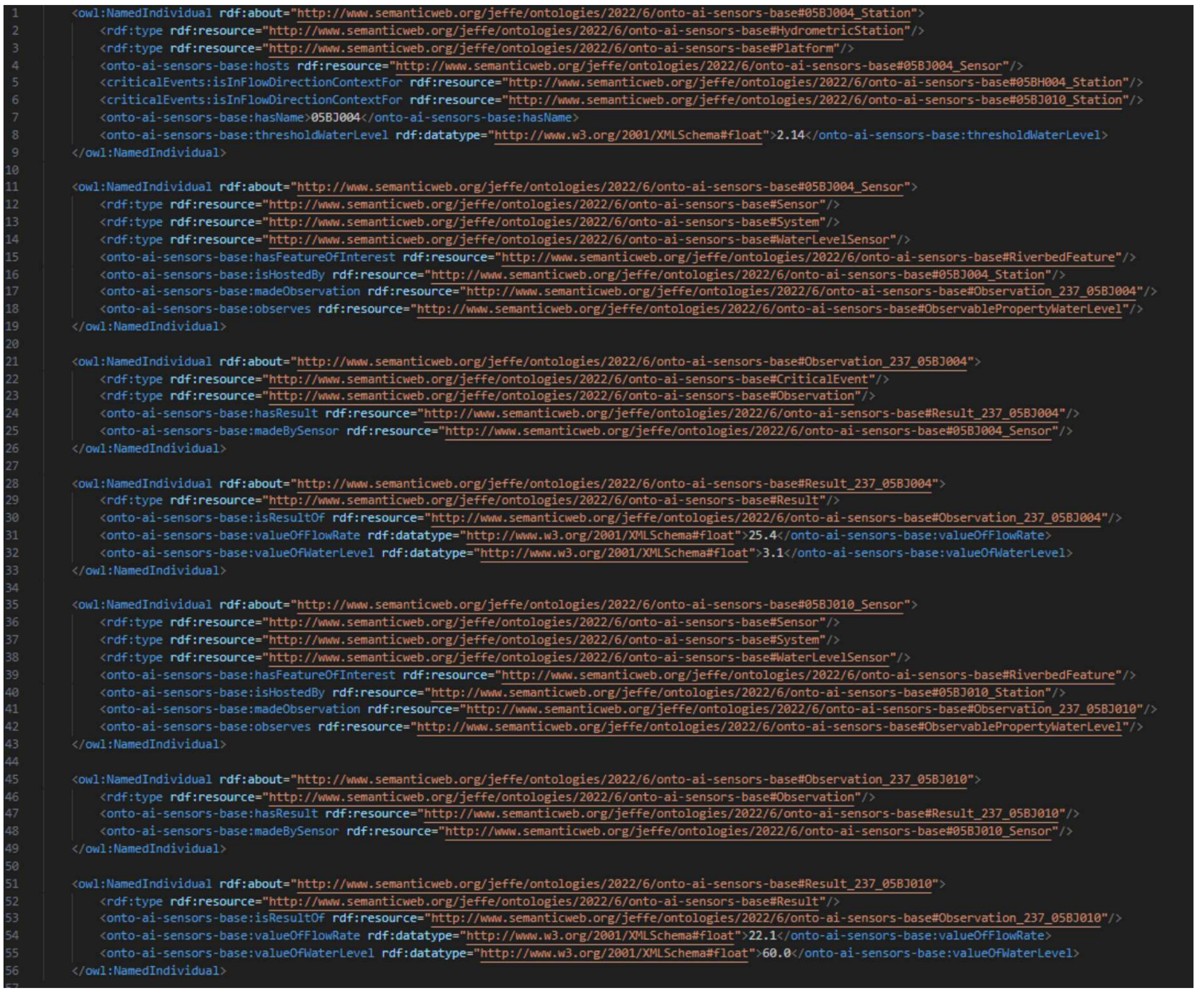
Our solution employs a context-aware input selection, where the LSTM model processes sensor data based on the identified context. We adapt the standard LSTM model to handle this context-dependent input selection. This means that, for a given context detected by the ontology model (such as `isInFlowDirectionContextFor` or `isInLocationContextFor`), the observations from the sensors involved in the detected context are transformed into a feature vector that serves as input to the input layer of the

LSTM model.

For example, consider a scenario with hydrometric stations. The ontology model detects that `05BJ004_Station` is in flow direction context for `05BJ010_Station`. This context is crucial for flood prediction. The observations from these stations might include:

- `05BJ004_Station`: Water level = 3.1 meters, Flow rate = 25.4 cubic meters per second, IsCriticalEvent = 1
- `05BJ010_Station`: Water level = 61 meters, Flow rate = 22.1 cubic meters per second, IsCriticalEvent = 0

The output of the ontology might look like the Figure 10.



```xml
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ004_Station">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#HydrometricStation"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Platform"/>
    <onto-ai-sensors-base:hosts rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ004_Sensor"/>
    <criticalEvents:isInFlowDirectionContextFor rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BH004_Station"/>
    <criticalEvents:isInFlowDirectionContextFor rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ010_Station"/>
    <onto-ai-sensors-base:hasName>05BJ004</onto-ai-sensors-base:hasName>
    <onto-ai-sensors-base:thresholdWaterLevel rdf:datatype="http://www.w3.org/2001/XMLSchema#float">2.14</onto-ai-sensors-base:thresholdWaterLevel>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ004_Sensor">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Sensor"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#System"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#WaterLevelSensor"/>
    <onto-ai-sensors-base:hasFeatureOfInterest rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#RiverbedFeature"/>
    <onto-ai-sensors-base:isHostedBy rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ004_Station"/>
    <onto-ai-sensors-base:madeObservation rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation_237_05BJ004"/>
    <onto-ai-sensors-base:observes rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#ObservablePropertyWaterLevel"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation_237_05BJ004">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#CriticalEvent"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation"/>
    <onto-ai-sensors-base:hasResult rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Result_237_05BJ004"/>
    <onto-ai-sensors-base:madeBySensor rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ004_Sensor"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Result_237_05BJ004">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Result"/>
    <onto-ai-sensors-base:isResultOf rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation_237_05BJ004"/>
    <onto-ai-sensors-base:valueOfFlowRate rdf:datatype="http://www.w3.org/2001/XMLSchema#float">25.4</onto-ai-sensors-base:valueOfFlowRate>
    <onto-ai-sensors-base:valueOfWaterLevel rdf:datatype="http://www.w3.org/2001/XMLSchema#float">3.1</onto-ai-sensors-base:valueOfWaterLevel>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ010_Sensor">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Sensor"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#System"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#WaterLevelSensor"/>
    <onto-ai-sensors-base:hasFeatureOfInterest rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#RiverbedFeature"/>
    <onto-ai-sensors-base:isHostedBy rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ010_Station"/>
    <onto-ai-sensors-base:madeObservation rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation_237_05BJ010"/>
    <onto-ai-sensors-base:observes rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#ObservablePropertyWaterLevel"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation_237_05BJ010">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation"/>
    <onto-ai-sensors-base:hasResult rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Result_237_05BJ010"/>
    <onto-ai-sensors-base:madeBySensor rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#05BJ010_Sensor"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Result_237_05BJ010">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Result"/>
    <onto-ai-sensors-base:isResultOf rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/onto-ai-sensors-base#Observation_237_05BJ010"/>
    <onto-ai-sensors-base:valueOfFlowRate rdf:datatype="http://www.w3.org/2001/XMLSchema#float">22.1</onto-ai-sensors-base:valueOfFlowRate>
    <onto-ai-sensors-base:valueOfWaterLevel rdf:datatype="http://www.w3.org/2001/XMLSchema#float">60.0</onto-ai-sensors-base:valueOfWaterLevel>
</owl:NamedIndividual>
```

Figure 10 – Ontology assertions example

In terms of implementation, this output is deserialized and converted to a JSON representation, as shown next:

```
{
  "station": "05BJ004_Station",
  "waterLevel": 3.1,
  "flowRate": 25.4,
  "isCriticalEvent": 1,
  "isInFlowDirectionContextFor": "05BJ010_Station"
},
{
  "station": "05BJ010_Station",
  "waterLevel": 61,
  "flowRate": 22.1,
  "isCriticalEvent": 0
}
```

Before being used as input for the LSTM model, the combined feature vector is normalized (except for the boolean values, which remain 0 or 1). Assuming min-max normalization, the normalized feature vector input to the LSTM model would be:

$$x_t = \left[\frac{3.1 - 2.0}{100.0 - 2.0}, \frac{25.4 - 20.0}{30.0 - 20.0}, 1, \frac{61 - 2.0}{100.0 - 2.0}, \frac{22.1 - 20.0}{30.0 - 20.0}, 0\right]$$

This normalization intends to ensures that all input features are on a similar scale. The normalized feature vector would be:

$$x_t = [0.0112, 0.54, 1, 0.6020, 0.21, 0]$$

This feature vector captures the relevant context-aware information, including the boolean values, allowing the LSTM model to make more accurate predictions regarding potential flood events. The LSTM model processes this input to predict whether a critical event, such as a flood, is likely to occur based on the historical patterns and current observations.

For prediction, the LSTM model outputs a probability. This probability is compared to a threshold (0.6). If the probability is greater than or equal to the threshold, the event is classified as critical (1), otherwise as non-critical (0).

The hyperparameters were chosen to optimize the prediction of flood events based on water level and precipitation sensor data. The number of LSTM units was set to 64. This value strikes a balance between capturing complex temporal patterns in the data and maintaining computational efficiency, without quickly leading to overfitting. A dropout rate of 0.2 was used to mitigate overfitting by randomly dropping 20% of the neurons during training, promoting generalization. The number of hidden layers was set to two, providing a balance between representational capacity and computational efficiency. Two layers allow for the capture of more complex and abstract temporal patterns in the data without making the model overly complex or difficult to train. The activation function for the output layer is ‘sigmoid’, which is suitable for binary classification tasks like this, as it converts the output into a probability between 0 and 1.

For training, the model was set to run for 10 epochs. This number of epochs allows for sufficient training iterations to learn the data patterns without overextending the training time or causing overfitting. The batch size was set to 256, a commonly used value that balances the trade-off between gradient estimation quality and computational efficiency, especially when utilizing GPU (Graphics Processing Unit) resources. The validation split was set to 0.2, meaning that 20% of the data is reserved for validation while 80% is used for training. This more conservative split intend to ensure that a larger portion of the data is available for training. It also provides a sufficiently large validation set to evaluate the model’s performance.

During the walk-forward validation process, a window size of 100 was chosen. This window size is large enough to capture relevant temporal dependencies in the data but not so large that it becomes computationally expensive. The step size of 1 intend to ensure a detailed and continuous evaluation of the model’s performance over time, though it increases computational demands. Overall, these hyperparameter choices are grounded in empirical evidence during our development, experimentation and common practices, aiming to balance training efficacy, computational efficiency, and model generalization capabilities.

The model utilized, consists of two LSTM layers with 64 hidden units in each layer. The output of the first LSTM layer ($h_t^{(1)}$) serves as the input to the second LSTM layer. The superscripts denote each LSTM layer’s parameters (weights and biases). The input at each time step (x_t) is processed by the first layer, and the hidden states ($h_t^{(1)}$) become the input to the second layer. The final hidden state ($h_t^{(2)}$) output is passed through an additional output layer for binary classification.

Layer 1

$$
\begin{aligned}
i_t^{(1)} &= \sigma(W_{ii}^{(1)} x_t + b_{ii}^{(1)} + W_{hi}^{(1)} h_{t-1}^{(1)} + b_{hi}^{(1)}) \\
f_t^{(1)} &= \sigma(W_{if}^{(1)} x_t + b_{if}^{(1)} + W_{hf}^{(1)} h_{t-1}^{(1)} + b_{hf}^{(1)}) \\
o_t^{(1)} &= \sigma(W_{io}^{(1)} x_t + b_{io}^{(1)} + W_{ho}^{(1)} h_{t-1}^{(1)} + b_{ho}^{(1)}) \\
g_t^{(1)} &= \tanh(W_{ig}^{(1)} x_t + b_{ig}^{(1)} + W_{hg}^{(1)} h_{t-1}^{(1)} + b_{hg}^{(1)}) \\
c_t^{(1)} &= f_t^{(1)} \odot c_{t-1}^{(1)} + i_t^{(1)} \odot g_t^{(1)} \\
h_t^{(1)} &= o_t^{(1)} \odot \tanh(c_t^{(1)})
\end{aligned}
$$

Layer 2

$$
\begin{aligned}
i_t^{(2)} &= \sigma(W_{ii}^{(2)} h_t^{(1)} + b_{ii}^{(2)} + W_{hi}^{(2)} h_{t-1}^{(2)} + b_{hi}^{(2)}) \\
f_t^{(2)} &= \sigma(W_{if}^{(2)} h_t^{(1)} + b_{if}^{(2)} + W_{hf}^{(2)} h_{t-1}^{(2)} + b_{hf}^{(2)}) \\
o_t^{(2)} &= \sigma(W_{io}^{(2)} h_t^{(1)} + b_{io}^{(2)} + W_{ho}^{(2)} h_{t-1}^{(2)} + b_{ho}^{(2)}) \\
g_t^{(2)} &= \tanh(W_{ig}^{(2)} h_t^{(1)} + b_{ig}^{(2)} + W_{hg}^{(2)} h_{t-1}^{(2)} + b_{hg}^{(2)}) \\
c_t^{(2)} &= f_t^{(2)} \odot c_{t-1}^{(2)} + i_t^{(2)} \odot g_t^{(2)} \\
h_t^{(2)} &= o_t^{(2)} \odot \tanh(c_t^{(2)})
\end{aligned}
$$

In the first LSTM layer (Layer 1), $i_t^{(1)}$, $f_t^{(1)}$, and $o_t^{(1)}$ represent the input gate, forget gate, and output gate at time step t. These gates, calculated using sigmoid activation, regulate the flow of information. The candidate cell state $g_t^{(1)}$, determined by the hyperbolic tangent activation function, signifies new information that could be stored in the cell state. The updated cell state $c_t^{(1)}$ considers the forget and input gates, along with the candidate cell state and the previous cell state ($c_{t-1}^{(1)}$), denoted by $\odot$ for element-wise multiplication. The updated hidden state $h_t^{(1)}$ is a function of the output gate and the hyperbolic tangent of the updated cell state.

The second LSTM layer (Layer 2) is analogous, with its own gates and states functioning similarly to those in the first layer. The final hidden state ($h_t^{(2)}$) is then passed through an additional layer designed for binary classification. The model's output is a real number between 0 and 1, representing the probability that a given event is a Critical Event.

For classification purposes, this probability is converted to discrete values: events with probabilities greater than or equal to a threshold are classified as Critical Events (1), and those below the threshold are classified as non-critical events (0). During testing, it was observed that some events near the boundary between critical and non-critical water level values were misclassified. To address this, a threshold of 0.6 was found to be the most effective for improving performance in these edge cases, resulting in a better distinction between critical and non-critical events.

To split and create the input sets for training, testing, and validating the model, we used the Walk-Forward Validation (WFV) technique. WFV is a time-series cross validation method useful for time-ordered data where temporal sequence matters. It provides a realistic evaluation of how well a model will generalize to future unseen data(32). Algorithm 1 defines the steps of the WFV algorithm, and Figure 11 illustrates the intuition behind the algorithm steps.

Algorithm 1 Walk-Forward Cross-Validation

```
Input: data, model, window_size, horizon, metric
Output: evaluation_scores
    Initialization: starting_index = 0
 1: evaluation_scores = []
 2: while starting_index + window_size + horizon $\leq$ length(data) do
 3:     training_data = data[starting_index : starting_index + window_size]
 4:     testing_data = data[starting_index + window_size : starting_index + window_size + horizon]
 5:     model.fit(training_data)
 6:     predictions = model.predict(testing_data)
 7:     score = metric(testing_data, predictions)
 8:     evaluation_scores.append(score)
 9:     starting_index = starting_index + 1
10: end while
11: return evaluation_scores
```

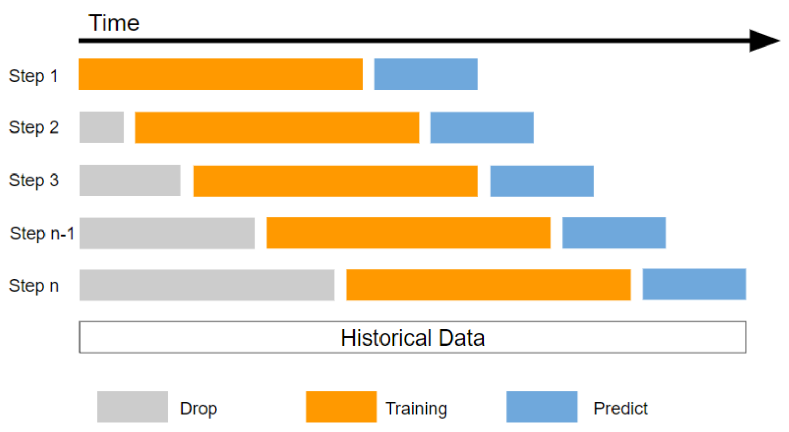


Figure 11 – Walk Forward Validation intuition diagram.

WFV repeatedly trains and tests the model on different subsets of the time series data, moving forward one step at a time. At the beginning of the algorithm, an empty list evaluation_scores is set up to store the evaluation results, and a variable starting_index representing the starting point of the sliding window over the time series data is initialized to 0.

The algorithm enters a loop that continues until the sliding window cannot move any further along the time series data. Within the loop, we split the data into two parts: *training_data* and *testing_data*. The training set consists of a fixed number of consecutive data points determined by the window_size parameter, while the testing set contains the data points immediately following the training set, up to a specified forecast *horizon*. The LSTM model *model* is trained on the training data using the $model.fit()$ function. After training, the trained model is used to make predictions on the testing data using the *model.predict*() function.

The performance of the model's predictions is evaluated using a specified evaluation *metric*, (in this case, precision, recall, and *f1-score*) comparing the predicted values with the actual values in the testing set. The obtained evaluation score is then appended to the evaluation_scores list.

Finally, the starting index is moved forward by one step to slide the window along the time series data. Once the loop completes, the algorithm evaluates the model's performance at each step of the walk-forward cross-validation process. It returns the list of evaluation scores *evaluation_scores*, providing insight into how well the model performs over different time periods within the dataset.

3.5 Alert Layer

The **Alert Layer** can be activated in two ways. Firstly, it activates when the ontology detects a Critical Event. This can occur if the measured value at a sensor station exceeds its defined threshold or if a Critical Event is detected at a station that has a context rule indicating its influence on another station. Secondly, the Alert Layer is activated when the AI layer predicts the occurrence of a Critical Event at a sensor station. These mechanisms ensure that alerts are both contextually relevant and based on predictive analytics, enhancing the reliability and responsiveness of the monitoring system.

3.6 Solution Model

We define the elements of our framework as follows:

$S = \{s_0, s_1, \ldots, s_n\}$ the set of Sensors in the domain of interest;

$O_{s_i} = \{o_0, o_1, \ldots, o_p\}$ the set of Observations performed by each sensor s_i;

$A_{o_k} = \{a_0, a_1, \ldots, a_q\}$ the set of Attributes associated to each observation o_k;

$\epsilon_{s_i} = \{\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_r\}$ the set of Critical Events associated to a sensor s_i;

A sensor is defined as: $s_i = \{\{o_0, o_1, ..., o_p\}, \{\varepsilon_0, \varepsilon_1, ..., \varepsilon_r\}\}$, such that $p, r \geq 0$.

As defined with the previous sets, each measure detected by a sensor is an Observation, and each sensor has its own set of Critical Events.

