Project Acronym	SILVANUS
Grant Agreement number	101037247 (H2020-LC-GD-2020-3)
Project Full Title	Integrated Technological and Information Platform for Wildfire Management
Funding Scheme	IA – Innovation action

DELIVERABLE INFORMATION

Deliverable Number	D5.4
Deliverable Name	Semantic information fusion framework
Dissemination level	Public
Type of Document	Demonstrator
Contractual date of delivery	30/09/2024
Deliverable Leader	CTL
Status & version	Final
WP/Task responsible	WP5
Keywords:	Wildfire management, Semantic Information Fusion Framework, Environmental sensors, Social media, Multilingual alert system, Decision support, Semantic knowledge base, Data integration
Abstract:	This framework is engineered to enhance wildfire prevention, detection, and restoration phases by synthesizing and analyzing diverse data streams from environmental sensors, social media, and other real-time sources into a dynamic and cohesive semantic knowledge base. These deliverable details the development of sophisticated tools and systems, including real-time alert mechanisms and AI-assisted decision-support models, which leverage this integrated data to provide timely and effective responses to wildfire incidents. The framework also introduces innovative semantic reasoning and multilingual communication capabilities to adapt to varying environmental conditions and linguistic needs across different regions.

Deliverable Leader:	CTL
Contributors:	HB, EAI, AMIKOM, CTL, VTG, UISAV

Reviewers:	UISAV, DELL, CMCC
Supervised by:	DELL

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No 101037247

Disclaimer

All information in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose.

The user there of uses the information at its sole risk and liability. For the avoidance of all doubts, the European Commission has no liability in respect of this document, which is merely representing the authors' view.

Document History			
Version	**Date**	**Contributor(s)**	**Description**
V0.1	10/6/2024	CTL	Assigned contributors; Table of Contents and basic structure
V0.2	24/7/2024	CTL	Initial text has been added by CTL, currently under construction.
V0.3	13/09/2024	HB, EAI, AMIKOM, CTL, VTG, UISAV	All the contributing partners have provided their inputs, and CTL has additionally integrated them, making the document ready for internal review.
Final	26/09/2024	CTL, CMCC,UISAV, DELL	Integration of review comments

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

Table 1: List of acronyms and abbreviations

ACRONYM	Description
AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
BA	Burn Area
BERTopic	Topic modeling using BERT embeddings
BFAM	Bidirectional Feature Aggregation Module
CASPAR	Structured Data Semantic Exploitation Framework
CNN	Convolutional Neural Network
DSS	Decision Support System
DSL	Domain-Specific Language
DTM	Dynamic Topic Models
DTR	Decision Tree Regression
EFFIS	European Forest Fire Information System
FWI	Canadian Fire Weather Index
GAN	Generative Adversarial Network
GFMC	Global Fire Monitoring Center
GIS	Geographic Information System
GPR	Gaussian Process Regression
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise
IoT	Internet of Things
JSON	JavaScript Object Notation
KDE	Kernel Density Estimation
KG	Knowledge Graph
LLMs	Large Language Models
LPCN-SR	Lossless Pooling Convolutional Networks for Super Resolution
LWS	Lightweight Semantics
NN	Neural Network
NWCG	National Wildfire Coordinating Group
OWL	Web Ontology Language

RDF	Resource Description Framework
RDF(S)	Resource Description Framework (Schema)
RF	Random Forest
NER	Named Entity Recognition
NLP	Natural Language Processing
SAL	Storage Abstraction Layer
SBERT	Sentence-BERT, a modification of the BERT model
SemKB	SILVANUS Semantic Knowledge Base
SPARQL	SPARQL Protocol and RDF Query Language
SVR	Support Vector Regression
t-SNE	t-Distributed Stochastic Neighbor Embedding
UI	User Interface
UP	User Product
UP3	User Product 3
UP4a	User Product 4a
UP9b	User Product 9b
UP9h	User Product 9h
VAE	Variational Autoencoder
VM	Virtual Machine

Table 2: List of beneficiaries

No	Partner Name	Short name	Country
1	UNIVERSITA TELEMATICA PEGASO	PEGASO	Italy
2	ZANASI ALESSANDRO SRL	Z&P	Italy
3	NETCOMPANY-INTRASOFT SA	INTRA	Luxembourg
4	THALES	TRT	France
5	FINCONS SPA	FINC	Italy
6	ATOS IT SOLUTIONS AND SERVICES IBERIA SL	ATOS IT	Spain
6.1	ATOS SPAIN SA	ATOS SA	Spain
7	EMC INFORMATION SYSTEMS INTERNATIONAL	DELL	Ireland
8	SOFTWARE IMAGINATION & VISION SRL	SIMAVI	Romania
9	CNET CENTRE FOR NEW ENERGY TECHNOLOGIES SA	EDP	Portugal
10	ADP VALOR SERVICOS AMBIENTAIS SA	ADP	Portugal
11	TERRAPRIMA - SERVICOS AMBIENTAIS SOCIEDADE UNIPESSOAL LDA	TP	Portugal
12	3MON, s. r. o.	3MON	Slovakia
13	CATALINK LIMITED	CTL	Cyprus
14	SYNTHESIS CENTER FOR RESEARCH AND EDUCATION LIMITED	SYNC	Cyprus
15	EXPERT SYSTEM SPA	EAI	Italy
16	ITTI SP ZOO	ITTI	Poland
17	Venaka Treleaf GbR	VTG	Germany
18	MASSIVE DYNAMIC SWEDEN AB	MDS	Sweden
19	FONDAZIONE CENTRO EURO-MEDITERRANEOSUI CAMBIAMENTI CLIMATICI	CMCC F	Italy
20	EXUS SOFTWARE MONOPROSOPI ETAIRIA PERIORISMENIS EVTHINIS	EXUS	Greece
22	Micro Digital d.o.o.	MD	Croatia
23	POLITECHNIKA WARSZAWSKA	WUT	Poland
24	HOEGSKOLAN I BORAS	HB	Sweden
25	GEOPONIKO PANEPISTIMION ATHINON	AUA	Greece
26	ETHNIKO KENTRO EREVNAS KAI TECHNOLOGIKIS ANAPTYXIS	CERTH	Greece
27	PANEPISTIMIO THESSALIAS	UTH	Greece
28	ASSOCIACAO DO INSTITUTO SUPERIOR TECNICO PARA A INVESTIGACAO E DESENVOLVIMENTO	IST	Portugal

29	VELEUCILISTE VELIKA GORICA	UASVG	Croatia
30	USTAV INFORMATIKY, SLOVENSKA AKADEMIA VIED	UISAV	Slovakia
31	POMPIERS DE L'URGENCE INTERNATIONALE	PUI	France
32	THE MAIN SCHOOL OF FIRE SERVICE	SGPS	Poland
33	ASSET - Agenzia regionale Strategica per lo Sviluppo Ecosostenibile del Territorio	ASSET	Italy
34	LETS ITALIA srls	LETS	Italy
35	Parco Naturale Regionale di Tepilora	PNRT	Italy
36	FUNDATIA PENTRU SMURD	SMURD	Romania
37	Romanian Forestry Association - ASFOR	ASFOR	Romania
38	KENTRO MELETON ASFALEIAS	KEMEA	Greece
39	ELLINIKI OMADA DIASOSIS SOMATEIO	HRT	Greece
40	ARISTOTELIO PANEPISTIMIO THESSALONIKIS	AHEPA	Greece
41	Ospedale Israelitico	OIR	Italy
42	PERIFEREIA STEREAS ELLADAS	PSTE	Greece
43	HASICSKY ZACHRANNY SBOR MORAVSKOSLEZSKEHO KRAJE	FRB MSR	Czechia
44	Hrvatska vatrogasna zajednica	HVZ	Croatia
45	TECHNICKA UNIVERZITA VO ZVOLENE	TUZVO	Slovakia
46	Obcianske zdruzenie Plamen Badin	PLAMEN	Slovakia
47	Yayasan AMIKOM Yogyakarta	AMIKOM	Indonesia
48	COMMONWEALTH SCIENTIFIC AND INDUSTRIAL RESEARCH ORGANISATION	CSIRO	Australia
50	FUNDACAO COORDENACAO DE PROJETOS PESQUISAS E ESTUDOS TECNOLOGICOS COPPETEC	COPPETEC	Brazil
51	Rinicom Ltd	RINICOM	UK

EXECUTIVE SUMMARY

This deliverable is part of WP5 and it reports on the SILVANUS project's work in improving wildfire management through the integration of semantic technologies. With semantics acting as the common thread connecting all the sections, D5.4 provides an in-depth description of the system and its outcomes.

More specifically, at the center of this effort is the sophisticated Semantic Knowledge Base (SemKB), which has been extensively developed to handle and analyze data from social media, environmental sensors, and other real-time sources. In order to maintain the information's relevance and speed for emergency responses, this knowledge base facilitates complex semantic mapping and data population procedures. On top of SemKB, the Integrated Data Insights part leverages the semantic framework to provide decision-support tools and real-time alert systems. An important development covered in D5.4 is the incorporation of Semantic Intensity Information to the semantic framework. The framework gains greater analytical power and can make more complex decisions by adding intensity measures to the semantic analysis. Furthermore, the use of lightweight semantics and the use of reasoning schemes are examined in the document. These techniques support the system's efficacy and adaptability in the face of shifting data requirements and environmental changes. Additionally, the Woode application monitors biodiversity and forest conditions using cutting-edge AI and semantics, greatly improving public outreach and educational efforts related to forest conservation. This innovative AI tool is crucial for evaluating the ecological effects of wildfires and supporting plans for adaptation and restoration after wildfire incidents. Last but not least, the creation of a Multilingual Forest Fire Alert System is also discussed in detail, highlighting the project's dedication to providing timely and easily understandable wildfire notifications across linguistically different terrain.

Considered together, these elements demonstrate SILVANUS's innovative strategy for using semantic technology to thorough and efficient wildfire management, establishing a fresh norm for related projects in the future.

1 Introduction

As the SILVANUS project, supported by the European Union's Horizon 2020 program, moves towards its conclusion, D5.4 represents a critical milestone, exemplifying the integration of semantic technologies into wildfire management. This deliverable embodies the collective expertise of an interdisciplinary team, and it marks the completion of a Semantic Information Fusion Framework that not only anticipates and manages wildfires but also substantially elevates decision-making capabilities through advanced semantic processing. D5.4 showcases significant advancements in semantic integration within the SILVANUS project through its detailed presentation of the Semantic Information Fusion Framework. It effectively assimilates a various heterogeneous data inputs, from environmental sensors to social media insights, into a unified semantic structure. This framework leverages these inputs to support dynamic, real-time decision-making across different linguistic and cultural contexts, demonstrating the framework's capacity to handle evolving semantic needs. This deliverable highlights significant achievements such as the implementation of a multilingual alert system and the application of AI technologies for biodiversity assessments, which showcase the project's innovative approach to managing and analyzing diverse datasets. The integration of semantic intensity information and evolving semantics further illustrates the sophisticated capabilities of the framework to adapt and respond to complex wildfire scenarios. Through D5.4, the SILVANUS project demonstrates a transformative approach to wildfire management, illustrating the robust potential of semantic technologies to enhance the accuracy and timeliness of decision-making processes essential for effective disaster management.

The rest of this document contains the following sections detailed below:

Section 2: *The Semantic Knowledge Base:* This section explores the SILVANUS semantic model in depth, detailing its ontology framework, diverse data sources, semantic mapping techniques, and real-time data population processes. It discusses how the model uses ontologies to systematically organize and interpret data from varied sources like environmental sensors and social media, enhancing the foundational UP9h, which is further elaborated in the subsequent section.

Section 3: *Integrated Data Insights:* This part describes systems that leverage the semantic knowledge base to enable real-time monitoring and decision-making. It highlights alert-based systems and case studies that demonstrate the practical application and integration of these systems within the broader SILVANUS platform through UP9h, alongside discussions on visualization capabilities that enhance user interaction and data interpretation.

Section 4: *Augmenting Semantic Space by Semantic Intensity Information:* This section investigates methods to enrich the semantic space with detailed intensity information, improving the framework's capability for deeper knowledge extraction and enhanced decision-making. It includes practical experiments in information fusion and methodologies to track and analyze fire intensity, sentiment, and semantics via social media data.

Section 5: *Semantic Reasoning Schemes and Evolving Semantics:* Here, the focus shifts to the adoption of lightweight semantic schemes as alternatives to traditional W3C standards, aimed at increasing the system's adaptability and responsiveness to changing and new semantic contexts.

Section 6: *AI Technologies for Modeling Biodiversity:* This section covers the application of advanced AI technologies to assess biodiversity impacts due to wildfires. It details specific technologies deployed, such as the Woode Application's backend technologies, super-resolution imaging, and classifiers for identifying tree species, which collectively enhance the ecological analysis within the project scope.

Section 7: *Multilingual Forest Fire Alert System:* It details the development and implementation of a multilingual alert system, emphasizing methods used for dataset preparation, preprocessing, and linguistic fine-tuning to cater to diverse language needs, enhancing accessibility and efficacy in emergency communications.

Section 8: *Conclusions:* The concluding section synthesizes the insights and developments described throughout the document, providing a cohesive overview of the deliverable’s achievements and outlining future directions for research and application within the SILVANUS project framework.

2 The Semantic Knowledge Base

In the development of a semantic information fusion framework for wildfire management, it is crucial to consider the existing body of knowledge and technologies that have contributed to the field. The theoretical background for the semantic information fusion framework in the SILVANUS project is anchored in several key concepts of semantic web technologies, knowledge representation, and data integration. These concepts are critical for unifying heterogeneous data sources into a coherent and interoperable system.

- *Ontologies and Semantic Models:* Ontologies are essential for structuring and integrating data from diverse sources. An ontology defines a common vocabulary and the relationships between terms, facilitating shared understanding across different systems and domains. In the context of wildfire management, ontologies help to model various entities such as sensors, weather conditions, fire incidents, and geographical locations. This structured approach ensures that data from different sources can be combined meaningfully, supporting advanced reasoning and analytics.
- *Knowledge Graphs:* A knowledge graph is a structured representation of semantic data, where entities (nodes) and their interrelations (edges) are captured. Knowledge graphs enable the integration of disparate data points into a unified model, which can be queried and analyzed to derive insights. In this framework, the knowledge graph continuously ingests new data in near real-time, maintaining an up-to-date and comprehensive view of the wildfire-related information. The dynamic nature of knowledge graphs supports real-time decision-making and situational awareness.
- *RDF and SPARQL:* The Resource Description Framework (RDF)[1] is a standard model for data interchange on the web. RDF facilitates data merging even if the underlying schemas differ, making it ideal for integrating heterogeneous data sources. SPARQL (SPARQL Protocol and RDF Query Language)[2] is used to query and manipulate RDF data. SPARQL's powerful querying capabilities enable complex searches over the knowledge graph, allowing users to extract specific information, perform analytics, and derive insights necessary for effective wildfire management.
- *Semantic Reasoning and Inference:* Semantic reasoning involves deriving new knowledge from existing data using logical rules and ontologies. This process allows for the inference of implicit information that is not directly stated in the data. For example, if the knowledge graph indicates dry weather conditions and high wind speeds in a specific region, semantic reasoning can infer an increased risk of wildfire in that area. This capability is crucial for proactive wildfire management, enabling the prediction and prevention of potential incidents.
- *Data Fusion and Integration:* Data fusion refers to the process of integrating multiple data sources to produce more consistent, accurate, and useful information than that provided by any individual data source. In the SILVANUS project, data fusion techniques combine sensor data, social media posts, and other relevant datasets. This integrated approach enhances the quality and completeness of the information, supporting comprehensive wildfire monitoring and management.
- *Microservices and Modular Architecture:* The framework's architecture is designed to be modular and scalable, using microservices. Each microservice is responsible for a specific function, such as data ingestion, semantic processing, or alert generation. This modular approach allows for flexibility and ease of maintenance, as individual components can be updated or replaced without affecting the entire system. The use of microservices also facilitates the integration of new data sources and functionalities as needed.
- *Message Brokers and Asynchronous Communication:* To handle real-time data ingestion and processing, the framework employs message brokers like RabbitMQ[3]. Message brokers manage the flow of data between microservices, ensuring that data is processed asynchronously and efficiently. This setup supports the near real-time updating of the knowledge graph, enabling timely alerts and responses to emerging wildfire threats.

[1] https://www.w3.org/RDF/
[2] https://www.w3.org/TR/sparql11-query/
[3] https://www.rabbitmq.com/

- *Event Logging and Auditing:* Event logging is a critical component for tracking system activities, diagnosing issues, and ensuring accountability. By recording system events, the framework can provide a detailed audit trail of data processing and decision-making processes. This transparency is essential for validating the system's performance and ensuring the reliability of the alerts and insights generated.

In summary, the theoretical foundations of the semantic information fusion framework encompass a range of concepts and technologies designed to integrate and analyze heterogeneous data sources effectively.

The application of semantic technologies in wildfire management is supported by several pioneering platforms and projects. These initiatives have demonstrated the potential of integrating diverse data sources and employing advanced data processing techniques to enhance wildfire monitoring, prediction, and response. Numerous projects have successfully integrated semantic technologies into wildfire management frameworks. The FireSense project[4,5,6] by NASA, for example, utilizes a combination of sensor networks, computer vision, and data fusion techniques to monitor and predict forest fires. By integrating data from thermal cameras, weather stations, and other sensors, FireSense can provide real-time detection and early warnings of wildfire incidents. Another notable system is IQ FireWatch[7], which employs automated early wildfire detection technology to identify smoke and hot spots in real time. This system uses multi-spectral sensing technology and AI-based algorithms to monitor forest areas continuously, providing crucial early warnings to help manage and mitigate wildfires. IQ FireWatch has been implemented in various regions globally, including Germany, Chile, and the United States, contributing significantly to early detection and wildfire management efforts

Effective wildfire management relies on the integration of diverse data sources. Practical implementations often involve the fusion of remote sensing data, weather information, social media feeds, and ground sensor data. The European Forest Fire Information System (EFFIS)[8] is a prime example of such an approach. EFFIS combines satellite imagery, meteorological data, and ground-based observations to deliver timely and accurate information on wildfire occurrences and risks across Europe. This integrated approach supports comprehensive wildfire monitoring and facilitates coordinated response efforts.

Projects like SILVANUS aims to build upon these foundations by developing a more robust and scalable semantic information fusion framework. By unifying heterogeneous data sources under a common semantic layer, this approach ensures consistent and meaningful data integration, enhancing the overall effectiveness of wildfire management strategies. Real-time monitoring and alert systems are vital components of effective wildfire management. The integration of semantic technologies enables the development of sophisticated alert systems that leverage real-time data ingestion and semantic reasoning. For instance, the PROMETHEUS system integrates GIS data, fire spread models, and real-time sensor data to predict and alert authorities about the spread of wildfires. This system utilizes semantic reasoning to enhance the accuracy and timeliness of alerts, providing actionable insights that aid in decision-making. Furthermore, RDFox[9], a knowledge graph and reasoning engine, enables incremental reasoning and real-time data processing, dynamically updating the knowledge graph as new data arrives. This capability is crucial for applications requiring real-time insights, such as wildfire alert systems, ensuring timely and accurate responses to emerging threats.

Thε rest of this chapter presents the SILVANUS Semantic Knowledge Base (SemKB) under T5.2 lead by CTL, which, in essence, is the semantic model presented in D3.1 populated with instance data and further discussed in subsection 2.1. SemKB also encompasses the repository (i.e., RDF triplestore) for persisting the populated model, as well as the ontology populating framework. The latter is presented in subsections 2.4 and 2.5, while Figure 1 gives an overview of SemKB.

[4] https://esto.nasa.gov/new-nasa-firesense-technology-program-will-help-first-responders-combat-wildfires/
[5] https://appliedsciences.nasa.gov/our-impact/news/nasa-makes-firesense
[6] https://science.nasa.gov/science-research/science-enabling-technology/nasa-wildfire-digital-twin-pioneers-new-ai-models-and-streaming-data-techniques-for-forecasting-fire-and-smoke/
[7] https://www.iq-firewatch.com/
[8] https://forest-fire.emergency.copernicus.eu/
[9] https://www.oxfordsemantic.tech/rdfox

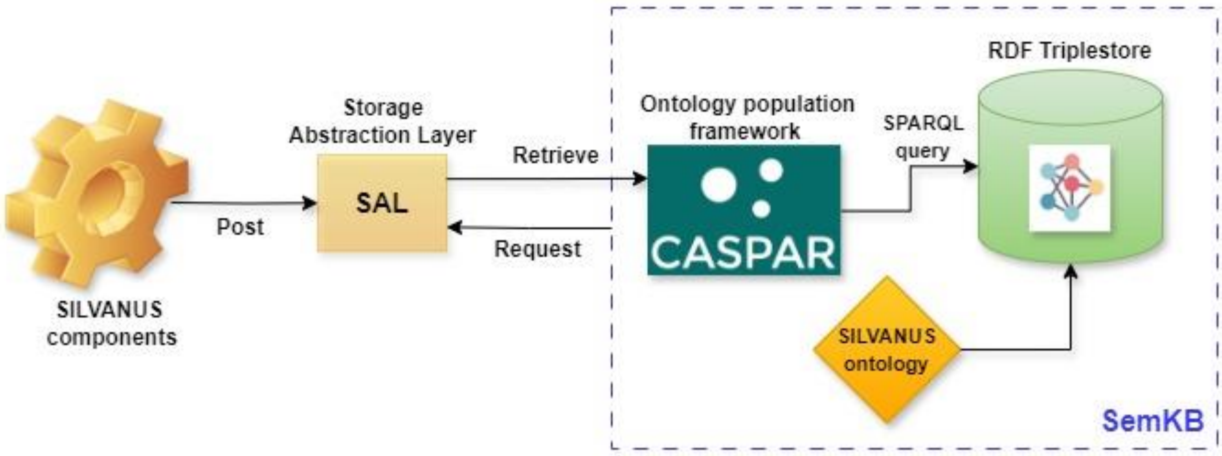


Figure 1 Semantic Knowledge Graph (SemKB) overview.

2.1 SILVANUS Semantic Model

The SILVANUS Semantic Model serves as a comprehensive framework for integrating and managing wildfire-related information through enhanced data fusion and interoperability. This chapter details the updates since submitting D3.1 and the utilization of the SILVANUS ontology as part of the SemKB. These updates, evaluation processes, and utilization strategies have been implemented to ensure the model's robustness, accuracy, and relevance. These efforts are essential for facilitating effective decision-making, improving situational awareness, and supporting various wildfire management activities within the SILVANUS project.

2.1.1 Ontology updates

The SILVANUS ontology has undergone significant updates to enhance its clarity, accuracy, and support specifically for the task of information fusion. These updates were driven both by project partners and by the feedback received during the first project review.

The first project review highlighted several areas where the SILVANUS ontology could be improved to ensure it serves as a robust and reliable framework for wildfire-related information. To address these concerns, the team undertook a comprehensive revision of the ontology, focusing on incorporating missing terms, aligning with established standards, and refining the ontology's structure.

One of the primary actions taken was the incorporation of several key terms that were previously missing from the ontology. This effort was needed to enhance the comprehensiveness of the ontology and ensure it could represent a wide range of wildfire-related concepts. The new terms added include "early warnings," "lookout towers," "aerial means," and "retardants." These terms were primarily sourced from the National Wildfire Coordinating Group (NWCG) Glossary, a recognized authority in wildfire terminology. By integrating these terms, the ontology now provides a more complete representation of critical elements in wildfire management.

Recognizing the importance of standardized terminology in fostering effective information fusion, the project team aligned the SILVANUS ontology with established wildfire domain standards. This alignment involved consulting several authoritative sources, including the NWCG, the Global Fire Monitoring Center (GFMC) FAO Fire Management Glossary[10], and the European Glossary for Wildfires and Forest Fires.

For instance, the term "fire speed" was replaced with the standardized term "fire spread rate" to ensure consistency with NWCG terminology, as shown in Figure 2. Additionally, the classification of wildfire causes was aligned with the European standard as defined by the European Forest Fire Information System (EFFIS). This alignment ensures that the ontology uses consistent and widely accepted terminology, which is crucial

[10] https://gfmc.online/literature/glossary.html

for interoperability and accurate data exchange during information fusion. Specific examples of terminological changes include:

"FireSeverity" changed to "BurnSeverity."
"ResponseResources/ResponseVehicle/Aircraft" changed to "Airtanker."
"ResponseResources" changed to "Resources."
"Volunteer" changed to "Volunteer firefighter."

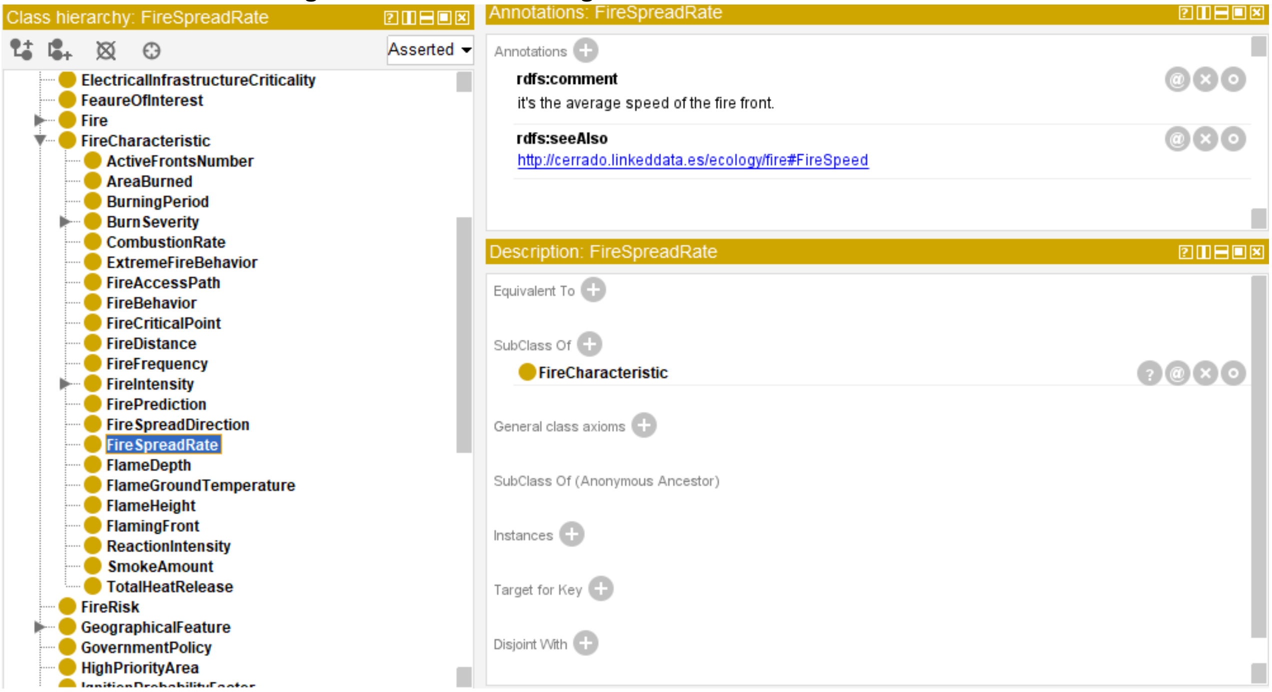


Figure 2: "Fire Spread Rate" class, former "Fire Speed"

Furthermore, new terms were added based on these sources to enhance the ontology's comprehensiveness. Examples of these terms include "Wildland Urban Interface", "Active fire", "Fuel model", "Assessment Fire Weather", "Beaufort wind scale", "Burning period", "Combustion rate", "Crown out", "Fuel dryness level", "Fuel size class", "Flaming front" and "Flame depth".

Beyond addressing the review feedback, the SILVANUS ontology was also updated to better support the task of information fusion. Information fusion involves integrating data from multiple sources to produce more consistent, accurate, and useful information. To enhance the ontology's capacity in this area, several new concepts and properties were added.

A key enhancement was the introduction of new concepts and properties designed to represent and integrate diverse information sources relevant to information fusion tasks. These additions include:

- **Media and Textual Representation:** Concepts such as hasAbsoluteMediaURL, AbsoluteMediaURL, hasTextCategory, TextCategory, hasTextualConcept, and TextualConcept were introduced to better handle media and textual data.
- **Event and Platform Categorization:** New properties like hasEventCategory, EventCategory, hasPlatform, and Platform were added to categorize events and platforms more effectively in the context of information sourced from Social Media.
- **Detection and Probability Indices:** To support the detection and probabilistic analysis of wildfire events, concepts and properties such as RaspberryPi Detection (highlighted in Figure 3), hasFireDetection, hasFireDetectionScore, hasFireProbabilityIndex, and FireProbabilityIndex were incorporated.
- **Social Media Integration:** Properties like isFake, isQuote, isRelevant, isRetweet, and PostEvent were added to facilitate the integration of social media data, which can facilitate real-time information fusion and benefit situational awareness.

- **Additional Metadata:** Other properties such as hasID, hasText, hasTimestamp, hasDescription, hasRaspberryPiSensor, RaspberryPiSensor, hasRelativeMediaPath, and RelativeMediaPath were introduced to enrich the metadata associated with various data points.

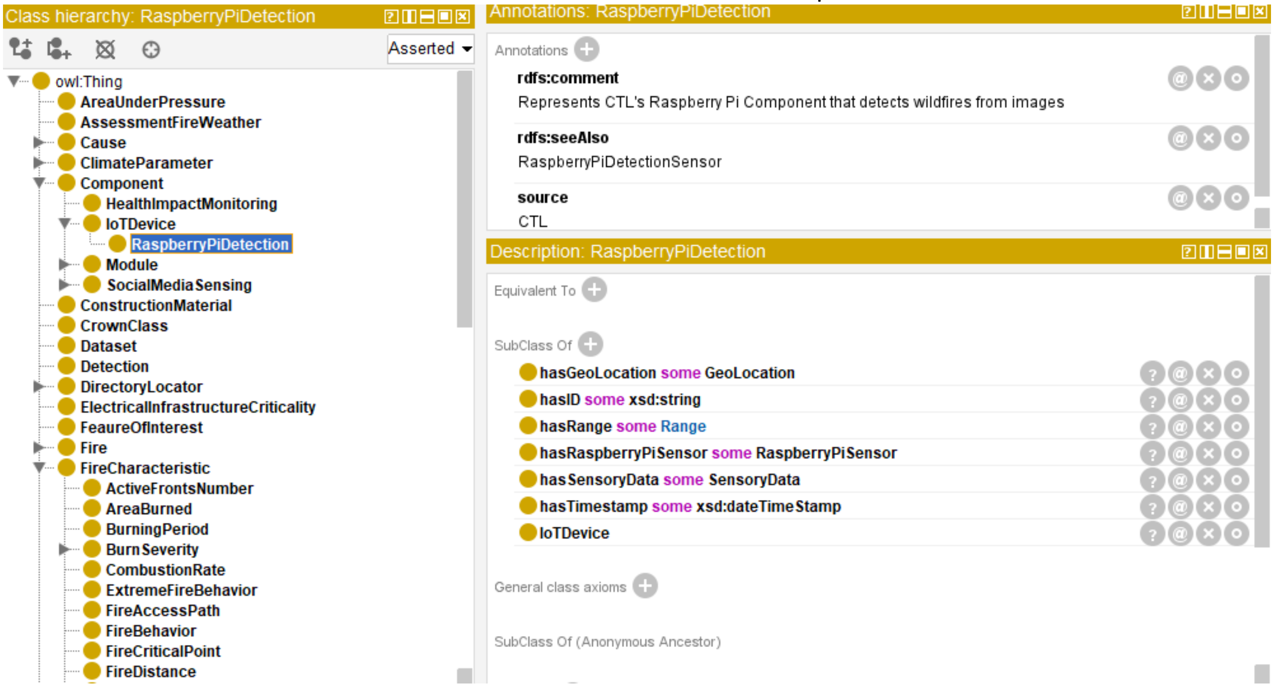


Figure 3: RaspberryPi Detection class

These additions are designed to enhance the ontology's ability to integrate and process diverse types of data, thereby supporting more effective information fusion. For example, the inclusion of properties related to media URLs and textual categories allows for a more detailed representation of media content, which can facilitate the analysis of wildfire-related information.
These updates to the SILVANUS ontology represent a step forward in improving its clarity, accuracy, and support for information fusion. By addressing the feedback from the first project review, incorporating missing terms, aligning with established standards, and refining the ontology's structure, the ontology's comprehensiveness and reliability were improved. Furthermore, the addition of new concepts and properties specifically designed for information fusion tasks ensures that the ontology is well-equipped to handle the complex and diverse data sources involved in wildfire management. These enhancements collectively contribute to a more robust and effective framework for representing and integrating wildfire-related information, thereby supporting the SILVANUS project's overarching goal of improving wildfire management and mitigation.