An Observation o_k , $\forall o \in O_{s_i}$, is given by:

$o_k = \{(a_0, v_{a_0}), (a_1, v_{a_1}), \ldots, (a_l, v_{a_q})\}$ such that $v_{a_q} \in \mathbb{R}$

An Critical Event ε_r , $\forall \varepsilon_{s_i} \in \epsilon$, is given by:

$\varepsilon_r = \{(a_0, vi_{a_0}, vf_{a_0}), (a_1, vi_{a_1}, vf_{a_1}), \ldots, (a_q, vi_{a_q}, vf_{a_q})\}$ such that $vi_{a_q}, vf_{a_q} \in \mathbb{R}$

For example, a given hydrometric sensor can have Observations such as:

o_0 ={(date, 2013-06-10), (water_level, 2.60), (flow_rate, 20.2)}
o_1 ={(date, 2013-06-10), (water_level, 2.85), (flow_rate, 21.7)}
o_2 ={(date, 2013-06-21), (water_level, 22.10), (flow_rate, 21.0)}

And some Critical Events defined as:

ε_0 ={(water_level, 0, 3),(flow_rate, 0, 15)}
ε_1 ={(water_level, 3, 4), (flow_rate, 15, 30)}
ε_2 ={(water_level, 4, 100)}

Considering vi and vf the initial and final attribute values that limit the classification of a critical event, and given $o_k = \{(a_0, v_{a_0}), (a_1, v_{a_1}), \ldots, (a_q, v_{a_q})\}$, and $\varepsilon_r = \{(a_0, vi_{a_0}, vf_{a_0}), (a_1, vi_{a_1}, vf_{a_1}),$
$\ldots, (a_q, vi_{a_q}, vf_{a_q})\}$, is said to be equivalence between o_k and ε_r ($o_k \equiv \varepsilon_r$) when:

$\exists\ \mathbf{M} \subset o_k \mid \mathbf{M} = \varepsilon_r$, wich means

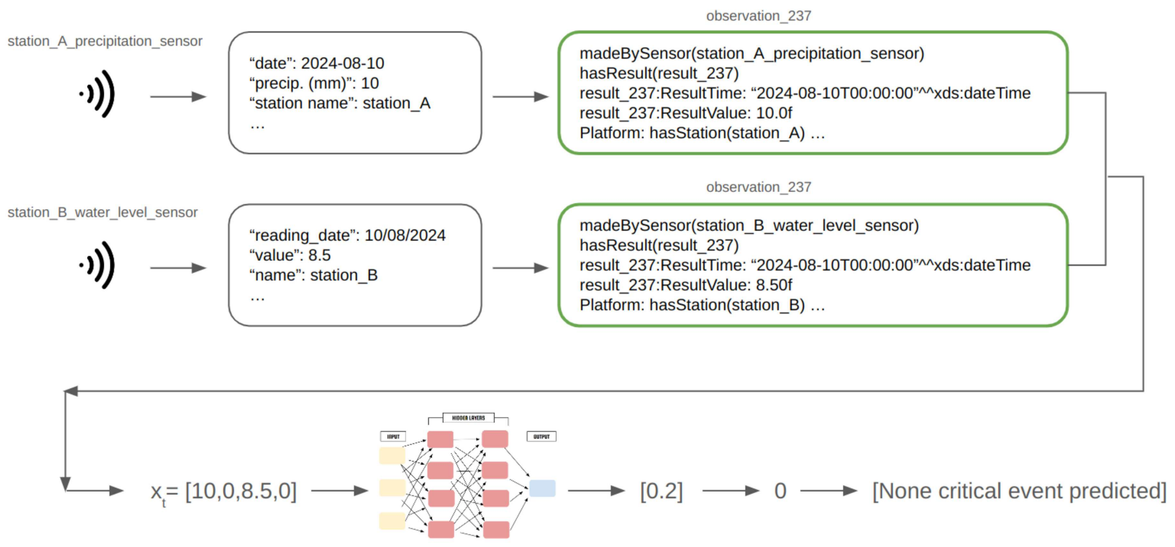
$\forall (a_q, vi_{a_q}, vf_{a_q}) \in \varepsilon_r\ \exists (a_q, v_{a_q}) \in o_k \mid vi_{a_q} \leq v_{a_q} \leq vf_{a_q}$

That means that, in the ontology layer, an observation is classified as a critical event if the values of all attributes in a defined critical event are detected in an observation. For example, the observation o_2 is classified as a critical event ε_2 just because its attribute *water_level* has a value greater than 4 and less than 100.

After the context rules are applied, all observations grouped by context are sent to the AI Layer. In the AI Layer, each feature of the observation is used as input to the AI model. The AI model then outputs the probability that this contextualized input means a critical event for the target station. This process intends to ensure that predictions are informed by both the raw data and its relevant contextual information, leading to more accurate and reliable assessments of potential critical events.

It's worth reinforcing that critical events detected directly by threshold rules are directly forwarded to the alert layer. Meanwhile, Observations that do not violate the Threshold Rules feed the context that serves as input for the prediction model.

In Figure 12, the system processes data from two sensors at a given moment, labeled as time "t." The sensor at Station A, responsible for monitoring precipitation, records a rainfall of 10 mm. Meanwhile, the sensor at Station B, which tracks water levels, registers a reading of 8.5. These raw data points are converted into observations, capturing the sensor information, results, timestamp, and station details. When these observations are processed by the system, no critical event is detected in either location. This means that, although the system has received data, neither Station A nor Station B is experiencing an event that exceeds the predefined thresholds for a critical situation. Furthermore, the system does not predict any upcoming critical event at this moment, as the probability generated by the model (0.2) remains too low to trigger any further actions. As a result, no alert is activated, since neither a detection of an ongoing event nor a prediction of a future event is warranted.


station_A_precipitation_sensor
station_B_water_level_sensor
observation_237
observation_237
x_t= [10,0,8.5,0]
HIDDEN LAYERS
INPUT
OUTPUT
[0.2]
0
[None critical event predicted]

```
"date": 2024-08-10
"precip. (mm)": 10
"station name": station_A
...
```

```
madeBySensor(station_A_precipitation_sensor)
hasResult(result_237)
result_237:ResultTime: "2024-08-10T00:00:00"^^xds:dateTime
result_237:ResultValue: 10.0f
Platform: hasStation(station_A) ...
```

```
"reading_date": 10/08/2024
"value": 8.5
"name": station_B
...
```

```
madeBySensor(station_B_water_level_sensor)
hasResult(result_237)
result_237:ResultTime: "2024-08-10T00:00:00"^^xds:dateTime
result_237:ResultValue: 8.50f
Platform: hasStation(station_B) ...
```

Figure 12 – Example of Observations with No Critical Event Detection or Prediction.

In Figure 13, the system processes new data from the same two sensors, this time at a later moment, called time "t+1." At this point, the sensor at Station A shows a dramatic increase in rainfall, recording 120 mm, which is above the threshold for a critical event. As a result, the system detects that a critical event is actively occurring at Station A, meaning that the situation has already escalated into a hazardous event. In parallel, the water level sensor at Station B shows an increase to 9.5. Based on the data from both stations, the system applies its prediction model, which calculates a high probability (0.8) that a critical event is likely to occur at Station B in the near future. Therefore, while Station A is already experiencing a critical event, Station B is not yet in a critical state, but there is a strong indication that it could soon be affected. Consequently, an alert is triggered, reflecting both the detection of the current critical event at Station A and the prediction of a possible upcoming critical event at Station B. This distinction between detection (an event already happening) and prediction (an event anticipated based on current data) is a key aspect of the system's functionality.

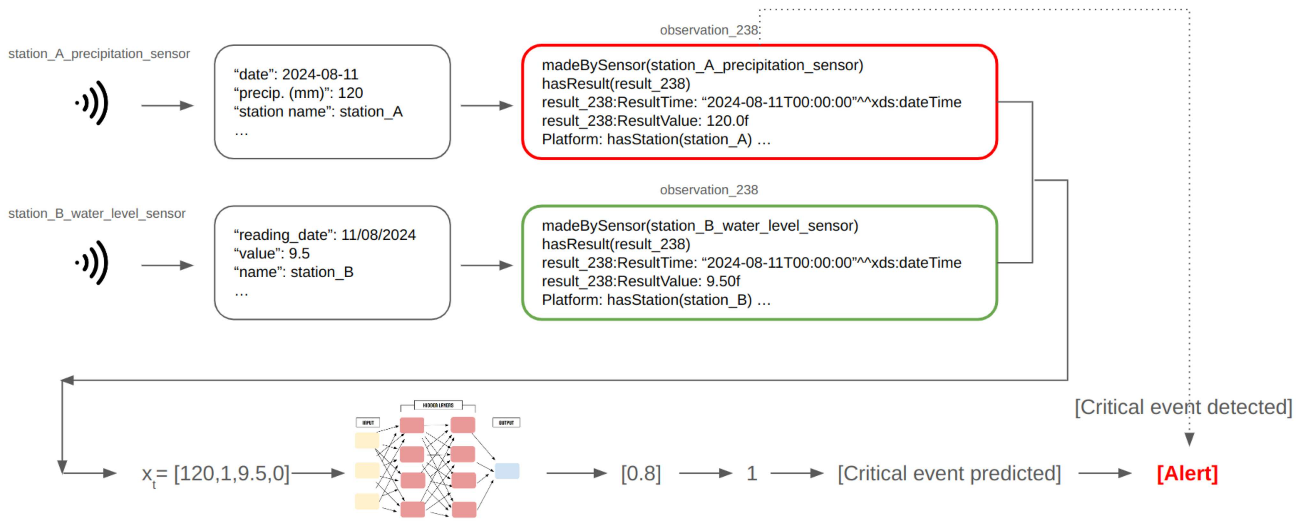

station_A_precipitation_sensor
"date": 2024-08-11
"precip. (mm)": 120
"station name": station_A
…
observation_238
madeBySensor(station_A_precipitation_sensor)
hasResult(result_238)
result_238:ResultTime: "2024-08-11T00:00:00"^^xds:dateTime
result_238:ResultValue: 120.0f
Platform: hasStation(station_A) …
station_B_water_level_sensor
"reading_date": 11/08/2024
"value": 9.5
"name": station_B
…
observation_238
madeBySensor(station_B_water_level_sensor)
hasResult(result_238)
result_238:ResultTime: "2024-08-11T00:00:00"^^xds:dateTime
result_238:ResultValue: 9.50f
Platform: hasStation(station_B) …
x_t = [120,1,9.5,0]
[0.8]
1
[Critical event predicted]
[Critical event detected]
[Alert]

Figure 13 – Example of Critical Event Detection and Prediction.

Once our proposal details are clarified, we can revisit the related works mentioned in Table 1 and update it with our proposed approaches for addressing the highlighted limitations of those works, as shown in Table 9.

Table 9 – Identified limitations of related works and our proposal's applied approaches

Study	Limitations and approaches
Gmati et al.(2019)	**Identified Limitations:** Difficulty in holistically considering complex interactions within the data. Challenges in addressing temporal dynamics. Dependence on high-quality historical data. Generalization across different domains can be difficult. **Proposal's approach:** The use of an extended ontology allows for a more comprehensive integration of diverse data sources, facilitating a richer contextual analysis. LSTM networks capture temporal dynamics and long-term dependencies, improving predictive accuracy. The framework is designed to generalize across different domains by adapting ontology definitions and context rules.
Zhao(2021)	**Identified Limitations:** Need for joint prediction of heterogeneous yet correlated outputs. Complexity in managing interdependencies among prediction outputs. Difficulty in continuously updating models to reflect real-time data changes. **Proposal's approach:** The extended ontology facilitates the integration of data from various sources, enabling the joint prediction of related events. The framework supports continuous model updates, allowing for real-time data integration and adaptation to new conditions.
Li et al.(2021)	**Identified Limitations:** Limited focus on specific LSTM architectures and their optimization for a particular hydrological dataset. Lack of exploration of broader applicability beyond specific dataset limitations. Emphasis on architectural comparison rather than practical deployment. **Proposal's approach:** The dissertation develops a flexible framework capable of adapting to different data types and domains, not restricted to hydrological data. The framework allows for practical deployment in various real-world scenarios, beyond just architectural optimization.
Al Qundus et al.(2022)	**Identified Limitations:** Dependence on relative values for disaster detection thresholds, making generalization across locations difficult. Requirement for specific historical data collection for new locations. Potential limitations in scalability and adaptability. **Proposal's approach:** The framework includes adaptive mechanisms for setting thresholds based on diverse data inputs, facilitating broader applicability. It minimizes dependence on specific historical data by utilizing a more generalized approach to data integration and analysis.
Nearing et al.(2024)	**Identified Limitations:** Global focus may overlook local nuances and precision in specific areas. Relies on extensive public datasets, which might not capture local specifics adequately. Challenges in providing actionable insights for localized flood predictions. **Proposal's approach:** Emphasizes localized data integration to enhance precision in predictions, addressing specific local challenges. The flexible framework supports various applications, offering context-specific predictions and insights across different domains, not limited to hydrology.

3.6.1 Final Considerations of the Chapter

In this chapter, we presented the proposed framework for sensor data integration and critical event detection and prediction. We outlined the key components of the framework, including the data, preprocessing, ontology, and AI layers, and explained how each layer contributes to the integration and processing of sensor data. The framework's ability to abstract, contextualize, and predict events from heterogeneous data sources was highlighted, demonstrating its potential for addressing the challenges of data diversity and real-time event prediction.

The architecture and processes defined in this chapter establish the foundation for the evaluation phase, which will be discussed in the following chapter. There, we will assess the performance of the proposed framework using real-world data and analyze its effectiveness in detecting and predicting critical events.

4 Feasibility Study

This chapter describes a feasibility study using real use case scenarios. The framework was utilized to integrate data and detect and predict events from sensor-produced data. In this feasibility study, the scope is defined within hydrological and hydrometric monitoring stations. The goal of a feasibility study is to assess whether a technology performs as claimed and is worth further development and investment (18).

This evaluation followed the GQM (Goal - Question - Metric) model (11), which determines that the study's objectives should be defined, followed by the research questions and metrics for evaluating the research questions. The scope of this evaluation and the Goal are described as follows: "**To analyze** the framework **for the purpose of** semantic data integration and event prediction **in relation to** sensors data **under the point of view of** decision-makers **in the context of** data from hydrological and hydrometric sensors".

The framework was evaluated considering both semantic data integration and event prediction. Based on the evaluation Goal and on the ontology's competence questions, one Research Question (RQ) was stated, and two secondary research questions (SRQ) were derived. The first secondary research question (SRQ1) is important to measure the capacity of semantically integrating data from multiple sources. The second secondary research question (SRQ2) investigates whether the decision-making system is achieving its goals by detecting and predicting events of interest.

(i) RQ – Can the framework semantically integrate data from multiple data sources to detect and predict events of interest from these data?
    - SRQ1 – Does using the proposed framework and implemented ontology model allow the integration of sensor data and add new data sources?
    - SRQ2 - Are the events of interest correctly detected and predicted?

Three metrics were used to answer the research questions. The metrics M1 and M2 aim to answer SRQ1, and M3 to answer SRQ2. The metrics are:

M1: The data representation of multiple and different sensors standardized as a semantic canonical representation.

M2: Number and diversity of sensor data sources integrated.

M3: AI model performance (precision, recall, and f1-score).

4.1 Data description

We obtained sensor data from hydrological and hydrometric domains. The data from hydrometric stations are available at https://wateroffice.ec.gc.ca/, and data from hydrological stations are available at https://acis.alberta.ca/acis/. Both are public data sources made available by the Government of Canada and Alberta. We also use data to validate event detection. The data is from a public data source available at https://www.gdacs.org/ and is made available by the Global Flood Detection System (GFDS)[1], which monitors floods worldwide using near-real-time satellite data.

The feasibility study considered data from seven hydrological stations (Jumpingpound Ranger, Elbow Ranger, Elbow Auto, Priddis, Calgary Springbank, Cop Upper, and Calgary International Airport), which monitor precipitation levels, and eight hydrometric stations (05BH004, 05BH010, 05BH015, 05BJ001, 05BJ004, 05BJ008, 05BJ010, and 05BM904), which monitor the water level of the riverbed to which they are associated. Figure 14 shows a map with the considered stations. The data is from 2005 to 2023. Table 10 shows the summary of the station's data, with the total number of observations and depicted atypical events.

The levels of precipitation considered "high" can vary depending on the region and specific context, such as local climate conditions and historical precipitation patterns. Generally, precipitation is categorized as follows: Light Rain: less than 10 mm per day; Moderate Rain: 10 mm to 35 mm per day; Heavy Rain: 35 mm to 50 mm per day; Very Heavy Rain: 50 mm to 100 mm per day; Extreme Rain: more than 100 mm per day. These categories are based on general meteorological guidelines and widely accepted standards by the scientific community and meteorological organizations, including the World Meteorological Organization (WMO) (79).

For water level classification, the Global Flood Detection System (GFDS), developed by the Joint Research Centre of the European Commission in collaboration with the Dartmouth Flood Observatory, classifies flood events by analyzing water surface metrics from satellite data. The system uses data from various Earth observation satellites to provide near-real-time information on flood signals, magnitude, and time series for around 10,000 observation sites worldwide (36, 25).

Both the sources and criteria cited above were used to label the data as precipitation-critical events and water-level-critical events (flood).

[1] https://www.copernicus.eu/en/global-flood-detection-system-data-products-specification

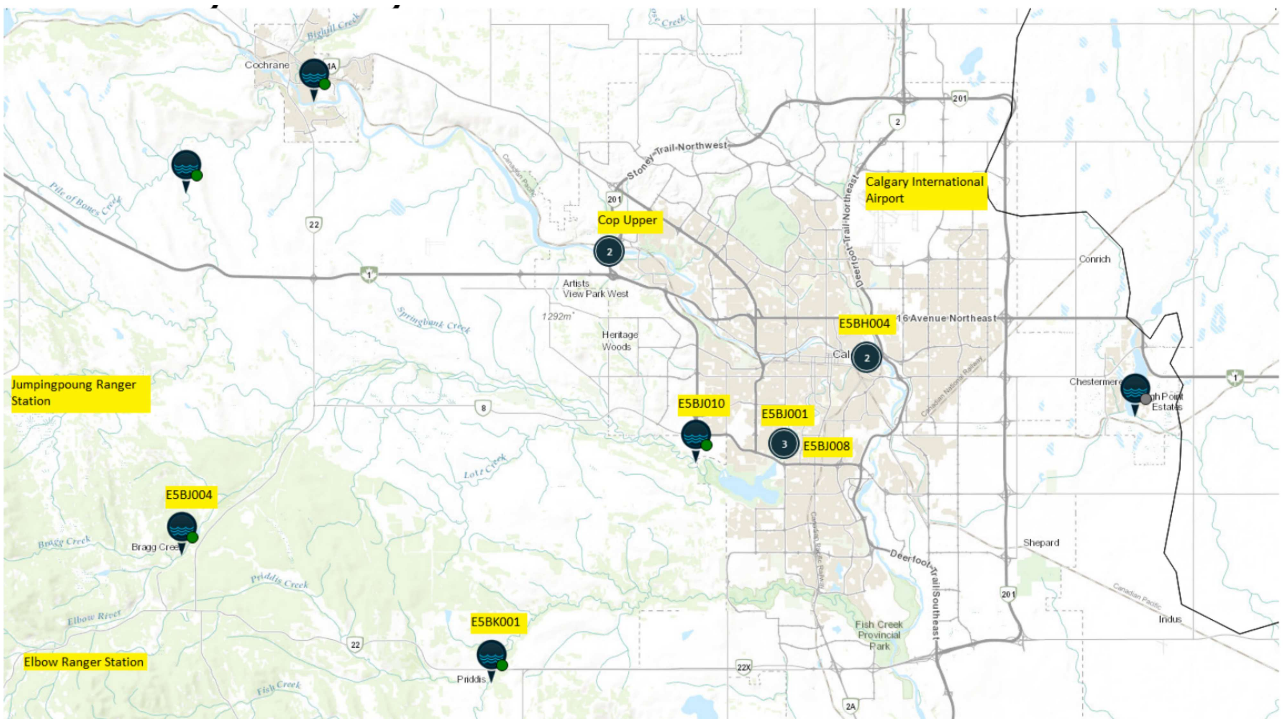


Figure 14 – Considered stations on the map. Source:(27)

Table 10 – Station's Observations Summary

Hydrological Stations			Hydrometric Stations		
Station	Total Observations	High Precipitation Observations	Station	Total Observations	High Water Level Observations
PriddisObservatory	6575	30	05BH004	6385	177
CICS	6575	34	05BJ001	6320	285
CSBA	6575	7	05BH010	6557	3
CopUpper	6575	36	05BM904	3798	0
ElbowRangerStation	6575	65	05BJ008	5993	1
JumpingpoundRangerStation	6575	40	05BJ004	6324	145
ElbowAuto	6575	41	05BH015	6130	346
			05BJ010	6030	708

Table 10 provides detailed data on the total number of observations and the instances of high precipitation and high-water levels recorded at hydrological and hydrometric stations.