2.1.2 Ontology Evaluation

In accordance with the guidelines provided by the Ontology Summit 2013 Communiqué[11], our ontology underwent a comprehensive evaluation process to ensure its quality, effectiveness, and suitability for its intended purpose.

The 2013 report is an effort in advancing the understanding and adoption of ontology evaluation practices. It discusses the challenges of evaluating ontologies, offers a model for the ontology life cycle and presents evaluation criteria in the context of the phases of said life cycle. The report recommends incorporating ontology evaluation strategies across all phases of the ontology life cycle and conducting evaluation against carefully identified requirements. The evaluation of our ontology encompassed multiple phases and criteria, as outlined below.

[11] https://ontologforum.com/index.php/OntologySummit2013

During the Requirements Development Phase, the rationale and expected benefits of the ontology were established. The ontology was deemed necessary to address the complex challenges posed by forest fires and to facilitate interdisciplinary collaboration and the integration of innovative technologies in fire management. Its intended usage encompasses fire prevention, detection, and restoration, serving as a knowledge framework and enabling interoperability among various stakeholders.

To determine the scope of the ontology and identify relevant concepts, a set of competency questions were formulated. These questions guided the development process and ensured that the ontology covered key aspects of fire management, such as sensors and devices, fire causes and characteristics, climatic parameters, resources and tools, critical infrastructure, biodiversity, and best practices. The competency questions were representative of the intended usages and provided a foundation for capturing the domain-specific knowledge.

The Ontological Analysis Phase aimed to verify the adequacy of the ontology in capturing relevant terms and entities within the defined scope. The documentation and unambiguous nature of the ontology were examined, along with the agreement of domain experts with the ontological analysis. The ontology successfully met these evaluation criteria, with feedback from experts helping refine the representation.

The Ontology Design Phase focused on the chosen ontology language, the expressiveness of the query language, and the adherence to design principles. The ontology was designed using the RDF language, which is widely adopted for ontology development. SPARQL, a query language, was chosen to formalize the competency questions. The design also emphasized the reuse of existing ontologies by selectively incorporating relevant parts and establishing loose links using the "seeAlso" property.

The System Design Phase examined the integration of the ontology within the overall system architecture. The operations to be performed using the ontology, the interfaces, data sources, and tools required for ontology development, evaluation, configuration management, and maintenance were considered. The system design phase ensured that the ontology would be compatible with the adjacent components and facilitated knowledge base population.

During the Ontology Development Phase, the informal and formal modeling results were evaluated. The ontology's fidelity, including the documentation of terms and the capture of entities within its scope, was verified.

The System Development and Integration Phase involved building the system as specified in the design phase and ensuring successful integration of the ontology with other components. A collaboration with project partners enabled the integration and operational adaptation of the ontology.

In the Deployment Phase, the ontology's compliance with requirements and its provision of new capabilities were evaluated. Although at the time of writing the project is still in development, no outstanding problems were identified that would disrupt the deployment of the ontology.

Through rigorous evaluation following the guidelines provided by the Ontology Summit 2013 Communiqué, the ontology demonstrated its fitness for purpose, capturing relevant knowledge, meeting requirements, and supporting interoperability in the domain of fire management and emergency response. Notably, out of the 53 evaluation questions suggested in the Ontology Summit 2013 Communiqué, 44 questions were answered positively, corresponding to an 83% of compliance with the Ontology Summit 2013 Communiqué guidelines. This level of compliance reinforces the ontology's robustness and adherence to best practices in ontology development, further validating its suitability for practical application in the challenging domain of wildfire management. Ongoing maintenance and collaboration efforts with project partners aim to further refine and adapt the ontology for operational use.

2.1.3 Ontology Utilization

The wildfire ontology, described in detail in deliverable D3.1, provides the foundational framework for the SemKB. In this deliverable, we focus on how the ontology is applied within the knowledge base to integrate and utilize data from various sources.

Several classes and properties within the existing wildfire ontology have been enhanced to accommodate the specific needs and heterogeneous data sources of the SILVANUS project. These enhancements include:

Sensor Data Integration: Classes for different types of sensors (e.g., temperature, humidity, smoke detectors) and their properties (e.g., measurement range, accuracy) have been expanded to allow for the integration of real-time sensor data.

Weather Data Representation: Classes for weather conditions have been enhanced to include properties such as temperature, humidity, wind speed, and precipitation. These enhancements are critical for modeling the environmental factors that influence wildfire behavior.

Fire Incident Details: Classes for fire incidents have been expanded to include detailed properties such as incident location, severity, spread rate, and affected areas, ensuring comprehensive documentation and analysis of wildfire events.

Geographical Information: Classes representing geographical locations have been refined to support detailed spatial data, including coordinates, region boundaries, and topographical features, facilitating accurate mapping and visualization of wildfire incidents and their impacts.

Social Media Integration: New classes have been introduced to capture data from social media platforms, including posts, user interactions, and metadata, enabling the incorporation of public reports and observations into the wildfire management framework.

The process of populating these fields within the ontology involves several steps. Data from various sources is collected and pre-processed to ensure quality and consistency. This data is then mapped to the corresponding classes and properties in the ontology using predefined templates. The CASPAR framework[12] facilitates this mapping process by transforming the data into SPARQL queries and populating the RDF triplestore.

Continuous ingestion and updating of the knowledge base are enabled by microservices and message brokers, ensuring that the information remains current and relevant. Semantic reasoning techniques are applied to infer new knowledge and generate insights, supporting real-time decision-making and alert systems. This approach not only enhances immediate response efforts but also contributes to long-term planning and prevention strategies within the SILVANUS platform.

2.2 Data Sources

As part of T5.2, three primary components have been identified and agreed upon as data sources for the SemKB. These data sources, which are also key user products of SILVANUS, provide outputs in predefined JSON formats and are critical for the ontology population process described in the following sections. Each data source is detailed below, with a focus on its role and significance within the SemKB framework.

2.2.1 UP9b – DSS - Health impact assessment

The UP9b – DSS - Health Impact Assessment developed by UTH provides observations related to health impacts, specifically focusing on air quality. More information can be found in D4.5 concerning pilot results and latest updates while D5.3 offers an initial description of device components. The JSON structure includes the following key components:

Header:

[12] https://caspar.catalink.eu/

- *client_key*: A predefined key, "silvanus_client_uth_key," used for authentication and identification.

Body:
- *health_impact_observations*: An array of observation objects, each containing:
  - *uuid*: A unique identifier for the observation.
  - *timestamp*: The date and time when the observation was recorded.
  - *sensor*: Type of sensor used (e.g., CO2).
  - *owner*: Ownership status of the sensor (e.g., static).
  - *air_quality*: Quality of air reported (e.g., good).
  - *location*: An array of location objects, each with:
    - *placename*: Name of the place.
    - *geometry*: Geometric details including type (e.g., MultiPoint) and coordinates (latitude and longitude).
  - *age_group*: Age group category of the individuals affected (e.g., adult).
  - *description*: Additional description of the observation.

Example JSON snippet:


```
{
    "header": {
        "client_key": "silvanus_client_uth_key"
    },
    "body": {
        "health_impact_observations": [
            {
                "uuid": "232832673137286",
                "timestamp": "2022-09-07T15:00:00Z",
                "sensor": "CO2",
                "owner": "static",
                "air_quality": "good",
                "location": [
                    {
                        "placename": "Example place polygon",
                        "geometry": {
                            "type": "MultiPoint",
                            "coordinates": [
                                {
                                    "lat": -73.9580,
                                    "lon": 40.8003
                                },
                                {
                                    "lat": -73.9580,
                                    "lon": 40.8003
                                },
                                {
                                    "lat": -73.9580,
                                    "lon": 40.8003
                                },
                                {
                                    "lat": -73.9580,
                                    "lon": 40.8003
                                }
                            ]
                        }
                    }
                ],
                "age_group": "adult",
                "description": "Example description"
```

```
            }
        ]
    }
}
```


2.2.2 UP4a – Fire/Smoke detection from IoT devices

The UP4a component developed by CTL outputs a data source that focuses on detections made by Raspberry Pis that also encompass various environmental sensors. This component was initially described in D4.1**,** with algorithmic developments detailed in D4.2 and updates provided in D5.1. The pilot demonstrations and results, along with any encountered limitations, are thoroughly discussed in D4.5**.** The JSON structure is as follows:

Header:

- *client_key*: A predefined key, "silvanus_client_ctl_key," used for authentication and identification.

Body:

- *raspberrypi_detections*: An array of detection objects, each containing:
  - *uuid*: A unique identifier for the detection event.
  - *sensor_type*: Types of sensors used (e.g., humidity, temperature, camera).
  - *range*: Measurement range including value and unit (e.g., meters).
  - *timestamp*: The date and time when the detection was recorded.
  - *location*: An array of location objects, each with:
    - *placename*: Name of the place.
    - *geometry*: Geometric details including type (e.g., Point) and coordinates (latitude and longitude).
  - *sensory_data*: Detailed sensory data including temperature, humidity, and gas sensor readings.
  - *visual_data*: Visual data analysis including fire, smoke detection scores and captured image.

Example JSON snippet:


```
{
    "header": {
        "client_key": "silvanus_client_ctl_key"
    },
    "body": {
        "raspberrypi_detections": [
            {
                "uuid": "01879875-7508-7b37-abc5-6f8df91c8ccc",
                "sensor_type": [
                    "humidity",
                    "temperature",
        "smoke_gas",
                    "camera"
                ],
                "range": {
                    "value": 5,
                    "unit": "meters"
                },
                "timestamp": "2023-04-19T07:41:39.463138Z",
                "location": [
                    {
                        "placename": "croatia-pilot",
                        "geometry": {
                                "type": "Point",
```

```
                                "coordinates": [
                                    {
                                        "lat": 45.470987,
                                        "lon": 14.237045
                                    }
                                ]
                        }
                    }
                ],
                "sensory_data": {
                    "temperature": {
                        "value": 13.4,
                        "unit": "celsius"
                    },
                    "humidity": {
                        "value": 55.5,
                        "unit": "percentage"
                    },
                    "gas_sensor": {
                        "smoke": 5.62555,
                        "liquid_petrolium_gas_lpg": 1.323912,
                        "methane": 2.795502,
                        "hydrogen": 2.823664,
                        "unit": "parts_per_million(ppm)"
                    }
                },
                "visual_data": {
                    "fire_detection": {
                        "contains_fire": false,
                        "fire_score": 0.02293,
                        "fire_probability": "none"
                    },
                    "smoke_detection": {
                        "contains_smoke": false,
                        "smoke_score": 0.0033
                        "smoke_probability": "none"
                    }
                     "image":
"data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAoAAAAHgCAIAAAC6s0uzAAAg
AElEQVR4AezBWY+lWXYe5vdde....  "
                }
            }
        ]
    }
}
```


Note that because the base64 encoding[13] of the image is too large to display, only part of it is displayed in the example above.

2.2.3 UP3 - Fire detection based on social sensing

The UP3 component, led by CERTH, outputs a data source that includes social media post events, particularly focusing on fire-related incidents. This component leverages social sensing techniques to detect and monitor fires in real-time through the analysis of social media data. An initial description of the UP3 component and its integration is provided in D4.2, which details the development of this UP for fire detection based on social media inputs. Once again, updates, pilot demonstrations, results, and limitations

[13] https://en.wikipedia.org/wiki/Base64

of this component are extensively discussed in D4.5. These deliverables offer comprehensive insights into the implementation, testing, and refinement of the social media sensing capabilities for fire detection, so it is avoided being repeated in the current deliverable. It is a massive subcomponents collection of service, creating a JSON file like the one described below:

Header:

- *client_key*: A predefined key, "silvanus_client_certh_key," used for authentication and identification.

Body:

- *post_events*: An array of post event objects, each containing:
    - *uuid*: A unique identifier for the post event.
    - *posts*: An array of individual posts, each with:
        - *uuid*: Unique identifier for the post.
        - *text*: Content of the post.
        - *timestamp*: The date and time when the post was made.
        - *media_url*: URLs of media associated with the post.
        - *media_type*: Type of media (e.g., image).
        - *is_retweet*: Boolean indicating if the post is a retweet.
        - *is_quote*: Boolean indicating if the post is a quote.
        - *language*: Language of the post.
        - *platform*: Platform where the post was uploaded.
        - *analysis_concepts*: Analysis concepts derived from the post, including:
            - *module_name*: Name of the analysis module.
            - *analysis_type*: Type of analysis (e.g., Visual, Textual).
            - *concepts*: Concepts identified in the analysis.
            - *metadata*: Additional metadata related to the concepts.
        - *text_is_relevant*: Relevance classification for the text.
        - *text_categories*: Categories assigned to the text.
        - *event_type*: Type of event recognized.
        - *extracted_locations*: Locations extracted from the text, each with:
            - *module_name*: Name of the extraction module.
            - *analysis_type*: Type of analysis (e.g., Textual).
            - *geolocation*: Geolocation details including placename and coordinates.

Example JSON snippet:


```
{
    "header": {
        "client_key": "silvanus_client_certh_key"
    },
    "body": {
        "post_events": [
            {
                "id": "T20221213190000",
                "posts": [
                    {
                        "id": "1602746121675218944",
                        "text":  "RT  @chris_avramidis:  RT  @chris_avramidis: Φωτιές και οδοφράγματα αυτή τη στιγμή στην περιοχή της Ροτόντας.\n\n#16χρονος #antireport https://t.co/6pvHbf27kzΦωτιές και οδοφράγματα αυτή τη στιγμή στην περιοχή της Ροτόντας.\n\n#16χρονος #antireport https://t.co/6pvHbf27kz",
                        "timestamp": "2022-12-13T19:23:47Z",
                        "media_url": [
                            "https://pbs.twimg.com/media/Fj4X3YuXgAE7qXp.jpg"
```

```
                          ],
                          "media_type": "image",
                          "is_retweet": true,
                          "is_quote": false,
                          "language": "el",
                          "platform": "twitter",
                          "analysis_concepts": [
                              {
                                  "module_name": "CTL_concept_analysis_Visual",
                                  "analysis_type": "Visual",
                                  "concepts": [
                                      "fire",
                                      "smoke"
                                  ],
                                  "metadata": [
                                      {
                                          "concept_name": "fire",
                                          "score": 0.9754
                                      },
                                      {
                                          "concept_name": "smoke",
                                          "score": 0.7423
                                      }
                                  ]
                              },
                              {
                                  "module_name":
"UISAV_concept_analysis_Textual",
                                  "analysis_type": "Textual",
                                  "concepts": [
                                      "vineyard_fire",
                                      "firefighters_fighting",
                                      "nearby_forest"
                                  ]
                              },
                              {
                                  "module_name": "HB_concept_analysis_Textual",
                                  "analysis_type": "Textual",
                                  "concepts": [
                                      "wildfire_state",
                                      "declares_wildfire",
                                      "state_emergency"
                                  ]
                              },
                              {
                                  "module_name": "EAI_concept_analysis_Textual",
                                  "analysis_type": "Textual",
                                  "concepts": [
                                      "city",
                                      "affected_asset",
                                      "address",
                                      "affected_human"
                                  ],
                                  "metadata": [
                                      {
                                          "concept_name": "city",
                                          "score": 1,
                                          "value": "Bari"
                                      },
                                      {
```

```
                        "concept_name": "affected_asset",
                        "score": 0.51,
                        "value": "bus"
                    },
                    {
                        "concept_name": "address",
                        "score": 0.999,
                        "value": "Bolt Creek"
                    },
                    {
                        "concept_name": "affected_human",
                        "score": 0.999,
                        "value": "autista"
                    }
                ]
            },
            {
                "module_name": "HB_concept_analysis_Visual",
                "analysis_type": "Visual",
                "concepts": [
                    "person",
                    "truck"
                ],
                "metadata": [
                    {
                        "concept_name": "truck",
                        "score": 0.9846,
                        "bounding_box": [
                            52,
                            78,
                            254,
                            648
                        ]
                    },
                    {
                        "concept_name": "person",
                        "score": 0.9906,
                        "bounding_box": [
                            364,
                            157,
                            736,
                            427
                        ]
                    }
                ]
            },
            {
                "module_name": "CERTH_concept_analysis_Visual",
                "analysis_type": "Visual",
                "concepts": [
                    "Sun",
                    "Outdoor",
                    "Nighttime",
                    "Sky",
                    "Explosion_Fire",
                    "Clouds",
                    "Amateur_Video"
                ]
            },
            {
```

```
                                "module_name":  "ATOS_concept_analysis_Textual-
Visual",
                                "analysis_type": "Textual-Visual",
                                "concepts": [
                                    "Fire_in_text",
                                    "Fire_in_image"
                                ]
                            }
                        ],
                        "text_is_relevant": [
                            {
                                "module_name":
"CERTH_relevance_classficiation",
                                "fire": true
                            },
                            {
                                "module_name": "HB_relevance_classficiation",
                                "fire": false
                            },
                            {
                                "module_name":
"AMIKOM_relevance_classficiation",
                                "fire": true
                            }
                        ],
                        "text_categories": {
                            "module_name": "UISAV_text_categories",
                            "categories": [
                                {
                                    "category_name": "urban_fire",
                                    "score": "0.7"
                                }
                            ]
                        },
                        "event_type": {
                            "module_name": "EAI_event_recognition",
                            "event_name": "fire",
                            "score": 0.6,
                            "positions": [
                                3,
                                5
                            ]
                        },
                        "extracted_locations": [
                            {
                                "module_name": "CERTH_location_extraction",
                                "analysis_type": "Textual",
                                "geolocation": {
                                    "placename": "Πάρκο Ροτόντας, Kamara, 1st
District of Thessaloniki, Thessaloniki Municipal Unit, Municipality of
Thessaloniki, Thessaloniki Regional Unit, Central Macedonia, Macedonia and
Thrace, Greece",
                                    "geometry": {
                                        "type": "Point",
                                        "crs": "WGS84",
                                        "coordinates": [
                                            {
                                                "lat": 40.6333609,
                                                "lon": 22.9529216142
                                            }
```

```
                                ]
                            }
                        }
                    },
                    {
                        "module_name": "UISAV_location_extraction",
                        "analysis_type": "Textual",
                        "geolocation": {
                            "placename": "Πάρκο Ροτόντας, Kamara, 1st
District of Thessaloniki, Thessaloniki Municipal Unit, Municipality of
Thessaloniki, Thessaloniki Regional Unit, Central Macedonia, Macedonia and
Thrace, Greece",
                            "geometry": {
                                "type": "Point",
                                "crs": "WGS84",
                                "coordinates": [
                                    {
                                        "lat": 40.6333609,
                                        "lon": 22.9529216142
                                    }
                                ]
                            }
                        }
                    },
                    {
                        "module_name": "AMIKOM_location_extraction",
                        "analysis_type": "Textual",
                        "geolocation": {
                            "placename": "Πάρκο Ροτόντας, Kamara, 1st
District of Thessaloniki, Thessaloniki Municipal Unit, Municipality of
Thessaloniki, Thessaloniki Regional Unit, Central Macedonia, Macedonia and
Thrace, Greece",
                            "geometry": {
                                "type": "Point",
                                "crs": "WGS84",
                                "coordinates": [
                                    {
                                        "lat": 40.6333609,
                                        "lon": 22.9529216142
                                    }
                                ]
                            }
                        }
                    },
                    {
                        "module_name": "ATOS_location_extraction",
                        "analysis_type": "Visual",
                        "geolocation": {
                            "placename": "Πάρκο Ροτόντας, Kamara, 1st
District of Thessaloniki, Thessaloniki Municipal Unit, Municipality of
Thessaloniki, Thessaloniki Regional Unit, Central Macedonia, Macedonia and
Thrace, Greece",
                            "geometry": {
                                "type": "Point",
                                "crs": "WGS84",
                                "coordinates": [
                                    {
                                        "lat": 40.6333609,
                                        "lon": 22.9529216142
                                    }
```

```
                                        ]
                                    }
                                }
                            }
                        ]
                    }
                ],
                "gallery": [
                    "https://pbs.twimg.com/media/Fj4X3YuXgAE7qXp.jpg"
                ],
                "tags": [
                    "Nighttime",
                    "Clouds",
                    "Sun",
                    "Amateur_Video",
                    "Sky",
                    "Explosion_Fire",
                    "Outdoor"
                ],
                "event_description": "Possible fire in location: Πάρκο Ροτόντας,
Kamara, 1st District of Thessaloniki, Thessaloniki Municipal Unit, Municipality
of Thessaloniki, Thessaloniki Regional Unit, Central Macedonia, Macedonia and
Thrace, Greece",
                "location": {
                    "module_name": "CERTH_location_extraction",
                    "analysis_type": "Textual",
                    "geolocation": {
                        "placename": "Πάρκο Ροτόντας, Kamara, 1st District of
Thessaloniki, Thessaloniki Municipal Unit, Municipality of Thessaloniki,
Thessaloniki Regional Unit, Central Macedonia, Macedonia and Thrace, Greece",
                        "geometry": {
                            "type": "Point",
                            "crs": "WGS84",
                            "coordinates": [
                                {
                                    "lat": 40.6333609,
                                    "lon": 22.9529216142
                                }
                            ]
                        }
                    }
                },
                "timestamp": "2022-12-13T19:00:00Z"
            }
        ]
    }
}
```


2.3 Semantic Mapping

The semantic mapping refers to the process of mapping each field of the JSON from the above sources into an ontology concept, such as an instance of a class or object type, and data type properties. This section details the semantic mappings for each of the above data sources, as visualized through graffoo[14] in the provided figures.

[14] https://essepuntato.it/graffoo/

2.3.1 UP9b mapping

U9b as described earlier, focuses on observations related to health impacts, particularly air quality. The ontology includes several key components around it. As observed in Figure 4, the class *HealthImpactMonitoring* is central, and it relates to the *AirQualitySensor*, *GeoLocation*, *Geometry*, *Coordinates*, *AirQualityIndex*, *AgeGroup*, and *Movement* classes. Observations are uniquely identified through the *hasID* relationship, which links to unique identifiers. The *hasTimestamp* relationship associates observations with the recorded date and time. The type of sensor used, such as CO2, is specified through the *hasSensor* relationship, while the ownership status of the sensor, such as static, is indicated by the *hasOwner* relationship. The air quality reported is denoted by the *hasAirQuality* relationship. The geographic location of the observation is detailed through the *hasLocation* relationship, which includes geometric details captured by the *hasGeometry* relationship and specific coordinates specified by the *hasCoordinates* relationship. The age group of the individuals affected is specified by the *hasAgeGroup* relationship, and additional descriptions are provided through the *hasDescription* relationship. Movement related to the observation is captured by the *hasMovement* relationship.

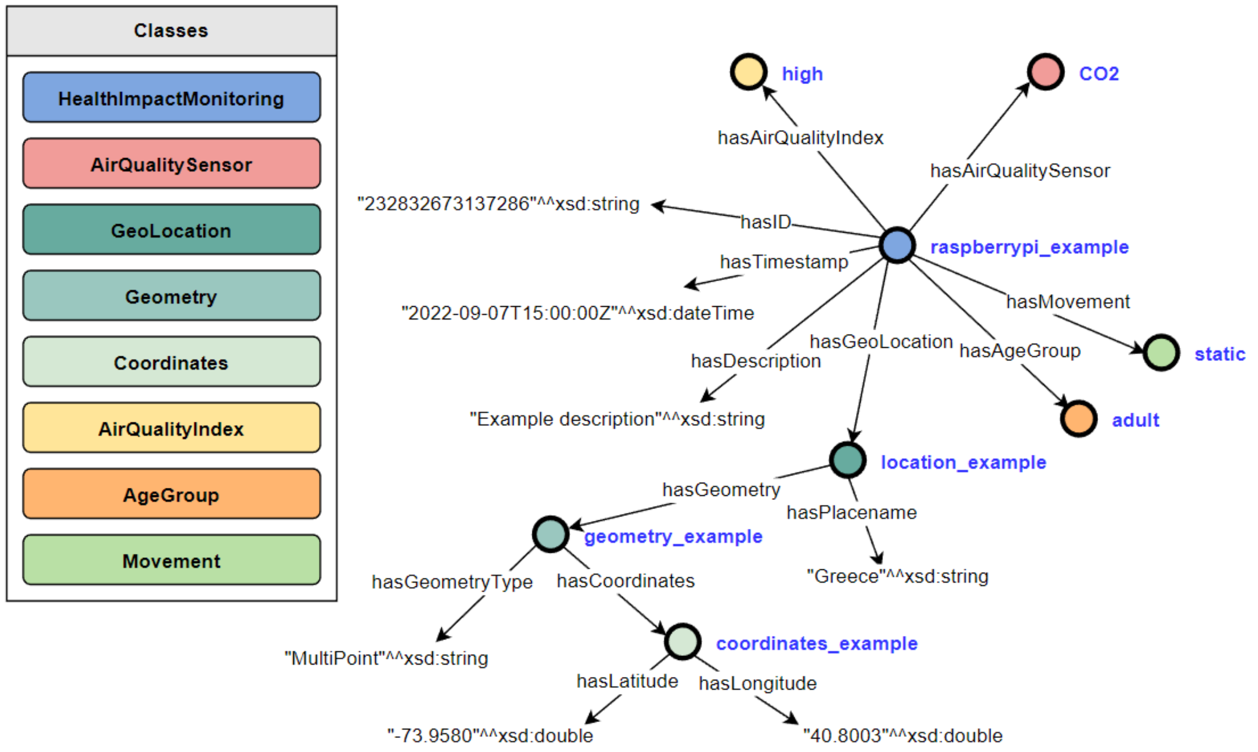


Figure 4: UP9b's example visualisation

2.3.2 UP4a Mapping

UP4a performs image-based fire/smoke detections and monitor various environmental parameters, using Raspberry Pis. The class *RaspberryPiDetection,* visualised in Figure 5, is central and connects to *RaspberryPiSensor*, *GeoLocation*, *Geometry*, *Coordinates*, *SensoryData*, *VisualSensoryData*, *Measurement*, *MeasureType*, *Unit*, and *Range* classes. Each captured event is uniquely identified through the *hasID* relationship, which links to unique identifiers. The types of sensors used, such as humidity, temperature, and camera, are specified through the *hasSensorType* relationship. The measurement range is detailed through the *hasRange* relationship. The date and time when the detection was recorded are captured through the *hasTimestamp* relationship. The geographic location of the detection is detailed through the *hasLocation* relationship, which includes geometric details captured by the *hasGeometry* relationship and specific coordinates specified by the *hasCoordinates* relationship. Sensory data, including temperature, humidity, and gas sensor readings, are detailed through the *hasSensoryData* relationship. Visual data analysis, including fire and smoke detection, is provided through the *hasVisualData* relationship.

Figure 5: UP4a's example visualisation

2.3.3 UP3 Mapping

The UP3's central class *PostEvent* connects to *Post*, *GeoLocation*, *Geometry*, *Coordinates*, *AbsoluteMediaURL*, *EventCategory*, *Concept*, *MediaFormat*, *TextualConcept*, *VisualConcept*, *Platform*, and *Language* classes as depicted in Figure 6. Each post event and individual post is uniquely identified through the *hasID* relationship. Individual posts are linked through the *hasPost* relationship. The content of the post is captured through the *hasText* relationship, and the date and time when the post was made are specified by the *hasTimestamp* relationship. Media associated with the post, including URLs, are detailed through the *hasMediaURL* relationship, with the type of media indicated by the *hasMediaType* relationship. The *isRetweet* and *isQuote* relationships indicate whether the post is a retweet or a quote, respectively. The language of the post is captured through the *hasLanguage* relationship, and the platform where the post was made is specified by the *hasPlatform* relationship. Analysis concepts derived from the post are detailed through the *hasAnalysisConcepts* relationship, and text categories are assigned through the *hasTextCategories* relationship. The type of event recognized is specified by the *hasEventType* relationship, and locations extracted from the text are detailed through the *hasExtractedLocations* relationship.

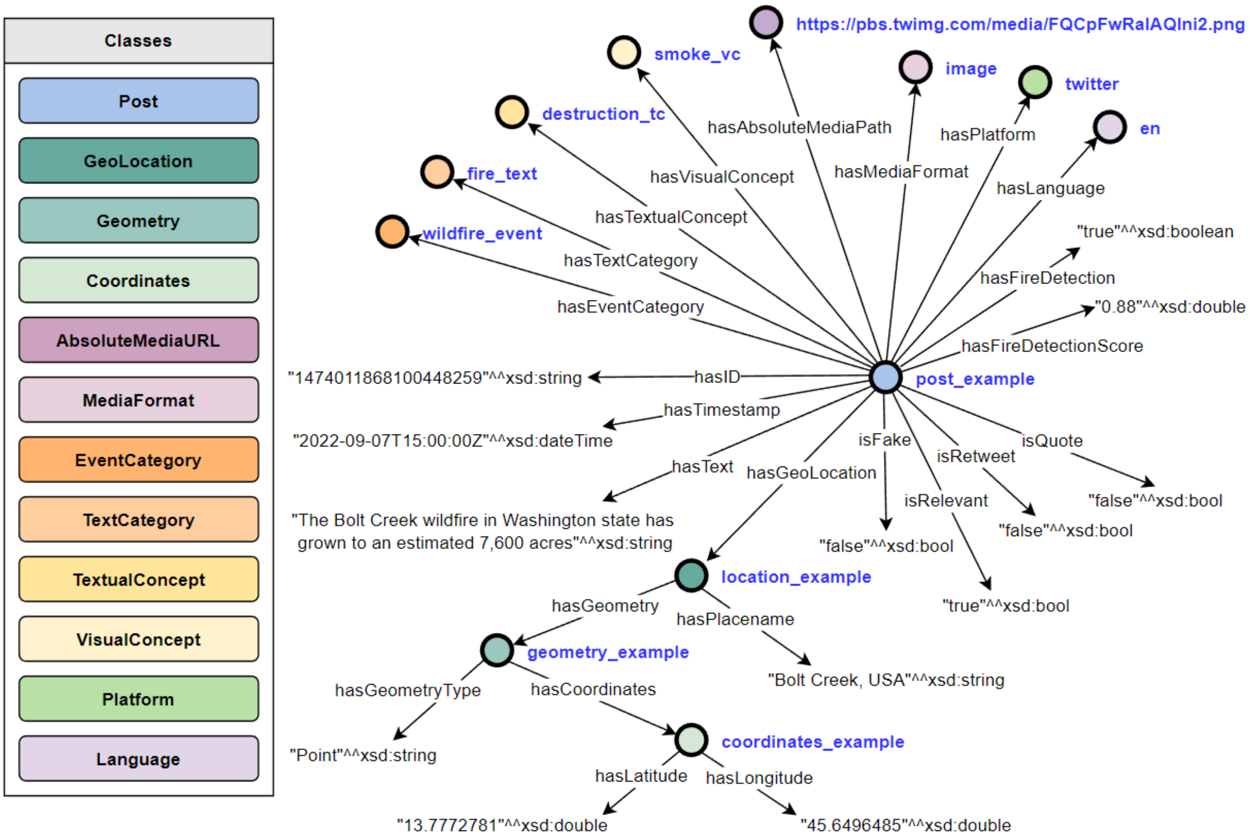


Figure 6: UP3's example visualisation

2.4 Ontology Data Population

Semantic data integration, also referred to as "semantic data fusion" or "ontology population," involves populating an initially empty semantic model with instance data derived from SILVANUS components (i.e., raw or higher-level data, described in section 2.5). This section presents the semantic data integration process adopted within SILVANUS.

The integration of diverse semantic data within the SILVANUS project into the SemKB is managed by CASPAR (Structured Data Semantic Exploitation Framework), CTL's powerful, domain-agnostic semantic data integration framework. CASPAR is being extended within SILVANUS to meet the project's data ingestion requirements. Figure 7 illustrates CASPAR's overall architecture.