For the hydrological stations, each station has consistently recorded a total of 6,575 observations. This means there is no absence of data on the observed period. The number of high precipitation observations varies across these stations. For example, CSBA had 7 high precipitation events observations, and Elbow Ranger Station reported the highest number of high precipitation observations with 65. This variability suggests differing levels of precipitation intensity or frequency at these locations, possibly influenced by local climatic conditions.

In contrast, the hydrometric stations exhibit variation in both the total number of observations and the count of high-water level events. Station 05BM904 had the lowest total observations at 3798 and did not record any high-water level events. Station 05BJ010 recorded the highest number of high-water level events at 708, indicating a higher susceptibility to flooding.

This summary indicates that the frequency of high precipitation varies for hydrological stations, suggesting localized weather patterns. Hydrometric stations show significant differences, reflecting diverse conditions and potential flood risks across different locations. These variations highlight the importance of tailored monitoring and response strategies to address specific regional and environmental conditions effectively.

Comparing Table 10 with Figure 15, it is evident that not all outliers are classified as Critical Events. For instance, in the case of station 05BJ008, Table 10 indicates only one critical event (a high-water level observation). However, Figure 15 shows a significant number of outliers for this station. This discrepancy highlights that outliers in the data do not necessarily correspond to Critical Events.

Analyzing the data behavior and its distribution over time, as displayed in the histograms in Figure 16, it is noticeable that the data obtained for station 05BJ010 exhibit a peculiar behavior. This behavior is characterized by two distinct clusters of frequency counts: one around the lower measured values and another around the higher measured values, with a significant gap in between. Upon examining the historical series through the graph in Figure 17, this peculiar behavior is confirmed. Delving deeper into the sensor data records associated with this station, the cause of this behavior could not be determined based on the obtained data. However, there is a disclaimer in the data repository [2] stating as a remark: "DATUM CHANGE OCCURRED ON APRIL 27, 2018. GAUGE WAS RELOCATED ON APRIL 27, 2018.". With this information in mind, we partitioned the data for this station, with the first partition from April 13, 2005, to April 26, 2018, and the second partition from April 27, 2018, to March 31, 2023. The boxplot in Figure 18 presents a visual partitioning of data for station 05BJ010, divided into two distinct periods: before and after the relocation of the gauge on April 27, 2018. This division helps account for potential shifts in measurement data caused by the relocation and a change in the datum, ensuring a more accurate interpretation of the sensor readings. By separating the data into these periods, the analysis captures any variations or anomalies that could have resulted from the gauge's move, providing clearer insights into water level behavior over time. Figure 19 depicts the historical data for station 05BJ010, illustrating the two partitions in the data. The data series is divided between the periods before and after the gauge relocation on April 27, 2018. This split highlights any changes in the data collection process and ensures that potential discrepancies, due to the gauge change, are considered during the analysis. This approach helps in understanding trends and differences in water level measurements across these two significant periods.

[2] Disponível em: `https://wateroffice.ec.gc.ca/report/remarks_e.html?type=h2oArc&stn=05BJ010&dataType=Daily&parameterType=Flow&first_year=1979&last_year=2022&mode=Graph&page=historical&year=2022&start_year=1850&end_year=2024`

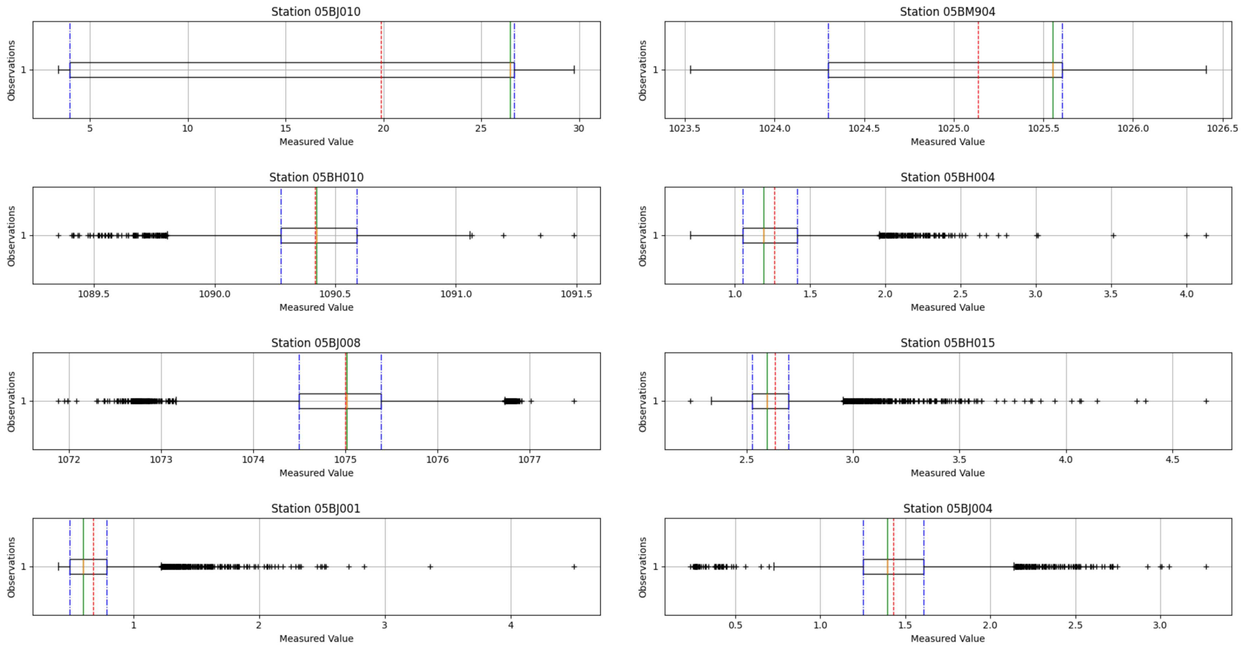


Figure 15 – Boxplot of Measured Values by station.

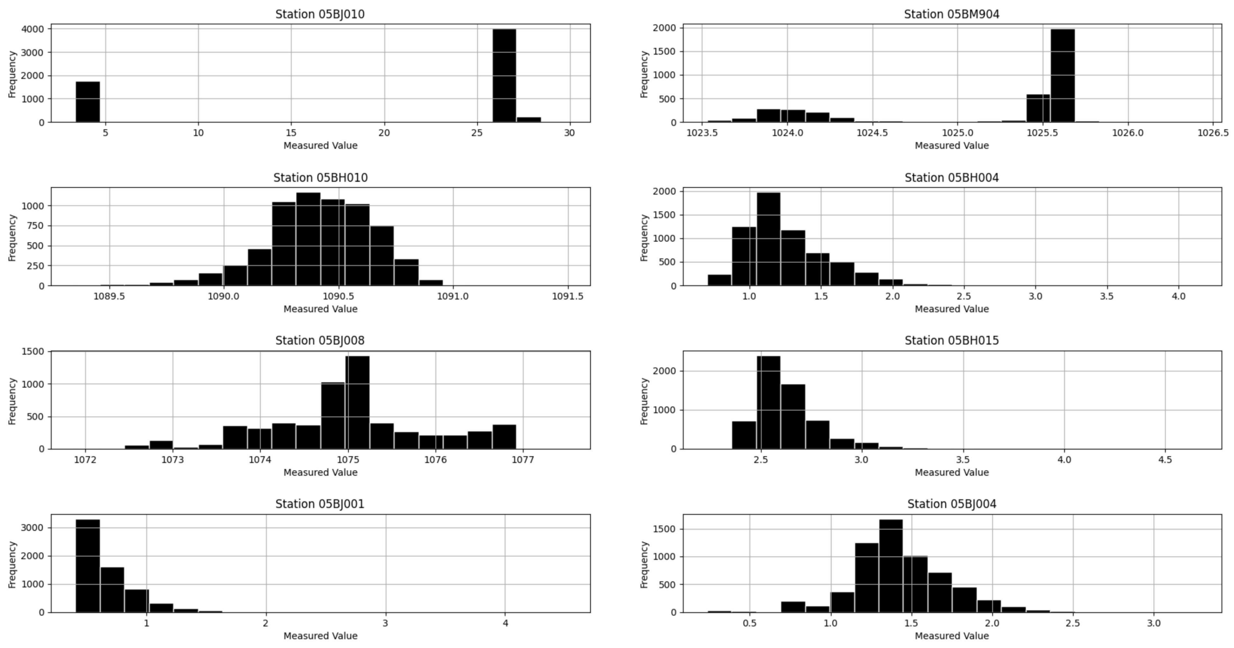


Figure 16 – Histogram of Measured Values by station.

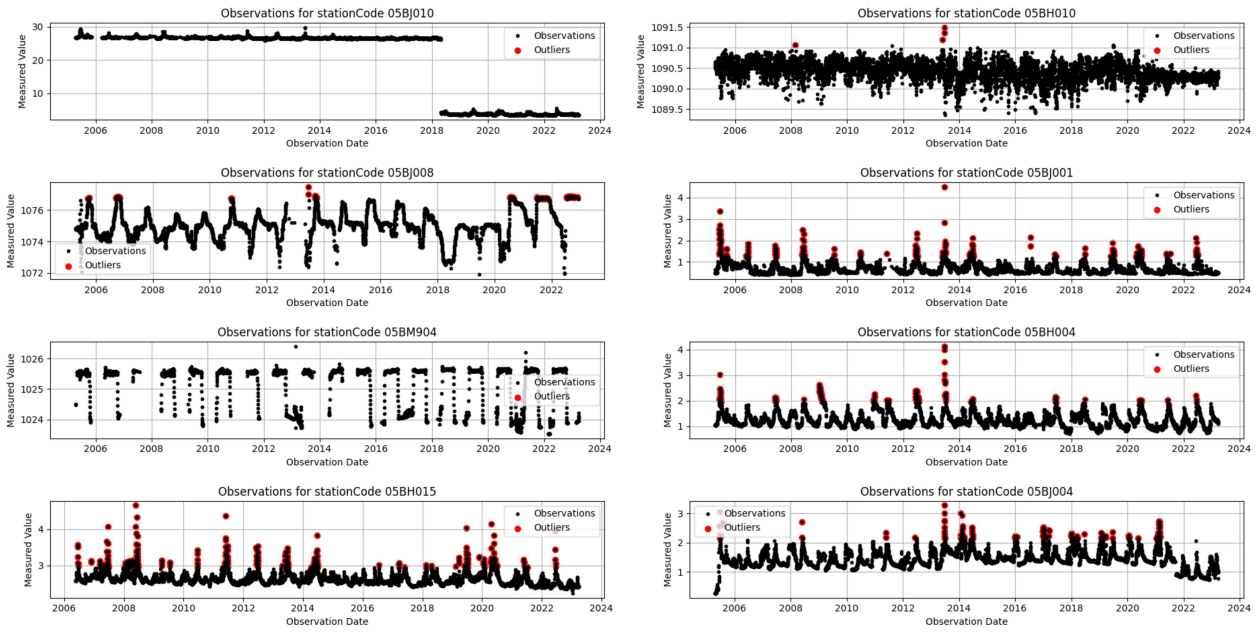


Figure 17 – Historical Data of Measured Values by station.

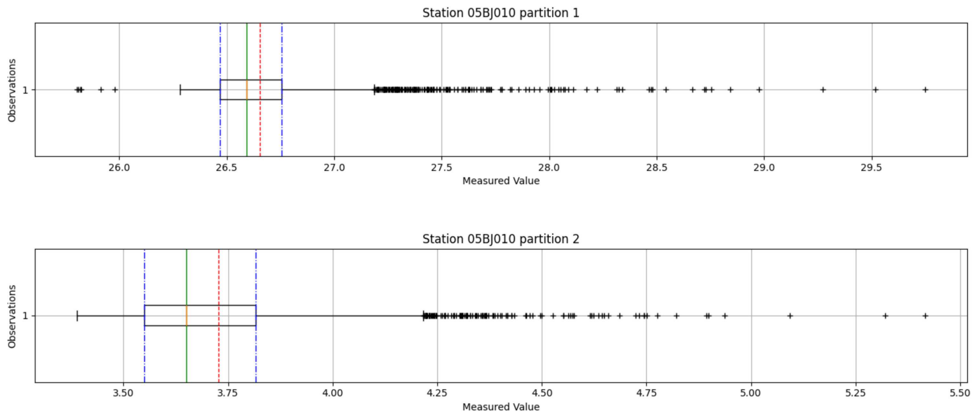


Figure 18 – Boxplot partitions for station 05BJ010.

Figure 20 presents a heatmap representing the occurrence of rainfall classified as heavy or above, based on the data from the observed hydrological stations. This heatmap allows visualization of daily rainfall patterns, highlighting the days and locations with the highest rainfall accumulations. The Elbow Ranger Station stands out significantly. This suggests that this station experiences more frequent and intense rainfall compared to other stations.

Finaly, Figure 21 displays a heatmap illustrating the occurrence of elevated water levels at the monitored hydrometric stations. The station 05BJ010 is particularly noteworthy. It has the highest number of high-water level events. This indicates a higher susceptibility to flooding at this station observed consistently over the monitoring period.

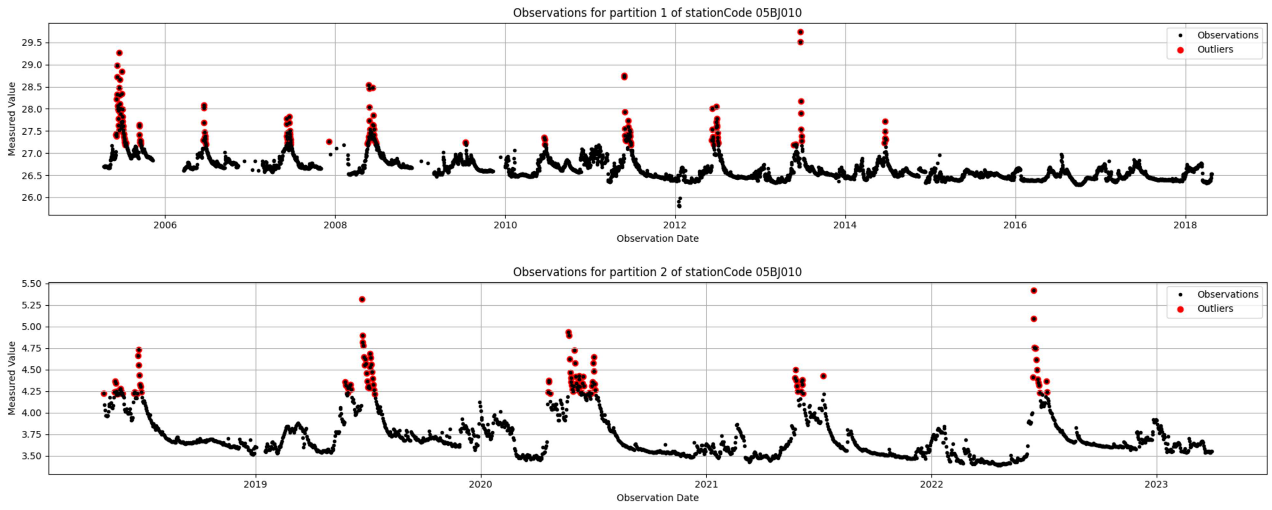


Figure 19 – Historical Data partitions for station 05BJ010.

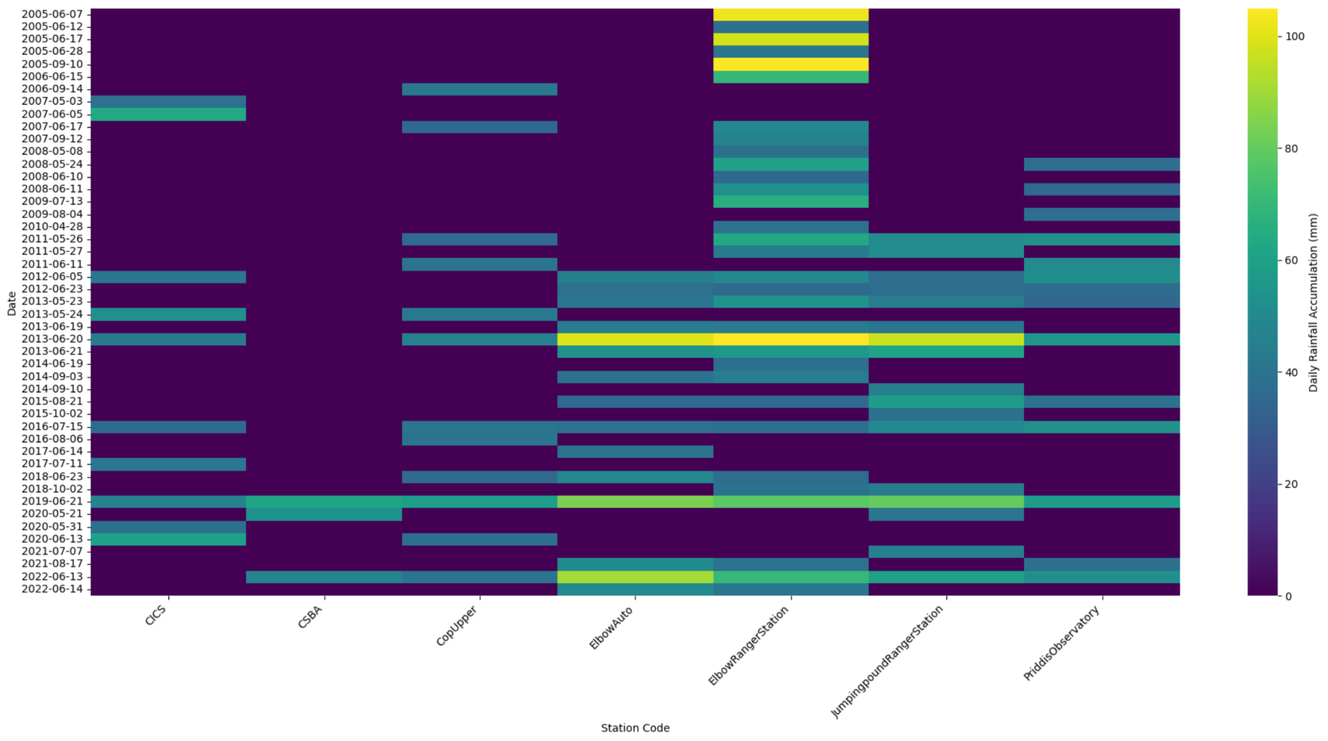


Figure 20 – Heatmap of Daily Rainfall Accumulation by Station.