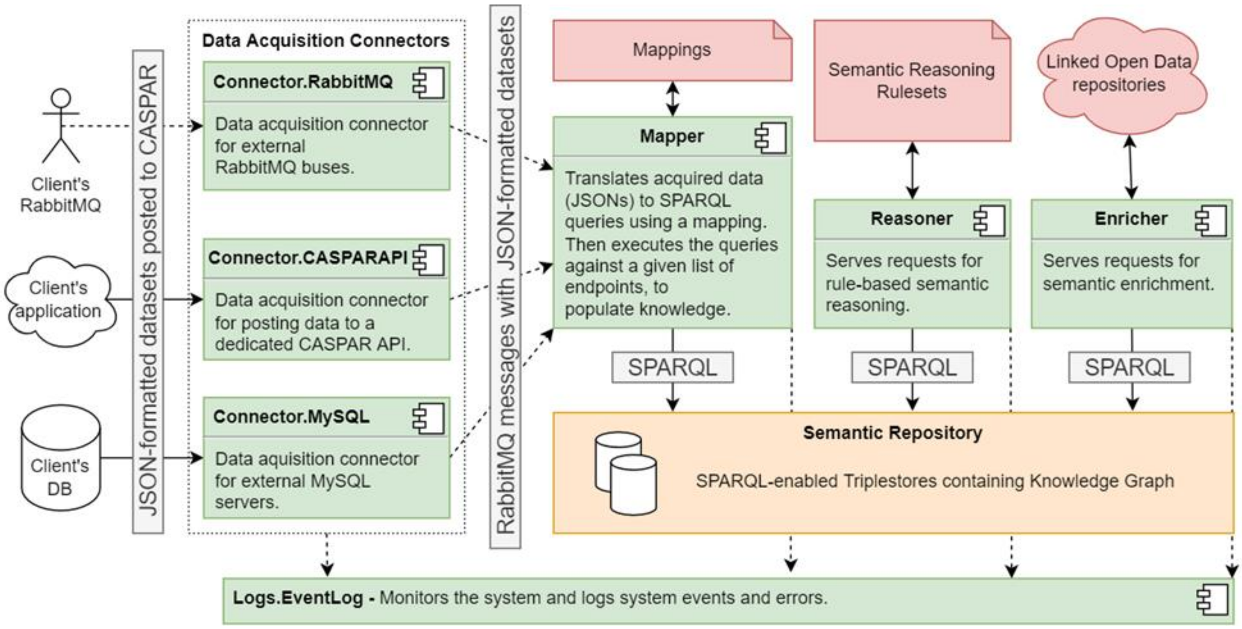


Figure 7: Block diagram portraying the CASPAR architecture

As shown in the figure, the tool deploys the following interconnected mechanisms for ingesting data into a semantic model:

- Automated acquisition of structured data from APIs, databases, and message buses.
- Mapping of input data fields to semantic entities (concepts, relationships, etc.).
- Semantic integration of knowledge into a semantic repository.
- Semantic enrichment of existing knowledge from Linked Open Data sources.
- Rule-based semantic reasoning to unveil underlying or generate new knowledge.

CASPAR defines mappings between input data fields and respective ontology concepts through a flexible methodology using a Domain-Specific Language (DSL) based on JSON syntax. The building blocks of a mapping are templates, individuals, and properties. A template focuses on specific parts of the input, allowing easier maintenance of the mapping itself. Since CASPAR can handle large pieces of input, defining several templates within a mapping that target specific parts is beneficial. A template contains a set of individuals, which declare the nodes to be created or updated in the Semantic KG. An individual, in the context of using an ontology as the schema of the KG, is an instance of one or more classes. A property indicates a desired association in the KG, connecting a node with another node or a literal value. Properties in mappings are defined by predicates, representing the relationship types of the ontology, and objects, which indicate the value to be assigned to the property.

In SILVANUS, we have defined a generic JSON structure for representing each input data, as described in section 2.5, and feeding it to CASPAR for automated integration into the KG as individuals/objects. Subsequently, data structured in this manner are provided to CASPAR's dedicated data connectors. CASPAR then maps the information to ontological entities (classes, relationships, and properties) and instantiates it into the KG by using the mapping described below, which is expressed in CASPAR's proprietary DSL syntax.

In essence, this mapping instructs CASPAR's mapper module to populate individuals representing the fire event, as described and visualized in the associated figures in section 2.3. The corresponding values are also populated as datatype and annotation properties. CASPAR can identify whether the fire event and relevant property instances already exist in the KG and update the existing individuals instead of creating new, redundant ones. This approach facilitates easier access to accumulated knowledge, such as the observation history of a fire event for specific locations. A sample instantiation of the KG based on the input of UP4a discussed in section 2.2.2 is shown below in Figure 8, coming from the triplestore CTL utilised.

Figure 8: Ontotext's triplestore visualising UP4a example

2.5 CASPAR setup

CASPAR uses RabbitMQ for asynchronous intercommunication among its components. To initialize CASPAR, a RabbitMQ instance is deployed and configured, which can be accessed through its management interface. Each CASPAR component has a RabbitMQ directory in its Python code, containing a Consumer and a Producer class, along with a RabbitMQ configuration file (JSON). The configuration file for each CASPAR component allows the user to specify the settings for the RabbitMQ instance, such as the IP address and port number which enables the component to use the selected RabbitMQ instance.

CASPAR's modular data connections, enable it to manage a wide range of structured data from multiple sources with efficiency, reflecting versatility. These connectors give clients exact control over how frequently data is posted, ensuring flexibility across a broad range of applications. Notably, by directly Mappings receiving messages from the user's RabbitMQ deployment, the RabbitMQ Connector effortlessly populates the Knowledge Graph (KG) in real-time. Figure 9 depicts how the CASPAR framework uses a Mapper module as a fundamental building block to transform external data retrieved by connectors into SPARQL queries and populate the RDF triplestore.

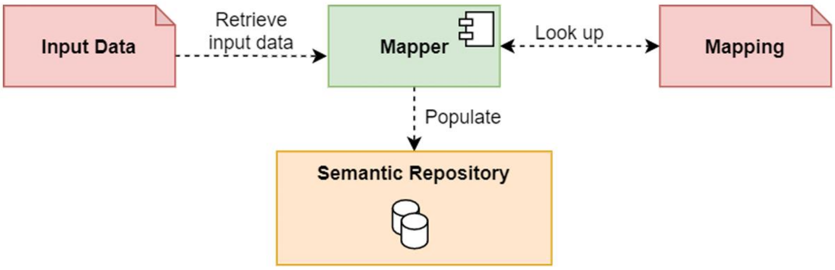


Figure 9: Mapper fuctionality

To achieve this, the Mapper requires predefined files called Mappings as described previously, which specify how data fields associate with concepts and relationships in the KG. Each Mapping can contain one or more templates, which operate independently and contain sections for prefixes and individuals. Prefixes are structural components of SPARQL searches that define relationships in the ontology. Templates contain individuals that declare nodes to be created or updated in the KG. An individual is an instance of one or more classes in the ontology and can be defined using datatype and object properties. Datatype properties are assigned literal values, whereas object properties connect individuals through attributes. Properties in mappings are established using predicates and objects, specifying the relationship type and the property's value. Below is the complete Raspberry Pi Detection mapping:

```
{
    "templates": [
        {
            "prefixes": [
                {
                    "prefix": "rdf",
                    "namespace": "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
                },
                {
                    "prefix": "rdfs",
                    "namespace": "http://www.w3.org/2000/01/rdf-schema#"
                },
                {
                    "prefix": "xsd",
                    "namespace": "http://www.w3.org/2001/XMLSchema#"
                },
                {
                    "prefix": "xml",
                    "namespace": "http://www.w3.org/XML/1998/namespace"
                },
                {
                    "prefix": "owl",
                    "namespace": "http://www.w3.org/2002/07/owl#"
                },
                {
                    "prefix": "silvanus",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#"
                }
            ],
            "individuals": [
                {
                    "path": "raspberrypi_detections.[*]",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:RaspberryPiDetection"],
                    "properties": [
                        {
```

```
                            "predicates": [
                                "silvanus:hasID",
                                "rdfs:label"
                            ],
                            "object": {
                                "path": "raspberrypi_detections.[*].uuid",
                                "datatype": "xsd:string"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasRaspberryPiSensor"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensor_type.[*]"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasRange"],
                            "object": {
                                "path": "raspberrypi_detections.[*].range"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasTimestamp"],
                            "object": {
                                "path": "raspberrypi_detections.[*].timestamp",
                                "datatype": "xsd:dateTime"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasGeoLocation"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*]"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasSensoryData"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.temperature"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasSensoryData"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.humidity"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasSensoryData"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasSensoryData"],
                            "object": {
```

```
                                "path":
"raspberrypi_detections.[*].visual_data"
                            }
                        }
                    ],
                    "update_on": [
                        {
                            "predicates": [
                                "silvanus:hasID",
                                "rdfs:label"
                            ],
                            "object": {
                                "path": "raspberrypi_detections.[*].uuid",
                                "datatype": "xsd:string"
                            }
                        }
                    ]
                },
                {
                    "path": "raspberrypi_detections.[*].sensor_type.[*]",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:RaspberryPiSensor"],
                    "properties": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensor_type.[*]",
                                "datatype": "xsd:string"
                            }
                        }
                    ],
                    "update_on": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensor_type.[*]",
                                "datatype": "xsd:string"
                            }
                        }
                    ]
                },
                {
                    "path": "raspberrypi_detections.[*].range",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:Range"],
                    "properties": [
                        {
                            "predicates": ["silvanus:hasRangeValue"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].range.value",
                                "datatype": "xsd:double"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasUnit"],
```

```
                        "object": {
                            "path": "raspberrypi_detections.[*].range.unit"
                        }
                    }
                ],
                "update_on": []
            },
            {
                "path": "raspberrypi_detections.[*].range.unit",
                "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                "classes": ["silvanus:Unit"],
                "properties": [
                    {
                        "predicates": ["rdfs:label"],
                        "object": {
                            "path":
"raspberrypi_detections.[*].range.unit",
                            "datatype": "xsd:string"
                        }
                    }
                ],
                "update_on": [
                    {
                        "predicates": ["rdfs:label"],
                        "object": {
                            "path":
"raspberrypi_detections.[*].range.unit",
                            "datatype": "xsd:string"
                        }
                    }
                ]
            },
            {
                "path":
"raspberrypi_detections.[*].sensory_data.temperature",
                "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                "classes":                            ["silvanus:Measure",
"silvanus:MeasureInDegrees",                         "silvanus:SensoryData",
"silvanus:TemperatureSensoryData"],
                "properties": [
                    {
                        "predicates":
["silvanus:hasTemperatureMeasurement"],
                        "object": {
                            "path":
"raspberrypi_detections.[*].sensory_data.temperature.value",
                            "datatype": "xsd:double"
                        }
                    },
                    {
                        "predicates": ["silvanus:hasUnit"],
                        "object": {
                            "path":
"raspberrypi_detections.[*].sensory_data.temperature.unit"
                        }
                    }
                ],
                "update_on": []
```

```
                },
                {
                    "path":
"raspberrypi_detections.[*].sensory_data.temperature.unit",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:Unit"],
                    "properties": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.temperature.unit",
                                "datatype": "xsd:string"
                            }
                        }
                    ],
                    "update_on": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.temperature.unit",
                                "datatype": "xsd:string"
                            }
                        }
                    ]
                },
                {
                    "path": "raspberrypi_detections.[*].sensory_data.humidity",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes":                              ["silvanus:Measure",
"silvanus:MeasureInPercentage",                         "silvanus:SensoryData",
"silvanus:HumiditySensoryData"],
                    "properties": [
                        {
                            "predicates": ["silvanus:hasHumidityMeasurement"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.humidity.value",
                                "datatype": "xsd:double"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasUnit"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.humidity.unit"
                            }
                        }
                    ],
                    "update_on": []
                },
                {
                    "path":
"raspberrypi_detections.[*].sensory_data.humidity.unit",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:Unit"],
```

```
                    "properties": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.humidity.unit",
                                "datatype": "xsd:string"
                            }
                        }
                    ],
                    "update_on": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.humidity.unit",
                                "datatype": "xsd:string"
                            }
                        }
                    ]
                },
                {
                    "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes":   ["silvanus:Measure",   "silvanus:SensoryData",
"silvanus:SmokeGasSensoryData"],
                    "properties": [
                        {
                            "predicates": ["silvanus:hasSmokeGasMeasurement"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.humidity.value",
                                "datatype": "xsd:double"
                            }
                        },
                        {
                            "predicates":
["silvanus:hasLiquidPetroliumGasMeasurement"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.liquid_petrolium_gas_lpg",
                                "datatype": "xsd:double"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasMethaneMeasurement"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.methane",
                                "datatype": "xsd:double"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasHydrogenMeasurement"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.hydrogen",
                                "datatype": "xsd:double"
```

```
                            }
                        },
                        {
                            "predicates": ["silvanus:hasUnit"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.unit"
                            }
                        }
                    ],
                    "update_on": []
                },
                {
                    "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.unit",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:Unit"],
                    "properties": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.unit",
                                "datatype": "xsd:string"
                            }
                        }
                    ],
                    "update_on": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].sensory_data.gas_sensor.unit",
                                "datatype": "xsd:string"
                            }
                        }
                    ]
                },
                {
                    "path": "raspberrypi_detections.[*].visual_data",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes":   ["silvanus:Measure",   "silvanus:SensoryData",
"silvanus:VisualSensoryData"],
                    "properties": [
                        {
                            "predicates": ["silvanus:hasFireDetection"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].visual_data.fire_detection.contains_fire",
                                "datatype": "xsd:boolean"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasFireDetectionScore"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].visual_data.fire_detection.fire_score",
                                "datatype": "xsd:double"
```

```
                              }
                         },
                         {
                              "predicates": ["silvanus:hasFireProbabilityIndex"],
                              "object": {
                                   "path":
"raspberrypi_detections.[*].visual_data.fire_detection.fire_probability"
                              }
                         },
                         {
                              "predicates": ["silvanus:hasSmokeDetection"],
                              "object": {
                                   "path":
"raspberrypi_detections.[*].visual_data.smoke_detection.contains_smoke",
                                   "datatype": "xsd:boolean"
                              }
                         },
                         {
                              "predicates": ["silvanus:hasSmokeDetectionScore"],
                              "object": {
                                   "path":
"raspberrypi_detections.[*].visual_data.smoke_detection.smoke_score",
                                   "datatype": "xsd:double"
                              }
                         },
                         {
                              "predicates":
["silvanus:hasSmokeProbabilityIndex"],
                              "object": {
                                   "path":
"raspberrypi_detections.[*].visual_data.smoke_detection.smoke_probability"
                              }
                         }
                    ],
                    "update_on": []
               },
               {
                    "path":
"raspberrypi_detections.[*].visual_data.fire_detection.fire_probability",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:FireProbabilityIndex"],
                    "properties": [
                         {
                              "predicates": ["rdfs:label"],
                              "object": {
                                   "path":
"raspberrypi_detections.[*].visual_data.fire_detection.fire_probability",
                                   "datatype": "xsd:string"
                              }
                         }
                    ],
                    "update_on": [
                         {
                              "predicates": ["rdfs:label"],
                              "object": {
                                   "path":
"raspberrypi_detections.[*].visual_data.fire_detection.fire_probability",
                                   "datatype": "xsd:string"
                              }
```

```
                        }
                    ]
                },
                {
                    "path":
"raspberrypi_detections.[*].visual_data.smoke_detection.smoke_probability",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:SmokeProbabilityIndex"],
                    "properties": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].visual_data.smoke_detection.smoke_probability",
                                "datatype": "xsd:string"
                            }
                        }
                    ],
                    "update_on": [
                        {
                            "predicates": ["rdfs:label"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].visual_data.smoke_detection.smoke_probability",
                                "datatype": "xsd:string"
                            }
                        }
                    ]
                },
                {
                    "path": "raspberrypi_detections.[*].location.[*]",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:GeoLocation"],
                    "properties": [
                        {
                            "predicates": ["silvanus:hasPlacename"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*].placename",
                                "datatype": "xsd:string"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasGeometry"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*].geometry"
                            }
                        }
                    ],
                    "update_on": []
                },
                {
                    "path": "raspberrypi_detections.[*].location.[*].geometry",
                    "namespace":                "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:Geometry"],
                    "properties": [
```

```
                        {
                            "predicates": ["silvanus:hasGeometryType"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*].geometry.type",
                                "datatype": "xsd:string"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasCoordinates"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*].geometry.coordinates.[*]"
                            }
                        }
                    ],
                    "update_on": []
                },
                {
                    "path":
"raspberrypi_detections.[*].location.[*].geometry.coordinates.[*]",
                    "namespace":               "https://silvanus-project.eu/wp-
content/uploads/2022/09/SILVANUS-Ontology.zip#",
                    "classes": ["silvanus:Coordinates"],
                    "properties": [
                        {
                            "predicates": ["silvanus:hasLatitude"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*].geometry.coordinates.[*].lat",
                                "datatype": "xsd:double"
                            }
                        },
                        {
                            "predicates": ["silvanus:hasLongitude"],
                            "object": {
                                "path":
"raspberrypi_detections.[*].location.[*].geometry.coordinates.[*].lon",
                                "datatype": "xsd:double"
                            }
                        }
                    ],
                    "update_on": []
                }
            ]
        }
    ]
}
```

In Figure 10, a part of the mapping is extracted to provide detailed explanations directly alongside it.

```
    ],
    "individuals": [
        {
            "path": "raspberrypi_detections.[*]", For each item found in the input data in the list of field
            "namespace": "http://www.semanticweb.org/SILVANUS_onto_demo#", Create an individual that starts with this namespace
            "classes": ["silvanus_demo:RaspberryPiDetection"], This individual should be of type RaspberryPiDetection
            "properties": [ This individual should have a set of properties in the Knowledge Graph
                {
                    "predicates": [
                        "silvanus_demo:hasID",   Should have a :hasID and a :label
                        "rdfs:label"             property
                    ],
                    "object": {
                        "path": "raspberrypi_detections.[*].id", This is the field where the value should be derived
                        "datatype": "xsd:string" The value should be noted as a string
                    }
                },
                {
                    "predicates": ["silvanus_demo:hasRaspberryPiSensor"],   Object property that connects the 'RaspberryPiDetection'
                    "object": {                                             individual to the "sensor_type" individual (which is declared
                        "path": "raspberrypi_detections.[*].sensor_type.[*]"  latter in the mapping)
                    }
                },
```

Figure 10: Sample of mapping individuals and properties (both object properties and datatype properties)

CASPAR's Mappings also include an optional update_on field for each individual, which allows for the declaration of unique property-value pairs. This enables the framework to update existing nodes in the KG instead of generating new ones, preventing the occurrence of duplicates and enabling precise knowledge extraction. The Mapper first searches for an existing node based on the unique property-value pairs, and if found, associates the new properties with it. If no node is found, the Mapper creates a new node and populates it with the new predicate-object pairs.

The Semantic Repository is a special database that stores the KG in the form of an ontology. It uses SPARQL and HTTP requests to allow users to perform queries on the KG. The repository can be hosted on the cloud or on premise, which ensures the safety and integrity of sensitive data. The Mapper can use multiple endpoints to create multiple KGs at the same time. In this work, we used Ontotext GraphDB instance deployed on CTL's premises. As one can see in Figure 11, CASPAR allows users to receive, verify, and publish data with time indicators while the process of data populating the graph happens.

```
JSONParser():  0:00:00.002007
Populator.create_triples_of_individual_populators():  0:00:00.009942
Populator.cluster_triples_to_queries():  0:00:00.008954
SPARQLQueryBatch():  0:00:00.004027
Populator():  0:00:00.026819
Total:  0:00:00.028826
Message received and validated
JSONParser():  0:00:00.001001
Populator.create_triples_of_individual_populators():  0:00:00.002959
Populator.cluster_triples_to_queries():  0:00:00.001507
SPARQLQueryBatch():  0:00:00.015375
Populator():  0:00:00.024772
Total:  0:00:00.025773
Message received and validated
JSONParser():  0:00:00.000484
Populator.create_triples_of_individual_populators():  0:00:00.000526
Populator.cluster_triples_to_queries():  0:00:00
SPARQLQueryBatch():  0:00:00.003496
Populator():  0:00:00.004284
Total:  0:00:00.004844
```

Figure 11: Sample of CASPAR populating Ontotext's GraphDB

As we have seen in this chapter, CASPAR plays a crucial role in the SILVANUS SemKB by providing a framework for knowledge management and enabling the construction of a knowledge graph. The use of SPARQL-enabled RDF triplestore and HTTP requests enables the deployment of the triplestore on the cloud or on-premises, allowing for the preservation of knowledge graphs wherever it is required.

2.6 SPARQL queries

SPARQL is critical for efficiently querying and managing the RDF-based SemKB. This allows for the smooth integration of diverse data sources required for comprehensive wildfire management, with SPARQL queries playing an important role in obtaining precise information quickly and efficiently. They enable users to perform thorough analytics and gain insights that help with both immediate actions and strategic planning in wildfire management.

The following are two examples of SPARQL queries developed for SemKB:

Query 1

Sample query: Fetch fire detections with associated humidity data

```
PREFIX : <https://silvanus-project.eu/wp-content/uploads/2022/09/SILVANUS-Ontology.zip#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?instance ?id ?time ?long ?lat ?temp ?humidity
WHERE {
  ?instance rdf:type :RaspberryPiDetection ;
          :hasID ?id ;
          :hasTimestamp ?time ;
          :hasGeoLocation/:hasGeometry/:hasCoordinates ?geolocation ;
          :hasSensoryData ?sensor.
  ?geolocation :hasLongitude ?long ;
            :hasLatitude ?lat.
  ?sensor rdf:type :TemperatureSensoryData ;
        :hasTemperatureMeasurement ?temp ;
        :hasHumidityMeasurement ?humidity.
}
```

Query 2

Sample query: Aggregate temperature readings by location

```
PREFIX : <https://silvanus-project.eu/wp-content/uploads/2022/09/SILVANUS-Ontology.zip#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?long ?lat (AVG(?temp) AS ?avgTemp)
WHERE {
  ?instance rdf:type :RaspberryPiDetection ;
          :hasGeoLocation/:hasGeometry/:hasCoordinates ?geolocation ;
          :hasSensoryData ?sensor.
  ?geolocation :hasLongitude ?long ;
            :hasLatitude ?lat.
  ?sensor rdf:type :TemperatureSensoryData ;
        :hasTemperatureMeasurement ?temp.
}
GROUP BY ?long ?lat
```

2.7 Communication via SAL

2.7.1 SemKB – SAL connection description

The SemKB is hosted on a virtual machine (VM) at CTL's premises and integrates three data sources as outlined in section 2.2, through the Storage Abstraction Layer (SAL). SAL serves as a centralized repository within the SILVANUS framework, receiving data from all SILVANUS components. For secure communication with SAL, it is essential to use the private SILVANUS VPN; without it, accessing or transmitting data is not possible. Detailed descriptions of the architecture, communications, and data inputs are documented in D5.1. Additionally, D8.3 provides updates on the architecture. The SemKB actively monitors multiple RabbitMQ queues, with one queue dedicated to each data source. When these queues receive new data, the system consumes it and processes it through to SemKB, as illustrated in Figure 1. The queues attached to SemKB are replicated versions of those used by the UP components mentioned earlier. This replication ensures that the original queues remain available for other components interacting with data from UP4a, UP3, and UP9b, without leaving them empty.

The implementation is written in Python and is similar for each of the three UP components. It operates as a service; whenever it detects new data in any of the queues, it consumes that data and updates the knowledge graph accordingly. Since the process is the same for all three data sources, the description below focuses only on UP4a, including detailed explanations, portions of the code, and illustrative figures.

2.7.2 SemKB – UP4a connection description

To enable seamless integration between the UP4a component powered by CTL and the CASPAR framework, a Python script has been developed as mentioned earlier. This script facilitates the following:

- Listening for Sensor Data: It connects to a local RabbitMQ server to receive data from Raspberry Pis (IoTs) deployed in the field.
- Data Transformation: It transforms the incoming data into a structured format that is compatible with the CASPAR framework's requirements.
- Data Transmission: It publishes the transformed data to the CASPAR framework's RabbitMQ server for further processing and analysis.

2.7.2.1 Overview of the Integration Script

The script performs three main functions:

1. Establishing Connections:
    - Local RabbitMQ Server: Connects to the local message broker where sensor data is queued.
    - CASPAR RabbitMQ Server: Establishes a connection to the CASPAR framework's message broker to send transformed data.
2. Processing and Transforming Data:
    - Message Handling: Listens for messages containing sensor data and handles them via a callback function.
    - Data Transformation: Wraps the received data into a JSON structure that includes necessary headers and formatting.
3. Publishing Transformed Data:
    - Message Publishing: Sends the transformed data to the CASPAR framework by publishing it to a specified exchange and routing key.

- Acknowledgment: Acknowledges the message to confirm successful processing.

2.7.2.2 Key Components of the Script

Note: For security reasons, actual credentials and sensitive information have been omitted. In practice, credentials should be securely stored and retrieved using environment variables or configuration files.

1. Dependencies:
   - Python Packages: The script utilizes the pika library for RabbitMQ communication and json for data handling.
   - Installation: Required packages can be installed using pip install pika.
2. Connection Parameters:
   - Local Connection: Uses credentials and connection parameters to connect to the local RabbitMQ server.
   - CASPAR Connection: Sets up connection details to the CASPAR framework's RabbitMQ server.
3. Data Transformation Function:
   - Purpose: Converts incoming sensor data into the format expected by the CASPAR framework.
   - Structure: Encapsulates data within a header and body, adding any required metadata.
4. Callback Function:
   - Message Processing: Triggered when a new message is received; it decodes and processes the message.
   - Metadata Handling: Extracts unique identifiers and handles any additional metadata.
   - Data Transmission: Calls the function to send transformed data to the CASPAR framework.
   - Acknowledgment: Sends an acknowledgment to the message broker upon successful processing.
5. Main Execution Block:
   - Consumer Setup: Initializes the connection to the local RabbitMQ server and starts consuming messages from the specified queue.
   - Graceful Shutdown: Handles interruptions and ensures connections are properly closed.

2.7.2.3 Workflow Description

1. Initialization:
   - The script imports necessary libraries and sets up connection parameters for both local and CASPAR RabbitMQ servers.
2. Listening for Data:
   - Connects to the local RabbitMQ server.
   - Waits for incoming messages on the designated queue, which receives data from the IoT devices (Raspberry Pis).
3. Processing Messages:
   - Upon receiving a message, the callback function is invoked.
   - The message is decoded from bytes to a string and then parsed as JSON.
   - Extracts relevant information such as uuid and any metadata.
4. Transforming Data:
   - The data is wrapped into a new JSON structure with appropriate headers.
   - Ensures the data format aligns with what the CASPAR framework expects.
5. Sending Data to CASPAR:
   - Establishes a connection to the CASPAR RabbitMQ server.
   - Publishes the transformed data to the specified exchange and routing key.

- Closes the connection after successful publishing.

6. Acknowledging Messages:
    - The script sends an acknowledgment to the local RabbitMQ server to confirm that the message has been processed.

In Figure 12 can be shown a part of this script described above

```
# Create a connection and start consuming messages
try:
    connection = pika.BlockingConnection(parameters)
    channel = connection.channel()

    queue_name = 'ctl.iotkb'

    print(f"Waiting for messages in {queue_name}. To exit, press CTRL+C")

    channel.basic_consume(queue=queue_name, on_message_callback=callback)
    channel.start_consuming()

except KeyboardInterrupt:
    print("Stopped by user")
    channel.stop_consuming()
    connection.close()
```

Figure 12: Part of the script for initiating RabbitMQ message consumption

3 Integrated Data Insights

The SILVANUS project employs an Integrated Data Insights user product under the Decision Support System (DSS) named UP9h, designed to leverage the power of a SemKB for wildfire management. UP9h enhances the capabilities of real-time monitoring and decision-making. This section details the architecture and functionality of this UP, emphasizing its role in improving wildfire management through timely, accurate data analysis and making the knowledge clearer. Key advancements in the SemKB have been discussed in the recent paper by Simone Martin Marotta, Vincenzo Masucci, Stelios Kontogiannis, and Konstantinos Avgerinakis, presented as "From Unified Ontology to Knowledge Base: Data Fusion for Enhanced Wildfire Management" at the 8th World Conference on Smart Trends in System, Security, and Sustainability (WorldS4 2024) [1]. Additionally, insights into the integration of data for enhanced decision support have been further elaborated in "Ontology Data Insights and Alerts for Wildfire Protection" by Marios Iacovou, Stelios Kontogiannis, and Konstantinos Avgerinakis, featured at the 13th EETN Conference on Artificial Intelligence (SETN 2024) [2]. These contributions underscore the pivotal role of UP9h in integrating data and producing insights critical for effective wildfire management. This section will further explore the advanced alerting mechanisms that enable proactive management responses to emerging wildfire threats.

3.1 Alert-Based System

The idea for creating this system is to utilize predefined rules and real-time data to generate alerts of varying severity levels. At the present status, the system continuously monitors data from two primary sources: UP9b and UP4a, as described in section 2.

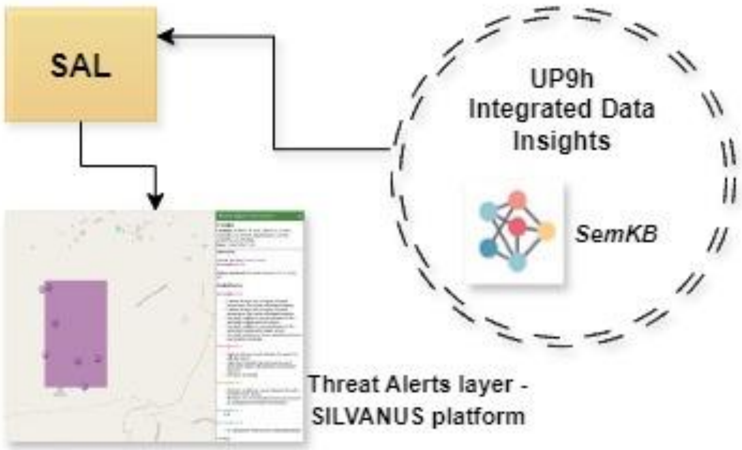


Figure 13: Overview of the SILVANUS platform integration with UP9h Integrated Data Insights through SemKB, SAL and Threat Alerts layer

The alert generation framework is designed to provide holistic real-time monitoring and alerts for potential wildfire threats. The framework, an overview of the general architecture seen in Figure 13, continuously analyzes integrated data residing in the SemKB to detect conditions indicative of potential wildfire threats by applying predefined rules to assess various environmental parameters. The alert system categorizes the severity of detected conditions into multiple levels, facilitating appropriate responses to emerging fire risks. For instance, UP4a's sensors gather data on temperature, humidity, gas concentrations (such as hydrogen, LPG, smoke, and methane), and visual indicators of fire and smoke. Additional information on air quality, such as CO_2, CO, NO_2, PM10, and PM2.5 levels, is provided by UP9b's monitoring sensors. UP9h utilizes a combination of data querying technologies, including SPARQL, to retrieve relevant information from the SemKB and generate alerts based on the severity of detected conditions.

3.1.1 Benefits

The integration of diverse data sources into the SemKB and the alert system under the umbrella of UP9h, offers several key benefits for wildfire management:

- **Enhanced Situational Awareness**: By aggregating data from multiple sources, the system provides a comprehensive view of the wildfire environment. This holistic perspective allows for better understanding and anticipation of wildfire behavior.
- **Improved Accuracy and Timeliness**: The use of real-time data and advanced semantic reasoning techniques ensures that alerts are generated promptly and accurately. This timely information is crucial for initiating rapid response measures and mitigating potential damage.
- **Automated Decision Support**: The alert system utilizes predefined rules and severity levels to provide automated decision support, significantly reducing the need for manual analysis. This automation enhances the efficiency and effectiveness of wildfire management operations. Users receive a comprehensive overview of the situation initially and can then delve into additional layers for more detailed information.
- **Scalability and Flexibility**: The system's modular design facilitates the seamless integration of additional data sources and the adaptation of rules to address evolving wildfire risks. This flexibility ensures the system remains relevant and effective as new technologies and data streams become available. For instance, in the final year of the SILVANUS project, the plan is to enhance UP9h by incorporating alerts related to UP3, utilizing social media data such as tweets.

3.1.2 Data retrieval and rule application

As an initial step, data is retrieved from the SemKB, focusing on key parameters such as temperature, humidity, gas levels, and visual detection of fire and smoke. Next, a series of predefined rules are applied to the retrieved data. The remaining rules utilize either fire/smoke detection data, health impact data, or a combination of both. Each rule targets specific conditions that could indicate the presence or potential outbreak of wildfires, ensuring comprehensive analysis and timely alerts to enhance wildfire management and response.
Current approach has 13 established rules, detailed below:

- Rule 1: *Dry Conditions* - Temperature and humidity evaluation to assess fire risk due to dry conditions.
- Rule 2: *Higher-Risk Areas* - Comparison of temperature and historical data for fire risk areas.
- Rule 3: *Fire Detection* - Utilization of visual fire detection data.
- Rule 4: *Smoke Detection* - Utilization of visual smoke detection data.
- Rule 5-8: *Combined Conditions* – Leverage combinations of temperature, fire detection scores, smoke detection scores, and gas levels to assess risk of fire ignition.
- Rule 9-12: *Air Quality and Gas Level*s – Evaluation of air quality metrics to identify the presence of combustible gases.
- Rule 13: *Overall Air Quality* – Takes into consideration multiple pollutants to determine poor air quality conditions.

3.1.3 Severity levels and assessment

The severity levels range from 1 to 4, with 4 indicating the highest level of risk.