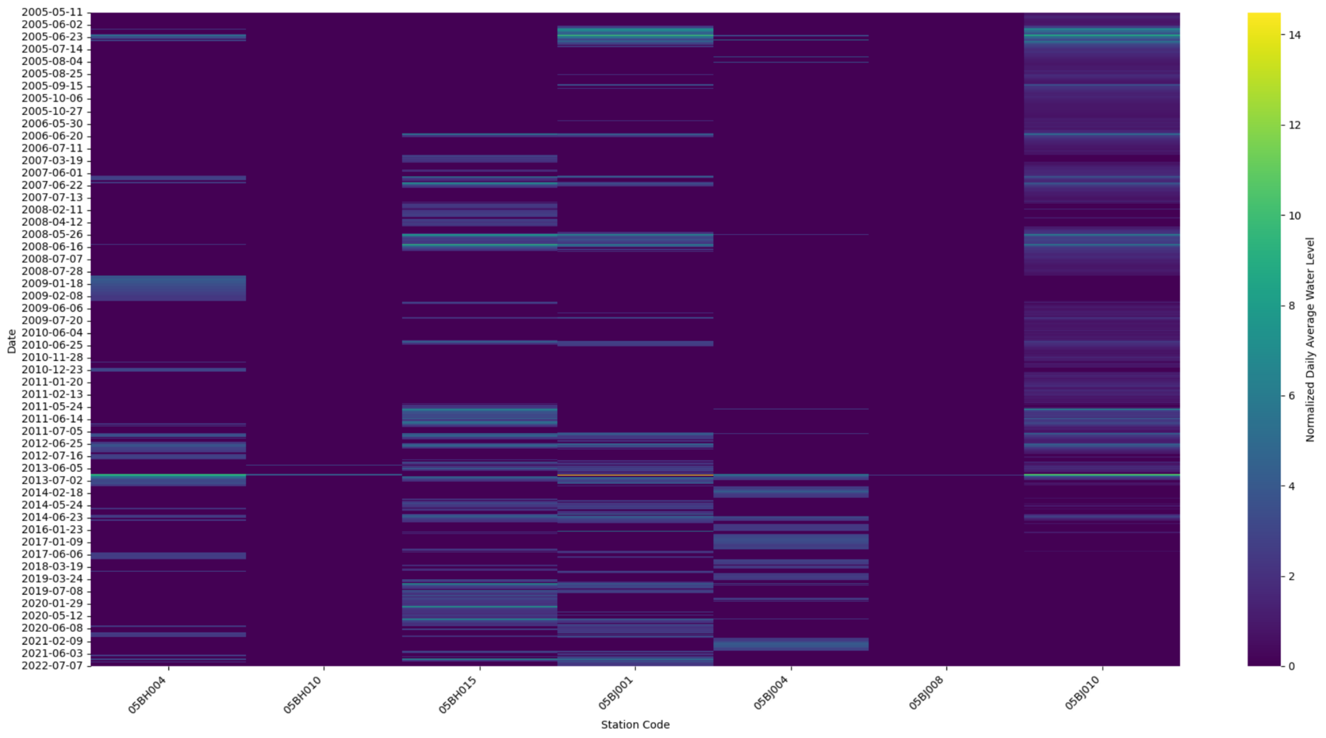


Figure 21 – Heatmap of Daily Average Water Levels by Station.

Data provided by the hydrological sensors contains the following attributes: *"Station Name", "Date (Local Standard Time)", "Precip. (mm)", "Precip. Accumulated (mm)"*, among others. The hydrometric sensors have the attributes: *ID, PARAM, Date, and Value*, where the *ID* feature represents the station identification and *PARAM* means the parameter monitored by the station (water level or discharge flow).

4.2 AI model definition

Given these data characteristics presented in the previous subsection, employing an LSTM model rather than statistical analysis techniques like ARIMA or SARIMA may be more suitable for the current scenario. Several studies (69, 22, 12, 21, 82) outline the advantages and disadvantages of these models comparatively, and Table 11 summarizes some of them.

- Linear: LSTMs are capable of capturing linear patterns in data, but their true strength lies in capturing non-linear and complex patterns. Therefore, their ability to model linear relationships can be considered good, but it's not their main advantage.
- Seasonal: While LSTMs can capture seasonality in time series data, they are more suited for modeling long-term patterns and long-range dependencies. Compared to statistical models like ARIMA and SARIMA, which are specifically designed for modeling seasonality, LSTMs can be considered good at this feature but perhaps not as specialized.

Table 11 – Comparison of Time Series Analysis Models

Data Characteristic	**ARIMA**	**SARIMA**	**LSTM**
Linear	Excellent	Excellent	Good
Seasonal	Moderate	Excellent	Good
Non-linear	Moderate	Moderate	Excellent
Large Volume of Data	Moderate	Moderate	Excellent
Computational Complexity	Low	Moderate	High
Interpretability	High	High	Low
Robustness to Outliers	Moderate	Moderate	Low
Handling of Missing Data	Moderate	Moderate	Low
Handling of Multivariate Series	No	No	Possible
Ease of Implementation	High	Moderate	Moderate
Manual Adjustment Requirements	Low	Moderate	Low
Generalization Capability	Low	Moderate	High
Adaptability to Changes	Low	Moderate	High
Efficiency in High-Frequency Data	Low	Moderate	High
Hyperparameter Sensitivity	Low	Moderate	High

Source: ChatPDF GmbH. **Summarized data characteristics when comparing ARIMA, SARIMA and LSTM models**. GPT-3.5. Artificial Intelligence. https://www.chatpdf.com/. Acessed on 2024-02-14

- Non-linear: One of the biggest advantages of LSTMs is their ability to model non-linear and complex patterns in time series data. This includes capturing long-term dependencies, non-linear relationships between variables, and irregular sequential patterns. Therefore, their classification as excellent in this feature is justified.

- Large Volume of Data: LSTMs are highly effective at handling large volumes of time series data. They are designed to automatically learn complex patterns from large datasets and have been widely used in applications with large data volumes, such as stock market prediction, traffic forecasting, and weather prediction.

- Computational Complexity: Training LSTM models can be computationally intensive, especially when working with large datasets and complex architectures. LSTM training typically requires robust hardware and can be time-consuming, especially if many training iterations are needed to achieve good performance.

- Interpretability: While LSTMs are powerful in modeling complex patterns, their internal structure is difficult to interpret. They are considered ̈black boxes,̈ where it's challenging to understand exactly how patterns are learned and represented. This can be a limitation in situations where interpretability is important.

- Robustness to Outliers: LSTMs are sensitive to outliers in data. Extreme values can distort predictions and introduce significant errors. Therefore, they may be considered less robust to outliers compared to some statistical models that can better handle such situations.

- Handling Missing Data: LSTMs do not naturally handle missing data. If time series data contains gaps or missing values, this can be a challenge for the LSTM to learn accurate patterns. Careful handling of missing data is required before feeding them into an LSTM.

- Handling Multivariate Series: Although LSTMs were originally designed for univariate time series, they can be extended to handle multivariate series. However, this requires modifications to the LSTM architecture and preprocessing of data to consider multiple input variables.

- Ease of Implementation: Implementing an LSTM requires knowledge of deep learning and can be more complex than implementing traditional statistical models like ARIMA. However, there are many deep learning frameworks and libraries available that facilitate LSTM implementation.

- Manual Tuning Requirements: Since LSTMs are capable of automatically learning complex patterns from data, manual tuning requirements are generally low compared to statistical models that may require manual parameter tuning.

- Generalization Capability: LSTMs have a natural ability to generalize and can be effective in time series prediction across different contexts and domains, provided they have been trained on representative and sufficiently diverse data.

- Adaptability to Changes: One of the advantages of LSTMs is their ability to adapt to changes in data patterns over time. They can adjust their predictions based on new observed information.

- Efficiency in High-Frequency Data: LSTMs are capable of efficiently handling high-frequency data, such as time series with a high temporal sampling rate. They can capture short and long-term patterns in such data effectively.

- Hyperparameter Sensitivity: The performance of LSTMs can be highly sensitive to model hyperparameters, such as the number of layers, memory size, learning rate, etc. Careful selection and tuning of these hyperparameters are necessary to achieve the best performance of the LSTM.

4.3 Implementation

Based on the solution presented in Chapter 3 and the data described in the previous section, we've outlined the structure and implementation of our solution. Since the proposal suggests a modular solution with independent yet interconnected parts, a microservices architecture is an ideal fit (44). The advantages of using microservices are summarized in Table 12.

Table 12 – Microservices using advantages

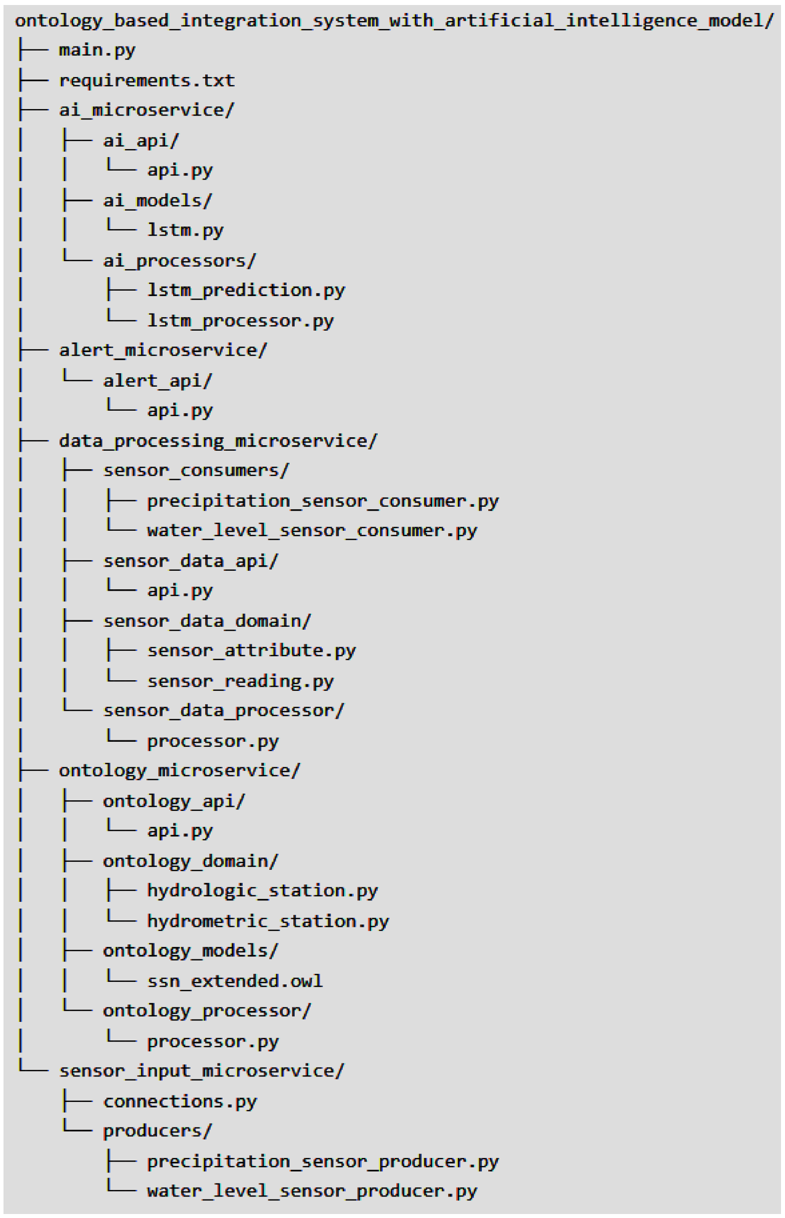
Characteristic	**Reasoning**
Modularity	Enable independent development and scaling of solution components.
Scalability	Specific parts of the application can be scaled independently, optimizing performance.
Technology Fit	Each microservice can use the most suitable technology stack for its functionality.
Maintenance	Allow for easier maintenance and continuous evolution of the solution.
Resilience	Failures are isolated to individual components, ensuring overall system resilience.
Integration	Simplified integration with external systems and adaptability to changing requirements.

By adopting a microservices approach, the system is broken down into smaller, standalone services that work together where each service focuses on a specific task.

The primary advantage is increased flexibility and ease of management. Each service can be developed, updated, and scaled independently without affecting others, allowing quick adaptation to change and the addition of new features without disrupting the entire system. This flexibility proved necessary during development, as evidenced by our switch from using Auto-encoders for the AI model without requiring changes beyond the AI microservice.

Microservices also enhance scalability. If a particular part of the system demands more resources, we can scale up that specific service rather than the entire system, optimizing performance and resource usage.

Furthermore, each service can utilize the most appropriate technology for its tasks. For instance, in our framework, the sensor_input_microservice is optimized for handling real-time data, while the ai_microservice handles complex calculations. This intends to ensure each part of the system performs optimally. Figure 22 depict the implemented structure of microservices.



```
ontology_based_integration_system_with_artificial_intelligence_model/
├── main.py
├── requirements.txt
├── ai_microservice/
│   ├── ai_api/
│   │   └── api.py
│   ├── ai_models/
│   │   └── lstm.py
│   └── ai_processors/
│       ├── lstm_prediction.py
│       └── lstm_processor.py
├── alert_microservice/
│   └── alert_api/
│       └── api.py
├── data_processing_microservice/
│   ├── sensor_consumers/
│   │   ├── precipitation_sensor_consumer.py
│   │   └── water_level_sensor_consumer.py
│   ├── sensor_data_api/
│   │   └── api.py
│   ├── sensor_data_domain/
│   │   ├── sensor_attribute.py
│   │   └── sensor_reading.py
│   └── sensor_data_processor/
│       └── processor.py
├── ontology_microservice/
│   ├── ontology_api/
│   │   └── api.py
│   ├── ontology_domain/
│   │   ├── hydrologic_station.py
│   │   └── hydrometric_station.py
│   ├── ontology_models/
│   │   └── ssn_extended.owl
│   └── ontology_processor/
│       └── processor.py
└── sensor_input_microservice/
    ├── connections.py
    └── producers/
        ├── precipitation_sensor_producer.py
        └── water_level_sensor_producer.py
```

Figure 22 – Implemented microservices directory structure.

However, the microservices architecture does present challenges. Managing multiple services can be complex, requiring advanced tools for orchestration and monitoring, as well as a robust infrastructure for inter-service communication, such as APIs and messaging systems. Communication between services can introduce latency and is more prone to network failures, necessitating strategies like retries and circuit breakers.

During the planning phase, the main functional components of the system were identified, and corresponding microservices were defined, evaluating the most appropriate technologies and frameworks for each service. Each microservice was developed in parallel implementing RESTful APIs for inter-service communication.

Testing involved both unit and integration tests for each microservice individually, as well as end-to-end integration tests to ensure correct interaction between services.

During the project, several design decisions were revisited and updated. This continuous process of review and adaptation is crucial in the software development lifecycle, especially in a microservices environment. The main reasons for these revisions included the evolution of requirements, feedback from implementation, new technologies, and performance and scalability considerations, once one of the framework goals is to allow adding new sensors to the solution.

Changes in project requirements or a deeper understanding of the problem often demand adjustments to the initial design decisions. Feedback from the implementation phase also played a significant role; issues identified during development and testing required revisiting and revising initial approaches to ensure the system's robustness and functionality.

Performance and scalability tests often revealed the need for optimizations or architectural adjustments to better support the system's scalability. Managing these revisions and updates was facilitated by agile methodologies, which allowed the project to adapt quickly to changes.

In the context of the proposed solution, the choice of a microservices architecture was guided by the specific quality attributes required for integrating sensor data and predicting critical events in a scalable, resilient, and maintainable manner. However, it is essential to acknowledge that alternative architectures could also offer different quality attributes and might be suitable under certain conditions.

For instance, a monolithic architecture could provide advantages, particularly in smaller systems or when dealing with limited resources. A monolithic structure consolidates all components into a single application, simplifying integration and reducing the need for managing inter-service communication. In solutions with lower scaling demands, a monolithic architecture may offer better performance due to reduced communication overhead between services. Furthermore, with all functionalities housed in a single codebase, monitoring, deployment, and troubleshooting can be streamlined, especially when the system is small or maintained by a smaller team. Similarly, an event-driven architecture could provide benefits such as real-time responsiveness and decoupled components that react to specific events. However, while these architectures may offer certain advantages, they do not necessarily address all the critical requirements identified for this project.

The quality attributes identified—scalability, resilience, independence of components, and maintainability—were not only necessary but also sufficient for the successful implementation of the proposed solution. Scalability ensures that the system can handle increasing data sources and demands without performance degradation, while resilience guarantees that failures in one component do not compromise the entire system. The independence of components facilitates easy modification and evolution of individual parts without disrupting the entire structure. Additionally, maintainability ensures that the system can be efficiently updated and monitored, which is crucial for long-term success. These attributes comprehensively address the primary concerns of data integration, real-time processing, and adaptability.

While alternative architectures may offer benefits in specific scenarios, the microservices approach was selected because it meets all the critical requirements for this project. By ensuring that each of these attributes is addressed, the proposed architecture provides a robust, scalable, and future-proof solution capable of handling the complexities of sensor data integration and event prediction. Thus, the chosen architecture not only aligns with the technical needs but also offers the flexibility to accommodate future system evolution.

Comparing the framework architecture in Figure 5 with the structure shown in Figure 22, we got:

- the ai_microservice implements the **AI Layer** functionalities;
- the data_processing_microservice implements the **Processing Layer**;
- the ontology_microservice stands for the **Ontology Layer**;
- the sensor_input_microservice contains the structure associated with the **Data Layer**
- the alert_microservice implements the **Alert Layer**.

The full implementation is available at the project's Bitbucket repository[3].

To implement this proposed solution, a combination of various tools and libraries was utilized. Apache Kafka[4] was employed to simulate real-time data streaming. MongoDB[5] was used to store historical data and ontology data, intending to ensure a robust structure for managing large volumes of data. The code logic was developed in Python[6], leveraging the extensive availability of scientific and data processing libraries. The Owlready2 library[7] was chosen for ontology implementation, while the Protégé platform[8] served as the development and management environment for these ontologies. NumPy[9] was used for numerical operations, and Pandas[10] for data manipulation and analysis. Visualizations and plots were generated using Matplotlib[11]. For machine learning tasks, Scikit-Learn (sklearn)[12] was employed, utilizing features such as StandardScaler for standardizing

[3] `https://bitbucket.org/JeffersonAmara/obiwan/src/main/`
[4] `https://kafka.apache.org/`
[5] `https://www.mongodb.com/`
[6] `https://www.python.org/`
[7] `https://owlready2.readthedocs.io/en/latest/`
[8] `https://protege.stanford.edu/`
[9] `https://numpy.org/`
[10] `https://pandas.pydata.org/`
[11] `https://matplotlib.org/`
[12] `https://scikit-learn.org/`

features by removing the mean and scaling to unit variance, train_test_split for splitting the dataset into training and testing sets, and SMOTE for handling class imbalance by oversampling the minority class. The TensorFlow[13] backend, used by Keras, was used for building and training neural networks. Finally, Pymongo[14] was used to interact with MongoDB databases, facilitating read and write operations in the database.

In summary, the microservices architecture was chosen for this framework due to its modularity, scalability, resilience, and technological flexibility. Although it presents operational and communication challenges, the advantages in terms of independent development, the ability to scale services individually, and ease of maintenance outweigh the disadvantages. Implementing this architecture, intends to ensure an adaptable system prepared for future evolutions, as we aim to do.

4.4 Experimental Results

We assume that the primary events of interest are linked to detecting high-water levels in hydrometric stations, as these events often signify potential flooding, posing a threat to people and buildings in the affected region. To identify these events, we leverage data from various sensors integrated within the framework of 'location' and 'flow direction' contexts.

This section presents the outcomes obtained when the previously described sensor data undergoes analysis within the framework. We aim to semantically integrate data from diverse sensors and detect and predict events, addressing the two research questions introduced in the Introduction Section.

Integrating data from hydrological and hydrometric sensors considers factors such as their installation location, flow value and direction, mutual influence, and the temporal aspects of this influence. Figure 23 illustrates the diagram depicting the locations of stations within the network of integrated sensors. Each dotted arrow represents contextual relationships established based on the flow direction context.

For instance, by integrating data from hydrological sensors at Cop Upper with hydrometric sensors at 05BH010 (considering they share the same location context), precipitation data can be a feature for predicting water levels at 05BH010.