- Level 1: Represents the lowest severity, indicating conditions that are concerning but not immediately dangerous.
- Level 2: Indicates moderate risk, requiring attention and potential action.
- Level 3: Signifies high risk, where conditions are likely to lead to significant issues.
- Level 4: Represents the highest severity, indicating critical conditions that require immediate and decisive action.

These severity levels inform the urgency and type of response required. An example of a rule generating a severity level given certain conditions is illustrated in Figure 14, where an appropriate severity level is returned according to a visual smoke detection score and particulate matter scores.

Rule 11: Elevated PM2.5/PM10 and visual indication for potential smoldering

```
Data: smoke_detection_score, PM25_air_quality, PM10_air_quality
Result: Severity level based on smoke detection and air quality
if smoke_detection_score ≥ 25 then
    if PM25_air_quality = "extremely poor" or PM10_air_quality =
      "extremely poor" then
        return 4
    else if PM25_air_quality = "very poor" or PM10_air_quality =
      "very poor" then
        return 3
    else if PM25_air_quality = "poor" or PM10_air_quality = "poor"
      then
        return 2
    else if PM25_air_quality = "moderate" or PM10_air_quality =
      "moderate" then
        return 1
return 0
```

Figure 14: Rule for Assessing Potential Smoldering Based on PM2.5/PM10 Levels and Smoke Detection Score

3.2 Case Studies and Example Scenarios

To demonstrate the capabilities of the UP9h system, two key scenarios have been developed: active fire detection and gas leakage detection. These scenarios showcase the system's ability to monitor environmental conditions continuously, apply predefined safety protocols, and issue alerts based on the severity of the detected conditions. The data used are synthetic, designed to emulate real-world conditions accurately and demonstrate the system's effectiveness in a controlled environment. This preliminary testing is crucial to ensure that the UP9h is fully operational and effective before its deployment in the pilots scheduled for SILVANUS last year.

3.2.1 Scenario 1: Active Fire Detection

In this scenario, the system monitors data from a device named "Greece-Device1," simulating conditions indicative of an active fire. The system evaluates environmental parameters such as temperature, humidity, and visual indicators of fire and smoke at 1-minute intervals. Based on the predefined rules, the system generates alerts and assesses their severity, escalating the severity levels as conditions worsen. This scenario demonstrates the system's capability to detect early signs of fire and provide timely alerts to stakeholders.

Setup: Synthetic data simulates conditions indicative of an active fire, including elevated temperature, humidity, and visual indicators of fire and smoke.
Process: The system collects data from "Greece-Device1" and applies the 13 rules to assess various environmental parameters. Alerts are generated based on the severity of detected conditions.
Time Increments: Data is collected and evaluated at 1-minute intervals, demonstrating the system's ability to monitor changes in real-time.

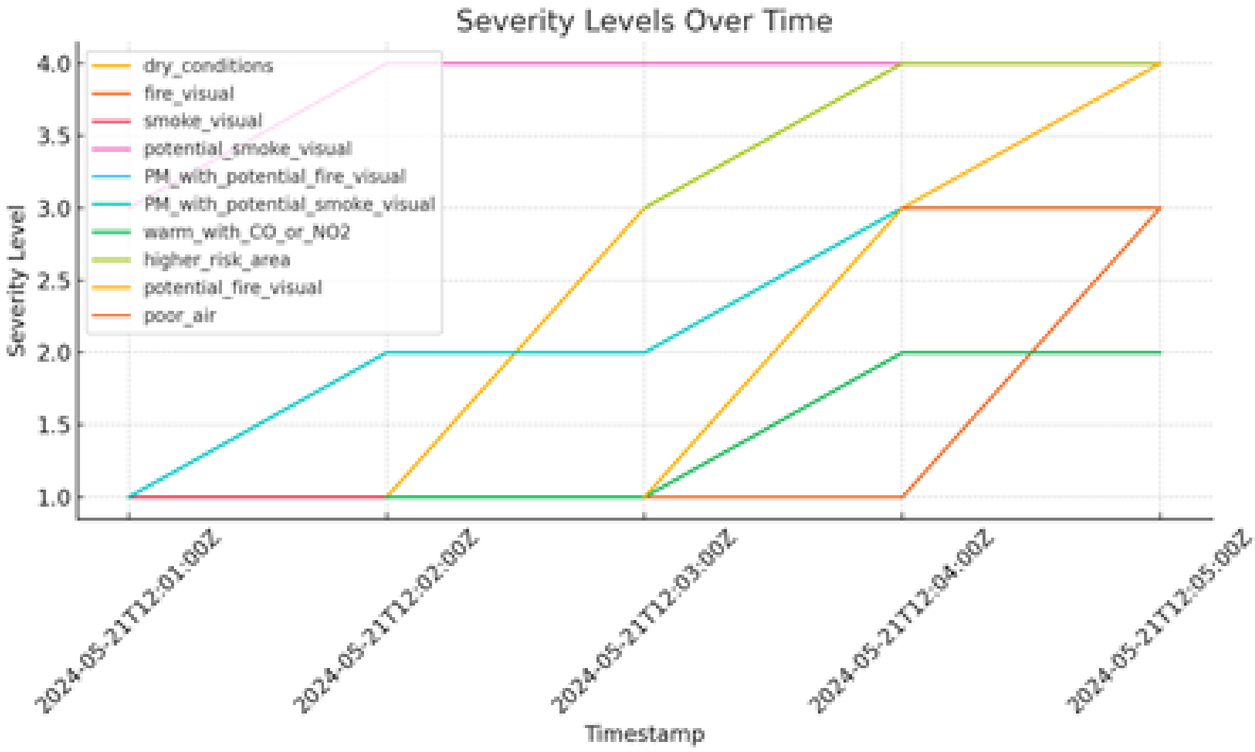


Figure 15: Severity levels over time for different alerts – indicating fire

Figure 15 illustrates the severity levels of various alerts over a series of timestamps, collected every minute from the device "Greece-Device1". Each line represents a specific condition, with severity levels ranging from 1 to 4. The conditions monitored include dry conditions, fire visual, smoke visual, potential smoke visual, PM with potential fire visual, PM with potential smoke visual, higher risk area, warm with CO or NO2, and poor air quality.

Results and Explanation: As the monitoring continues, the severity of these alerts escalates, reflecting worsening conditions and an increased risk of fire. Initially, in this scenario, dry conditions are detected with a severity level of 1, indicating mild dryness. As the data is continually assessed, the severity escalates to level 4, signifying extreme dryness, which poses a higher risk of wildfire. This progressive escalation in severity illustrates the system's capability to dynamically adjust the risk level based on real-time data. Fire visual indicators are consistently detected throughout the monitoring period, with their severity increasing over time. This consistent detection and escalation demonstrate the presence of visual signs of fire, which become more severe as conditions worsen. Similarly, smoke visual indicators follow the same pattern, starting at a lower severity and escalating as the conditions become more conducive to fire. The potential smoke visual indicators show an increasing severity, indicating a heightened risk of smoldering, which is a precursor to active fire. This highlights the system's ability to detect early signs of fire and escalate the alert levels accordingly.

PM readings combined with potential fire and smoke visual indicators show how environmental metrics correlate with fire risk. As the PM levels increase, they coincide with the detection of visual signs of fire and smoke, further validating the system's assessments. The detection of higher risk areas indicates that the device is located in a historically high-risk zone, which contributes to the overall severity of the alerts. This historical data is crucial for understanding the context of the detected conditions and their potential implications. Additional risks are considered through the metrics of warm conditions combined with elevated levels of CO or NO2, and poor air quality. These factors add another layer of complexity to the assessment, ensuring that all relevant environmental conditions are considered.

Overall, the results demonstrate the system's effectiveness in continuously monitoring and assessing environmental data to provide timely and accurate alerts in an automated manner, eliminating the need for human intervention to analyze and combine disparate data sources. The dynamic escalation of severity levels ensures that stakeholders are promptly informed about the increasing risks, enabling them to take appropriate actions to mitigate potential wildfire threats.

3.2.2 Scenario 2: Gas Leakage Detection

This scenario focuses on detecting gas leakage conditions using the same alert system and methodology. The system monitors data from "Greece-Device1," applying the predefined rules every minute to assess gas-related parameters. Synthetic data simulates conditions indicative of gas leaks, including elevated levels of various gases and their combinations with other environmental factors. The system's ability to dynamically escalate alerts based on the severity of conditions is demonstrated, providing stakeholders with critical real-time information about potential gas leak risks.

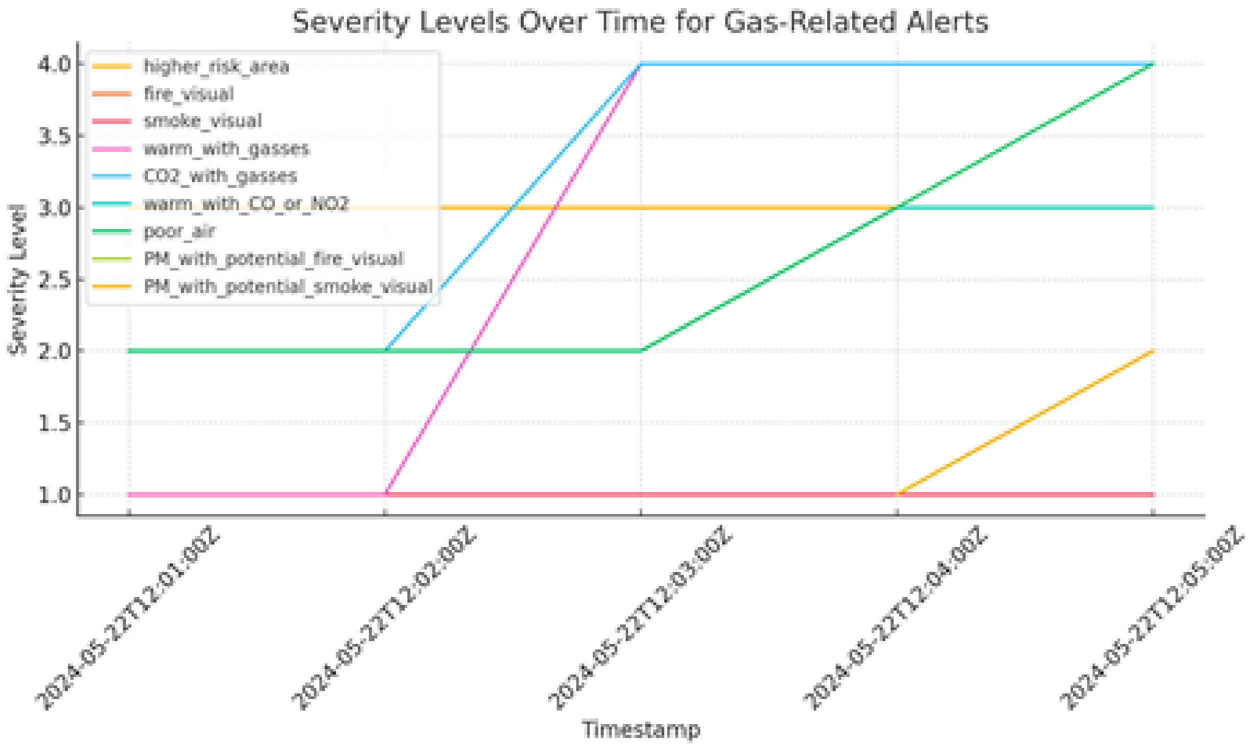


Figure 16: Severity levels over time for gas-related alerts

The results of the gas leakage detection scenario are shown in Figure 16. This figure displays the severity levels of various gas-related alerts over time, collected every minute from the device "Greece-Device1". The monitored conditions are as follows: Higher risk area, Fire visual, Smoke visual, Warm with gases, CO2 with gases, Warm with CO or NO2, Poor air quality. As time progresses, the severity levels of these alerts increase, indicating worsening conditions and heightened risk.

Figure 16 demonstrates the system's ability to detect and escalate alerts based on the severity of conditions. The severity levels evolve as follows: Initially, the device identifies that it is situated in a historically high-risk area, consistently detected at a severity level of 3. This ongoing detection underscores the device's location in a zone that has a history of heightened risk. Visual indications of fire and smoke are both identified, starting at a severity level of 1. These initial detections suggest early visual signs of fire and smoke presence. The presence of warm conditions combined with gases starts at a severity level of 1 and escalates to 4. This increase reflects a growing concentration of combustible gases coupled with rising temperatures, indicating a significant risk. CO2 levels combined with other gases show a severity increase from 2 to 4, indicating elevated levels of CO2 and other gases, which signal an escalating risk. Conditions of warmth combined with CO or NO2 increase from a severity level of 2 to 3, showing a rising combination of

elevated temperatures and hazardous gas levels. Finally, poor air quality conditions rise from severity level 2 to 4. This escalation demonstrates deteriorating air quality, which poses significant health risks to the surrounding area.
In summary, Figure 16 highlights the framework's capability to monitor and escalate alerts dynamically, providing stakeholders with critical, real-time information about the evolving environmental conditions and associated risks.

3.3 Integration with SAL and SILVANUS Dashboard

The integration of UP9h with the SAL facilitates the seamless transmission of alert data and metadata to SAL, following the procedures used also by the rest of the UPs, as outlined in D5.2 and also described in section 2.7. This ensures that the Threat Alerts layer dashboard is accurately populated by pushing data to the designated queue. Each alert data and metadata file is uniquely identified, containing comprehensive information on wildfire risks. The credentials required for pushing data to the queue include:

- amqp$vhost: The virtual host for the message queue, 'ctl'.
- amqp$exchange: The exchange name for routing messages, 'ctl.ex'.
- amqp$routingKey: The routing key to direct messages to the appropriate queue, 'ctl.up9idi'.

A new UUID is generated for each alert to ensure uniqueness. The metadata is then updated with the new UUID and expiry period. The prepared data and metadata are sent to the SAL queue using a POST request. The SAL processes the incoming data, and the SILVANUS dashboard, which monitors this queue, updates the Threat Alerts layer accordingly.

This integration allows stakeholders to view the latest alerts and detailed analyses, enabling informed decision-making. Below is shown an example of UP9h's data push to 'ctl.up9idi':

Example of data ingested to SAL from UP9h:


```
{
    "uuid": "0190cb98-8269-74e7-aedd-cf7a5eb541ab",
    "timestamp": "2024-09-24T10:00:12.651893Z",
    "index_range": 4,
    "overall_warning": {
        "index": 3,
        "label": "Danger",
        "color": "#ff5050"
    },
    "dry_conditions": {
        "index": 2,
        "label": "Extreme Caution",
        "color": "#e69138",
        "message": "Moderate drought due to elevated temperature (41.34C) and
decreased humidity (38.12%)"
    },
    "higher_risk_area": {
        "index": 3,
        "label": "Danger",
        "color": "#ff5050",
        "message": "Elevated temperature (41.34C) in category 3 higher-risk area"
    },
    "fire_visual": {
        "index": 0,
        "label": "No Warning",
        "color": "#50ccaa",
        "message": "No visual indication for active fire with fire sensor"
    },
    "smoke_visual": {
        "index": 0,
```

```
        "label": "No Warning",
        "color": "#50ccaa",
        "message": "No visual indication for active fire with smoke sensor"
    },
    "potential_fire_visual": {
        "index": 1,
        "label": "Caution",
        "color": "#76a5af",
        "message": "Mild confidence visual indication for fire potentially starting with fire sensor (36%, temperature 41.34C)"
    },
    "potential_smoke_visual": {
        "index": 1,
        "label": "Caution",
        "color": "#76a5af",
        "message": "Mild confidence visual indication for fire potentially starting with smoke sensor (41%, temperature 41.34C)"
    },
    "smoke_spread": {
        "index": 3,
        "label": "Danger",
        "color": "#ff5050",
        "message": "High indication of smoke spread detection (41%) in low humidity (38.12%) conditions"
    },
    "warm_with_gases": {
        "index": 0,
        "label": "No Warning",
        "color": "#50ccaa",
        "message": "No combustion warning from combustible gases"
    },
    "PM_with_potential_fire_visual": {
        "index": 2,
        "label": "Extreme Caution",
        "color": "#e69138",
        "message": "Moderate risk of combustion due to presence of particulate matter (PM2.5 36 micrograms pcm, PM10 55 micrograms pcm) with potential visual fire detection (36%)"
    },
    "CO_with_gases": {
        "index": 0,
        "label": "No Warning",
        "color": "#50ccaa",
        "message": "No combustion warning from combustible gases with CO"
    },
    "PM_with_potential_smoke_visual": {
        "index": 2,
        "label": "Extreme Caution",
        "color": "#e69138",
        "message": "Moderate risk of combustion due to presence of particulate matter (PM2.5 36 micrograms pcm, PM10 55 micrograms pcm) with potential visual smoke detection (41%)"
    },
    "warm_with_CO_or_NO2": {
        "index": 1,
        "label": "Caution",
        "color": "#76a5af",
        "message": "Mild risk of combustion due to presence of air pollutants (CO 3 micrograms pcm, NO2 79 micrograms pcm) and elevated temperature (41.34C)"
    },
```

```
    "poor_air": {
        "index": 2,
        "label": "Extreme Caution",
        "color": "#e69138",
        "message": "Poor air quality (PM2.5 36 micrograms pcm, PM10 55 micrograms
pcm, CO 3 micrograms pcm, NO2 79 micrograms pcm)"
    },
    "location": [
        {
            "placename": "Italy",
            "bbox": [
                15.9384146,
                41.8839116,
                15.9450696,
                41.8871981
            ],
            "geometry": {
                "type": "Polygon",
                "coordinates": [
                    [
                        [
                            15.9384146,
                            41.8839116
                        ],
                        [
                            15.9450696,
                            41.8839116
                        ],
                        [
                            15.9450696,
                            41.8871981
                        ],
                        [
                            15.9384146,
                            41.8871981
                        ],
                        [
                            15.9384146,
                            41.8839116
                        ]
                    ]
                ]
            }
        }
    ],
    "devices": {
        "fire_smoke_detection_IoT": [
            "IoT_1",
            "IoT_2",
            "IoT_3",
            "IoT_4"
        ],
        "air_quality_IoT": [
            "Weather Station 1",
            "Weather Station 2",
            "Weather Station 3"
        ]
    }
}
```


Explanation of the above JSON

- uuid: A unique identifier for the alert, ensuring each alert is distinct and can be traced individually.
- timestamp: The exact date and time when the alert was generated, allowing for precise tracking and chronological ordering of events.
- index_range: Indicates the maximum possible value for the warning level.

Overall Warning

- overall_warning: Provides a general warning that summarizes the situation:
- index: A numerical value indicating the most severe identified warning.
- label: A descriptive term (e.g., "Danger") that conveys the severity level.
- color: A color code that visually represents the severity level (e.g., red for high danger).

Detailed Analysis

dry_conditions: Details about the drought conditions present in the alert area:

- index: Severity level of the drought.
- label: Descriptive term indicating the severity (e.g., "Extreme Caution").
- color: Associated color code (e.g., orange for caution).
- message: Provides specific information about the drought conditions, such as elevated temperature and decreased humidity.

higher_risk_area: Information about regions within the alert area that are at higher risk due to elevated temperatures:

- index: Severity level for the high-risk area.
- label: Descriptive term indicating the severity (e.g., "Danger").
- color: Associated color code (e.g., red).
- message: Describes the specific risk, such as high temperatures in a category 3 high-risk area.

Visual Detection Status

fire_visual: Status of fire detection through visual means:

- index: Severity level for visual fire detection.
- label: Descriptive term indicating the severity (e.g., "No Warning").
- color: Associated color code (e.g., green for no warning).
- message: Explanation indicating whether any visual signs of fire were detected.

smoke_visual: Status of smoke detection through visual means:

- index: Severity level for visual smoke detection.
- label: Descriptive term indicating the severity (e.g., "No Warning").
- color: Associated color code (e.g., green for no warning).
- message: Explanation indicating whether any visual signs of smoke were detected.

potential_fire_visual: Potential risk of fire based on visual indicators:

- index: Severity level for potential fire detection.
- label: Descriptive term indicating the severity (e.g., "Caution").
- color: Associated color code (e.g., blue for caution).
- message: Provides details on the potential risk, such as mild confidence visual indications and specific temperature readings.

potential_smoke_visual: Potential risk of smoke based on visual indicators:

- index: Severity level for potential smoke detection.
- label: Descriptive term indicating the severity (e.g., "Caution").
- color: Associated color code (e.g., blue for caution).
- message: Provides details on the potential risk, such as mild confidence visual indications and specific temperature readings.

Risk Detection

smoke_spread: Detection of smoke spread in the area:

- index: Severity level for smoke spread detection.
- label: Descriptive term indicating the severity (e.g., "Danger").
- color: Associated color code (e.g., red).
- message: Explanation of the smoke spread risk, such as high indication of smoke spread detection in low humidity conditions.

warm_with_gases: Combustion risk based on the presence of gases:

- index: Severity level for gas-related combustion risk.
- label: Descriptive term indicating the severity (e.g., "No Warning").
- color: Associated color code (e.g., green for no warning).
- message: Provides details about the presence or absence of combustible gases.

PM_with_potential_fire_visual: Combustion risk from particulate matter combined with visual fire detection:

- index: Severity level for this combined risk.
- label: Descriptive term indicating the severity (e.g., "Extreme Caution").
- color: Associated color code (e.g., orange for extreme caution).
- message: Explanation of the combined risk, such as moderate risk due to particulate matter and potential visual fire detection.

CO_with_gases: Combustion risk from gases combined with CO presence:

- index: Severity level for this combined risk.
- label: Descriptive term indicating the severity (e.g., "No Warning").
- color: Associated color code (e.g., green for no warning).
- message: Explanation of the combined risk, such as no combustion warning from combustible gases with CO.

PM_with_potential_smoke_visual: Combustion risk from particulate matter combined with visual smoke detection:

- index: Severity level for this combined risk.
- label: Descriptive term indicating the severity (e.g., "Extreme Caution").
- color: Associated color code (e.g., orange for extreme caution).
- message: Explanation of the combined risk, such as moderate risk due to particulate matter and potential visual smoke detection.

warm_with_CO_or_NO2: Combustion risk from elevated temperatures and pollutants:

- index: Severity level for this combined risk.
- label: Descriptive term indicating the severity (e.g., "Caution").
- color: Associated color code (e.g., blue for caution).
- message: Explanation of the combined risk, such as mild risk due to presence of air pollutants and elevated temperature.

poor_air: General air quality conditions:

- index: Severity level for air quality.
- label: Descriptive term indicating the severity (e.g., "Extreme Caution").
- color: Associated color code (e.g., orange for extreme caution).
- message: Explanation of the air quality conditions, such as poor air quality due to various pollutants.

Location and Devices

location: Geographical details of the alert:

- placename: The name of the place.
- bbox: Bounding box coordinates defining the area.
- geometry: Polygon coordinates that outline the specific area affected.

devices: List of IoT devices involved in detection:

- fire_smoke_detection_IoT: Devices used for detecting fire and smoke.
- air_quality_IoT: Devices used for monitoring air quality.

Example of metadata file ingested to SAL from UP9h:


```
{
    "descriptor": {
        "uuid": "0190cb98-8269-74e7-aedd-cf7a5eb541ab",
        "obj-class": "SemKB",
        "format": {
            "type": "json"
        },
        "access": "default",
        "dataset-type": "IDI",
        "created": "1727172012.651893",
        "expiry-period": "1721834688.0"
    },
    "lineage": {
        "source": [
            "2f7c0113-3e78-413e-8702-1e9948afd2e4",
            "90fe14a9-d1c5-4ae3-ac04-e68d72c3dfe4",
            "c8a708c5-62fd-4816-aba1-a07f26741cf9",
            "1228b345-39bd-48a3-bc8c-34c5d22e6997",
            "a782f349-dd46-4694-96dd-7480cca799fe",
            "7c62ea99-9d26-41c7-b398-96f5b42d9129",
            "332e828a-5540-4735-9b35-037ba23dfab0"
        ],
        "owner": "CTL",
        "processing": "UP9h"
    },
    "spatial": {
        "type": "Polygon",
```

```
        "coordinates": [
            {
                "lon": 15.9384146,
                "lat": 41.8839116
            },
            {
                "lon": 15.9450696,
                "lat": 41.8839116
            },
            {
                "lon": 15.9450696,
                "lat": 41.8871981
            },
            {
                "lon": 15.9384146,
                "lat": 41.8871981
            },
            {
                "lon": 15.9384146,
                "lat": 41.8839116
            }
        ],
        "wkt":  "POLYGON  ((15.9384146  41.8839116,  15.9450696  41.8839116,
15.9450696 41.8871981, 15.9384146 41.8871981, 15.9384146 41.8839116))",
        "pilot": "gargano"
    },
    "temporal": {
        "datetime": "1727172012.651893"
    },
    "tag": {
        "devices": [
            "fire_smoke_detection_IoT",
            "air_quality_IoT"
        ],
        "create_alert": true
    }
}
```


3.4 SILVANUS Platform visualization

The visualization of UP9h features a map-based UI that presents alerts to stakeholders via the platform through a layer named Alert Threats, as is the case for most of UPs of SILVANUS. Adding this new layer to the existing ones provides stakeholders with an easy way to navigate and a clear, intuitive representation of wildfire risks, facilitating quick and informed decision-making. Additionally, color-coding is implemented to indicate the seriousness of alerts, adhering to the color-coding of the other UPs, which enhances the clarity and urgency of the displayed information. As can be seen in Figure 17 and Figure 18, Threat Alerts layer visualization on the SILVANUS dashboard provides a comprehensive overview of potential wildfire threats in a specific geographical area.

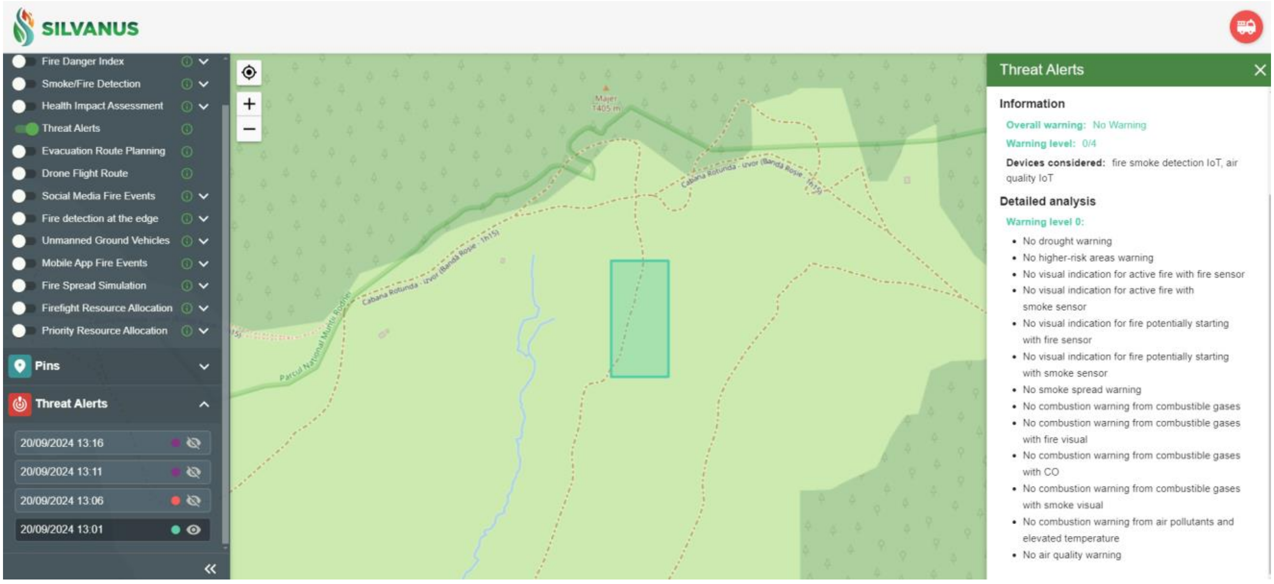


Figure 17: Threat Alerts example visualisation warning level 0 (Romanian pilot)

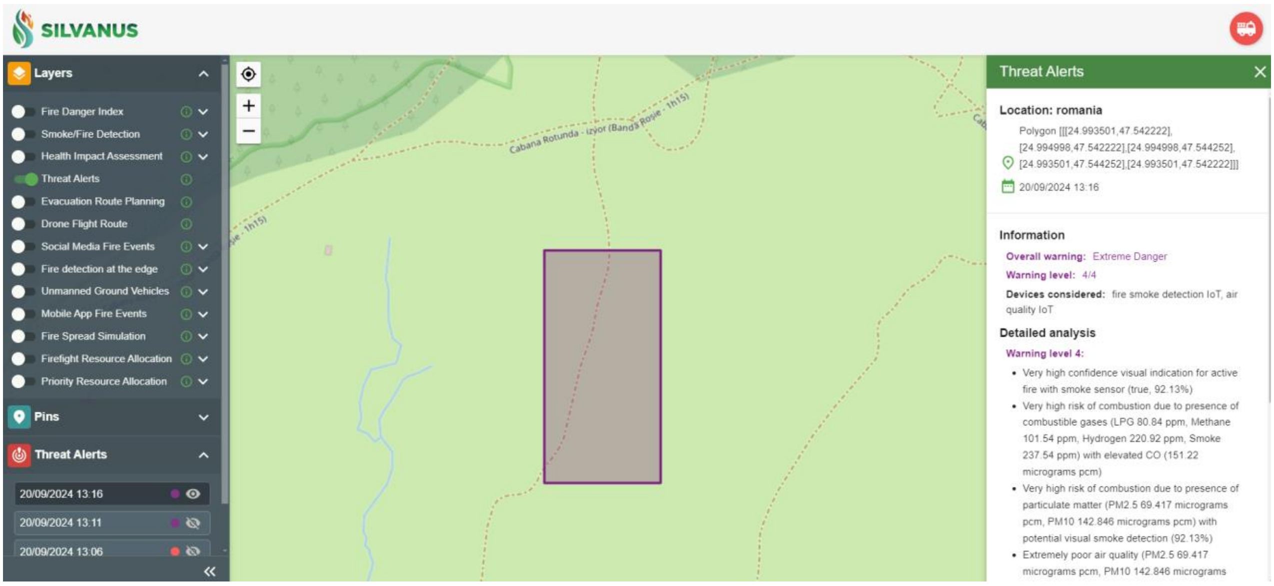


Figure 18: Threat Alerts example visualisation warning level 4 (Romanian pilot)

Here's a detailed description of the various components and functionalities displayed in Figure 18 (the same applies for Figure 17):

Map Overview

- **Geographical Area**: The map focuses on a specific area in Romania, highlighted with a purple polygon indicating the assessment region.
- **Icons and Pins**: Various icons on the map represent the locations of IoT devices that monitor environmental conditions. These pins are used to identify specific types of sensors and their respective locations.

Layers Control Panel

- **Toggle Layers**: The left-hand panel allows users to toggle various data layers such as Fire Danger Index, Smoke/Fire Detection, Health Impact Assessment, and more. This feature helps users customize the map view according to the specific type of data they wish to analyse.
- **Pins Section**: The pins section can be expanded to show more detailed information about each pin and its associated data.
- **Threat Alerts**: Located on the left-hand side under "Pins," this section allows users to view historical alert data and past threat notifications.

Information Panel

The information panel on the right provides detailed insights into the current health impact assessment for the highlighted area:

- **Location Details**: The panel specifies the exact location of the assessment, including the geographical coordinates (latitude and longitude) and the date and time of the assessment.
- **Overall Warning**: Displays the general risk level with a color-coded warning level (e.g., "Extreme Danger" with a purple label). The overall warning level is rated as 4/4, indicating a critical threat level.
- **Devices Considered**: Lists the types of IoT devices used for data collection in the assessment, such as fire smoke detection IoT and air quality IoT.

Detailed Analysis

The detailed analysis section is divided into various warning levels, providing specific information about different environmental conditions and their associated risks:

Warning Level 4: Indicates extreme danger with conditions such as:
  - Extreme drought due to highly elevated temperature and decreased humidity.
  - Very high confidence visual indication for fire potentially starting with fire and smoke sensors.
  - High indication of smoke spread detection in low humidity conditions.

Warning Level 3: Highlights high-risk conditions, including:
  - High confidence visual indication for active fire with fire sensor.
  - High risk of smoldering due to particulate matter with potential visual smoke detection.
  - Very poor air quality.

Warning Level 2: Details moderate risk conditions:
  - Moderate confidence visual indication for active fire with smoke sensor.
  - Moderate risk of combustion due to air pollutants and elevated temperature.

Warning Level 1: Not applicable in the current scenario.

Warning Level 0: Indicates no warning:
  - No combustion warning from combustible gases.

Additional Information

- **No Data**: Sections where data is not available are marked as N/A, indicating that there is no relevant information for those categories.

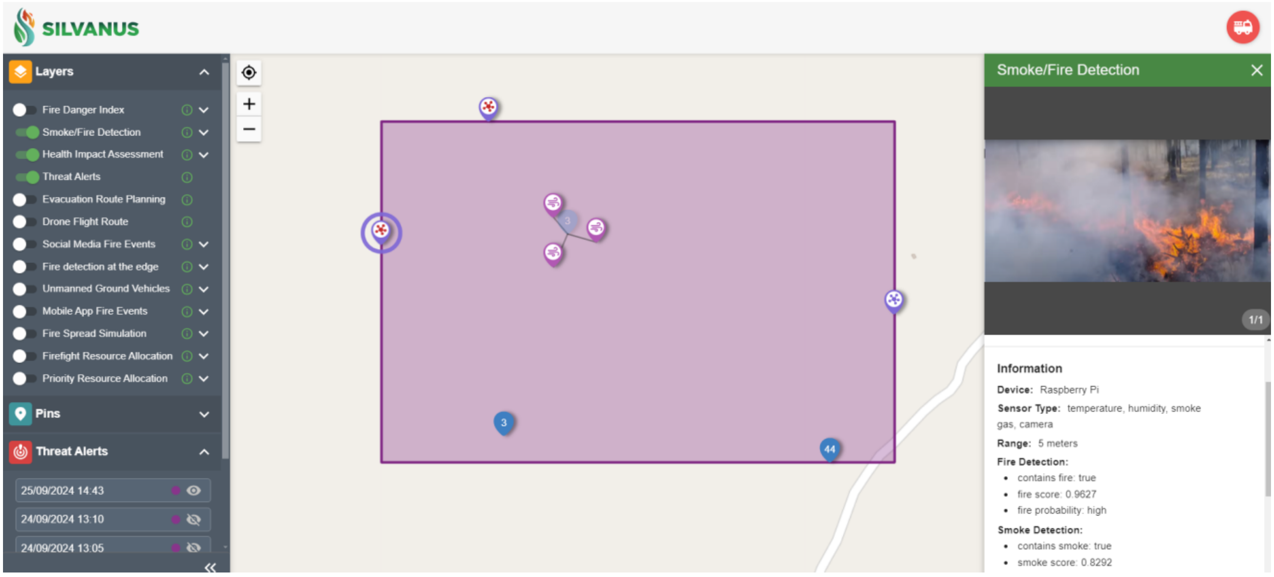


Figure 19: Multi-layered Fire Monitoring Dashboard

Last but not least, Figure 19 highlights the integration of Threat Alerts, Smoke/Fire Detection, and Health Impact Assessment layers within the SILVANUS platform. Upon activating the Threat Alerts layer, users can access additional information layers, which include Smoke/Fire Detection and Health Impact Assessment. This multi-layered visualization is designed to provide an initial overview of the area within the bounding box, marked by various pins that each correspond to different layers of data.

By selecting different layers, users can explore more specific aspects of the area's environmental conditions. For instance, the detailed Smoke/Fire Detection information shown on the right provides real-time visuals and a comprehensive analysis of the fire, indicating its severity and spread. This approach not only enhances initial situational awareness but also allows users to delve deeper into more specific data sources, thereby gaining a richer understanding of the situation. Such detailed insight is crucial for effective decision-making and supports the functionality of DSS, improving response strategies and overall safety measures in wildfire management.

3.5 Discussion of the results, limitations and future work

3.5.1 Results

The scenarios detailed above demonstrate the utilization of SemKB within the UP9h system to effectively escalate alerts based on the intensity and combination of detected conditions. This real-time capability is crucial for effective wildfire management, enabling stakeholders to make immediate and informed decisions. The system significantly enhances the user experience by automating and centralizing alert processes. Users receive comprehensive, actionable alerts without the need to manually synthesize data from various sources, and they have the option to delve deeper into additional data layers for more detailed statistics. The integration of the SILVANUS dashboard as the Alert Threats layer visually enhances comprehension and promotes timely actions in response to potential threats.

3.5.2 Limitations

Despite its potential, the UP9h system faces several challenges:

- **Rule Design**: The alert system's rules, primarily designed through discussions with end-users like environmental managers and firefighters, often lack detail and technical precision. This highlights the urgent need for continuous collaboration between these end-users and technical teams to refine and validate the rules, ensuring they are both practical and effective.
- **System Flexibility**: The rules for alert generation are currently hardcoded and predefined, limiting the system's ability to adapt to new conditions or incorporate user feedback dynamically. This inflexibility could impede the system's responsiveness to evolving wildfire patterns and emerging data insights.
- **Data Quality and Synchronization**: Accurate alert generation depends on high-quality, well-synchronized data from various sensors. Issues with spatial and temporal discrepancies in data collection can compromise the effectiveness of the alerts generated.

3.5.3 Future Work

The SILVANUS project is committed to continually enhancing the UP9h system. Planned developments include:

- **Integration of Social Media Data through UP3**: This will incorporate real-time, social-sensing data from platforms like Twitter, enhancing the comprehensiveness and immediacy of the alert system.
- **Continuous Refinement of Data Visualization**: Efforts will be made to improve how data is visualized, making it more accessible and useful for end-users.
- **Usability Testing and User Feedback**: We will conduct extensive usability testing and gather feedback from stakeholders, including firefighters and civil protection agencies, to refine the system and ensure it meets diverse needs.
- **Pilot Testing with Real Data**: Further pilot tests with real-world data will be conducted to validate and fine-tune the system, ensuring its readiness for deployment in wildfire-prone regions.

By addressing these challenges and implementing the planned enhancements, SemKB and UP9h aim to establish a robust and scalable decision-support system tailored for wildfire management. These improvements are designed to significantly enhance SILVANUS capabilities, fostering a more resilient response framework for managing wildfire threats. This will not only improve community safety but also strengthen disaster response strategies across the board, ensuring that the system can adapt and scale according to evolving needs and technologies.

4 Augmenting semantic space by semantic intensity information

4.1 Information fusion experiments with the semantic intensity and social significance of messages

With the expected co-occurrence of the next El Niño phenomenon and climate change [3], more natural disasters with increasing frequency are looming on the horizon [4]. As exemplified also by recent cases [5] [6] [7], wildfires manifest catastrophes with huge social fallout, and to fight them by means of a holistic digital solution, from their onset through crisis management to habitat recovery, is the overall target of the SILVANUS EU project. Our mid-term vision is to consider wildfires as a combination of natural and social disasters [8] [9], and to feed forward news analytics to a network of command centres for crisis management and post-event recovery.

Enriched information as raw material adds to the usability of tools for firefighting. Such enrichment is the endeavour of information fusion [10]. Below we explain two contributions which, by means of compounding different aspects of word vs. sentence meaning captured by social media sensing, can add new facts to the SILVANUS ontology for KG building.

In RP2 HB accomplished the following new developments:

- On test data (4K tweets) provided by CERTH, we enriched the semantic space of tweets by sentiments and centrality values indicating the social relevance of messages;
- On the same dataset, we drilled down in content and identified high-value adjective-noun phrases (bigrams) for the indexing of tweets;
- We demonstrate that semantic space contains evolving topics which, in a loose sense, can be modelled on fields in classical mechanics;
- We converted the above results into a new web service module for T4.4 with extended capabilities.

The above were worked out on two tracks. The first track used the Orange workflow designer [11]. The second one implemented tested findings in Python as a set of API endpoints for web service module upgrade with JSON output. Figure 20 sums up the respective workflow.

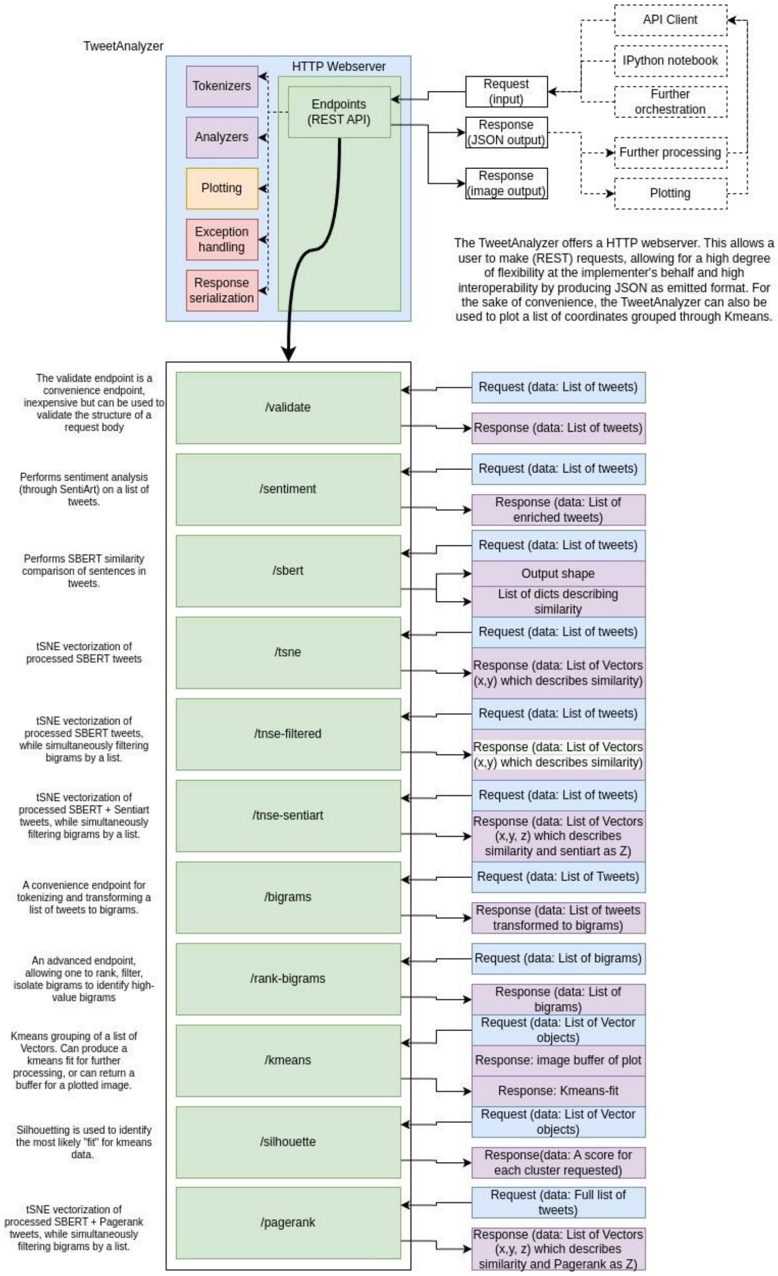


Figure 20: Flowchart of the updated HB tweet analyzer workflow component

Below first we provide a detailed description of background considerations having led to our theoretical inspiration, then proceed via experiment design and results.

4.1.1 Background considerations

Recent developments in DL and NLP [12] [13] hint at the existence of a shared conceptual vocabulary underlying different languages [14] [15] [16], manifest in different models of multilingual embeddings (Grave et al 2018). In this sense, word embeddings can be considered as high-dimensional locations in a geometric or topological vocabulary whose structural changes over time will become an important research question. As indicated in our earlier publications [17] [18] [19], vector spaces addressing change naturally lead to a next theoretical framework, vector fields, opening up new directions for methodology development such as the application of gravity to model content agglomeration [20] [21] [22] [23] [24].

In our current effort we studied static vs. dynamic content behavior in semantic spaces subjected to information fusion with other aspects of word and document meaning. In this frame of thought, there are two contributing factors we want to highlight. The first one is the changing nature of semantic content, typically studied as semantic drift [25] vs. concept drift [26]. Separately, another subfield is studying the evolving nature of semantic spaces. Dynamic Topic Models [27] and respective embedding methods, e.g. BERTopic [28] or HDBSCAN [29] [30] address this problem. The second factor worth mentioning is that evolving semantic spaces practically equal vector fields with changes both in location and direction vector values that map content trends to topology. Such trends could be related semantic drift and possibly formalized to model topic flows [31] [32] (Figure 21).

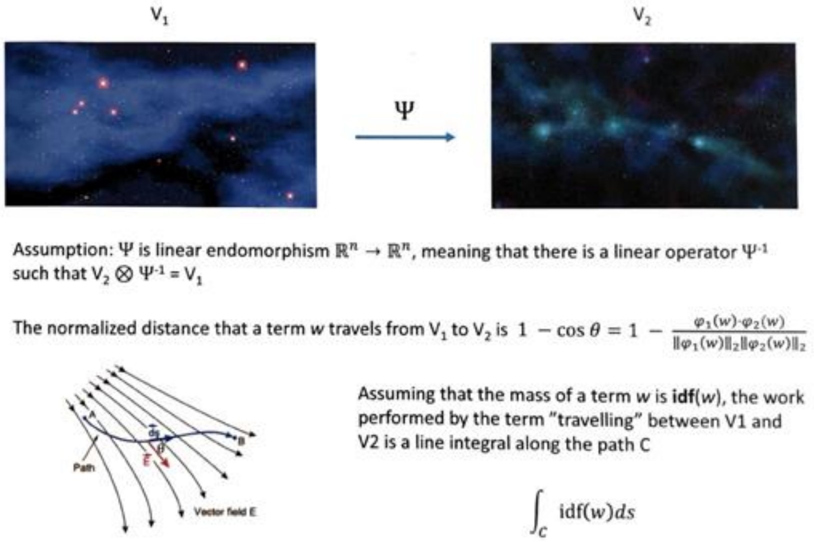


Figure 21: Field transformations can model semantic drift

4.1.2 Reasoning

Inspired by the idea of fields used both in physics and semantics, the above invites to test the applicability of classical mechanics as a frame of thought [19]. In the law of universal gravitation,

$$F = G\, m_1 m_2/d^2$$

F is the gravitational force acting between two objects, m_1 and m_2 are the masses of the objects, d is the distance between the centers of their masses, and G is the gravitational constant. In our current attempt, Δ replaced G in a social setting:

$$F = \Delta\, m_1 m_2/d^2$$

In other words, we anticipated that in a virtual environment covariant distance and sentiment vs. centrality values could be conceived as being co-dependent on Δ, with their momentum $p = mv$, i.e., $F = dp/dt$. This remedies the difference between a natural vs. a virtual environment where, in the former, only distances change.

Momentum is a key concept that applies to all systems, with force being its complementary. In other words, $F = dp/dt$ is the rate of change of momentum in the more general version of Newton's second law, $F = ma$, where a stands for acceleration. Once the time derivative tells us that we can identify some property of a system that changes, that property is its momentum whose development will be associated with a force. This expression of force in terms of the change in the momentum is universally applicable.

Below, two tests of the above frame of thought add sentiment and social importance as “momenta” to 2-dimensional semantic space. The resulting joint spaces manifest information fusion and exemplify how different pragmatic aspects of word and sentence meaning can be integrated into automatically monitoring communicated content.

Having considered the measurement aspects of word and sentence behavior, next we refer to the “charge” or ‘”loading” aspects of located content. Before contrasting a non-evolving vs. an evolving model of information fusion, we considered the semantic intensity of communication as our first additional measurable.

4.1.3 Semantic intensity

As the phrase “heated exchange” indicates, natural language has a thermometer-like component. This aspect is called its semantic intensity [33] [34]. Ranking words, phrases (from a linguistic perspective adjective-noun pairs, i.e., bigrams), sentences, or complete documents such as tweets by their semantic intensity involves analyzing the context and meaning behind each expression to determine how strongly they are related or the degree of emotion they convey [35] [36]. This is a complex task that typically requires advanced NLP techniques and access to a large corpus of text for context. To this end one might use:

- Sentiment analysis methods to evaluate the emotional intensity of each unit of content [37] [38]. This gives one a rough idea of the intensity in terms of positivity or negativity (Figure 22);
- Word embeddings like Word2Vec or GloVe, which can provide a numerical representation of words based on their context in a large corpus. The cosine similarity between the vectors of each word in a bigram or n-gram estimates their semantic relatedness [39]employs machine learning for sentiment embeddings [40], or uses databases like AffectVec (Raji & de Melo, 2020);
- If one has a specific set of criteria for “semantic intensity,” s/he can manually score each bigram or n-gram based on those criteria.

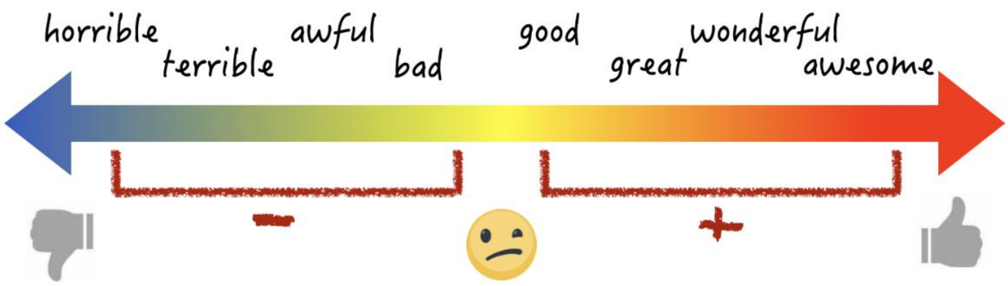


Figure 22: Full scale of adjectives describing positive and negative sentiment at different degrees from the SO-CAL dataset (Taboada et al., 2011)

In our experiments we defaulted on a combination of all three above components by using SBERT embeddings on a manually selected starter kit of sentiment ranked bigrams based on Roget’s Thesaurus (Figure 23).

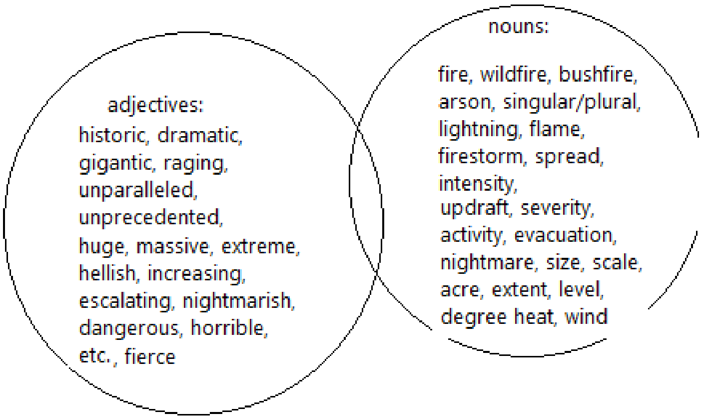


Figure 23: Sample ingredients for firefighting-related adjective-noun pairs

Below we show a rudimentary ranking based on the potential negative sentiment that bigrams might convey, considering that they are related to forest fires and evacuations:

- ('forest', 'fires'): Strong negative sentiment due to the association with uncontrolled fires.
- ('fires', 'becoming'): Slightly negative sentiment, suggesting an increase in fires.
- ('heavy', 'forest'): Negative sentiment, implying severe conditions.
- ('fire', 'smoke'): Negative sentiment, as smoke is a hazardous result of fires.
- ('smoke', 'forces'): Negative sentiment, indicating that smoke is causing actions to be taken.
- ('forces', 'three'): Neutral to negative sentiment, depending on what is being forced.
- ('communities', 'evacuate'): Strong negative sentiment due to the implication of danger.
- ('new', 'evacuation'): Negative sentiment, as evacuations are typically in response to threats.
- ('evacuation', 'history'): Neutral to negative sentiment, could imply a significant event.

4.1.4 Page Rank as a centrality measure

We considered the social significance of communicated content as our second measurable for information fusion. In its well-known form implemented by PageRank [41]. However, the mathematics of PageRank is entirely general and applies to any graph or network in any domain. In its simplest definition, the PageRank score for an item V in a graph G is defined as the weighted sum of the PageRank scores for the items linking to V, the weights being determined by the outdegree of each linking item. Thus, PageRank is now regularly used in bibliometrics, social and information network analysis, and for link prediction and recommendation. It is used for systems analysis of road networks, and in biology, chemistry, neuroscience, and physics [42]. With regard to text mining, PageRank is utilized in the TextRank algorithm that is used for keyword extraction and text summarization [43]. The TextRank algorithm is based on the creation of a weighted graph of text elements (e.g., words, phrases, or sentences) where the weight of the edge between elements is computed on the basis of verbal overlap or semantic similarity. Eventually our test opens the arena for the study of different kinds of centrality with an eye on node influence metrics as a more precise concept for information fusion. Furthermore, the concept of node centrality can be applied to dynamic graphs as well [44].

As ontologies and respective KGs are also known to evolve [45], [21], [46]albeit at a much slower pace than semantic spaces computed from quickly developing topic components [32], we considered these two measurables as candidates for enriching the SILVANUS semantic framework.

A suitable general mathematical framework for formulating and studying the "limit" of dense graphs, as those induced by semantic networks, social networks, and knowledge graphs is that of graphons (see e.g. [47]), which can be considered the continuous generalization of discrete adjacency matrices associated with graphs, as exemplified in Figure 24. In our upcoming work, we intend to formulate a procedure for inducing graphons from specific graph structures such as semantic networks of words and bigrams.

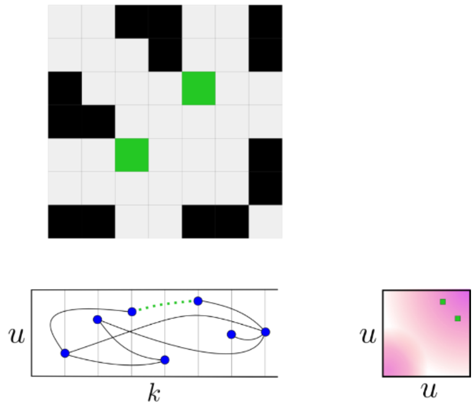


Figure 24: Example of a graphon corresponding to a discrete adjacency matrix. The image "Exchangeable random graph from graphon.png" by Victor Veitch is licensed under CC BY 3.0

4.1.5 Design of the experiment

The dataset provided by CERTH for experimentation on a proof-of-concept level consisted of 4K tweets. Building on the workflows of the Relevance Classification and Concept Extraction textual analysis modules HB contributed to T4.4 *Social media sensing and concept extraction*, we partly reused and augmented analytical tools to comprehend an Information Fusion module as follows.

To uncover latent topics in social media posts both in static and evolving topical environments, and provide a structured understanding of the key concepts and discussions surrounding wildfires, the workflow was composed as a sequence of document embeddings by means of sentence transformers (multiple pretrained models, [13]) using large language models (LLMs), topology-preserving dimensionality reduction (two-dimensional data projection by *t*-distributed stochastic neighbor embedding (*t*-SNE, [48])), unsupervised machine learning (*k*-means clustering ([49])) combined with silhouette-based quality estimation to identify the optimal number of clusters ([50]), sentiment analysis [38], network node centrality analysis [41], and field visualization both by contour maps and scalar fields representing the topical structure by means of inverted contour maps. For predictive purposes regression analysis (model the time series using vector auto-regression [51]can be added, because one formalism adapted in this work was to treat sequences of tweets related to the same events as time series of vector data.

Our approach analyzed the static vs. dynamic (longitudinal, time-dependent) aspects of content agglomeration on two levels:

- Considering tweets as documents that consist of a few short sentences, and turning sentence content into the basis of content grouping;

- Going one level deeper for concept extraction by screening the corpus for tweets with either semantically intensive, or socially important (i.e., retweeted) bigrams (adjective-noun phrases).

We used the results to test an interpretation framework that combines 2-dimensional groups of semantic content with sentiments and/or centralities as a third dimension by means of information fusion. Our purpose was to explain the resulting static vs. dynamic contour maps as vector fields, a step departing from vector spaces proper.

For bigram indexing, we screened Roget's Thesaurus[15] for an eventually conceivable set of adjective-noun phrases an excited reporter of a fire incident might use. This set was sentiment analyzed and gradually reduced to a shortlist of ranked phrases, subjected to further analysis and visualization. The same procedure was repeated to assign Page Rank values to the complete dataset. Every step of the above workflow is parametrizable.

4.1.6 Results

4.1.6.1 Static scenario

In the static approach to the dataset, i.e., considering all 4K tweets and their relations as elements of the same snapshot, both versions of joint space showed clusters of tweets augmented with sentiments vs. centralities to indicate semantic intensity vs. social significance in 3-d (Figure 25). We sentiment ranked 28 bigrams and tagged 117 tweets with them. The inverted landscape reminds one of the gravitational potential whose local gradient is the strength of the experienced attractive force. This strength can be visualized by a quiver field where arrows point at the strongest attractors, in our context the joint content with strongest "gravitation" (Figure 26). Messages with high sentiment or importance values are primary candidates for such sinks.

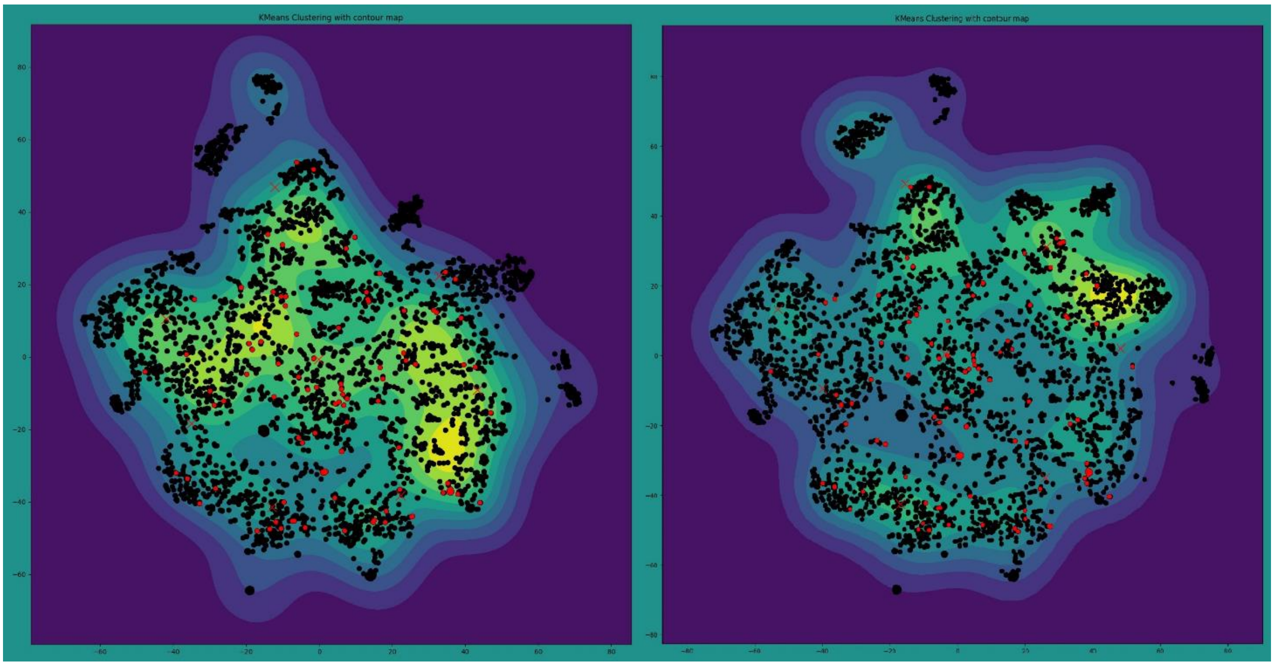


Figure 25: Left: Semantic intensity contour map of 4K tweets, highlighting 117 high-sentiment tweets in red. Right: Social importance contour map of 4K tweets, with elevation indicating perceived importance (e.g., retweets)

[15] https://www.gutenberg.org/ebooks/22

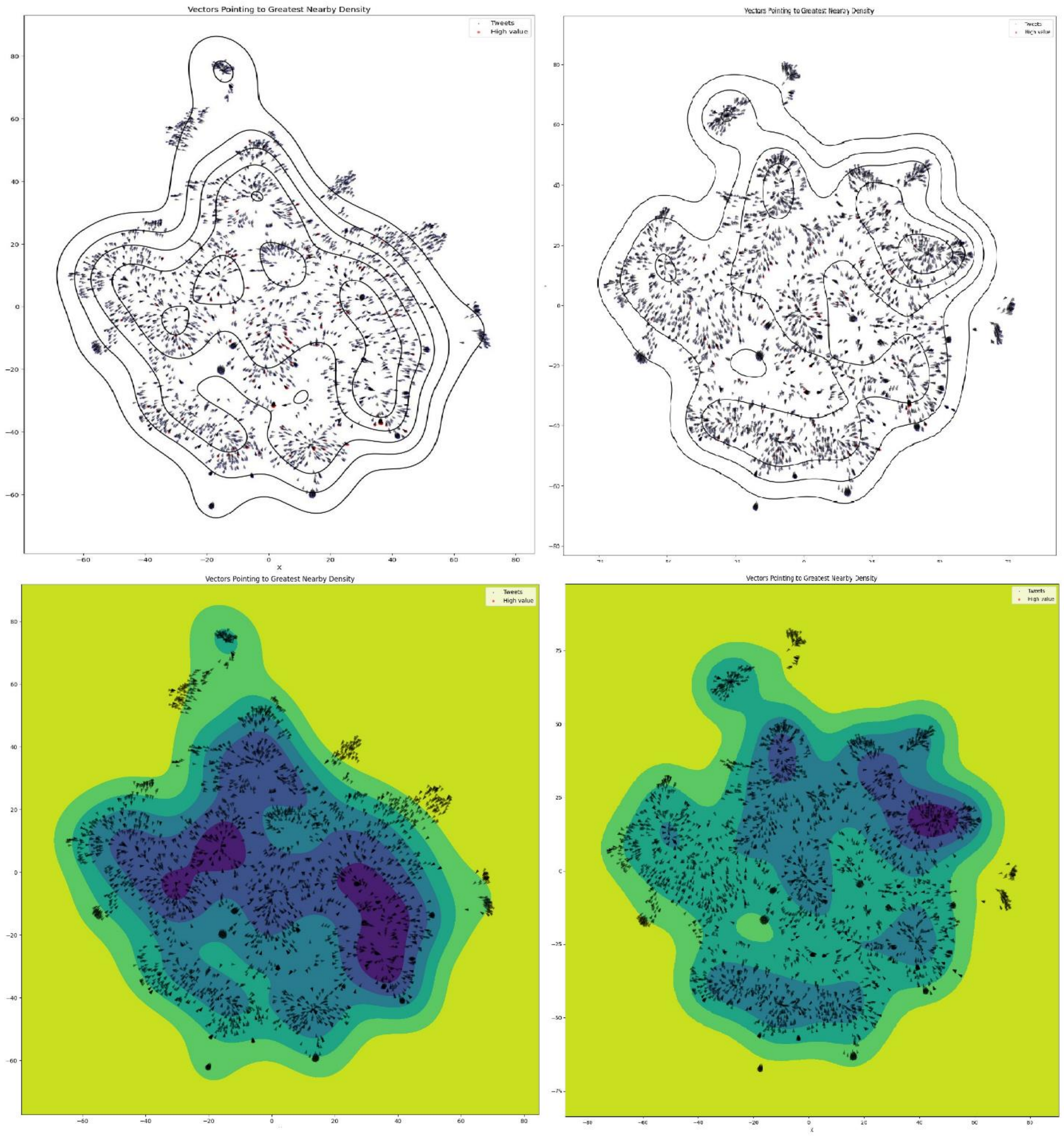


Figure 26: Static scenario. The two scalar fields are different. In the inverted contour map tweets with strongest fear (left) or highest social relevance (right) populate the basins

The procedure was as follows. With probability density assessed by Gaussian kernel density estimation (KDE), we computed a gravity-inspired elevation landscape where density is applied as a gradient to each plotted tweet, effectively pointing an arrow from each tweet to closest areas of greatest probability or "mass". Since force in classical mechanics is the negative gradient of potential, i.e. $F(x) = -dU/dx$, this potential corresponds to the local steepness of the inverted contour landscape, with the steeper the slope the stronger the corresponding "force" expressed as a field like structure.

4.1.6.2 Dynamic scenario

We selected a battery of 44 nouns pertinent to fire incidents and filtered the complete dataset for adjectives co-occurring with any of them. This resulted in 49 applicable adjectives to 14 occurring nouns out of the 44, and 66 adjective-noun bigrams, including ‘forest fire’, with 323 occurrences of these bigrams in the dataset. Next, we separated the data in four equal segments of 1000 tweets each according to timestamps and extracted 4 constant adjective-noun pairs (*Canadian wildfires*, *active wildfire*, *new fire*, *unprecedented wildfire*) which occurred in all four periods. The bigrams were sentiment vs. centrality ranked for joint space construction. In Strategy A, we computed the joint space separately for all four of the 1K tweet segments with the constant bigram basis marked in red (Figure 27). This was a non-evolving strategy whose knowledge was not accumulating over the periods. In strategy B, period 1, period 1+2, period 1+2+3 were observed and compared with the static picture we knew from the complete dataset, i.e., period 1+2+3+4 (Table 3). Here the changes in the *xyz* coordinate values over time reflected the results of accumulating statistics aka knowledge used for event classification (Figure 28). Next, we repeated the experiment with social importance as the height component of joint space, with the respective visuals in Figure 29 and Figure 30.