[13] `https://www.tensorflow.org/`
[14] `https://pymongo.readthedocs.io/`

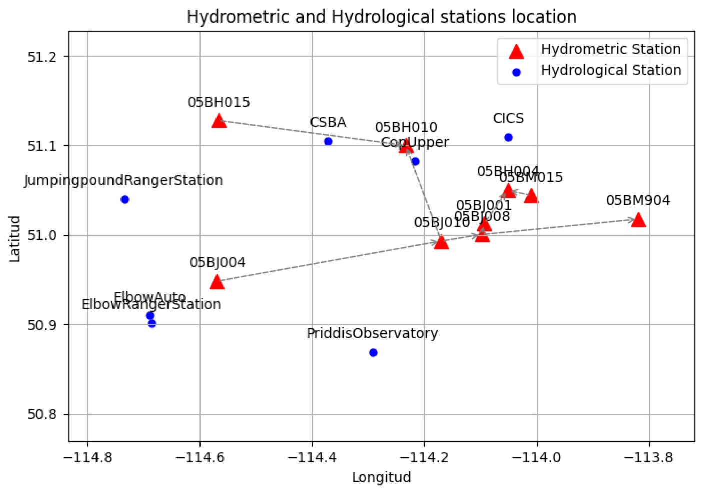


Figure 23 – Network location of stations.

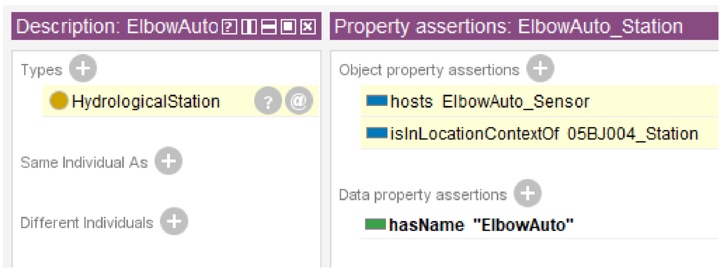
At this point, the framework cannot measure how much each hydrological station influences the hydrometric station, although there are mutual influences between hydrometric stations. To define the influence context from hydrological stations, we defined if there is influence or not based on the previous simultaneity of high precipitation and high-water level and station metadata. We also based our context rules in the study (63) on which the authors investigate causes, assessment, and damages in the area where data originated. Then, based on these data, we defined the context rules, as exhibited in Table 8 and described on Table 13. The definition of more accurate context rules depends on domain experts or context detection automation, which are points to be improved in future works.

Table 13 – Pre-defined context among stations

Context based on flow direction	Context based on location
05BH015, 05BH010	CopUpper, 05BH010
05BJ004, 05BJ010	CSBA, 05BH010
05BJ010, 05BH010	CSBA, 05BH015
05BJ010, 05BJ008	CICS, 05BH004
05BJ008, 05BJ001	ElbowAuto, 05BJ004
05BJ008, 05BM904	ElbowRanger, 05BJ004
05BJ001, 05BH004	
05BM015, 05BH004	

Through Figure 24, we revisit the ontology competence questions presented in Chapter 3, demonstrating the ontology's capacity to detect location and flow direction contexts. Figure 24 illustrates the location context assertions from the "ElbowAuto" station to the "05BJ004" station, addressing the first ontology competence question (CQ1), which asks: **What are the platforms in the same location context?** Additionally, Figure 25 allows us to revisit the second ontology competence question (CQ2) by showcasing the ontology's ability to determine flow direction context, specifically from station '05BJ010' to stations '05BH010' and '05BJ008', answering the question: **What are the platforms**

in the same flow direction context? Finally, the third ontology competence question (CQ3) is explored in Figure 26, where the combined assertions on lines 4, 7, and 10 demonstrate the ontology's ability to detect critical events, addressing the question: **What are the critical events detected when observation results trespass the platform threshold?**.



Description: ElbowAuto

Types
- HydrologicalStation

Same Individual As

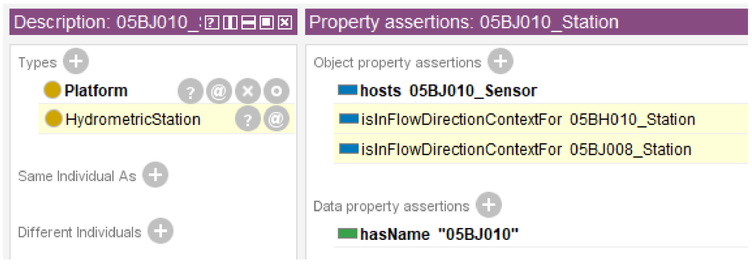
Different Individuals

Property assertions: ElbowAuto_Station

Object property assertions
- hosts ElbowAuto_Sensor
- isInLocationContextOf 05BJ004_Station

Data property assertions
- hasName "ElbowAuto"

Figure 24 – Location context assertion example.



Description: 05BJ010_

Types
- **Platform**
- HydrometricStation

Same Individual As

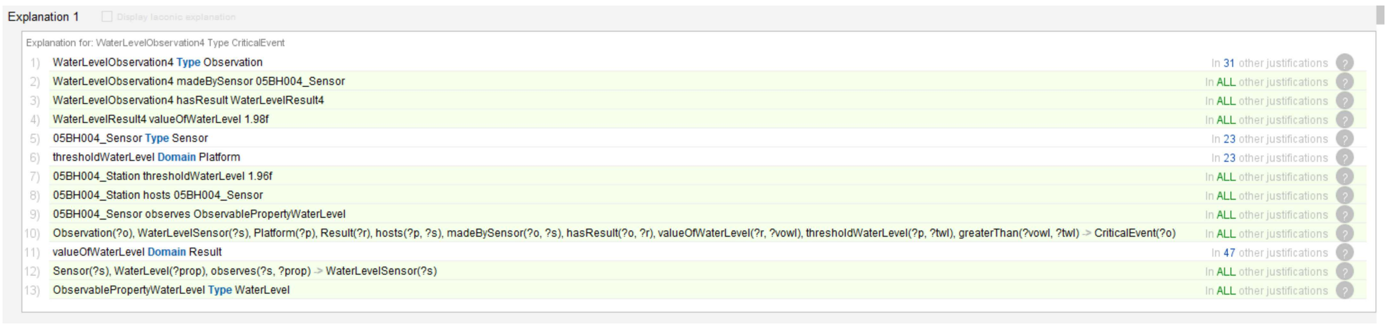
Different Individuals

Property assertions: 05BJ010_Station

Object property assertions
- **hosts 05BJ010_Sensor**
- isInFlowDirectionContextFor 05BH010_Station
- isInFlowDirectionContextFor 05BJ008_Station

Data property assertions
- **hasName "05BJ010"**

Figure 25 – Flow context assertion example.

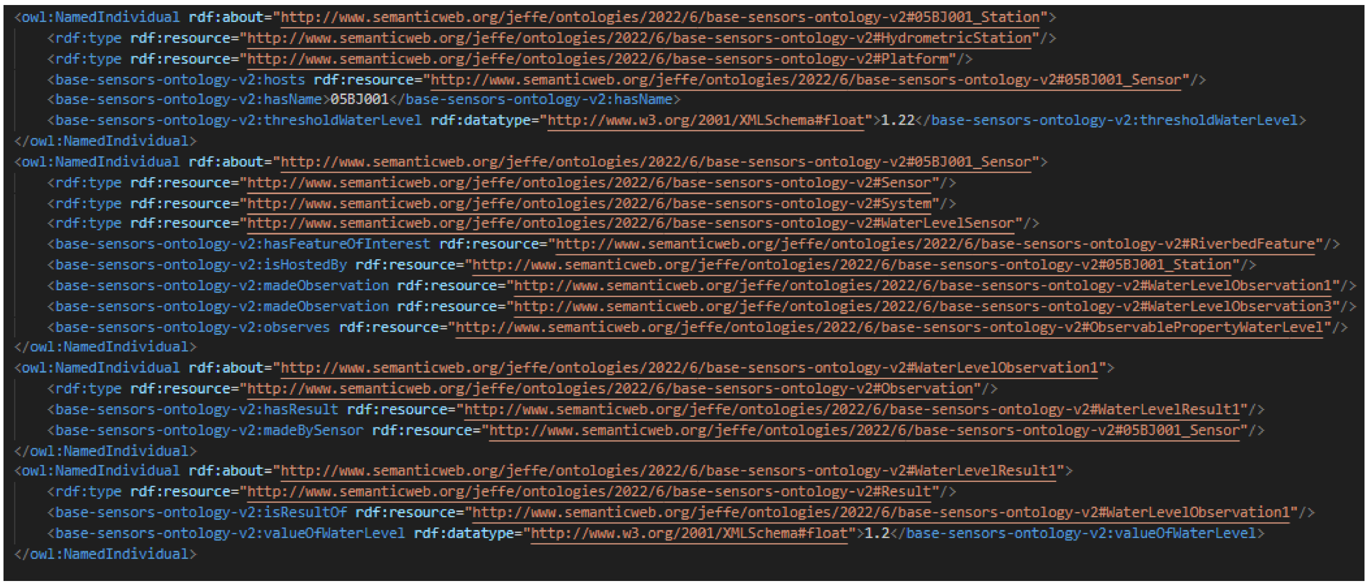


Explanation 1 ☐ Display laconic explanation

```
Explanation for: WaterLevelObservation4 Type CriticalEvent
1)  WaterLevelObservation4 Type Observation                                  In 31 other justifications
2)  WaterLevelObservation4 madeBySensor 05BH004_Sensor                       In ALL other justifications
3)  WaterLevelObservation4 hasResult WaterLevelResult4                       In ALL other justifications
4)  WaterLevelResult4 valueOfWaterLevel 1.98f                                In ALL other justifications
5)  05BH004_Sensor Type Sensor                                               In 23 other justifications
6)  thresholdWaterLevel Domain Platform                                      In 23 other justifications
7)  05BH004_Station thresholdWaterLevel 1.96f                                In ALL other justifications
8)  05BH004_Station hosts 05BH004_Sensor                                     In ALL other justifications
9)  05BH004_Sensor observes ObservablePropertyWaterLevel                     In ALL other justifications
10) Observation(?o), WaterLevelSensor(?s), Platform(?p), Result(?r), hosts(?p, ?s), madeBySensor(?o, ?s), hasResult(?o, ?r), valueOfWaterLevel(?r, ?vowl), thresholdWaterLevel(?p, ?twl), greaterThan(?vowl, ?twl) -> CriticalEvent(?o)   In ALL other justifications
11) valueOfWaterLevel Domain Result                                          In 47 other justifications
12) Sensor(?s), WaterLevel(?prop), observes(?s, ?prop) -> WaterLevelSensor(?s)   In ALL other justifications
13) ObservablePropertyWaterLevel Type WaterLevel                             In ALL other justifications
```

Figure 26 – Critical event detected by threshold limit assertion.

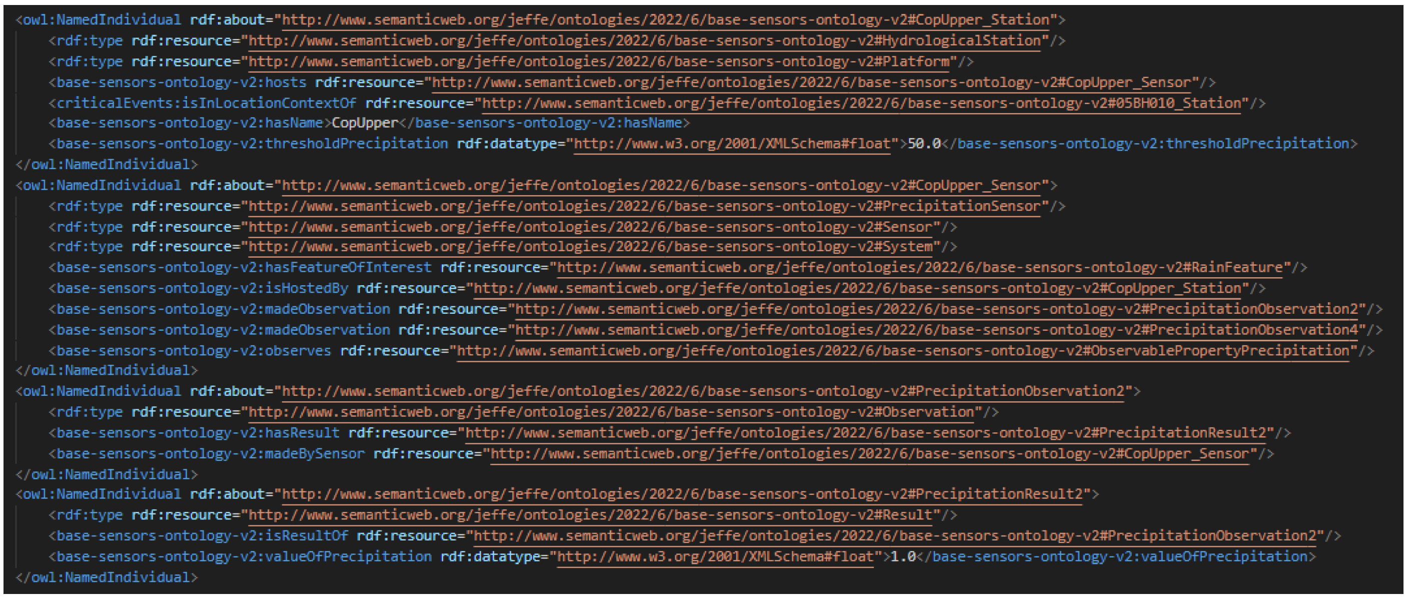
To assess Metric M1, we compared the input data from hydrological and hydrometric station sensors with their representations post-data integration. The instances of integrated data representation preserve the original values from each input dataset while incorporating semantic values related to context. Figure 27 exemplifies one of this representation.



```xml
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#05BJ001_Station">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#HydrometricStation"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Platform"/>
    <base-sensors-ontology-v2:hosts rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#05BJ001_Sensor"/>
    <base-sensors-ontology-v2:hasName>05BJ001</base-sensors-ontology-v2:hasName>
    <base-sensors-ontology-v2:thresholdWaterLevel rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.22</base-sensors-ontology-v2:thresholdWaterLevel>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#05BJ001_Sensor">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Sensor"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#System"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelSensor"/>
    <base-sensors-ontology-v2:hasFeatureOfInterest rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#RiverbedFeature"/>
    <base-sensors-ontology-v2:isHostedBy rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#05BJ001_Station"/>
    <base-sensors-ontology-v2:madeObservation rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelObservation1"/>
    <base-sensors-ontology-v2:madeObservation rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelObservation3"/>
    <base-sensors-ontology-v2:observes rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#ObservablePropertyWaterLevel"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelObservation1">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Observation"/>
    <base-sensors-ontology-v2:hasResult rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelResult1"/>
    <base-sensors-ontology-v2:madeBySensor rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#05BJ001_Sensor"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelResult1">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Result"/>
    <base-sensors-ontology-v2:isResultOf rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#WaterLevelObservation1"/>
    <base-sensors-ontology-v2:valueOfWaterLevel rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.2</base-sensors-ontology-v2:valueOfWaterLevel>
</owl:NamedIndividual>
```

Figure 27 – Hydrometric Station Semantic Ontology Representation Example.

Following our feasibility study, where data from eight hydrological and nine hydrometric sensors were successfully integrated, Metric M2 is deemed accomplished. This success is attributed to incorporating all contextual combinations between hydrological and hydrometric sensors and among hydrometric sensors themselves, all of this under the same canonical representation, as show in Figure 28.



```xml
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#CopUpper_Station">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#HydrologicalStation"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Platform"/>
    <base-sensors-ontology-v2:hosts rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#CopUpper_Sensor"/>
    <criticalEvents:isInLocationContextOf rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#05BH010_Station"/>
    <base-sensors-ontology-v2:hasName>CopUpper</base-sensors-ontology-v2:hasName>
    <base-sensors-ontology-v2:thresholdPrecipitation rdf:datatype="http://www.w3.org/2001/XMLSchema#float">50.0</base-sensors-ontology-v2:thresholdPrecipitation>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#CopUpper_Sensor">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationSensor"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Sensor"/>
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#System"/>
    <base-sensors-ontology-v2:hasFeatureOfInterest rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#RainFeature"/>
    <base-sensors-ontology-v2:isHostedBy rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#CopUpper_Station"/>
    <base-sensors-ontology-v2:madeObservation rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationObservation2"/>
    <base-sensors-ontology-v2:madeObservation rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationObservation4"/>
    <base-sensors-ontology-v2:observes rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#ObservablePropertyPrecipitation"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationObservation2">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Observation"/>
    <base-sensors-ontology-v2:hasResult rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationResult2"/>
    <base-sensors-ontology-v2:madeBySensor rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#CopUpper_Sensor"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationResult2">
    <rdf:type rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#Result"/>
    <base-sensors-ontology-v2:isResultOf rdf:resource="http://www.semanticweb.org/jeffe/ontologies/2022/6/base-sensors-ontology-v2#PrecipitationObservation2"/>
    <base-sensors-ontology-v2:valueOfPrecipitation rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.0</base-sensors-ontology-v2:valueOfPrecipitation>
</owl:NamedIndividual>
```

Figure 28 – Hydrological Station Semantic Ontology Representation Example.

Figure 27 exemplifies the semantic representation of a hydrometric station after data integration. This process transforms heterogeneous sensor inputs into a unified canonical structure, not only organizing the data but also adding essential semantic context to identify critical events.

Figure 28 expands this logic to hydrological stations, illustrating how semantic integration helps correlate precipitation data with water level measurements. Critical event detection is enhanced by cross-referencing information between sensors. For example, high precipitation levels detected by a hydrological station may be correlated with a subsequent rise in water levels monitored by a hydrometric station. Without the ontology to facilitate the combined interpretation of these data streams, such correlations might be missed.

Integration was crucial in situations where multiple sensors provided data that, when considered in isolation, would not have revealed a critical event. For instance, when there is a gradual increase in water levels but no immediate correlation with precipitation data, integration allows checking if other factors (such as data from neighboring stations) indicate an impending event. Without semantic integration, these signals could be overlooked, or the AI models might generate false positives.

By answering the ontology competency questions and accomplishing metrics M1 and M2, we can validate the key role the ontology performs in the framework. The ontology allows data from different sensors, as we can see in Figures 27 and 28, heterogeneous in format and content, to be integrated uniformly. This is achieved by defining a canonical data model that represents the various elements of the domain and their relationships. As a result, the ontology facilitates the creation of a unified view of the data, making it semantically enriched and easier to interpret and analyze. From Figures 24, 25, and 26, we can see the effectiveness of the ontology's SWRL rules in establishing context, based on the context rules, and detecting critical events, based on the threshold rules.

To answer Metric M3, we need to analyze the AI model's performance. Figure 29 shows the comparative average precision performance of the AI model over 10 executions. Precision is the proportion of true positives over all predicted as positive. From the graph, we can observe the following:

The AI model exhibits a consistently higher precision when context is included across all stations. The precision scores with context (represented by the black bars) are significantly higher compared to those without context (represented by the gray bars). This improvement in precision may indicate that the inclusion of contextual information allows the model to more accurately identify true positive events.

Furthermore, it is interesting to note that the greatest improvements in precision are for stations 05BH004 and 05BJ004, for which we used 2 context rules each, while for the other stations, only 1 context rule was used for each. This suggests that a greater number of context properties may be associated with higher precision, at least for these two stations.

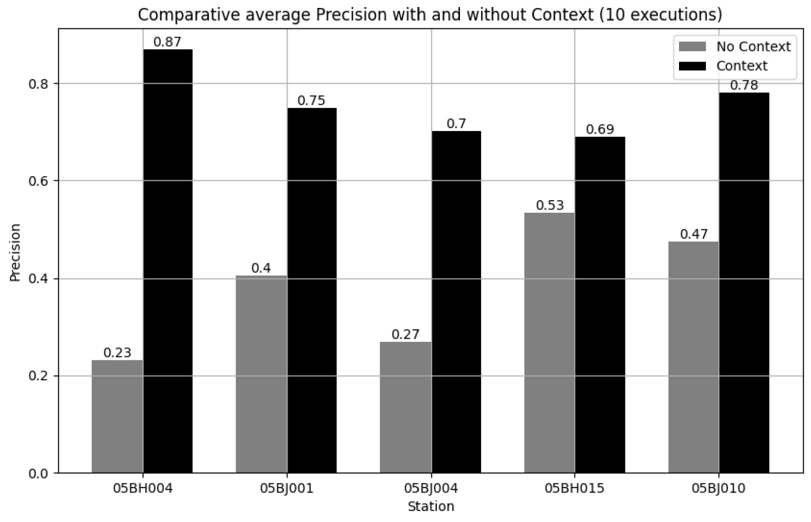


Figure 29 – Comparative Average Precision with and without context over 10 executions.