Table 3: Index term occurrences over observation periods in strategies A and B

Occurrences of Canadian wildfire, active wildfire, new fire, unprecedented wildfire	*Strategy A (non-evolving content, a sequence of static spaces)*	*Strategy B (evolving content)*
Period 1	24	24
Period 2	36	60
Period 3	25	85
Period 4	33	118

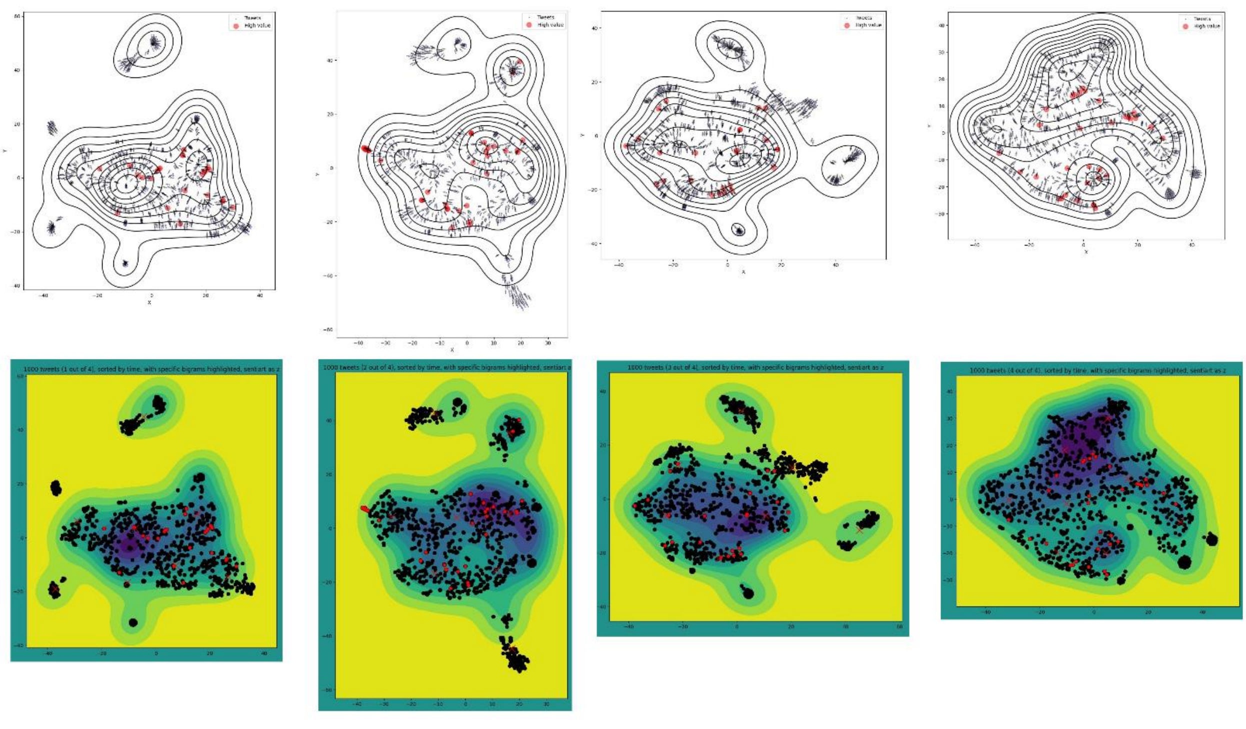
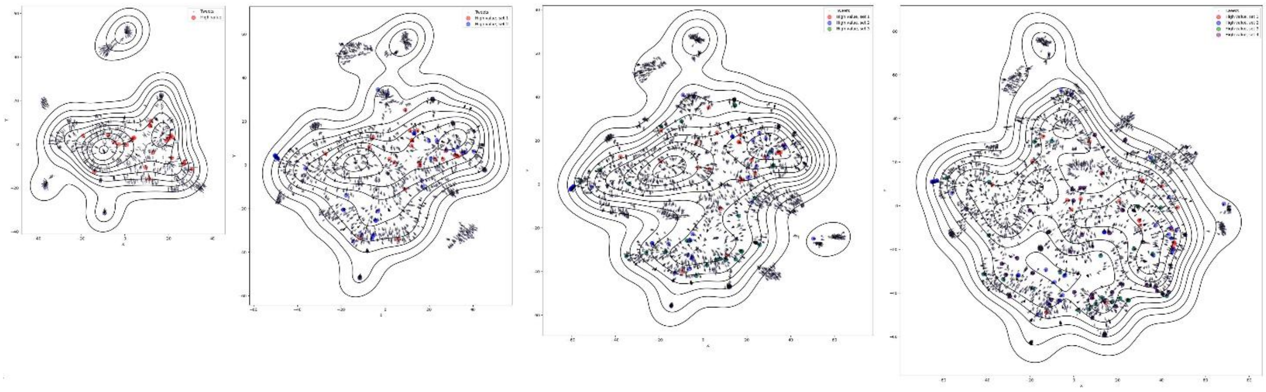


Figure 27: Joint semantic-sentiment space content and field structure, strategy A (four independent/static spaces, from left to right over periods 1 to 4)



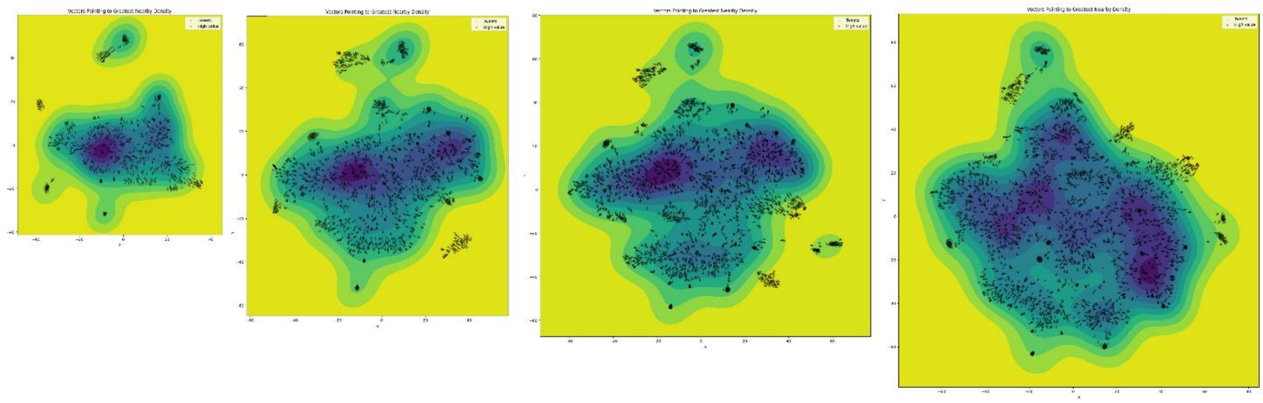


Figure 28: Joint semantic-sentiment space content and field structure, strategy B (four evolving spaces, from left to right over periods 1 to 4)

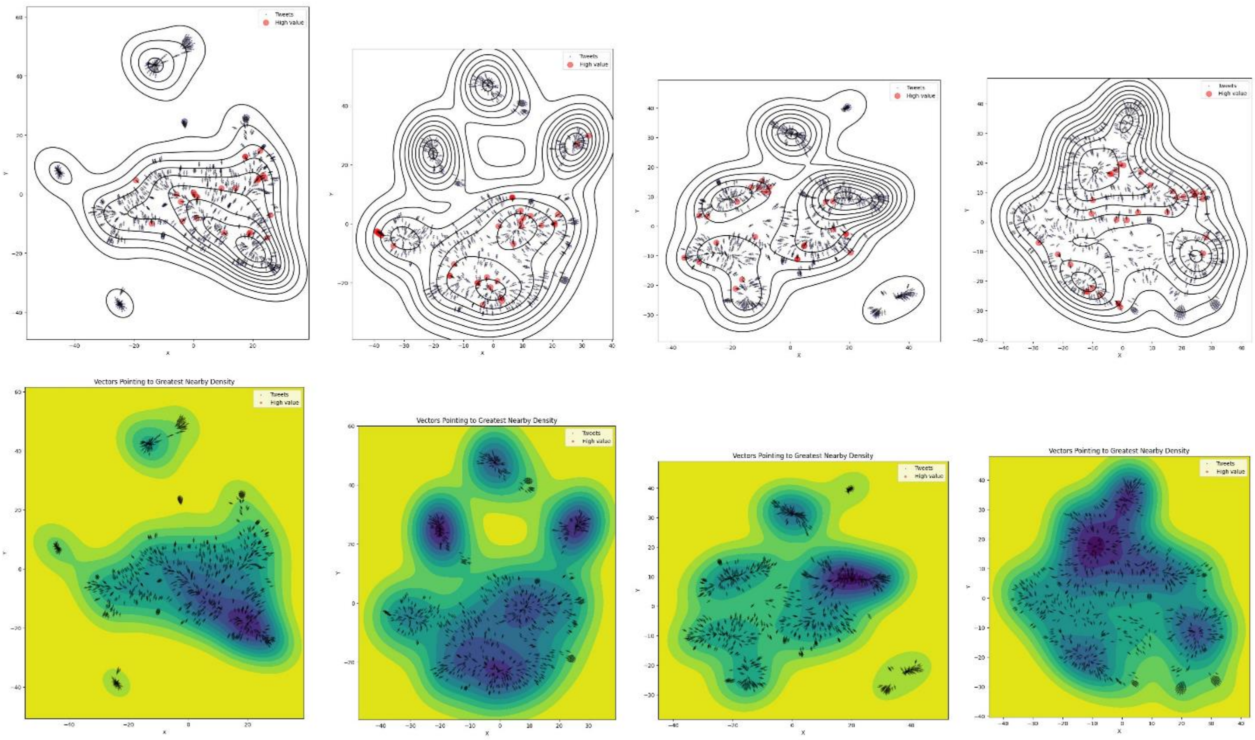


Figure 29: Joint semantic-centrality space content and field structure, strategy A (four independent/static spaces, from left to right over periods 1 to 4)

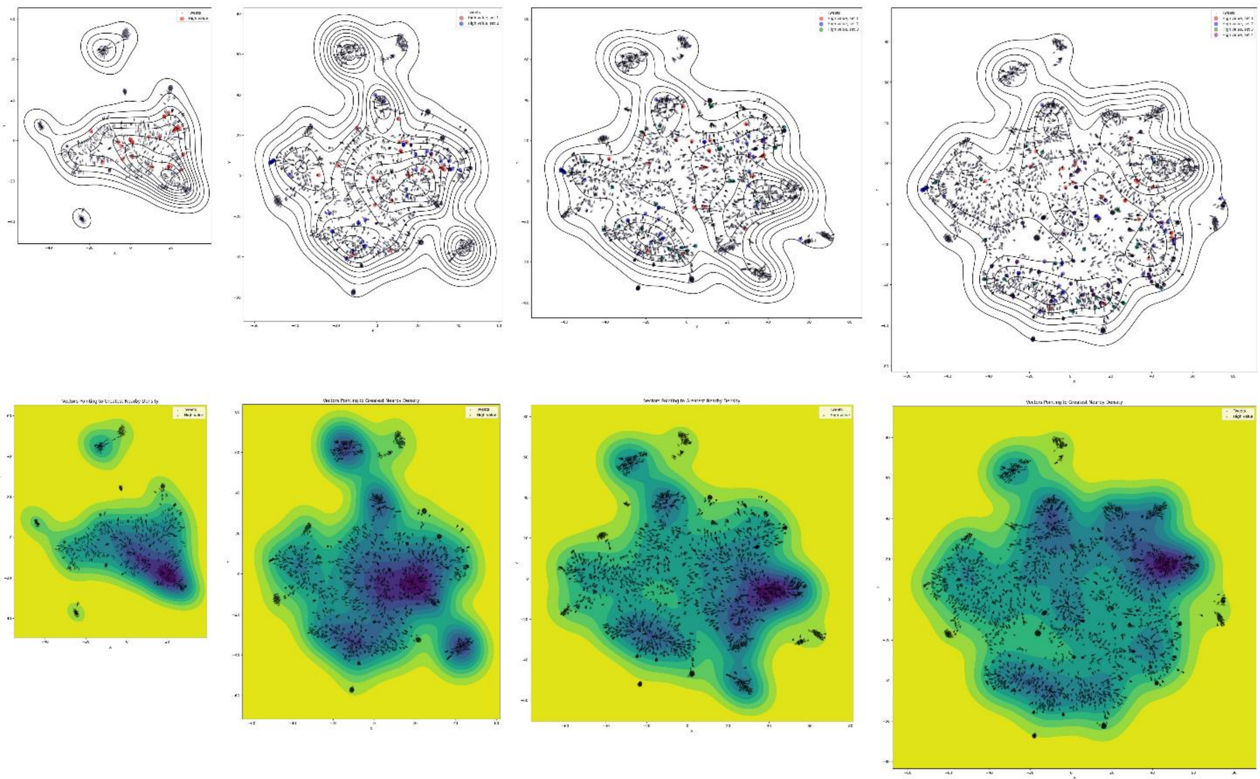


Figure 30: Joint semantic-centrality space content and field structure, strategy B (four evolving spaces, from left to right over periods 1 to 4)

As is evident from strategy B, accumulating examples expand semantic space and modify the knowledge structure. This could be key for a better understanding of a dynamic situation where tweets are coming in and getting obsolete on a one-by-one basis. The underlying reason of the above is that with accumulating tweets over the same four index terms, the semantic relations expressed by distance maps also keep on evolving (Figure 31).

Period 1	*Canadian wildfires*	*active wildfire*	*new fire*	*unprecedented wildfire*
Canadian wildfires		4,00	2,65	2,65
active wildfire	4,00		4,12	4,12
new fire	2,65	4,12		2,83
unprecedented wildfire	2,65	4,12	2,83	
Period 1+2	*Canadian wildfires*	*active wildfire*	*new fire*	*unprecedented wildfire*
Canadian wildfires		6,40	5,75	5,20
active wildfire	6,40		5,66	5,10
new fire	5,75	5,66		4,24
unprecedented wildfire	5,20	5,10	4,24	
Period 1+2+3	*Canadian wildfires*	*active wildfire*	*new fire*	*unprecedented wildfire*
Canadian wildfires		7,87	6,71	6,25
active wildfire	7,87		6,56	6,08
new fire	6,71	6,56		4,47
unprecedented wildfire	6,25	6,08	4,47	
Period 1+2+3+4	*Canadian wildfires*	*active wildfire*	*new fire*	*unprecedented wildfire*
Canadian wildfires		9,16	8,00	8,06
active wildfire	9,16		6,93	6,56
new fire	8,00	6,93		5,39
unprecedented wildfire	8,06	6,56	5,39	

Figure 31: Evolving distance structure among the four constant index terms over the four observation periods

4.1.7 Summary

It is possible to apply spacetime clustering to semantic content in interaction with sentiments or estimated social relevance in a message.

We have found a useful framework for expressing content attraction by information fusion. In its dynamic variant, evolving vector spaces can model the structuration capacity of semantics from a longitudinal perspective, suitable for modelling. In that framework, content dislocations can be expressed as a series of spaces with the observed entities subjected to virtual forces, driving them ahead according to topical tensions due to event formation, a manifestation of semantic evolution.

On a proof-of-concept level, messages in social media can be ranked according to their semantic intensity and/or perceived social importance, and this distinction as new metadata can enrich ontology and KG building. Research and development implications are twofold:

- *Firefighting*. Signal processing relies on data describing the actual states of physical fields. If messages pertain to fire events, and not to their political overtones (as is often the case in social

media), information fusion-induced semantic fields can be coupled with relevant physical field data. Cf. DTM linked with non-text climate[16] and fire danger[17] data for terminology calibration.
- *Ontologies.* With the topical composition of semantic space changing over time, the distance structure of shared index terms over periods will change as well. Because the distance structure of concepts in an ontology encodes their semantic relatedness, this finding will have to be studied from a vector field perspective.

Meeting both goals holds the promise to develop vector field representations of coupled sensor data, ontology concepts, and topics. The represented dynamics would go beyond social media and could be applied e.g. to disaster management communication. Via crisis-specific bigram collections to measure semantic intensity, these studies could be expanded to the complete spectrum of disaster management [52]).

4.1.8 Weaknesses and unfinished business

We note in passing that our accomplishments could be continued and extended e.g. based on the following observations:

- Currently bigrams with high sentiment/importance values were not used beyond indexing tweets containing them but could be. Bigram indexing based on crisis-specific vocabularies augmented with semantic intensity vs. social significance value scales could be one way to go.
- In a live setting, tweets come in one by one, not in bulk, making real time classification difficult and rendering static vs. dynamic measurements impossible, but dataset-based precomputed classifications and transfer learning might help.
- Ontologies represent logical semantics; information fusion is based on statistics and the distributional hypothesis [53]to align their findings is important.
- Timestamped messages about a particular event in joint space can be considered as a narrative with unfolding intensity vs. importance values.
- To create event specific multifocal datasets with coupled fire history, evolving physical data, professional communication, and social media coverage would help event modelling for firefighting.
- We recommend the notion of graphons and how such functions can be used to characterize large, dense information networks such as semantic graphs and knowledge graphs.

4.2 Tracking Fire Intensity Semantics and Sentiment in Social Media (EAI)

EAI supported HB in advancing the task of tracking evolving semantics, focusing on integrating and analyzing diverse data sources to enhance wildfire management strategies. This collaboration primarily targets the integration of semantic analysis of social media data, specifically tweets, into the SILVANUS platform. A significant aspect of the work involved mapping linguistic cues from tweets to physical measurements, which entailed analyzing the language used to describe fire events and exploring how this information can be added to real-time sensor data.

One of the main challenges addressed was eliciting fire intensity from tweets. EAI and HB worked together to understand how users express fire intensity on social media and align these expressions with concepts in the SILVANUS ontology. The team also focused on improving sentiment analysis, which initially faced issues, returning null results for certain tweets. This necessitated troubleshooting and refinement, leading

[16] https://drought.emergency.copernicus.eu/tumbo/edo/map/?id=1000
[17] https://forest-fire.emergency.copernicus.eu/apps/fire.risk.viewer/

to the identification and resolution of a bug in the sentiment analysis pipeline. After resolving this issue, the sentiment analysis performed as expected. There was also debate about assigning sentiment to individual concepts within a sentence versus assigning sentiment to the entire document, such as a tweet. It was decided to assign sentiment to the entire tweet so that each concept could weigh on the overall sentiment.

Content-related challenges were encountered with the tweet dataset from CERTH, mainly due to the content and genre of the tweets, which often centered on political opinions and blame rather than factual descriptions of fire events. This made it difficult to extract meaningful information that could be directly linked to the ontology and physical measurements. However, since the dataset was relatively small (4,000 tweets), the team continued to explore this avenue with more bigrams and linguistic expressions related to fire intensity. These were incorporated into the concept extraction module to provide such information when available or inferable.

Following the resolution of these initial challenges, the proposed next steps for collaboration between HB, EAI, and CERTH included the creation a joint visualization, like a contour map, to illustrate the distribution of SILVANUS concepts and sentiment values extracted from the tweets. This visualization aimed to demonstrate the potential of combining these analyses. Furthermore, the teams planned to expand the analysis of fire intensity, investigating if other, more indirect, language cues could be used to infer intensity levels from social media data.

By leveraging semantic technologies, EAI aimed to refine these tools, ensuring they provide accurate and actionable insights critical for proactive wildfire management. This approach not only supports data fusion but also supports the development of a dynamic SemKB, integral to improving predictive capabilities and real-time responsiveness to wildfires, ultimately contributing to the prevention, detection, and management of wildfires.

5 Semantic reasoning schemes – Lightweight semantics

5.1 Introduction

In opposition to classic semantic approaches e.g. using ontologies (as described in Chapter 3), we have exploited also a little bit simpler, Lightweight semantics (LWS), approach, used in Web 2.0 applications [54][[55] [56] mainly in social media. This approach utilizes tags (commonly known as hashtags) and operates on a flexible system that doesn't require defining a comprehensive taxonomy for every possible object within the application. Instead, it relies on the assignment of specific tags to organize and categorize content effectively.

Semantic interoperability can also be addressed by standards from semantic web such as RDF(S) or OWL (Chapter 3) [57]. These semantic technologies offer interesting possibilities, however, they face several unresolved problems which prevent them to be widely adopted. These issues include the exponential complexity of inference techniques in rich models; unavailability of exhaustive semantic description of the problem area; and the problem of contradictory knowledge. To overcome these obstacles, ongoing research is focusing on developing more efficient inference algorithms, establishing standardized ontologies, and enhancing methods for resolving inconsistencies.

In contrast, we can create LWS in the form of tags and annotations. Such LWSs may be more appreciated by end users, since it is easier to manage and can deal with contradictory knowledge, as well as provide scalable inference based on graph algorithms. To process LWS we can also exploit a wide range of graph mining methods developed in recent years to analyse the semantic networks created by users, as well as several interoperability activities or semi-automatic IE (Information Extraction) [58]. A key distinction between ontologies and LWSs lies in their approaches: ontologies typically employ a top-down methodology, whereas LWSs utilize a bottom-up approach.

5.2 Implementation

This LWS approach is developed together with Task 4.4. Social media sensing and concept extraction for Slovak pilot and Slovak texts. This task crawls and extract Named Entities by using the Facebook Crawler and a deep neural network based on SlovakBert model. A pre-trained model extracts: 6+1 different types/classes of Named Entities described in Table 4.

Table 4: All annotated classes

Named Entity Class	*Description*	*Example*
Keyword	action regarding the fire (verb, noun, adjective)	inhale, extinguish, burn, fire, flame, smoke
Time	time of the event	11:15, today, yesterday, before nine
Unit	unit which was used during the response	volunteer fire brigades, mountain rescue service, soldiers, policemen, HaZZ[18]
Fire area	what was on fire (noun)	grass, hall, flat, house, chimney, garage, cathedral, straw
Equipment	what equipment were used during the response	dron, vehicle, rank, helicopter, pump, jet
Location	location of the event	cities, villages, hills, factory areas, countries
Other	everything else	

[18] HaZZ means Firefighters and Rescue Units

We have used fir annotation the IOB2 format (with tagging unaffected by stop word filtering) as an extension of IOB (short for inside, outside, beginning), commonly referred to as BIO format (Figure 32).


```
12. B-time
novembra I-time
2022 I-time
vypukol O
na O
hrebeni O
pohoria O
Malá B-location
Fatra I-location
neďaleko O
sedla B-location
Pod I-location
hromovým I-location
požiar B-keyword
trávy B-fire_area
, O
ktorý O
zasiahol O
plochu O
o O
rozlohe O
2 O
hektáre O
. O
```

Figure 32: Post annotated in IOB2 format[19]

As depicted in Figure 32, it is necessary to concatenate several consecutive word chunks into a single object representing recognized named entities. The output is generated as an array of JSON objects. Each JSON object includes references to its start, end, type, and the value of the entity detected by the NER (Named Entity Recognition) deep neural network. Figure 33 illustrates the array of JSON objects produced.


```
 [{
      "end":17,
      "entity":"time",
      "object":"12. novembra 2022",
      "start":0
},
{
      "end":25,
      "entity":"keyword",
      "object":"vypukol",
      "start":18
},
{
      "end":55,
      "entity":"location",
      "object":"Malá Fatra",
      "start":45
},
{
      "end":83,
      "entity":"location",
      "object":"sedla Pod hromovým",
      "start":65
},
{
      "end":90,
      "entity":"keyword",
```


[19] The text refers to: 12th of November 2022 on Malá Fatra hills next to the Pod hromovým saddle the grass fire started …

```
        "object":"požiar",
        "start":84
},
{
        "end":97,
        "entity":"fire_area",
        "object":"trávy,",
        "start":91
}]
```


Figure 33: JSON object generated by NER neural network

Here is a brief explanation of the objects extracted from the example. 'Time' is self-explanatory. 'Vypukol' translates to 'started.' The first location is straightforward, while the second refers to a landform saddle named 'Pod hromovým saddle.' The keyword 'požiar' means 'fire,' and the fire area 'tráva' translates to 'grass.' This JSON array also includes a document ID, which is a unique identifier for the originating social media post. The data is then sent through the SAL queue to the main Lightweight Semantic (LWS) component, where it is stored in a graph structure. The document ID is also used as another property of the post within the graph. We utilize the Neo4j[20] graph database to store the data. If an object from a new post already exists in the graph database, it is not duplicated; instead, a new edge is created from the existing object to the newly stored post. An excerpt of such a graph can be seen in Figure 34. In the center of the image, our example is visible, connected through 'požiar' (fire) to other posts.

[20] https://neo4j.com/

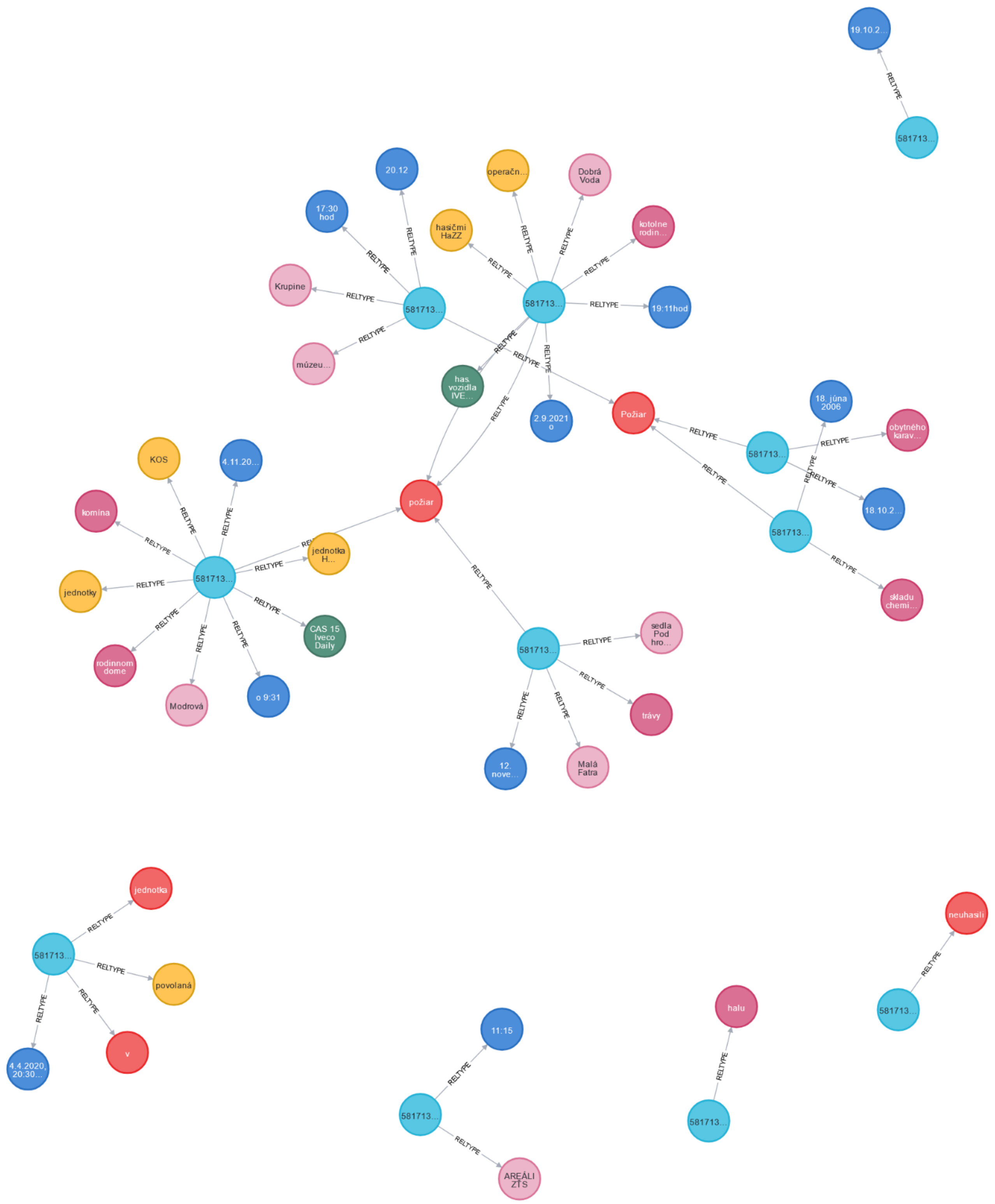


Figure 34: Excerpt of the graph stored in the neo4j graph database

Legend:

- cyan – post ID
- dark blue: date
- dark red: fire area
- light red: keyword
- pink: location

- yellow: unit
- dark green: equipment

The entire graph can be utilized to infer interesting facts, for example, by using SPARQL queries as described in Section 2. A Semantic Search Engine, built on top of the Lightweight Semantic (LWS) component, is integrated through SAL and is designed to process queries that infer facts from the Neo4j graph database. The format for these queries is illustrated in Figure 35.

```
{
    "depth": "3",
    "operator": "AND",
    "search": [
        {
            "object": "požiar",
            "type": "keyword"
        },
        {
            "object": "Bratislave",
            "type": "location"
        }
    ],
    "return": [
        {
            "type": "unit"
        },
        {
            "type": "time"
        },
        {
            "type": "fire_area"
        }
    ]
}
```

Figure 35: Request to the Semantic Reasoning Engine

To delve deeper into the capabilities of the Neo4j graph database, let's explore how it uses a query language known as Cypher[21], which is somewhat akin to SPARQL used in other semantic technologies. Cypher is designed for efficient querying of graph data structures, allowing for complex relationships and pattern recognition within the network of nodes and edges.

In the example provided, the Cypher query illustrated in Figure 36 leverages the Neo4j database to search for specific patterns and connections.

```
MATCH(k: Keyword{keyword: 'požiar'})-[]-(x1)-[*1..3]-(y1), (l: Location{location: 'Bratislave'})-[]-(x2)-[*1..3]-(y2) WHERE (x1.id=x2.id) RETURN y1.unit, y2.unit, y1.time, y2.time, y1.fire_area, y2.fire_area
```

Figure 36: Request to the Semantic Search Engine in Neo4j language

Explanation of the query:

- *Pattern Matching*: The MATCH clause specifies the patterns to find in the graph. Here, it looks for nodes labeled Keyword and Location with properties that match specific values ('požiar' and 'Bratislava'). These nodes are used as starting points to traverse the graph.

[21] https://neo4j.com/docs/cypher-manual/current/introduction/

- *Traversal*: The -[]-(x1)-[*1..3]-(y1) portion indicates a traversal from the keyword node through any kind of relationship (-[]-) to another node x1, and from there, it allows for a variable depth traversal ([*1..3], meaning 1 to 3 hops) to a node y1. A similar pattern is followed for the location node to y2.
- *Condition*: The WHERE clause filters the results to only those pairs of nodes (x1 and x2) that have the same id, ensuring that the traversal is comparing or combining data from logically connected elements.
- *Output*: The RETURN statement specifies the properties of the nodes y1 and y2 that the query will output, including units, time, and fire area.

As can be seen, crafting queries like the one shown in Figure 36 would be quite burdensome for users. Therefore, it is much more convenient to utilize a proxy to our Semantic Search Engine. This proxy simplifies the interaction, allowing users to leverage complex queries without needing to write them directly. To clarify the query illustrated: it is designed to return units, times, and fire areas associated with the keyword 'požiar' (fire) and the location 'Bratislava', while traversing the graph to a maximum depth of three. This depth parameter determines how far from the initial nodes—those containing the objects {keyword: 'požiar'} and {location: 'Bratislava'}—the algorithm should explore. The results of these queries are displayed in Figure 37.


```
"time": [
      "Nočná",
      "pred hodinou",
      "dnes okolo 12:30",
      "Práve",
      "18. hodinou"],
"unit": [
      "Hasičom",
      "momentálne",
      "hasiči",
      "bezpečnostné zložky"],
"fire_area": [
      "bytu",
      "opustenej budovy",
      "budovy pri",
      "bytovke",
      "Hotela Atrium",
      "garzónke",
      "druhom poschodí vchod č. 25",
      "byt",
      "bytovka",
      "10. poschodí",
      "Les"]
```


Figure 37: Returned objects of query from Figure 32

The meaning of the time objects' values in the order from the first to the last are: Nightly, hour before, today around 12:30, right now, 18^{th} hour. For units (again in order as they appear in the listing): Firefighters, right now (this is an error of NER tool), firefighters, rescue responders. And for the fire area: flat, abandoned building, building, Hotel Atrium, 1 room flat, second floor at number 25., flat, building, 10^{th} floor, forest.