Figure 30 shows the comparison on recall, which is the proportion of true positives over all the predicted that are really positive.

For some stations (like 05BH004 and 05BJ001), the recall is higher without the context rules. For others (like 05BH015 and 05BJ010), the recall is higher using context rules. The variability in recall among the stations may indicate that the effectiveness of using context may depend on specific characteristics of each station, not captured on context rules, such as topography, local hydrology, or even the quality of contextual rules. Also, it is important to highlight that the model without context may be 'guessing' more events as critical, and this decreases the number of false negative.

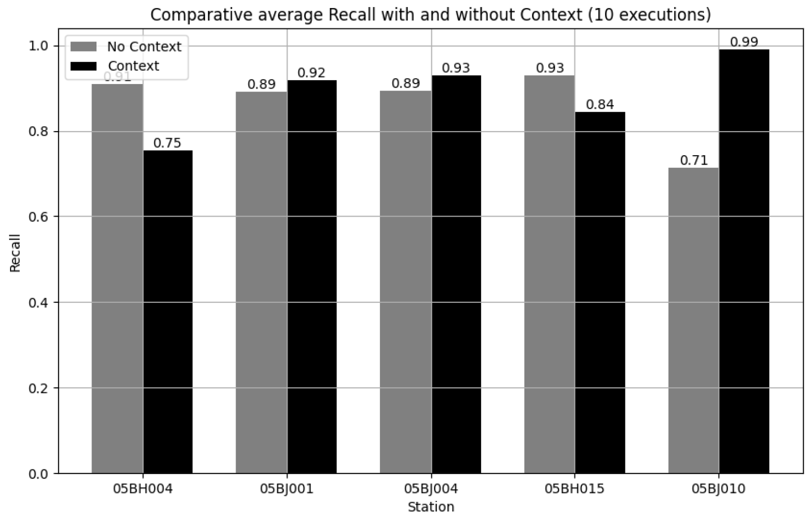


Figure 30 – Comparative Average Recall with and without context over 10 executions.

F1-Score, which is the harmonic mean of precision and recall. It is particularly useful when a balance between precision and recall is desired, and when avoiding the extremes of one metric at the expense of the other is important. Figure 31 shows that the greatest improvements are observed in stations such as 05BH004 and 05BJ004, where context seems to have a particularly strong impact. In stations such as 05BJ001, the improvement is still significant but less pronounced, suggesting that the relevance and amount of contextual information may also vary depending on the station and on the quality of context rule.

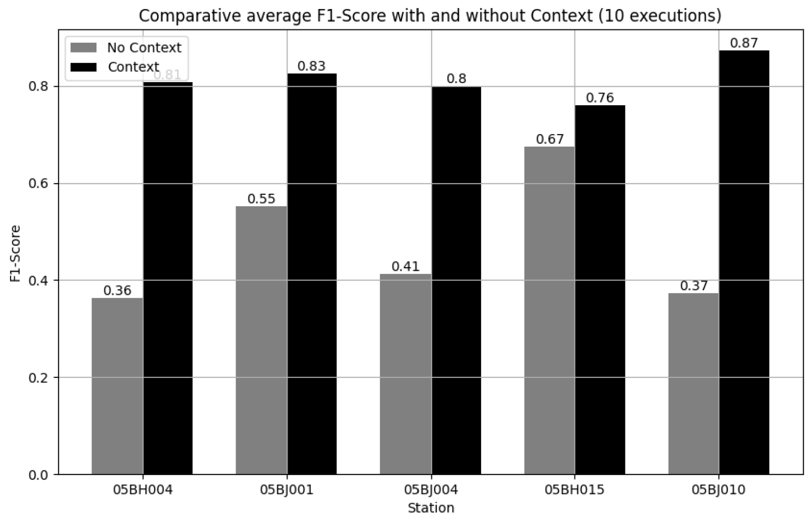


Figure 31 – Comparative Average F1-Score with and without context over 10 executions

It is noteworthy again the contribution provided by the ontology in establishing context and relationships among stations, which improves the AI model's capabilities.

We also considered the variation of prediction according to 'lead time' perspective, that is, what is the model's behavior when trying to predict critical events 3 and 2 days ahead. This evaluation is useful to see how the model can perform trying to anticipate even more the critical event prediction. Figure 32 depicts a considerable decrease of performance when trying to predict critical events with lead time of 2 and 3 days. This result suggests that event the context rules provided are not sufficient to improve predictions capabilities for these lead times.

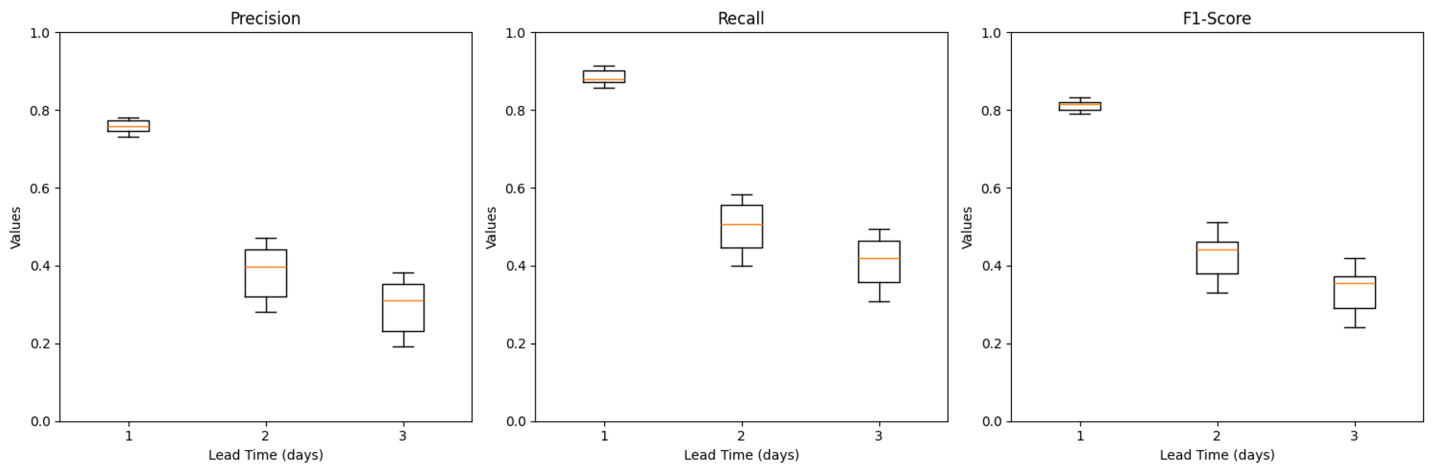


Figure 32 – Average performance considering lead time.

Compared to the work on (54), we can see in Figure 33 that maybe due to the specific data and specialized context rules, the performance to 1 day of lead time shows considerable better performance compared to that proposal. For lead times of 2 and 3 days, our proposal does not performs well. F1-Score, despite being less variable, does not present useful results for predict critical events 2 or 3 days ahead, even using specific data and specialized context rules.

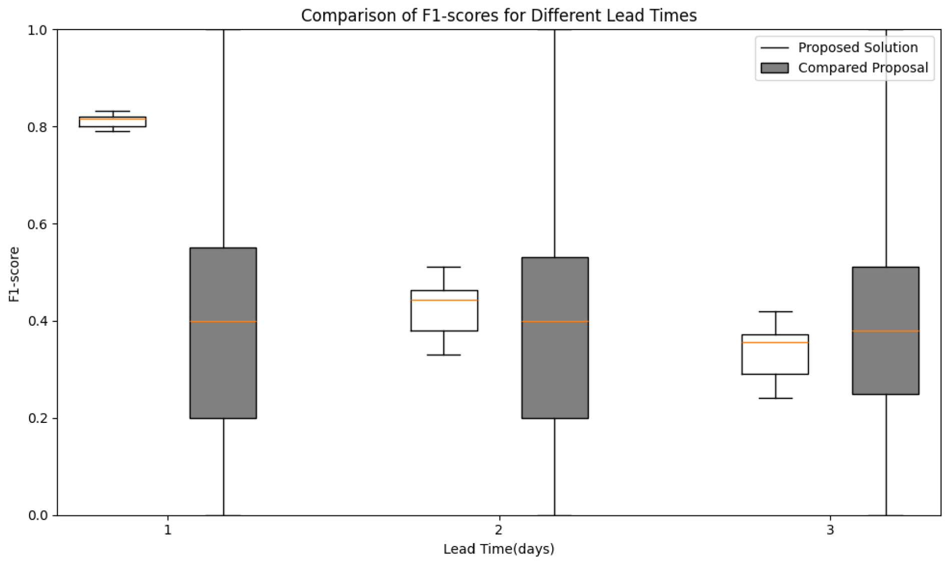
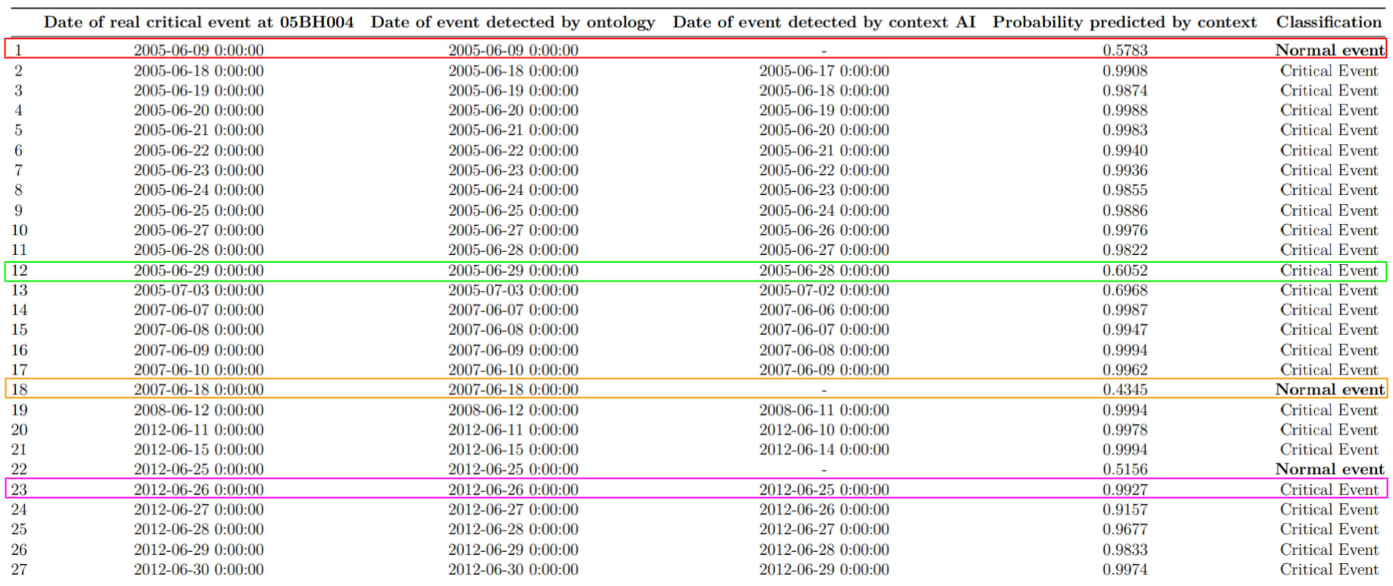


Figure 33 – Comparative Average F1-Score with lead time.

Figure 34 illustrates some of the critical events detected using the contextual rule defined between 05BJ001 and 054BH004. For example, in rows 12 and 23, critical events were detected by the ontology layer and predicted with probabilities of 0.6052 and 0.9927, respectively, one day in advance based on the AI layer.

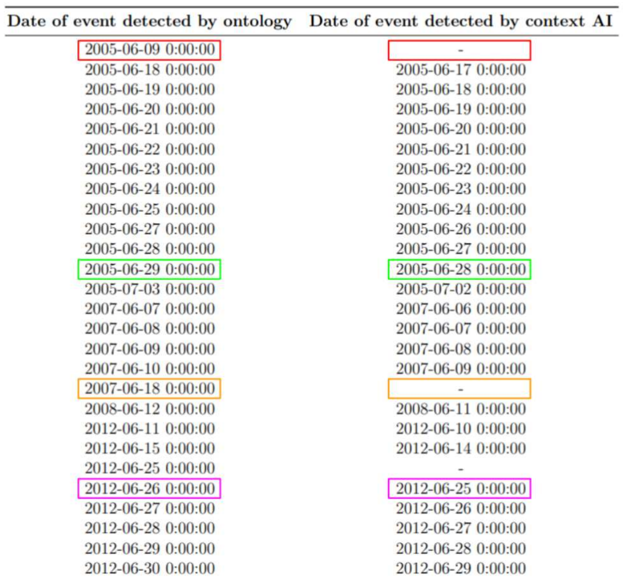
In rows 1 and 18, where the probabilities are 0.5783 and 0.4345, the AI layer, with this contextual rule, does not predict these events as critical. Therefore, they are only detected on the day they occur.



	Date of real critical event at 05BH004	Date of event detected by ontology	Date of event detected by context AI	Probability predicted by context	Classification
1	2005-06-09 0:00:00	2005-06-09 0:00:00	-	0.5783	**Normal event**
2	2005-06-18 0:00:00	2005-06-18 0:00:00	2005-06-17 0:00:00	0.9908	Critical Event
3	2005-06-19 0:00:00	2005-06-19 0:00:00	2005-06-18 0:00:00	0.9874	Critical Event
4	2005-06-20 0:00:00	2005-06-20 0:00:00	2005-06-19 0:00:00	0.9988	Critical Event
5	2005-06-21 0:00:00	2005-06-21 0:00:00	2005-06-20 0:00:00	0.9983	Critical Event
6	2005-06-22 0:00:00	2005-06-22 0:00:00	2005-06-21 0:00:00	0.9940	Critical Event
7	2005-06-23 0:00:00	2005-06-23 0:00:00	2005-06-22 0:00:00	0.9936	Critical Event
8	2005-06-24 0:00:00	2005-06-24 0:00:00	2005-06-23 0:00:00	0.9855	Critical Event
9	2005-06-25 0:00:00	2005-06-25 0:00:00	2005-06-24 0:00:00	0.9886	Critical Event
10	2005-06-27 0:00:00	2005-06-27 0:00:00	2005-06-26 0:00:00	0.9976	Critical Event
11	2005-06-28 0:00:00	2005-06-28 0:00:00	2005-06-27 0:00:00	0.9822	Critical Event
12	2005-06-29 0:00:00	2005-06-29 0:00:00	2005-06-28 0:00:00	0.6052	Critical Event
13	2005-07-03 0:00:00	2005-07-03 0:00:00	2005-07-02 0:00:00	0.6968	Critical Event
14	2007-06-07 0:00:00	2007-06-07 0:00:00	2007-06-06 0:00:00	0.9987	Critical Event
15	2007-06-08 0:00:00	2007-06-08 0:00:00	2007-06-07 0:00:00	0.9947	Critical Event
16	2007-06-09 0:00:00	2007-06-09 0:00:00	2007-06-08 0:00:00	0.9994	Critical Event
17	2007-06-10 0:00:00	2007-06-10 0:00:00	2007-06-09 0:00:00	0.9962	Critical Event
18	2007-06-18 0:00:00	2007-06-18 0:00:00	-	0.4345	**Normal event**
19	2008-06-12 0:00:00	2008-06-12 0:00:00	2008-06-11 0:00:00	0.9994	Critical Event
20	2012-06-11 0:00:00	2012-06-11 0:00:00	2012-06-10 0:00:00	0.9978	Critical Event
21	2012-06-15 0:00:00	2012-06-15 0:00:00	2012-06-14 0:00:00	0.9994	Critical Event
22	2012-06-25 0:00:00	2012-06-25 0:00:00	-	0.5156	**Normal event**
23	2012-06-26 0:00:00	2012-06-26 0:00:00	2012-06-25 0:00:00	0.9927	Critical Event
24	2012-06-27 0:00:00	2012-06-27 0:00:00	2012-06-26 0:00:00	0.9157	Critical Event
25	2012-06-28 0:00:00	2012-06-28 0:00:00	2012-06-27 0:00:00	0.9677	Critical Event
26	2012-06-29 0:00:00	2012-06-29 0:00:00	2012-06-28 0:00:00	0.9833	Critical Event
27	2012-06-30 0:00:00	2012-06-30 0:00:00	2012-06-29 0:00:00	0.9974	Critical Event

Figure 34 – Event detection and prediction days ahead for station 05BH004

Figure 35 depicts in more detail, comparing the detection by the ontology and the prediction by the AI model, we observe that the contextual rule between station 05BJ001 and 05BH004 is not sufficient for early prediction, as highlighted in orange.



Date of event detected by ontology	Date of event detected by context AI
2005-06-09 0:00:00	-
2005-06-18 0:00:00	2005-06-17 0:00:00
2005-06-19 0:00:00	2005-06-18 0:00:00
2005-06-20 0:00:00	2005-06-19 0:00:00
2005-06-21 0:00:00	2005-06-20 0:00:00
2005-06-22 0:00:00	2005-06-21 0:00:00
2005-06-23 0:00:00	2005-06-22 0:00:00
2005-06-24 0:00:00	2005-06-23 0:00:00
2005-06-25 0:00:00	2005-06-24 0:00:00
2005-06-27 0:00:00	2005-06-26 0:00:00
2005-06-28 0:00:00	2005-06-27 0:00:00
2005-06-29 0:00:00	2005-06-28 0:00:00
2005-07-03 0:00:00	2005-07-02 0:00:00
2007-06-07 0:00:00	2007-06-06 0:00:00
2007-06-08 0:00:00	2007-06-07 0:00:00
2007-06-09 0:00:00	2007-06-08 0:00:00
2007-06-10 0:00:00	2007-06-09 0:00:00
2007-06-18 0:00:00	-
2008-06-12 0:00:00	2008-06-11 0:00:00
2012-06-11 0:00:00	2012-06-10 0:00:00
2012-06-15 0:00:00	2012-06-14 0:00:00
2012-06-25 0:00:00	-
2012-06-26 0:00:00	2012-06-25 0:00:00
2012-06-27 0:00:00	2012-06-26 0:00:00
2012-06-28 0:00:00	2012-06-27 0:00:00
2012-06-29 0:00:00	2012-06-28 0:00:00
2012-06-30 0:00:00	2012-06-29 0:00:00

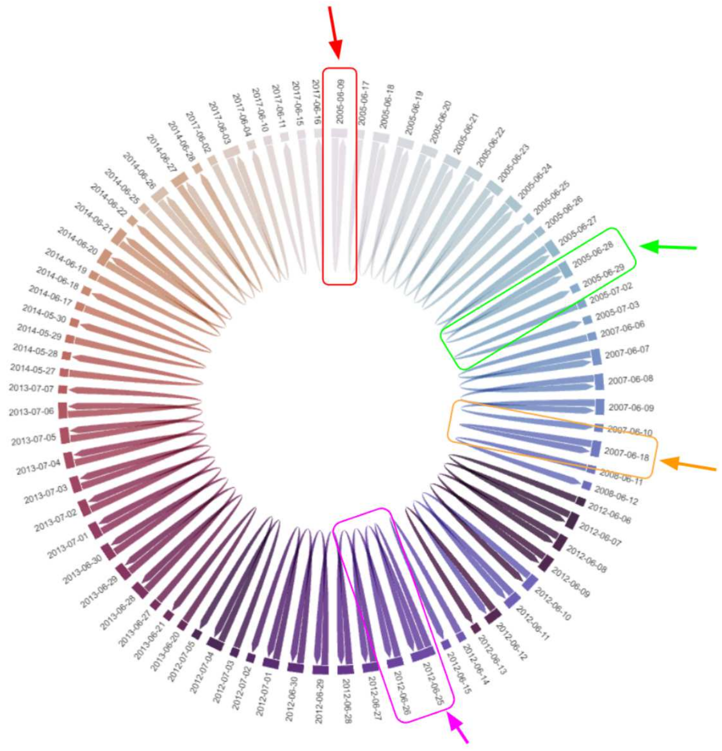
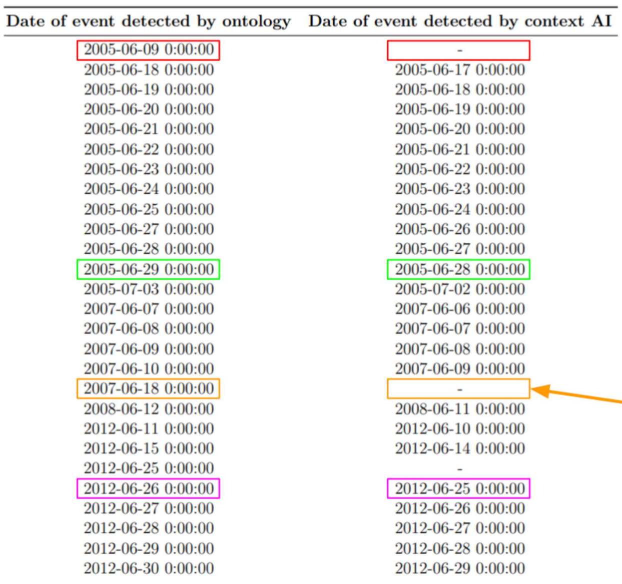


Figure 35 – Event detection and prediction days ahead for station 05BH004 in detail

However, another contextual rule (exihibited in Figure 36), established between the CICS station and station 05BH004, enables the AI model to predict the event on 18-06-2007 one day in advance, on 17-06-2007. This demonstrates how contextual rules have complementary potential, which is useful for event prediction. In this case, it fills a gap left by the contextual rule from station 05BJ001.