As mentioned earlier, all these components: (i.e. Facebook Crawler and NER tool from WP4, Lightweight Semantic and Semantic Search Engine) are integrated through SAL. There are 3 different queues created for that purpose:

- **uisav.fb-crawler** queue is used by Facebook crawler and NER tool (in WP4)
- **uisav.ner-lws** queue is used by NER tool and LWS to submit extracted entities with their types to Lightweight Semantic Engine

- **uisav.lws** queue can be used by any tool and LWS (Semantic Search Engine part) for inferencing and searching over stored data in Neo4j graph engine

6 AI technologies for modelling biodiversity

In this chapter, the AI technologies that have been developed in SILVANUS for monitoring the state of forest and collecting the biodiversity information through crowdsourcing has been outlined. In particular, the scope of the research being presented is to summarise the natural assets that are present in the forest. To this end, the section outlines the technical details of Woode application and the different modules that are integrated into the application.

6.1 Woode Application backend technologies

In this section a more detailed design of the three tools integrated within the Woode application is summarised. In this chapter the architectural design of each tool will be presented along generic diagrams while a detailed analysis and screenshots of the tools or application are also provided As mentioned before, SILVANUS development of Woode application relays on the latest research outcomes from Generative AI-models and Deep-Learning including Kernel-Based Adaptive and Collaborative Flows [1], Interpretability and Pruning of Vision-Based Neural Networks [2] and Lossless Pooling Convolutional Networks for video processing [3-4]. The first important technical development of Woode relates to AI-based AR to enable citizen engagement and promote the significance of the forest resources that needs to be protected. The corresponding envisaged tool enables automated generation of high-resolution images of real forest environments. Image or video upsampling involves increasing the resolution of low-resolution images. Traditional methods (such as bicubic interpolation) often result in blurry or unrealistic details. With the advent of deep learning technology a quantum leap was achieved and realistic super resolution images can be obtained exploiting the latest developments in the field. In this project the two most promising super resolution models are being implemented. First, stable diffusion based super resolution and lossless pooling convolutional networks for super resolution (LPCN-SR).

The first model leverages diffusion-based techniques to enhance the resolution of images, providing high-quality results. Diffusion models are a class of generative models that simulate the gradual spread of information or noise within an image. They operate by iteratively updating pixel values based on neighboring pixels, leading to coherent and realistic results. Stable diffusion models focus on stability during the diffusion process. They address challenges such as gradient explosion or vanishing gradients, ensuring smooth and reliable updates. Stable diffusion models take a low-resolution image as input. Guided by a text prompt (such as a description), they iteratively enhance pixel values. The diffusion process ensures coherent transitions and sharp details creating realistic scenes, modifying existing content, or enhancing details. The second approach exploits the down sampled self-replicas of the input, low resolution image. It is based on the theoretical work presented in [3-4], where lossless pooling layers that create several highly correlated down sampled replicas from the input image, which can then be fused in the SR network. The corresponding process is outline in Figure 38

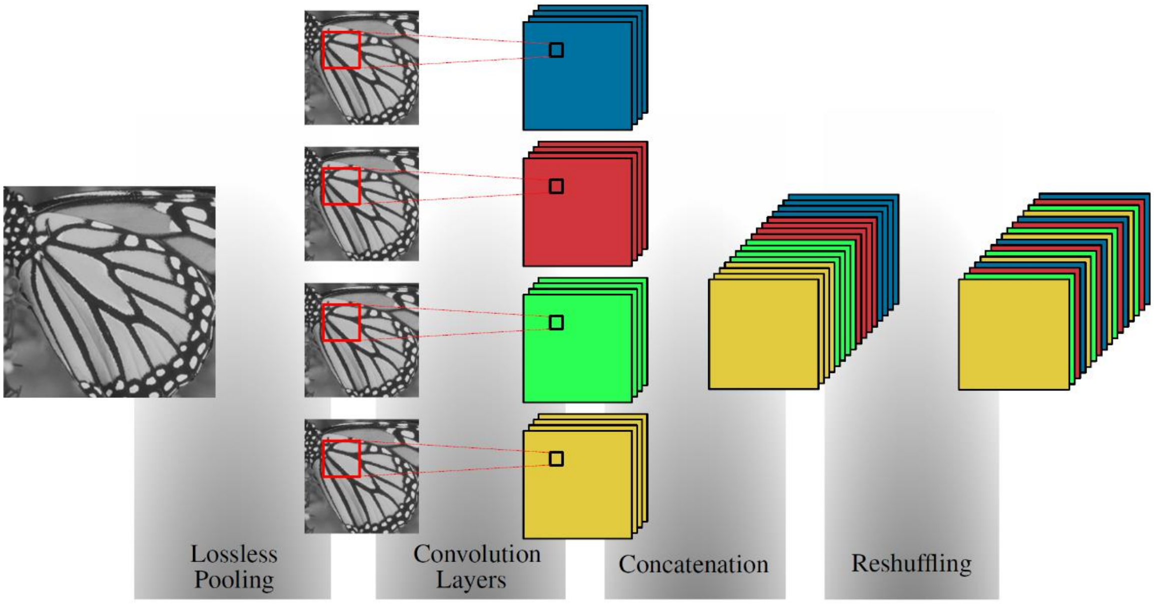


Figure 38: The process of fusing the self-replicas in CNNs by concatenation and reshuffling of the convolution outputs

A second significant contribution of Woode App is the provision of automatically generated virtual content for generating awareness and showcasing the threat of wildfire on the forest region. Although research on AI-based realistic image generation is already mature, the generation of videos is much more challenging and related research more embryonic. The starting point of our developments is the work by Tero Karras et al.reported in [5]. In this seminal paper the authors thoroughly analyse Diffusion-based generative models for image generation, in both unconditional and conditional settings. They also cement the fact that diffusion models overcome the limitations of more classic Generative Adversarial Networks (GANs). Indeed, GANs are also deep learning models that uses two networks called generator and discriminator competing against each other, to create realistic virtual images. A comprehensive review of available GAN techniques and their performance is presented in [6]. Although GANs are still a popular research topic for image processing and synthesis, their limitations led us to focus on Diffusion Models. Other techniques from the literature include Variational lower bound maximization or VAE models and Flow-based models. Since the performance of these models is much lower than advanced stable diffusion, these are not further considered in the developments of Woode app. The impact of the work presented in [5] is mostly due to the fact that the authors reduce previous heavily analytic and to some extend unnecessarily elaborated research work to their fundamental and pure practical characteristics. By doing so they introduce a design space that clearly separates the concrete design choices.

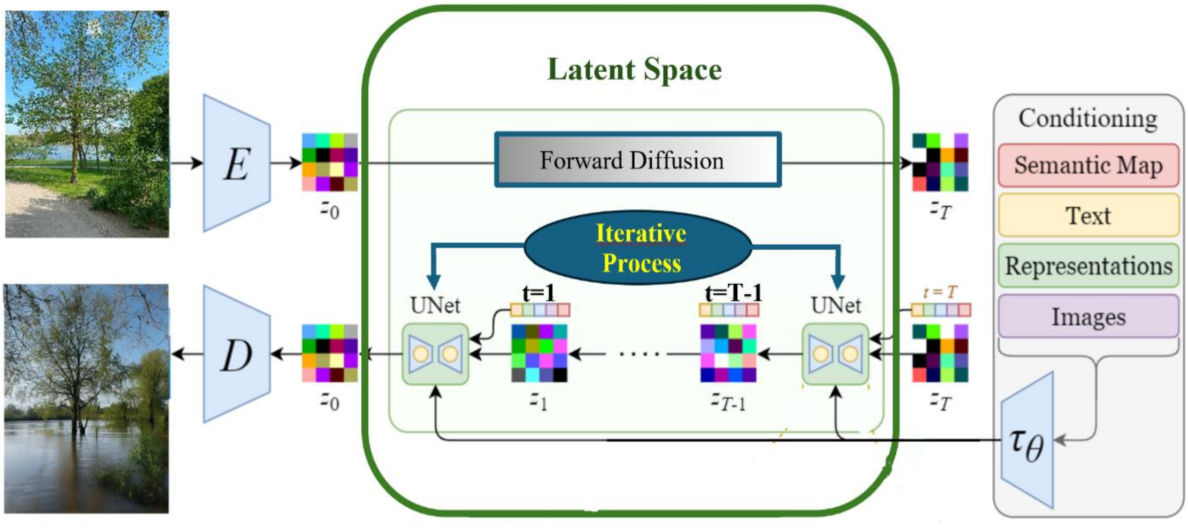


Figure 39: Woode architecture for stable diffusion

At implementation level, we adopt the standard architecture for stable diffusion as outlined in Figure 39. This model offers a new dimension of possibilities in the realm of virtual image generation since it combines semantic driven AI with learning in the image space. At the hearth of the iterative process is the well-established U-Net developed at the University of Freiburg [7]. However, inspired by the research presented in [5] and from initial testing and validation, our initial architecture leverages a larger U-Net backbone.

Its fundamental architecture is depicted in Figure 40. Additional technical features include attention blocks and greater cross-attention context for the semantic encoder and the refinement module to improve the visual fidelity of samples generated by the post-hoc image-to-image technique also introduced in [5]. The initial architecture of our basis implementation is depicted in Figure 39, supported by U-Net (Figure 40). However, to have a flexible and highly adaptive process pipeline, we start by exploring the same model only in the context of highly realistic single image synthesis. Then, we will move to expand the same pipeline for video clip generation. For this, a refiner network will be needed, to perform just refinements in the latent space.

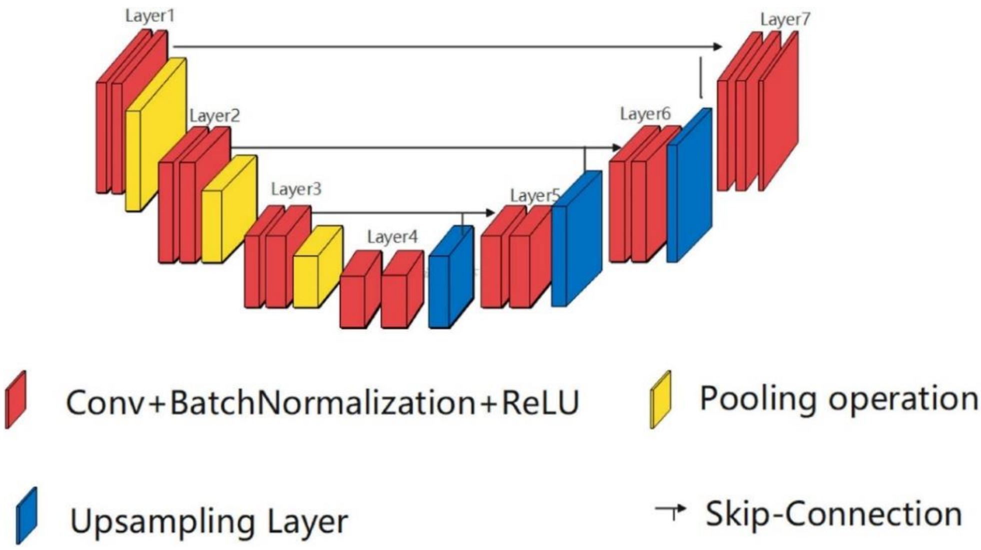


Figure 40: The U-Net model underpinning the adopted stable diffusion architecture in Fig. 6.

The idea is to take an image as an input of the processing pipeline and "refine" it. However, a VAE-Decoder will be also needed along with the refiner network to perform refinement in the latent space. This initial processing pipeline will be the base for our latent diffusion model. This is in principle an adaption to stable diffusion to allow for gradual diffusion of an image, which in turns allows modifications according to external requirements, not only in terms of performance but also in terms of quality of generated visual content (realism).

Initial experiments result in virtual images with a reasonable degree of realism. However, in consecutive refinements towards video clips, low jittery framerates are clearly perceived. To overcome this critical issue frame interpolation will be required and exploited.

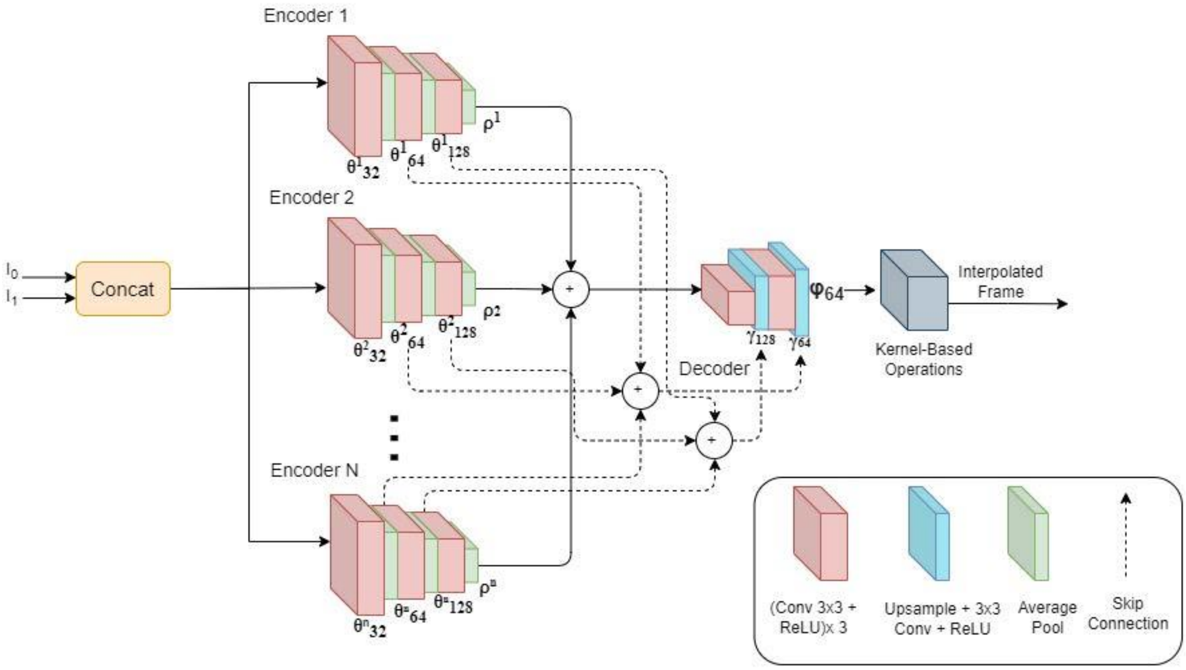


Figure 41: Overview of the proposed Woode app architecture of the multi-encoder network for frame.

Video frame interpolation involves the synthesis of new frames from existing ones. Convolutional neural networks (CNNs) have been at the forefront of the recent advances in this field. One popular CNN-based approach involves the application of generated kernels to the input frames to obtain an interpolated frame. Despite all the benefits interpolation methods offer, underlying networks require a large number of parameters and a heavier computational burden. Hence, we build on our recent research based on parameter reduction for a flow-less kernel-based network ACF (Adaptive Collaboration of Flows) [1]. To improve the performance, work is being conducted to integrate local and global features in the kernel-based interpolation network. The focus here will be on ensuring that runtime and memory usage is greatly reduced. The aim is to ensure that the proposed methods are easy, intuitive and straightforward to implement whilst achieving competitive performance to facilitate the adoption in the X-Alfy application. An overview of the targeted multi-encoder network is given in Figure 41.

This approach is orthogonal to other methods that reduce the complexity of networks such as pruning and quantisation [2]. However, it will be clear to the expert in the field that to achieve the required latency pruning and quantization will need to be considered too. In this context, X- Alfy's approach will be based on the exploitation of the universal method to reduce the complexity of neural networks introduced in [2]. It is based on a universal method to understand the "black box" nature of underlying networks. Then, using this information, redundant components from the neural networks can be removed, this is commonly called neural network pruning. This process leads to a significant decrease in the computing power required to deploy a neural network, reducing inference, deployment costs and allowing for neural networks to be used on smaller edge devices like phones [2].

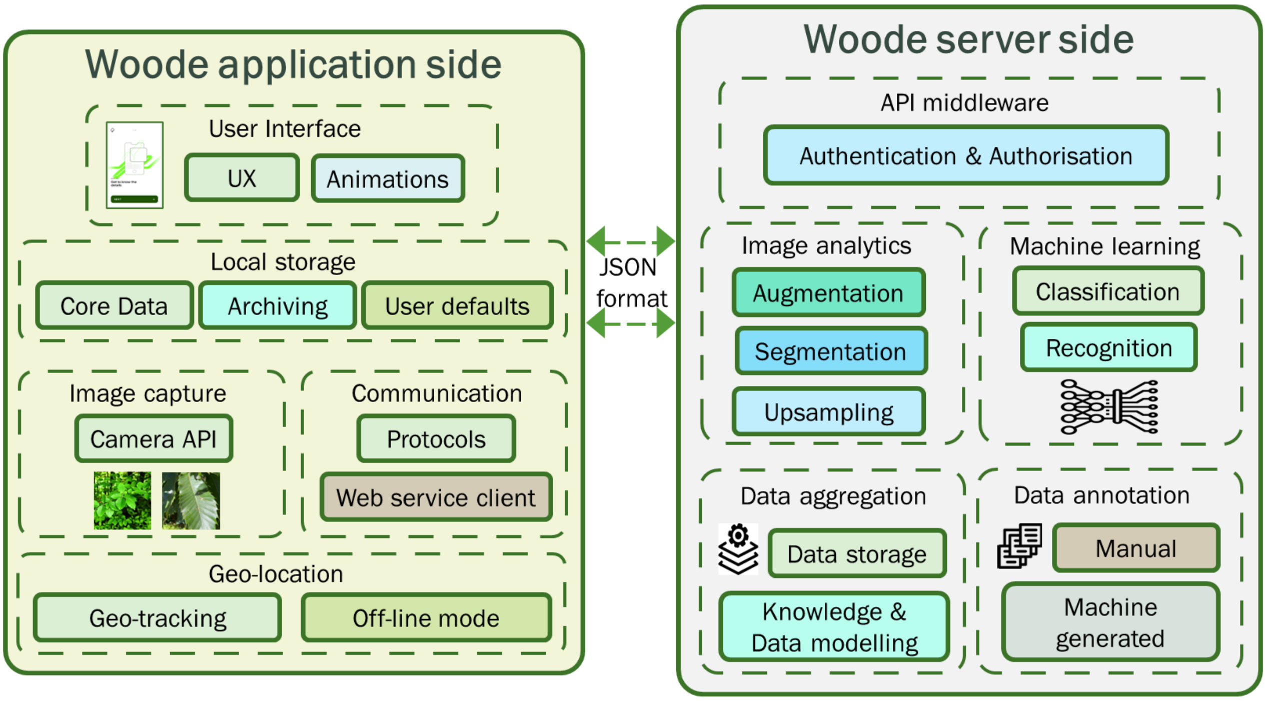


Figure 42: Outline architecture of the mobile application.

Finally, a short outline of the envisaged mobile app architecture is provided in Figure 42. This architecture should be based on a high-level structure and organization of its components, ensuring efficient functionality and performance. As depicted in Figure 42, it comprises several layers, including the presentation layer responsible for the user interface and user experience, the application layer handling logic and data processing, and the data layer managing storage and retrieval of information. Communication between these layers is facilitated through well-defined interfaces and protocols. The application architecture will be designed to allow for modular, scalable, and loosely coupled components, referred to as a micro-service. Cloud-based services, APIs, and middleware solutions are integrated to support features like data synchronization, user authentication, and real-time updates. Since a picture is worth thousand words, we have embedded remaining descriptions in Figure 42. In the remaining part of the section, a summary of super resolution algorithm is presented, followed by the tree species classifier and the description of the mobile application.

6.2 Super Resolution

The first Tool instantiates AI-based technology to automatically generate high-resolution images of real forest environments suitable for the creation of 360° panorama shots and VR simulation. Figure 43 shows the envisaged application and expected outcomes of this tool. Such super-resolution representations will be used as supplementary virtual content for the enrichment of didactic elements needed for the establishment of AR-enhanced teaching and learning environments. The aim is to use the super resolution component for the generation of visual data for the provision of appealing forest training courses in a virtual teaching and learning format. In such an AR enhanced course, users would be able to access silvicultural, timber harvesting or nature conservation lessons. They can then virtually learn tree species, assess the status of forest areas, identify valuable habitats and/or plan timber harvesting according to sustainable forestry protocols.

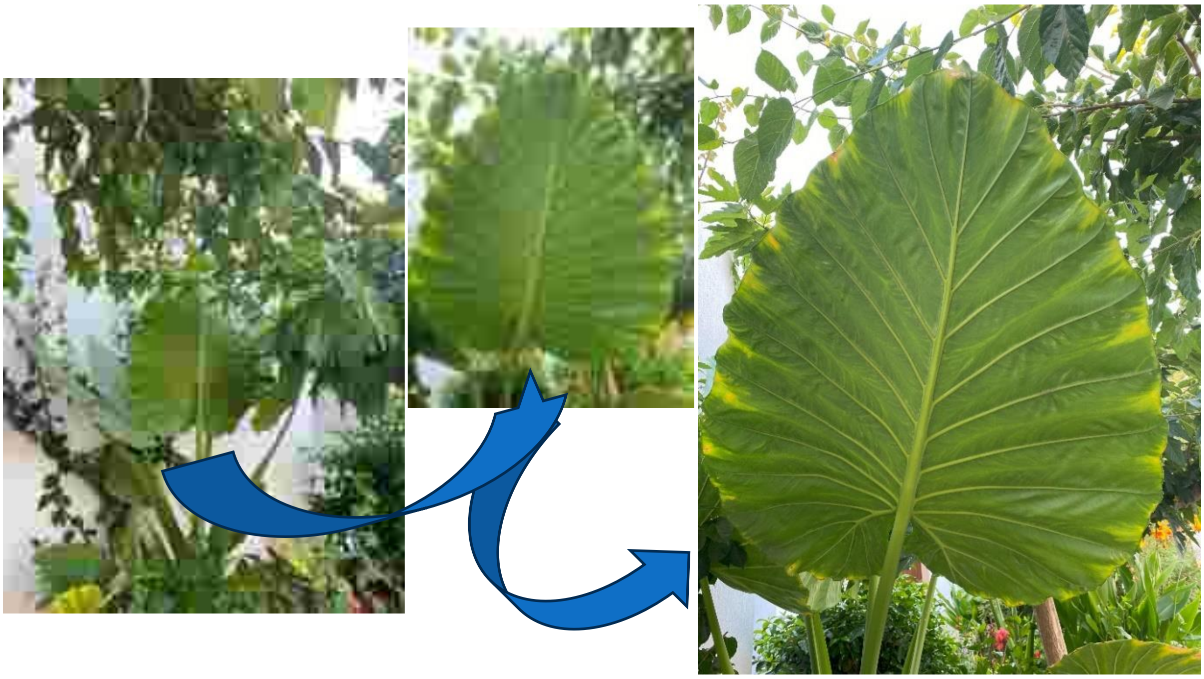


Figure 43: Super Resolution toolkit

Recent advances in machine learning, and application of deep neural networks, have resulted in major improvements in various computer vision applications. Super-resolution is not an exception, and it is amongst the popular topics that have been affected significantly by the emergence of deep learning. Employing modern machine learning solutions has made it easier to perform super-resolution in both images and videos and has allowed professionals from different fields to upgrade low resolution content to higher resolutions with visually appealing picture fidelity. In spite of that, there remain many challenges to overcome in adopting deep learning concepts for designing efficient super-resolution models.

The tool being developed for the generation of super-resolution content is based on both, emerging complex generative AI techniques and Lossless Pooling Convolutional Networks for video processing [3-4]. The underlying model only exploits down sampled self-replicas. It creates several highly correlated down sampled replicas from the input image, that are fused by the SR network [3].

The models are trained on large datasets, learning to capture high-frequency information. Clearly the final results are strongly dependent on both, quantity and quality of used training data. To address the quantity data challenge, in this project crowdsourcing of high-quality data is targeted. Hence both quality and quantity are addressed. The more people use the App the better the results of the provided tools. This is an important aspect considered in the implementation of the underlying mobile application, see additional related descriptions below.

6.3 Tree species classifier

This also an AI-based tool for automated classification of tree species. This tool will be used to generate complementary semantic descriptions to the visual AR content generated with super resolution. It will assist in the creation of semantic metadata for visual information. The aim is to further augment visual content with important semantic information to be exploited in teaching and learning courses.

The main objective is to enable automated classification and recognition of plant species by only few labelled samples. This tool is also key for crowdsourcing of information needed for both training of the AI models being developed and for educational purposes. Hence, with time it is expected that the image database will significantly grow enabling better re-training and classifications results.

The implemented technique relays on a local-foreground selection attentive bilateral feature reconstruction network for few-shot plant species classification. Classification of plant species is a fine-grained classification problem where different species have high intra-class variations and low inter-class variations. To achieve better classification accuracy, intra-class variations is minimized, while inter-class variations is maximized by exploiting local-foreground attention. During the reconstruction process, only the feature sequences that contribute to the foreground plant object is selected and the image parts belonging to background are rejected. In this way, the intra-class differences are reduced by removing the effect of background noise or cluttering. By extracting local spatial details, the inter-class differences are increased and more discriminative features are considered

6.4 Mobile application

Complementing the tools as outlined above, a mobile application for data gathering and testing, as well as, for showcasing the addressed educational and awareness scenarios, will be implemented (Figure 44). The aim is to enable: a) easy capturing and sharing of low resolution pictures and short clips of forest landscapes, trees and leaves; and b) showcasing the capabilities of the proposed tools for the generation of virtual and augmented visual content.

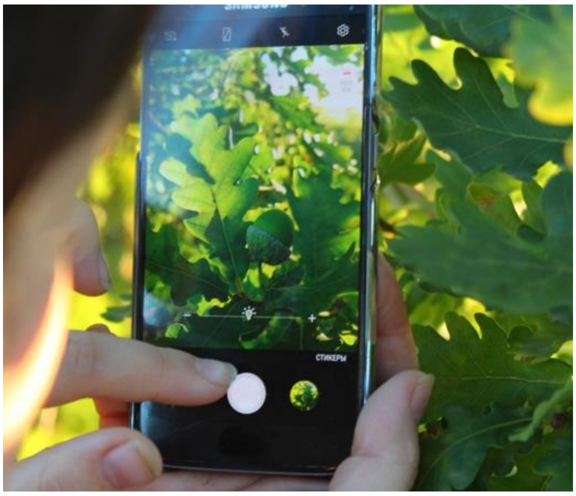


Figure 44: Woode mobile App

The aim is to exploit the pervasive use of handheld devices for the crowdsourcing of low resolution visual content in forestry. In addition, the application will provide the front-end features needed to demonstrate the capabilities of the previous tools outlined. Specifically, generation of highly realistic and suitable virtual content to be used in VR and AR-enhanced teaching and learning. Furthermore, the application will be key to show the potential of VR in the context of forestry conservation and to improve people's awareness, thus contributing to the reduction of human triggered forest fires. Finally, and critically, the application will be also instrumental for the monetisation of the developed technology, as base for a solid exploitation plan and as means to showcase the corresponding outputs to potential customers.

The information aggregated through the mobile app is then processed for the presentation to the foresters and end-users using semantic model. An overview of the point data representation of the location where the information of the tree species has been collected. The data about the tree species is then transformed into a GeoJSON which is then placed on the map as shown in Figure 45. The semantic information of the tree and the associated natural habitat are being presented in the GIS layer, as shown in Figure 46. The information about the natural resources and the habitat will be useful for the firefighters to consider a response coordination to mitigate against the loss of natural resources.

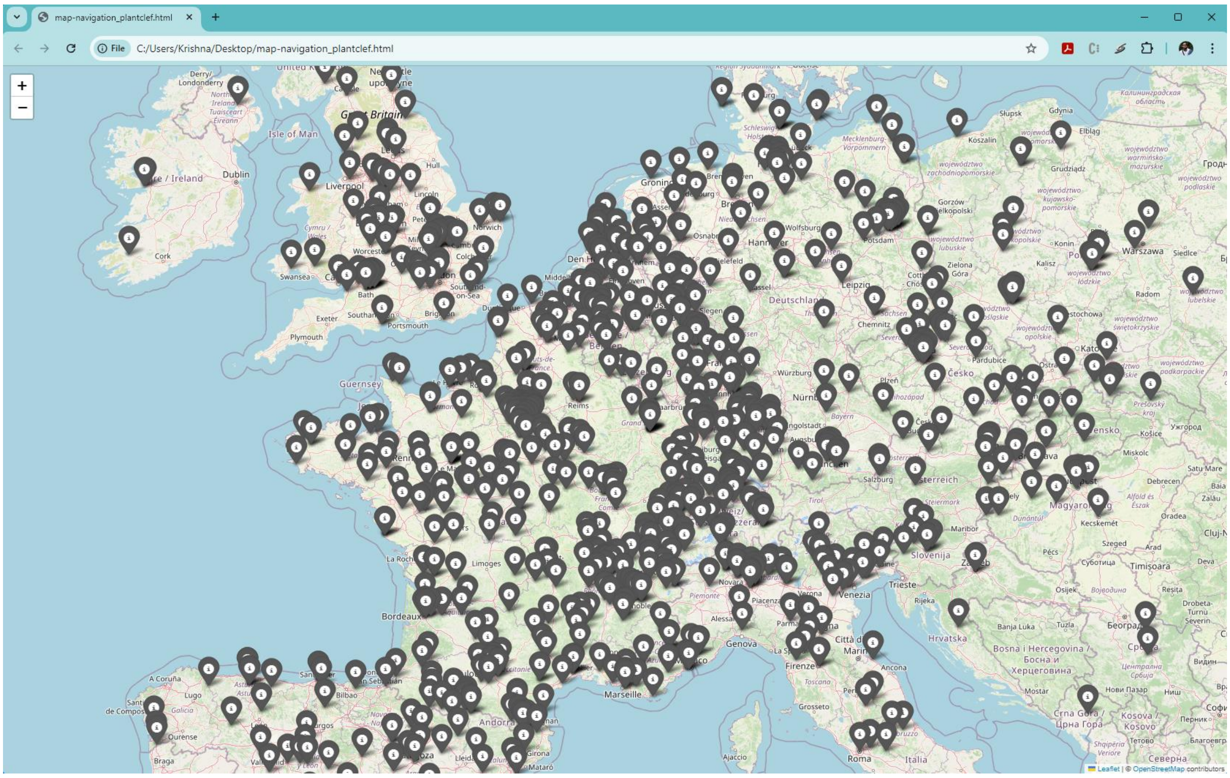


Figure 45: An overview of the tree species distributed across EU

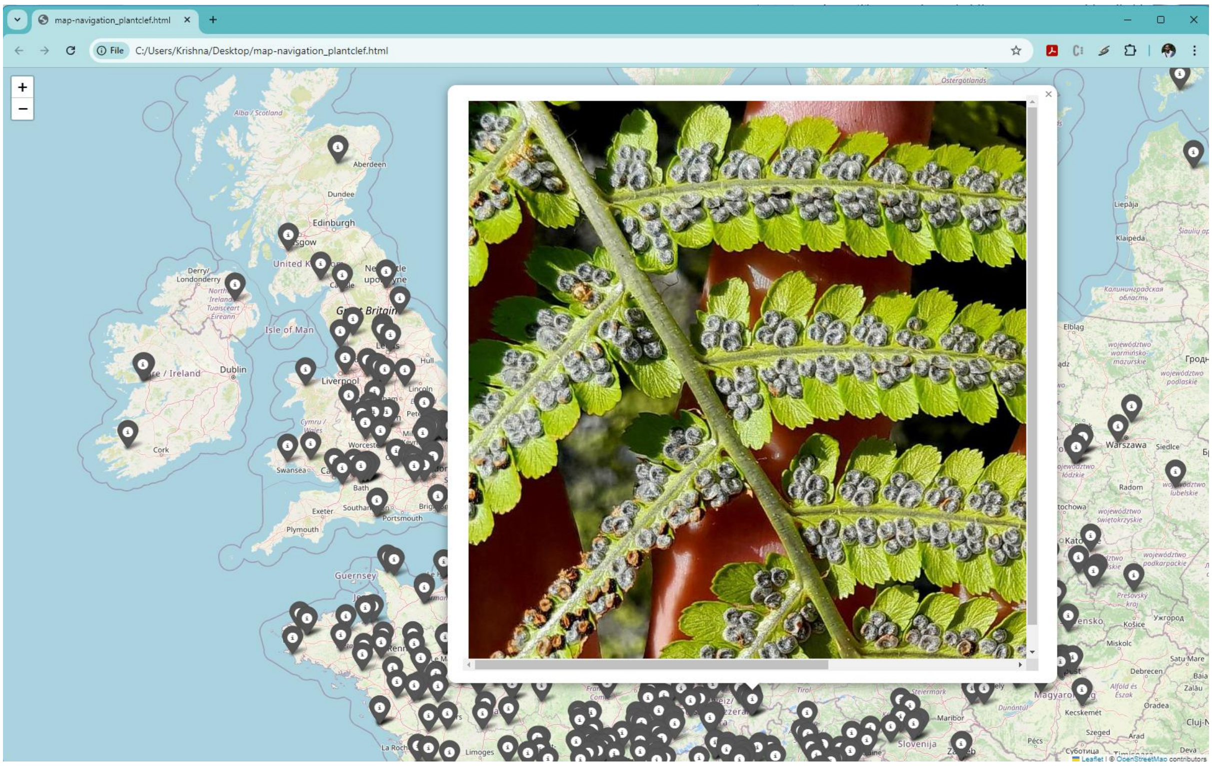


Figure 46: A view of the point data representing the specific tree species

7 Multilingual Forest Fire Alert System

7.1 Introduction

In recent years, the ubiquity of social media platforms has transformed the landscape of information dissemination, offering real-time insights into various societal phenomena, including natural disasters. Forest fires pose significant challenges due to their rapid spread and devastating impact on ecosystems and human settlements [59]. Traditional forest fire detection and monitoring methods often rely on satellite imagery and ground-based sensors, which may suffer from limitations such as latency and coverage gaps [60] [61]. Many approaches have investigated forest fire detection. The most used method to detect forest fire is a multi-sensor system that detects temperature, smoke, and CO2 controlled by an IoT system. The other method is using visuals or video by installing cameras in the forest areas at high fire risk. The remote sensing method is also quite popular with researchers. The satellite image from sentinel-1, sentinel-2, lansat-7, and lansat-8 is often used to detect fire events and burning areas. The last approach is using social media data to detect the fire event. The social media data is ingested by a system using each social media application programming interface (API) and a classification model to classify the text into two classes (fire event or non-fire event).