Date of event detected by ontology	Date of event detected by context AI
2005-06-09 0:00:00	-
2005-06-18 0:00:00	2005-06-17 0:00:00
2005-06-19 0:00:00	2005-06-18 0:00:00
2005-06-20 0:00:00	2005-06-19 0:00:00
2005-06-21 0:00:00	2005-06-20 0:00:00
2005-06-22 0:00:00	2005-06-21 0:00:00
2005-06-23 0:00:00	2005-06-22 0:00:00
2005-06-24 0:00:00	2005-06-23 0:00:00
2005-06-25 0:00:00	2005-06-24 0:00:00
2005-06-27 0:00:00	2005-06-26 0:00:00
2005-06-28 0:00:00	2005-06-27 0:00:00
2005-06-29 0:00:00	2005-06-28 0:00:00
2005-07-03 0:00:00	2005-07-02 0:00:00
2007-06-07 0:00:00	2007-06-06 0:00:00
2007-06-08 0:00:00	2007-06-07 0:00:00
2007-06-09 0:00:00	2007-06-08 0:00:00
2007-06-10 0:00:00	2007-06-09 0:00:00
2007-06-18 0:00:00	-
2008-06-12 0:00:00	2008-06-11 0:00:00
2012-06-11 0:00:00	2012-06-10 0:00:00
2012-06-15 0:00:00	2012-06-14 0:00:00
2012-06-25 0:00:00	-
2012-06-26 0:00:00	2012-06-25 0:00:00
2012-06-27 0:00:00	2012-06-26 0:00:00
2012-06-28 0:00:00	2012-06-27 0:00:00
2012-06-29 0:00:00	2012-06-28 0:00:00
2012-06-30 0:00:00	2012-06-29 0:00:00

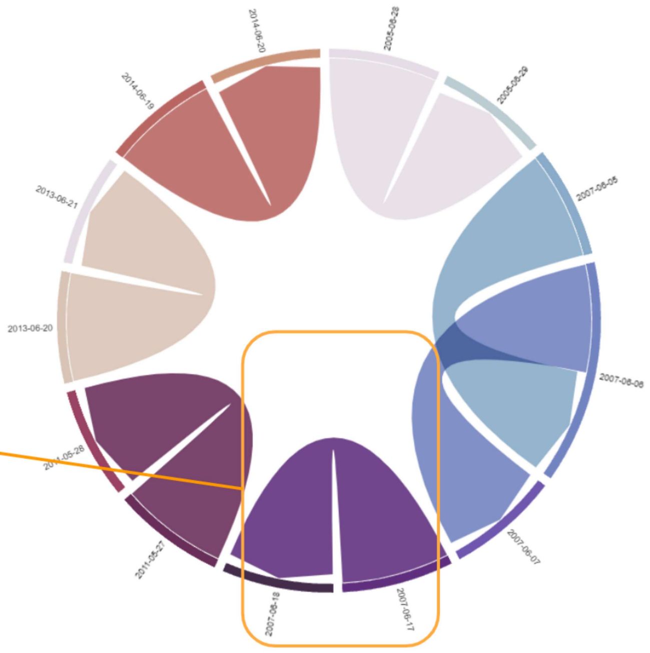


Figure 36 – Event detection and prediction days ahead for station 05BH004

This behavior suggests that the quality and quantity of context rules can influence more accurate detection and prediction of critical events. Furthermore, it reinforces the assertion of the capability to detect events, even with 0 days of advance notice, considering the redundancy between ontology detection and AI model predictions.

Based on these evidences, we can illustrate the synergy and complementarity between the ontology and the AI model. The ontology provides a canonical data representation that harmonizes the heterogeneous data sources, performs the detection of critical events based on threshold rules, and establishes rich, semantically contextualized information. Meanwhile, the AI model utilizes this contextualized data to predict critical events with lead time, enhancing performance. Notably, there are instances where certain events may not be detected by the ontology layer but can still be predicted earlier by the AI layer, and vice versa. This interplay highlights the strengths of both components, ensuring a comprehensive and robust approach to event detection and prediction.

4.4.1 Final Considerations of the Chapter

In this chapter, we conducted a feasibility study to evaluate the proposed framework using real-world data from hydrometric and hydrological sensors. The study covered data description, AI model definition, and implementation, followed by an analysis of the experimental results. These results demonstrated the framework's capability to integrate sensor data and predict critical events effectively, highlighting the potential of combining ontologies and AI models for accurate event detection and prediction.

This evaluation serves as a crucial step in validating the proposed framework and provides the necessary insights to further refine and optimize its performance. In the next chapter, we will perform a detailed analysis of the results, including the performance of semantic integration and AI models, to better understand the overall impact and efficiency of the solution.

5 Result Analysis

In this chapter, we present a detailed analysis of the results obtained from the feasibility study of the proposed framework for semantic integration of sensor data and event prediction. We evaluate the framework's effectiveness based on the defined metrics and discuss the implications of the findings.

5.1 Semantic Integration Performance

Semantic integration of data from multiple sensors was one of the primary objectives of this work. The results demonstrated that using ontologies to standardize and contextualize data was successful, meeting the expectations defined by metrics M1 and M2.

5.1.1 Metric M1: Canonical Data Representation

The M1 metric was evaluated by comparing the input data from hydrological and hydrometric sensors with their representations after data integration. Data integration was performed using an extended SSN (Semantic Sensor Network) ontology, which extracts syntactic, semantic, and contextual knowledge from the sensor data. Each sensor reading was mapped to a canonical model through the LCS Wrapper, which defines the transformation rules necessary to convert the sensor data into a canonical representation.

As shown in the previous chapter, for a hydrometric sensor, the original reading may include attributes such as date, water level, and flow rate. After transformation, these data are represented in an ontology that defines the classes, data properties, and object properties, allowing for semantic understanding and integration with other data.

For this, M1 metric was successfully met, as the sensor data were standardized into a canonical semantic representation, enabling data integration and contextualization.

5.1.2 Metric M2: Diversity of Integrated Data Sources

The Metric M2 focus on evaluate the framework's ability to integrate data from various sensor sources and add new data sources. The feasibility study considered data from eight hydrological stations and nine hydrometric stations, with data ranging from 2005 to 2023. The data sources included precipitation measurements and water levels from various stations, geographically distributed, and with different water flow contexts. As exhibited in the feasibility results, precipitation data from hydrological stations were integrated with water level data from hydrometric stations. The integration considered contexts such as the location of the stations and flow direction, allowing for a comprehensive and contextualized data analysis. We consider the M2 metric was met, as the framework successfully integrated data from various sensor sources, covering a wide range of measurements and contexts.

5.2 Metric M3: AI Model Performance

The AI model's performance was evaluated based on precision, recall, and f1-score metrics, as defined in metric M3. The analysis focused on the model's ability to detect and predict critical events.

Precision: The results showed that the model's precision increased significantly when context was included. This suggests that contextualizing data allows the model to more accurately identify true positive events.

Recall: The recall analysis revealed variations between stations, indicating that the effectiveness of context rules may depend on specific characteristics of each station. In some cases, the absence of context resulted in a higher number of false negatives, reducing recall.

F1-Score: The f1-score, which balances precision and recall, also showed improvements with the inclusion of context. This metric is particularly important to ensure the model does not favor one metric at the expense of the other.

5.3 Impact of Context Rules

Context rules were crucial to the model's performance. However, the dependency on these rules also posed challenges:

- **Performance Variation**: The effectiveness of context rules varied between stations, suggesting the need for specific adjustments for each scenario.
- **Limited Generalization**: Creating context rules specific for the target domain may limit the framework's generalization to new domains or different types of data.

5.4 Long-term Prediction

The ability to predict events with greater lead time (2-3 days) was limited, indicating that the current context rules are insufficient for long-term predictions. Future improvements should focus on:

- **Enhancing Context Rules**: Developing automated methods to detect and define more complex contexts.
- **Improving AI Models**: Investigating more robust AI models and advanced techniques to handle long-term prediction.

5.5 Discussions

The primary findings of this study demonstrate that the proposed framework effectively integrates data from multiple sources to detect and predict events of interest. The use of a modular architecture with microservices allows for flexibility and scalability, while the integration of local data enhances the precision and relevance of predictions. The ontology model and context rules facilitate the semantic integration of diverse data sources, leading to improved event detection and prediction.

The primary research question asks **(RQ)Can the framework semantically integrate data from multiple data sources to detect and predict events of interest from these data?**. The proposed framework achieves this by utilizing an ontology model that standardizes and contextualizes information from various sensors. Experimental results show that this integration improves the detection and prediction of events. For example, the framework was able to combine weather data, soil moisture readings, and river levels to predict potential flooding events with high accuracy. The use of Long Short-Term Memory (LSTM) networks further enhances the prediction capabilities by capturing temporal dependencies in the data.

Regarding the first secondary research question **(SRQ1)Does using the proposed framework and implemented ontology model allow the integration of sensor data and add new data sources?**, the framework demonstrates success. This is evidenced by metrics M1 (Canonical Data Representation) and M2 (Diversity of Integrated Data Sources). In the experiments, data from weather stations, and river gauges sensors were integrated, demonstrating the framework's capability to handle diverse data types. The ontology model provided a unified schema, making it easier to add new data sources without significant modifications to the system.

The second secondary research question asks **(SRQ2)Are the events of interest correctly detected and predicted?** . The results confirm that the events were correctly detected and predicted, as supported by performance metrics such as precision, recall, and F1-score (Metric M3). The AI model, particularly the LSTM networks, demonstrated high accuracy in predicting events like floods and droughts. The inclusion of contextual information from the ontology model significantly improved prediction performance. The results confirm that the framework not only integrates data effectively but also enhances the predictive accuracy of the AI models using context rules. Future research should explore automating context rules and improving long-term prediction capabilities.

5.6 Threats to Validity

In this section, we discuss potential threats to the validity of the results and findings presented in this work. These threats are categorized into four main types: internal validity, external validity, construct validity, and conclusion validity. By addressing these threats, we aim to provide transparency and robustness to the research outcomes and demonstrate the steps taken to mitigate their impact.

5.6.1 Internal Validity

Internal validity refers to the extent to which the results of the study can be attributed to the proposed approach rather than other factors. Several factors could potentially threaten internal validity:

- **Data Quality and Noise**: The sensor data used in this research could contain noise, missing values, or inconsistencies due to hardware limitations or environmental conditions. Although data cleansing and preprocessing techniques were applied, there is a risk that not all errors were fully addressed, which could affect the predictive accuracy of the AI models.
- **Bias in AI Model Training**: The performance of the LSTM model and other machine learning algorithms could be influenced by biases in the training data. If certain event types are overrepresented or underrepresented in the dataset, the model may not generalize well to unseen data. To mitigate this risk, cross-validation and data partitioning techniques were employed, but the inherent distribution of real-world events may still pose a challenge.

5.6.2 External Validity

External validity concerns the generalizability of the research findings to other contexts, domains, or datasets. Several factors affect the extent to which the results can be generalized:

- **Domain-Specific Dataset**: The dataset used for evaluating the framework is derived from hydrometric and hydrological sensors, specifically targeting flood prediction scenarios. While the proposed framework was designed to be adaptable to other domains (such as smart cities or healthcare), its performance in these areas has not yet been tested. Further studies are needed to confirm the applicability of the framework in different contexts.
- **Geographical and Temporal Scope**: The sensor data used in the feasibility study were collected from specific regions over a limited period. Environmental conditions, such as rainfall patterns and river behavior, may differ in other regions or seasons, which could affect the ability of the framework to predict events in different locations or times. Future studies should consider datasets from diverse geographic and temporal contexts to assess the framework's broader applicability.

5.6.3 Construct Validity

Construct validity refers to how well the proposed solution measures the intended constructs and achieves its stated goals. In this study, construct validity could be threatened by:

- **Definition of Critical Events**: The framework's performance hinges on the accurate definition of critical events, such as flood thresholds. If the thresholds used to classify events are not well-calibrated or contextually appropriate for different regions, the framework might produce inaccurate predictions. Although predefined thresholds were selected based on domain knowledge, these definitions may need to be adjusted for different applications or scenarios.
- **Semantic Ontology Extension**: The SSN-based ontology was extended to capture domain-specific knowledge for the hydrological domain. While the ontology was rigorously designed and evaluated, the construct validity of the semantic model could be questioned if its underlying assumptions do not align with real-world complexities. More validation and refinement may be required to ensure that the ontology remains applicable across diverse sensor systems.

5.6.4 Conclusion Validity

Conclusion validity deals with the reliability of the conclusions drawn from the results of the study. The main threats to conclusion validity in this work include:

- **Statistical Significance**: The results of the framework's performance were derived from experiments using specific datasets and metrics. While statistical methods were employed to evaluate the models, the sample size and variability of the sensor data might not be sufficient to claim strong statistical significance for all findings. To mitigate this, multiple runs of the experiments were conducted, but further replication with larger datasets is necessary to strengthen the conclusions.
- **Model Performance Metrics**: The performance of the event prediction model was evaluated using standard metrics such as precision, recall, and F1-score. However, these metrics might not fully capture all aspects of real-time critical event detection, especially when dealing with complex environmental interactions. Additional metrics, such as real-time response effectiveness or system latency, could provide a more comprehensive understanding of the framework's practical performance.

5.6.5 Mitigation Strategies

To address these threats to validity, several mitigation strategies were implemented. First, the use of cross-validation, walk-forward validation, and multiple experimental runs helped reduce the impact of data bias and variance. Second, the design of the framework allows for flexibility in extending the ontology and adapting the AI model, which enhances generalizability. Third, ongoing efforts to evaluate the framework in different domains and contexts will provide further insights into its external validity and robustness.

In conclusion, while the presented framework shows promising results, acknowledging these potential threats to validity is crucial for interpreting the findings and planning future work. Further experimentation, testing in different domains, and collaboration with domain experts will help reduce the impact of these threats and ensure that the framework can be broadly applied with confidence.

5.7 Final Considerations of the Chapter

In this chapter, we analyzed the results of the proposed framework, focusing on the performance of semantic data integration and the AI models used for event prediction. The results demonstrated the framework's ability to effectively integrate heterogeneous sensor data and improve prediction accuracy through the use of contextual rules. Additionally, the impact of context-aware semantic rules on event detection and long-term prediction was discussed, further validating the robustness and scalability of the framework.

The insights gained from this analysis underscore the framework's potential for addressing complex event prediction tasks, particularly in dynamic and data-rich environments. In the final chapter, we will summarize the key findings of this research, discuss its contributions to the field, and explore potential avenues for future work.

6 Conclusion and Future Works

The integration of sensor data is crucial in various scenarios, such as environmental monitoring, smart cities, and disaster management. These systems consist of devices that collect and transmit data in real-time. However, the heterogeneity and dispersion of sensor data pose significant challenges in identifying events of interest and predicting future occurrences. The complexity involved in integrating data from diverse sources and the need for contextual analysis to extract meaningful insights highlight the importance of developing solutions that can efficiently unify and analyze this data. This is essential for proactive decision-making and the implementation of preventive actions.

In this work, we propose a framework for sensor data integration and event prediction, utilizing an ontology-based approach and artificial intelligence techniques. The framework consists of layers, starting with data acquisition and preprocessing, where wrappers transform raw data into a standardized canonical format, intending to ensure consistency and quality. Next, an extended ontology based on the Semantic Sensor Network (SSN) is applied to add semantic context, allowing for a unified and enriched representation of the data. The artificial intelligence layer employs Long Short-Term Memory (LSTM) networks for event prediction, capturing temporal dependencies in the data. This layered architecture enables a holistic approach to sensor data integration and analysis, addressing challenges such as data heterogeneity and prediction accuracy. The proposed solution aims to provide a robust and scalable methodology, adaptable to various domains and applications.

The feasibility study aimed to evaluate the proposed framework's ability to semantically and efficiently integrate diverse data sources and accurately predict critical events. The study utilized real-world scenarios, specifically data from hydrological and hydrometric monitoring stations, to assess the framework's performance. Structured according to the Goal-Question-Metric (GQM) model, the evaluation focused on three key metrics: semantic canonical representation of data, diversity and number of integrated data sources, and the performance of the AI model. The ontology model was crucial in standardizing and contextualizing the data, which was then processed by LSTM networks for event prediction.

The study's findings demonstrated that the ontology effectively provided a canonical representation of sensor data, facilitating integration and contextualization (M1 metric). The framework successfully handled diverse data sources, including hydrological and hydrometric data (M2 metric), showcasing its capacity to integrate a wide range of sensor data sources. The performance of the AI model, evaluated using precision, recall, and F1-score metrics (M3), showed significant improvement in predicting critical events like floods due to the inclusion of contextual information from the ontology. However, the study also identified limitations, such as varying recall across different stations and challenges in long-term predictions, suggesting the need for further refinement in context rules and the AI model.

In summary, the proposed framework successfully integrated data from various sources and provided accurate, context-specific predictions. These results indicate that the framework is viable and adaptable to different domains, demonstrating potential for broader application beyond the hydrological context. However, certain challenges remain, such as the reliance on manual context rules and performance variation between different stations, highlighting areas for future improvement. The key contributions of the framework include achieving semantic integration of heterogeneous sensor data and enhancing event prediction through data contextualization. The modular and scalable structure allows for the seamless addition of new data sources and specific adjustments, showcasing its adaptability and robustness.

Future work should focus on automating the definition and application of context rules, reducing reliance on manual interventions, and enhancing AI models for improved long-term prediction accuracy. Specifically, for the domain evaluated in the feasibility study, future developments should include incorporating more attributes, such as regional topography and geographical features, to provide more comprehensive data for analysis. Additionally, constructing broader contexts that encompass more than just pairs of stations will help in capturing wider patterns and relationships within the data, further enhancing the framework's predictive capabilities. Expanding the framework to encompass new domains and data types will ensure greater generalization and applicability, making it more versatile and effective across various scenarios. Emphasizing real-time processing will enable the framework to handle data streams more effectively, ensuring timely and accurate predictions in dynamic environments. By addressing these areas, the framework can become more robust, efficient, and applicable to a wider range of scenarios, significantly enhancing its practical utility.

7 Appendix

Table 14 – Forward snowballing: Qualitative synthesis

Reference	Title	Journal	Main Contributions	Methodology	Conclusions
(7)	A Survey on Event Prediction Methods from a Systems Perspective: Bringing Together Disparate Research Areas	ACM Computing Surveys	Integrates event prediction methods across disciplines into a unified systems perspective.	Reviewed literature, created a taxonomy, and assessed event prediction methods.	Identifies challenges and suggests future directions for event prediction research.
(2)	Comparison of Different Artificial Intelligence Techniques to Predict Floods in Jhelum River, Pakistan	Water	Evaluates the performance of AI models for predicting floods in the Jhelum River.	Employed various AI models, comparing them using statistical performance measures.	The LLR model outperformed others, offering a basis for an early warning system.
(34)	Optimization of LSTM Parameters for Flash Flood Forecasting Using Genetic Algorithm	Water Resources Management	Developed a method to optimize LSTM parameters for flood forecasting using genetic algorithms.	Optimized LSTM parameters with genetic algorithms and tested using historical flood data.	The optimized LSTM model significantly improved forecasting accuracy over baseline models.
(41)	Comprehensive Overview of Flood Modeling Approaches: A Review of Recent Advances	Hydrology	Reviews current flood modeling approaches and their advances.	Conducted a literature review to categorize and evaluate various flood modeling techniques.	Discusses the strengths and weaknesses of models, identifies challenges, and suggests future research directions.

Reference	Title	Journal	Main Contributions	Methodology	Conclusions
(56)	Enhancing Flood Prediction using Ensemble and Deep Learning Techniques	International Arab Conference on Information Technology (ACIT)	Developed and compared four machine learning models for flood prediction in Ghana.	Trained and evaluated models using historical rainfall and water level data.	XGBoost showed superior performance, highlighting machine learning's potential in flood prediction.
(66)	Daily flow discharge prediction using integrated methodology based on LSTM models: Case study in Brahmani-Baitarani basin	HydroResearch	Developed a model for daily flow discharge prediction using LSTM models.	Utilized hydrological data and predictive modeling with LSTM.	The integrated model enhanced accuracy in flow discharge predictions for water management.
(76)	Deep learning in hydrology and water resources disciplines: concepts, methods, applications, and research directions	Journal of Hydrology	Reviewed deep learning applications in hydrology and water resources.	Discussed deep learning methods to address hydrological data and challenges.	Emphasized deep learning's transformative potential for water resource management.
(77)	Real-time flood forecasting based on a general dynamic neural network framework	Stochastic Environmental Research and Risk Assessment	Explored real-time flood forecasting using dynamic neural networks.	Used hybrid training algorithms to enhance forecasting models.	NARX models showed high accuracy, proving effective in real-time flood forecasting.

Reference	Title	Journal	Main Contributions	Methodology	Conclusions
(45)	Real-time flood forecasting based on a general dynamic neural network framework	Sustainable Development	Overviewed the use of deep learning for flood modeling and forecasting.	Combined bibliometric analysis and qualitative research to define research directions.	Suggested improvements and innovations in deep learning applications for flood research.
(24)	Explainable artificial intelligence in disaster risk management: Achievements and prospective futures	International Journal of Disaster Risk Reduction	Explored the application of explainable AI in disaster risk management.	Reviewed XAI applications using bibliometric and qualitative analyses.	Highlighted the potential of XAI to enhance transparency and management practices in disaster risk scenarios.
(64)	Multivariate Hydrological Modeling Based on Long Short-Term Memory Networks for Water Level Forecasting	Information	Developed a multivariate LSTM network model for water level forecasting.	Utilized a model with multiple inputs and outputs, compared against other models.	The LSTM model demonstrated higher accuracy, emphasizing the role of advanced ML techniques in forecasting.
(68)	Hydrometeorology Forecasting	Hydrometeorology	Detailed discussion on hydrological forecasting techniques, focusing on model configurations, uncertainties, and operational considerations.	Explores various hydrological forecasting methods including rainfall-runoff models, hydrological flow routing, and hydrodynamic modeling. Discusses model setup, calibration, validation, and the management of forecast uncertainties.	Highlights the integration of traditional methods with modern technologies like remote sensing and GIS to enhance forecast accuracy and utility. Emphasizes the importance of forecast verification in improving hydrological forecasting techniques.

Bibliography

1 Mohamed Aboualola, Khalid Abualsaud, Tamer Khattab, Nizar Zorba, and Hossam S Hassanein. Edge technologies for disaster management: A survey of social media and artificial intelligence integration. *IEEE Access*, 2023.

2 Fahad Ahmed, Ho Huu Loc, Edward Park, Muhammad Hassan, and Panuwat Joyklad. Comparison of different artificial intelligence techniques to predict floods in jhelum river, pakistan. *Water*, 14(21):3533, 2022.

3 Jamal Al Qundus, Kosai Dabbour, Shivam Gupta, Régis Meissonier, and Adrian Paschke. Wireless sensor network for ai-based flood disaster detection. *Annals of Operations Research*, pages 1–23, 2022.

4 Jefferson Amará, Victor Ströele, Regina Braga, and Michael Bauer. Sensor data integration using ontologies for event detection. In *International Conference on Advanced Information Networking and Applications*, pages 171–183. Springer, 2023.

5 Jefferson Amará, Victor Ströele, Regina Braga, and José Maria N David. From context to forecast: Ontology-based data integration and ai for events prediction. In *International Conference on Advanced Information Networking and Applications*, pages 349–361. Springer, 2024.

6 Muhammad Asfand-E-Yar and Ramis Ali. Semantic integration of heterogeneous databases of same domain using ontology. *IEEE Access*, 8:77903–77919, 2020.

7 Janik-Vasily Benzin and Stefanie Rinderle-Ma. A survey on event prediction methods from a systems perspective: Bringing together disparate research areas. *arXiv preprint arXiv:2302.04018*, 2023.

8 James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

9 Camila Bezerra, Fred Freitas, and Filipe Santana. Evaluating ontologies with competency questions. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 3, pages 284–285. IEEE, 2013.

10 S Borsch and Y Simonov. Operational hydrologic forecast system in russia. In *Flood Forecasting*, pages 169–181. Elsevier, 2016.

11 Victor R Basili1 Gianluigi Caldiera and H Dieter Rombach. The goal question metric approach. *Encyclopedia of software engineering*, pages 528–532, 1994.

12 Shobhit Chaturvedi, Elangovan Rajasekar, Sukumar Natarajan, and Nick McCullen. A comparative assessment of sarima, lstm rnn and fb prophet models to forecast total and peak monthly energy demand for india. *Energy Policy*, 168:113097, 2022.

13 Animesh Choudhury, Avinash Chand Yadav, and Stefania Bonafoni. A response of snow cover to the climate in the northwest himalaya (nwh) using satellite products. *Remote Sensing*, 13(4):655, 2021.

14 Sai Sree Laya Chukkapalli, Sudip Mittal, Maanak Gupta, Mahmoud Abdelsalam, Anupam Joshi, Ravi Sandhu, and Karuna Joshi. Ontologies and artificial intelligence systems for the cooperative smart farming ecosystem. *Ieee Access*, 8:164045–164064, 2020.

15 Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. *Journal of Web Semantics*, 17:25–32, 2012.

16 Ümit Demirbaga, Gagangeet Singh Aujla, Anish Jindal, and Oğuzhan Kalyon. Big data storage solutions. In *Big Data Analytics: Theory, Techniques, Platforms, and Applications*, pages 127–153. Springer, 2024.

17 AnHai Doan, Alon Halevy, and Zachary Ives. *Principles of data integration*. Elsevier, 2012.

18 Rodrigo Pereira dos Santos. *Managing and monitoring software ecosystem to support demand and solution analysis*. PhD thesis, Ph. D. thesis, Universidade Federal do Rio de Janeiro, 2016.

19 Haizhou Du, Yan Zhou, Yunpu Ma, and Shiwei Wang. Astrologer: Exploiting graph neural hawkes process for event propagation prediction with spatio-temporal characteristics. *Knowledge-Based Systems*, 228:107247, 2021.

20 Li Du, Xiao Ding, Yue Zhang, Ting Liu, and Bing Qin. A graph enhanced bert model for event prediction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2628–2638, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.206. URL `https://aclanthology.org/2022.findings-acl.206`.

21 Ashutosh Kumar Dubey, Abhishek Kumar, Vicente García-Díaz, Arpit Kumar Sharma, and Kishan Kanhaiya. Study and analysis of sarima and lstm in forecasting time series data. *Sustainable Energy Technologies and Assessments*, 47:101474, 2021.

22 Taha Falatouri, Farzaneh Darbanian, Patrick Brandtner, and Chibuzor Udokwu. Predictive analytics for demand forecasting–a comparison of sarima and lstm in retail scm. *Procedia Computer Science*, 200:993–1003, 2022.

23 Mariano Fernández-López, Asunción Gómez-Pérez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. 1997.

24 Saman Ghaffarian, Firouzeh Rosa Taghikhah, and Holger R Maier. Explainable artificial intelligence in disaster risk management: Achievements and prospective futures. *International Journal of Disaster Risk Reduction*, 98:104123, 2023.

25 Global Disaster Alert and Coordination System (GDACS). Flood alerts and warnings. `https://www.gdacs.org/Knowledge/models_fl.aspx`, 2024. Accessed: 2024-04-11.

26 Fatma Ezzahra Gmati, Salem Chakhar, Wided Lejouad Chaari, and Mark Xu. A taxonomy of event prediction methods. In *Advances and Trends in Artificial Intelligence. From Theory to Practice: 32nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2019, Graz, Austria, July 9–11, 2019, Proceedings 32*, pages 12–26. Springer, 2019.

27 Government of Alberta. Alberta floods. `https://floods.alberta.ca/`, 2024. Accessed: 2024-04-13.

28 Nicola Guarino. *Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy*, volume 46. IOS press, 1998.

29 Nicola Guarino, Daniel Oberle, and Steffen Staab. What is an ontology? *Handbook on ontologies*, pages 1–17, 2009.

30 Armin Haller, Krzysztof Janowicz, Simon D Cox, Danh Le Phuoc, Kerry Taylor, and Maxime Lefrançois. Semantic sensor network ontology, 2017.

31 Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

32 Michael Y Hu, Guoqiang Zhang, Christine X Jiang, and B Eddy Patuwo. A cross-validation analysis of neural network out-of-sample performance in exchange rate forecasting. *Decision Sciences*, 30(1):197–216, 1999.

33 Yu Huang and Tie-jian Luo. Nosql database: A scalable, availability, high performance storage for big data. In *Pervasive Computing and the Networked World: Joint International Conference, ICPCA/SWS 2013, Vina del Mar, Chile, December 5-7, 2013. Revised Selected Papers*, pages 172–183. Springer, 2014.

34 You-Da Jhong, Chang-Shian Chen, Bing-Chen Jhong, Cheng-Han Tsai, and Song-Yue Yang. Optimization of lstm parameters for flash flood forecasting using genetic algorithm. *Water Resources Management*, 38(3):1141–1164, 2024.

35 Ralph E Johnson and Brian Foote. Designing reusable classes. *Journal of object-oriented programming*, 1(2):22–35, 1988.

36 Joint Research Centre of the European Commission and Dartmouth Flood Observatory. Global flood detection system: Data products specification. `https://floodobservatory.colorado.edu/AMSR-E%20Gaging%20Reaches/About%20GFDS.htm`, 2016. Accessed: 2024-04-11.

37 Staffs Keele et al. Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, ver. 2.3 ebse technical report. ebse, 2007.

38 A Khatoon, Y Hafeez, S Asghar, and T Ali. An ontological framework for requirement change management in distributed environment. *The Nucleus*, 51(2):291–301, 2014.

39 Yubin Kim, Xuhai Xu, Daniel McDuff, Cynthia Breazeal, and Hae Won Park. Health-llm: Large language models for health prediction via wearable sensor data. *arXiv preprint arXiv:2401.06866*, 2024.

40 Rajalakshmi Krishnamurthi. An overview of iot sensor data processing, fusion, and analysis techniques. *Sensors*, 20(21):6076, 2020.

41 V Kumar, KV Sharma, T Caloiero, DJ Mehta, and K Singh. Comprehensive overview of flood modeling approaches: a review of recent advances. hydrology 10 (7): 141, 2023.

42 Vijendra Kumar, Naresh Kedam, Kul Vaibhav Sharma, Darshan J Mehta, and Tommaso Caloiero. Advanced machine learning techniques to improve hydrological prediction: A comparative analysis of streamflow prediction models. *Water*, 15(14): 2572, 2023.

43 Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246, 2002.

44 James Lewis and Martin Fowler. A definition of this new architectural term, 2014.

45 Hongyang Li, Mingxin Zhu, Fangxin Li, and Martin Skitmore. Solving flood problems with deep learning technology: Research status, strategies, and future directions. *Sustainable Development*, 2022.

46 Wei Li, Amin Kiaghadi, and Clint Dawson. Exploring the best sequence lstm modeling architecture for flood prediction. *Neural Computing and Applications*, 33: 5571–5580, 2021.

47 Zhipeng Li, Shanshan Feng, Jun Shi, Yang Zhou, Yong Liao, Yangzhao Yang, Yangyang Li, Nenghai Yu, and Xun Shao. Future event prediction based on temporal knowledge graph embedding. *Computer Systems Science & Engineering*, 44(3), 2023.

48 Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

49 Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

50 Jin Liu, Yunhui Li, Xiaohu Tian, Arun Kumar Sangaiah, and Jin Wang. Towards semantic sensor data: an ontology approach. *Sensors*, 19(5):1193, 2019.

51 Yutaka Matsubara. Fast mining and forecasting of complex time-stamped events. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 271–279, 2012.

52 Merriam-Webster. Definition of sensor. `https://www.merriam-webster.com/dictionary/sensor`, n.d. Accessed: 2024-07-12.

53 Amnai Mohamed, Choukri Ali, Yossef Fakhri, Gherabi Noreddine, et al. A survey on the challenges of data integration. In *2022 9th International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pages 1–6. IEEE, 2022.

54 Grey Nearing, Deborah Cohen, Vusumuzi Dube, Martin Gauch, Oren Gilon, Shaun Harrigan, Avinatan Hassidim, Daniel Klotz, Frederik Kratzert, and Asher Metzger. Global prediction of extreme floods in ungauged watersheds. *Nature*, 627(8004): 559–563, 2024.

55 Nadia Nedjah, Dong Xu, and Feng Ye. A hydrological data prediction model based on lstm with attention mechanism. *Water*, 15(4):670, 2023. doi: 10.3390/w15040670.

56 Isaac Kofi Nti, Owusu Nyarko-Boateng, Samuel Boateng, Faiza Umar Bawah, Promise Ricardo Agbedanu, Nicodemus Songose Awarayi, Peter Nimbe, Adebayo Felix Adekoya, Benjamin Asubam Weyori, and Vivian Akoto-Adjepong. Enhancing flood prediction using ensemble and deep learning techniques. In *2021 22nd International Arab Conference on Information Technology (ACIT)*, pages 1–9. IEEE, 2021.

57 Gustavo Nuñez, Carlos Ezequiel Buckle, and Marcos Daniel Zárate. Integrating oceanographic sensor data using ssn/sosa ontology. In *Facultad de Informática*, pages 53–57. Universidad Nacional de La Plata, 2023. ISBN 978-950-34-2271-7. URL `http://sedici.unlp.edu.ar/handle/10915/155442`.

58 Inès Osman, Sadok Ben Yahia, and Gayo Diallo. Ontology integration: approaches and challenging issues. *Information Fusion*, 71:38–63, 2021.

59 Oxford Languages. Definition of sensor. `https://www.lexico.com/definition/sensor`, n.d. Accessed: 2024-04-12.

60 M Tamer Özsu and Patrick Valduriez. *Principles of distributed database systems*, volume 2. Springer, 1999.

61 M. Tamer Özsu and Patrick Valduriez. *Principles of Distributed Database Systems*. Springer International Publishing, 2020. ISBN 9783030262532. doi: 10.1007/978-3-030-26253-2. URL `http://dx.doi.org/10.1007/978-3-030-26253-2`.

62 Nuno Pessanha Santos. The expansion of data science: Dataset standardization. *Standards*, 3(4):400–410, 2023.

63 John W. Pomeroy, Ronald E. Stewart, and Paul H. Whitfield. The 2013 flood event in the south saskatchewan and elk river basins: Causes, assessment and damages. *Canadian Water Resources Journal/Revue canadienne des ressources hydriques*, 41 (1-2):105–117, 2016.

64 Jackson B Renteria-Mena, Douglas Plaza, and Eduardo Giraldo. Multivariate hydrological modeling based on long short-term memory networks for water level forecasting. *Information*, 15(6):358, 2024.

65 Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.

66 Abinash Sahoo, Swayamshu Satyapragnya Parida, Sandeep Samantaray, and Deba Prakash Satapathy. Daily flow discharge prediction using integrated methodology based on lstm models: Case study in brahmani-baitarani basin. *HydroResearch*, 7:272–284, 2024.

67 Hend A Selmy, Hoda K Mohamed, and Walaa Medhat. A predictive analytics framework for sensor data using time series and deep learning techniques. *Neural Computing and Applications*, 36(11):6119–6132, 2024.

68 Kevin Sene. Hydrological forecasting. In *Hydrometeorology: Forecasting and Applications*, pages 167–215. Springer, 2024.

69 Uppala Meena Sirisha, Manjula C Belavagi, and Girija Attigeri. Profit prediction using arima, sarima and lstm models in time series forecasting: A comparison. *IEEE Access*, 10:124715–124727, 2022.

70 Anushka Srivastava and Manoranjan Rai Bharti. Hybrid machine learning model for anomaly detection in unlabelled data of wireless sensor networks. *Wireless Personal Communications*, 129(4):2693–2710, 2023.

71 Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

72 Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Boris Villazón-Terrazas. How to write and use the ontology requirements specification document. In *On the Move to Meaningful Internet Systems: OTM 2009: Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009, Vilamoura, Portugal, November 1-6, 2009, Proceedings, Part II*, pages 966–982. Springer, 2009.

73 Wang-Chiew Tan. Deep data integration. In *SIGMOD Conference*, page 2, 2021.

74 Saurabh Singh Thakur, Shabbir Syed Abdul, Hsiao-Yean Chiu, Ram Babu Roy, Po-Yu Huang, Shwetambara Malwade, Aldilas Achmad Nursetyo, and Yu-Chuan Li. Artificial-intelligence-based prediction of clinical events among hemodialysis patients using non-contact sensor data. *Sensors*, 18(9):2833, 2018.

75 R Thirumahal, G Sudha Sadasivam, and P Shruti. Semantic integration of heterogeneous data sources using ontology-based domain knowledge modeling for early detection of covid-19. *SN Computer Science*, 3(6):428, 2022.

76 Kumar Puran Tripathy and Ashok Kumar Mishra. Deep learning in hydrology and water resources disciplines: Concepts, methods, applications, and research directions. *Journal of Hydrology*, page 130458, 2023.

77 Xinyu Wan, Qingyang Wu, Zhenyu Cao, and Yan Wu. Real-time flood forecasting based on a general dynamic neural network framework. *Stochastic Environmental Research and Risk Assessment*, 37(1):133–151, 2023.

78 Lei Wang, Zijie Chen, Hailin Zou, Dongsheng Huang, Yuanyuan Pan, Chak-Fong Cheang, and Jianqing Li. A deep learning-based high-temperature overtime working alert system for smart cities with multi-sensor data. *Nondestructive Testing and Evaluation*, 39(1):164–184, 2024.

79 Wenyue Wang, Klemens Hocke, and Christian Mätzler. Physical retrieval of rain rate from ground-based microwave radiometry. *Remote sensing*, 13(11):2217, 2021.

80 Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, pages 1–10, 2014.

81 Liang Zhao. Event prediction in the big data era: A systematic survey. *ACM Computing Surveys (CSUR)*, 54(5):1–37, 2021.

82 Kun Zhou, Wen Yong Wang, Teng Hu, and Chen Huang Wu. Comparison of time series forecasting based on statistical arima model and lstm with attention mechanism. In *Journal of physics: conference series*, volume 1631, page 012141. IOP Publishing, 2020.