In research [62], several machine learning methods process the Canadian Fire Weather Index (FWI) to predict the fire event. The FWI contains climate and weather information, such as temperature, relative humidity, rain, etc. Random Forest (RF) can achieve 100% accuracy. Otherwise, the [63]result suggests that about 20% of the mapped forest disturbances can be linked to fire activity, which confirms a strong relationship between forest disturbances and fires in Riau, Indonesia. In [60] the satellite smoke scene dataset named USTC-SmokeRS was processed using a bidirectional feature aggregation module (BFAM) that can learn multi-level, multi-granularity features to achieve 98.33% accuracy. On [64]the size of the burned area is predicted using several machine learning methods, which are Neural Network (NN), Decision tree regression (DTR), Gaussian process regression (GPR), and support vector regression (SVR), bagging, and boosting. Rao et al. [65]processed incline (o), point, tallness (m), show up utilize, Normalized Difference Vegetation Index, detachment to way (m), partition to living plan region (m), heat (0C), wind speed (m/s) and precipitation (mm) into Wildfire Susceptibility Index. Based on the [66]the observed Burn Area (BA) spatial patterns were heterogeneous across the island, with 32 % of the grasslands burning annually, in contrast to other land cover types, such as the dense tropical forest, where <2 % burned every year. Xu et al. [67]found that Low spatial resolution of meteorological satellite images hinders accurate forest fire detection. The imbalance in individual samples affects the classification accuracy in detecting forest fires. In this research [68]Forest BA trends increased annually in the Middle Volga region. Coniferous forests were the most fire-prone, contributing to 59% of BA. Temperature had a more significant impact on BA than precipitation and wind. Lee et al. [69]use Forest-masking shapefile creation. Exponential analysis with Otsu algorithm. U-Net and HRNet DL models for forest fire detection.

This research [70]used mathematical modelling based on Reynolds equations for forest fire initiation and spread. Development of a fire spread model using the CA framework and Monte Carlo. In this research [71]IoT devices using LoRaWAN network architecture are based on a star topology with gateways. They were testing in a natural environment with network coverage and sensor operation. They found large plant masses that caused a large dispersion in the signal. Researchers [72]employed sensors like MQ135 and MQ3 for gas detection and utilized the NRF24L01 module for wireless data transmission. They found that WSN detects forest fires efficiently, alerting officials in real time. Sadia et al. [73]developed a LoRa-based system that detects forest fires using wireless networks and sensors. Real-time data transmission to the cloud for analysis and alerting authorities. Long-range communication, low power consumption, and early fire detection. Alvarez et al. [74]developed. Al-Dahoud et al. [75]developed a system to detect forest fires accurately with low false alarms. Real-time alerts are provided to forest authorities and users via mobile app. The proposed FogFire [76]model achieves 94% fire detection accuracy. Modified Greedy Forwarding outperforms RPL in IoT routing protocols.

The researchers [77] GoogLeNet-TL provides 96% accuracy and 97% F1 score in classifying forest fire images. The googLeNet-based TL model outperforms the MobileNet-based TL model. PSO quickly reduces the cost function to zero, which is suitable for forest fire classification. This research [78]uses IoT sensors for forest monitoring. Based on sensor data, the system does a real-time data transfer to cloud server and automated fire extinguishment. Anuar et al. [79]Developed a real-time forest fire detection, monitoring, and alert system. The system monitored temperature, humidity, and CO2 levels accurately in real time. The system provided an accurate location of the forest fire detection device. However, there is limited focus on sensor data analysis in existing forest fire systems, which lack direct connections to user monitoring and alert mechanisms. Jayasingh et al. [80]proposed a model that achieved 95.11% accuracy in detecting forest fires.

Hasan et al. [81]created a TwitterNews+ detected 30 out of 31 ground truth events. It achieved the highest precision of 0.89 among event detection systems. It outperformed baselines in detecting minor events with soft burst detection. The proposed system showed significant improvement in recall and precision. Saeed et al. [82]developed a Document Pivot and Feature Pivot techniques for event detection. Unsupervised approaches like graph-based event detection using Twitter and Sina Weibo. Papadimos et al. [83], using Kernel Density Estimation for Density (KDECD) outperforms STALTA and Z-Score with 0.9589 accuracy. It was used on the Spanish fire-related tweets dataset in 2019. However, Fixed search terms may lead to insufficient or excessive tweets. Bozas et al. [84]developed Disaster event detection, user identity linkage, relevance classification, and community detection. However, there is a lack of ground truth datasets for user account identification. It is also under-exploited by multimedia content due to retrieval challenges. Almeida et al. [85]proposed a method that achieved 98.71 accuracy and 98.97 F1-score. The classification time per sample is 30 ms for Tesla T4.

Akyol [86]Bi-LSTM network achieved 97.37% accuracy in fire detection in image data. The proposed model showed improved classification performance using pre-trained CNN models. The stacking ensemble learning model offers high test accuracies. Mondal et al. [87]successfully created a fire detection algorithm with an accuracy of 95.26% with a 91.61% true positive rate. It improved precision to 7.95% and accuracy to 9.43% over existing algorithms. Vikram Sinha [88]created a Neuro-fuzzy model tested on sensor data. CNN model evaluated on image dataset. Integrated model performance analyzed for sensor and image data. However, the existing models lack image data, affecting fire detection accuracy. There is limited discussion on the scalability and real-time implementation challenges. Verma Bakthula [89]used a Fusion model strategy that combines Inceptionv3, MobileNetV2, and ResNet50v2 features. Fused models achieved 94.33% accuracy, outperforming individual models significantly. Akyol [90]successfully implemented the DNN-3 classifier, achieving 97.11% accuracy on ResNet50 deep features. DNN-3 classifier had 96.84% sensitivity and a 3.16% false negative rate. NB and SVM showed high classification accuracies of 92.89. Paidipati et al. [91]used the FFDNet model and excelled in distinguishing fire and non-fire images. FFDNet algorithm showed improved performance with increased TACC and VACC. FFDNet technique achieved maximum TACC outcomes.

Our contribution lies in developing a novel system that harnesses the vast volume of user-generated content on platforms like X to provide nearly real-time early warnings to stakeholders, facilitating timely response and mitigation efforts. The detailed contributions are as follows:

1. We develop extraction and analysis of multilingual social media data streams to detect forest fire incidents.
2. We utilize advanced natural language processing (NLP) techniques, such as sentiment analysis, keyword extraction, and geolocation tagging, to identify relevant posts about forest fire incidents.
3. We use the Bidirectional Encoder Representations from Transformers (BERT) based method to detect the type of language, detect a wildfire, and extract location.
4. We provided fire probability information from a data fusion application to ensure that fire probability data corroborate the occurrence of fire events.

This report presents a pioneering effort in leveraging multilingual social media data and decision support systems for early forest fire detection. By integrating advanced NLP techniques and machine learning algorithms, we aim to enhance the efficiency and effectiveness of forest fire monitoring and response strategies, ultimately contributing to preserving natural ecosystems and protecting human lives and property.

7.2 Methods

By harnessing the vast volume of user-generated content on platforms like X, our system aims to provide nearly real-time early warnings to stakeholders, facilitating timely response and mitigation efforts. The core functionality of our proposed system centers around the extraction and analysis of multilingual social media data streams. Through advanced natural language processing (NLP) techniques, including sentiment analysis, keyword extraction, and geolocation tagging, we aim to identify relevant posts indicative of forest fire incidents. This process is augmented by BERT method trained on annotated datasets to improve the accuracy and reliability of fire detection. Furthermore, our decision support system integrates additional contextual information, such as fire probability. We aim to provide stakeholders with comprehensive situational awareness and actionable insights. A key aspect of our approach is the support for multilingual capabilities, enabling effective fire detection and localization in linguistically diverse regions. This is achieved through the development of language-agnostic NLP models and geolocation algorithms capable of processing textual content in multiple languages.

The process of system is following:

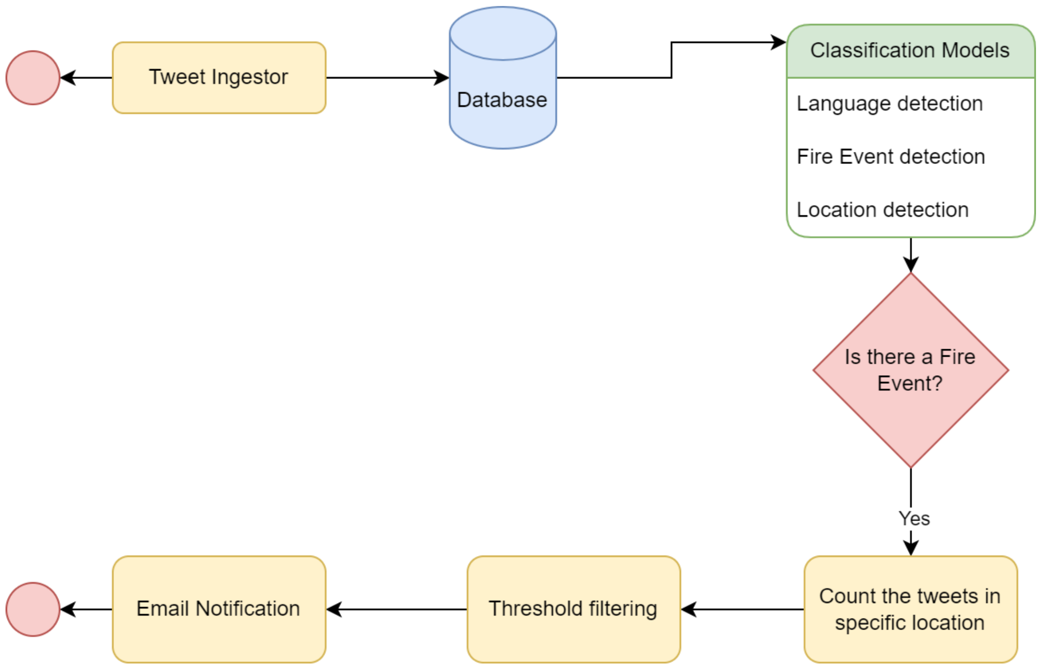


Figure 47: System Architecture

The process of system is following:

1. The tweets are collected via API every 6 minutes.
2. The tweets are included in the database.
3. The classification models are implemented sequentially.
4. If there is a fire event, then the tweets will be counted incrementally for specific location.
5. Each user has various tweet number thresholds based on the urgency of each occupation. For example, the fire fighter has the lowest tweet threshold as the front liner of the disaster mitigation.

6. When the number of tweets in each 6 minutes cross over the threshold then the email notification script will be triggered, and the email will be delivered to the user.

The workflow of the proposed method for multilingual named entity recognition model for location and time extraction of forest fire tweets can be seen in Figure 47.

7.2.1 Dataset

The common_language dataset is a multilingual public dataset that includes a variety of languages without slovak. However, in this study we only used 4 languages, namely Indonesian, English, Spanish, and Italian. It is designed to facilitate language detection tasks and contains text samples from diverse sources, providing a rich resource for training language models. Sample data from the common_language dataset can be seen in Table 5.

Table 5: Sample of common_language dataset

No	*Text*	*Language*
1	*Aku baik-baik saja.*	Indonesian
2	*It is a charity school whose fees are calculated on a means test.*	English
3	*Es ante todo una fuente mitológica.*	Spanish
4	*Conosce Renato Fucini, Giovanni Marradi, Mario Rapisardi e comincia ad occuparsi di studi marxisti.*	Italian

The common_lang dataset consists of 3 subsets with the amount of data for each set being 1,771 in the training set, 433 in the validation set and 442 in the testing set. The distribution of data for each language in the common_lang dataset can be seen in Table 6.

Table 6: Data distribution of Indonesian, English, Spanish and Italian

No	*Language*	*Train Set*	*Validation Set*	*Test Set*
1	Indonesian	595	166	150
2	English	414	79	98
3	Spanish	389	96	94
4	Italian	373	92	100

In addition, we also use the Kiviki/SlovakSum dataset for the Slovak language. The kiviki/SlovakSum dataset is focused on providing textual data in the Slovak language, particularly summaries of various topics. This dataset is valuable for fine-tuning language models to recognize and process Slovak text. Sample data from the kiviki/SlovakSum dataset can be seen in Table 7.

Table 7: Sample of kiviki/SlovakSum dataset

No	*Text*	*Language*
1	*Provincia Idlib v severozápadnej Sýrii je poslednou baštou povstalcov.*	Slovak

The kiviki/SlovakSum dataset consists of 3 subsets with the amount of data for each set being 165,298 in the training set, 20,663 in the validation set and 20,662 in the testing set. However, we curated all three subsets to balance them with the common_language dataset. As a result, the amount of data in each set is 414 in the training set, 69 in the validation set and 69 in the testing set. Next, we combined the common_language and kiviki/SlovakSum datasets and created a data distribution as shown in Table 8.

Table 8: Data distribution of language detection model

No	*Language*	*Train Set*	*Validation Set*	*Test Set*
1	Indonesian	595	166	150
2	English	414	79	98
3	Spanish	389	96	94
4	Italian	373	92	100
5	Slovak	414	69	69
	Total	2,185	502	511

The forest fire tweet dataset was obtained using scraping techniques from Twitter to collect tweet data in Indonesian based on the keyword "*kebakaran hutan*" on November 22nd, 2023. The data collected was 2,330 tweet data. Data labeling was done manually with two classes, namely fire and not fire. The fire class means that the tweet represents the occurrence of a forest fire in a location. Meanwhile, the not fire class means the opposite. As a result, the data distribution in the fire and non-fire classes was 660 and 1,204 respectively. Table 9 displays the results of scraped and labeled tweet data.

Table 9: Sample of labeled tweets

No	*Tweet*	*Language*	*Class Label*
1	*Kebakaran hutan dan lahan di Gunung Penanggungan, Kabupaten Mojokerto, Jawa Timur, menjalar ke Gunung Sarang Kelapa dan Puncak Bayangan.*	Indonesian	Fire
2	*Green forest fire alert in Australia: On 18/11/2023, a forest fire started in Australia, until 20/11/2023.*	English	Fire
3	*: Incendio Forestal Cerró Domingo Ortiz de Rosas. Bomberos en el lugar. Noticia en desarrollo .*	Spanish	Fire
4	*Ancora incendi a Palermo. Vigili del fuoco e forestali impegnati a Monte Pellegrino e anche in provincia*	Italian	Fire
5	*Ostrov Madeira sužujú lesné požiare. Ohrozujú aj turistov*	Slovak	Fire

Nergrit Corpus is a public dataset created by PT. Gria Innovation Technology (GRIT) from Indonesian news site articles. This dataset was specifically created for general-purpose natural language processing tasks such as named entity recognition with the BIO (Beginning-Inside-Outside) annotation format, where B indicates the beginning of an entity, I indicate part of an entity, and O means not an entity (non-entity). Data annotation in this dataset was done manually, involving several annotators. Furthermore, the sentences in this dataset are composed of alphabetical, numeric, and special characters that represent the source of the original news, while emoticons and links have been programmatically eliminated. Examples of data in the Nergrit Corpus dataset are shown in Table 10. This dataset is used in order to investigate its effectiveness for forest fire domain purposes.

Table 10: Sample data in the Nergrit Corpus dataset

Token	Martahan	Sohuturon	,	CNN	Indonesia	...	8	:	20	WIB
Label	B-PER	I-PER	O	B-ORG	I-ORG	...	B-TIM	I-TIM	I-TIM	I-TIM

The Nergrit Corpus dataset consists of 17,452 data which are separated into training, test, and validation data of 12,532, 2,399, and 2,521, respectively. The dataset also has 19 entity labels, including Cardinal (CRD), Date (DAT), Event (EVT), Facility (FAC), Geopolitical Entity (GPE), Legal Entity (LAW), Location (LOC), Money (MON), Organization Politics (POL), Ordinal (ORD), Organization (ORG), Person (PER), Percent (PRC), Product (PRD), Quantity (QTY), Religion (REG), Time (TIM), Work of Art (WOA), and Language (LAN). Figure 48 shows the distribution of entity labels in the Nergrit Corpus dataset.

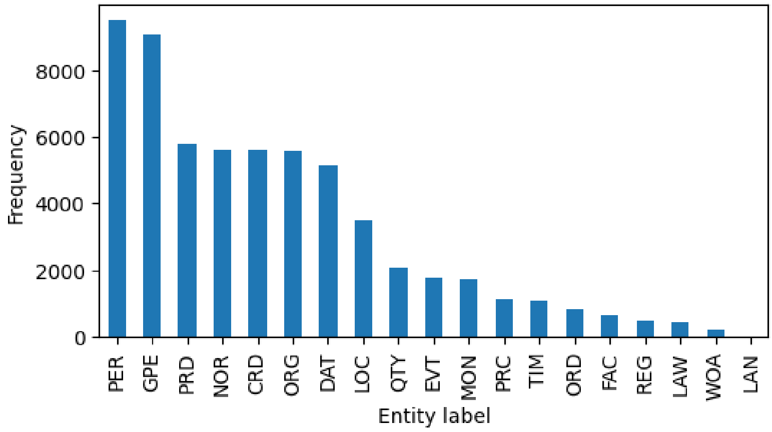


Figure 48: Distribution of entity labels in the Nergrit Corpus dataset

To extract location and time from forest fire tweets, we use four related entity labels: Location (LOC), Geopolitical Entity (GPE), Date (DAT), and Time (TIM), which represent the location of non-administrative areas (e.g., name of mountain, forest, park, road, etc.), location of administrative areas (e.g., name of country, province, district, village, etc.), date indicator, and time indicator.

We merge the Location (LOC) and Geopolitical Entity (GPE) entity labels into Location (LOC) exclusively to obtain comprehensive data on forest fire locations, considering the locations of both administrative and non-administrative areas. Meanwhile, Date (DAT) and Time (TIM) entity labels are maintained to obtain time information separately. We changed the remaining 15 entity labels to non-entity (O) because they were no longer needed. Figure 49 and Figure 49 show the distribution of entity labels and sample data after data labeling.

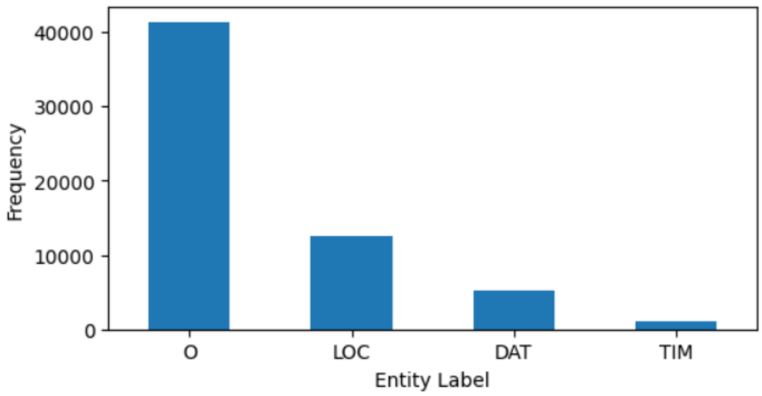


Figure 49: Distribution of entity labels after data labeling

Next, we performed data cleaning on the Nergrit Corpus dataset, eliminating multiple rows of tokens labeled as non-entity (O), created during the data labeling phase. We do this to prevent tokens that share the same context in a sentence but have different entity labels Table 11. After the data cleaning process, the proportion of data in the dataset now amounts to 9,090, with the proportion of train, test, and validation data being 6,611, 1,228 and 1,251, respectively.

Table 11: Sample data after data labeling

Token	Martahan	Sohuturon	,	CNN	Indonesia	...	8	:	20	WIB
Label	O	O	O	O	O	...	B-TIM	I-TIM	I-TIM	I-TIM

7.2.2 Preprocessing

The data collected is then preprocessed to clean the data from noise that can appear in tweet data such as hashtag symbols, user mentions, emoticons and URLs. However, we retain punctuation so that large-scale language models can get a better representation of sentence context and reduce ambiguity between one sentence and another. Table 12 shows the results of data preprocessing.

Table 12: Data Preprocessing Results

No	*Tweet*	*Language*	*Class Label*
1	*Kebakaran hutan dan lahan di Gunung Penanggungan, Kabupaten Mojokerto, Jawa Timur, menjalar ke Gunung Sarang Kelapa dan Puncak Bayangan.*	Indonesian	Fire
2	*Green forest fire alert in Australia: On 18/11/2023, a forest fire started in Australia, until 20/11/2023.*	English	Fire
3	*: Incendio Forestal Cerró Domingo Ortiz de Rosas. Bomberos en el lugar. Noticia en desarrollo .*	Spanish	Fire
4	*Ancora incendi a Palermo. Vigili del fuoco e forestali impegnati a Monte Pellegrino e anche in provincia*	Italian	Fire
5	*Ostrov Madeira sužujú lesné požiare. Ohrozujú aj turistov*	Slovak	Fire

We randomly separated the data into three subsets, namely training, validation and test sets. The distribution of data in each subset is 80% for training, 10% for validation and 10% for model testing. Train and validation data are used to fine-tune the model, while test data are used to evaluate the model. As a result, the amount of data in the training, validation and testing sets is 1864, 233, and 233 respectively.

7.2.3 Fine Tuning

Fine-tuning refers to the process of taking a pre-trained language model and adapting it to a specific downstream task, such as text classification. This involves training the pre-trained model on a labeled dataset text classification task while retaining the learned parameters from the initial pre-training phase. By leveraging the vast amounts of linguistic knowledge encoded in the pre-trained model, fine-tuning allows for efficient and effective adaptation to specialized tasks, typically requiring significantly less data and computational resources than training a model from scratch. This technique has been proven to enhance performance and generalization across various NLP applications. In this study, we fine-tuned the XLM-RoBERTa pre-trained model using a labeled dataset for forest fire tweet classification.

XLM-RoBERTa is a transformer-based language model designed to handle multiple languages effectively by leveraging a vast multilingual corpus. It was pre-trained on the CommonCrawl corpus (https://commoncrawl.org/), which contains 2.5 TB of text data in 100 languages, making it one of the largest and most diverse corpora used for pre-training. The model employs byte-pair encoding (BPE) for tokenization, resulting in a vocabulary of 250,000 tokens that can represent a wide array of linguistic features across different languages. XLM-RoBERTa comes in several configurations, with the base model consisting of 12 transformer layers, 768 hidden units per layer, and 12 attention heads, totaling approximately 270 million parameters. This extensive pre-training and robust architecture enable XLM-RoBERTa to capture nuanced language patterns and semantics, facilitating superior performance in various cross-lingual NLP tasks. Figure 50 displays the fine-tune model architecture of the XLM-RoBERTa pre-trained model.

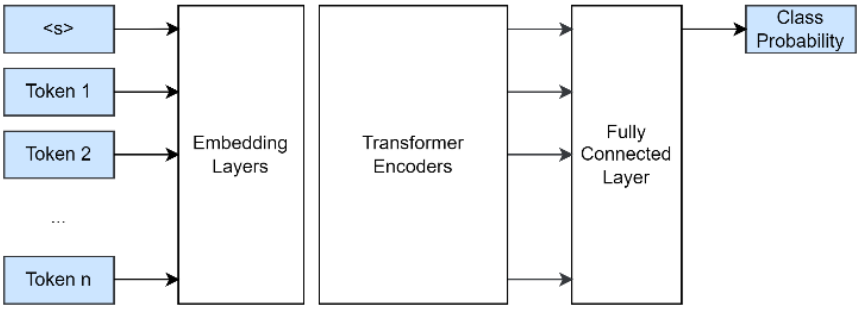


Figure 50: XLM-RoBERTa model architecture

Figure 50 depicts the technique for fine-tuning a model based on BERT architecture. The token sequence contains snippets of words from a sentence. To format this sequence, the special token <s> is used to indicate the beginning of sentences. The process of embedding inputs involves two methods: token embedding, which converts tokens into ids and then into vectors by retrieving the embedding value from a pre-trained word embedding matrix, and positional embedding, which incorporates positional information into the token representation. Next, input the data into the transformer encoders to obtain a contextual embedding for each word. This encoder captures the context of each word using multiple attention heads in each layer. The fully connected layer simultaneously receives the data to determine the class probability of fire versus non-fire. The highest-class probability appears as a predicted result.

Fine-tuning involves adapting a pre-trained model to a specific downstream task or domain. The pre-trained model has undergone extensive training on a vast amount of text data to acquire a deep understanding of language representations. Although pre-trained multilingual models such as mBERT and XLM-R possess extensive language knowledge, they may lack specialization in specific domains or tasks as shown in Figure 51. Consequently, a transfer learning approach such as fine-tuning can significantly enhance their performance for tasks, like recognizing location and time entities in forest fire tweets. Through the process of fine-tuning, the model can utilize the knowledge and representations it acquired during pre-training by training it on a smaller labeled dataset. This knowledge transfer improves the performance of a fine-tuned model, greatly reduces the amount of effort required, and is relatively inexpensive. Moreover, the fine-tuned model exhibits the capacity to effectively apply learned knowledge to unseen data, even across multiple languages.

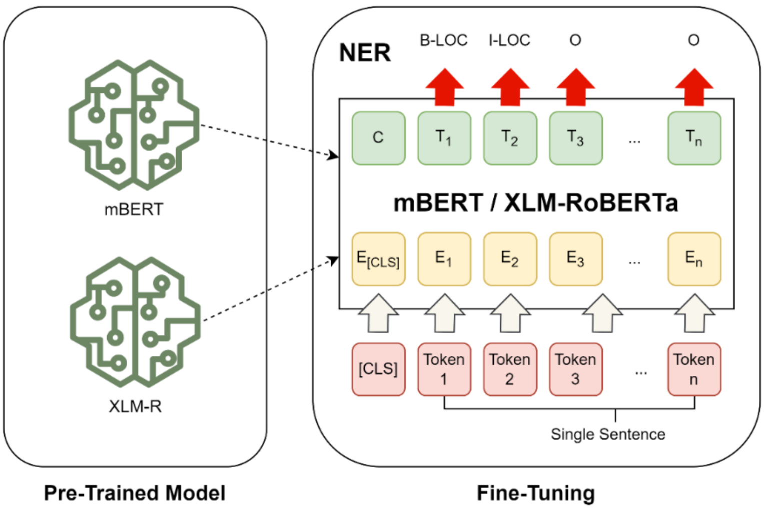


Figure 51: The mBERT and XLM-R fine-tuning

A token sequence contains snippets of words from a single sentence, is defined as an input sequence, and is represented as $\{\text{Token 1}, \text{Token 2}, \ \ldots \ \text{Token n}\}$. To format this token sequence, special tokens such as [CLS] are used to indicate the beginning of sentences. To represent textual input data, mBERT and XLM-R use three types of embeddings: positional embedding, to represent the position of each token in the input sequence; token type embedding or segment embedding, to determine whether a sentence follows another sentence logically; and token embedding, to convert the input sequence into a list of token IDs. The embedding layers are represented as $\{E_{[CLS]}, E_1, E_2, E_3 \ldots \ E_n\}$. Next, transformer encoders capture the context of the input sequence through multiple attention heads in each layer and represent it as $\{T_1, T_2, T_3 \ldots \ T_n\}$. Simultaneously, the input sequence is also passed to the fully connected layer to obtain the output for each token in the BIO annotation scheme.

7.3 Implementation

The implementation of the Multilingual Forest Fire Alert System (DSS-MFAS) involves a comprehensive integration of data streaming, processing, and analysis components designed to provide real-time alerts for forest fire events. The system harnesses the power of social media platforms, specifically leveraging the vast amount of user-generated content on platforms like X (formerly known as Twitter), to detect and monitor fire incidents. Data is continuously streamed from social media through APIs, processed by advanced natural language processing (NLP) models, and stored in a robust backend infrastructure. The processing pipeline includes several stages, from keyword searching and data retrieval to classification, geolocation tagging, and notification dispatch as shown in Figure 52. By implementing a highly scalable and fault-tolerant architecture, DSS-MFAS ensures timely and accurate detection of forest fire events, facilitating immediate response and mitigation efforts.

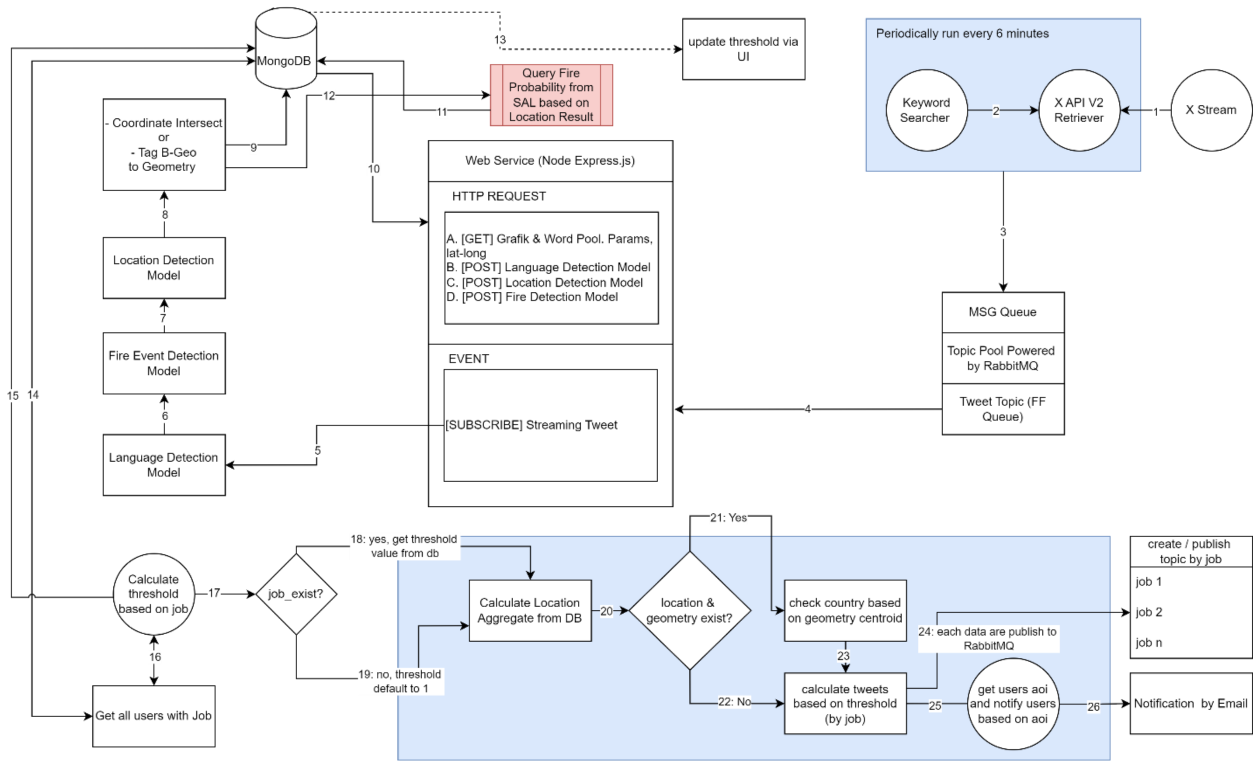


Figure 52: Multilingual Forest Fire Alert System (DSS-MFAS) Flow

The detailed step-by-step process is explained for each of the 26 arrows, representing the sequence from beginning to end. The following subsections will elucidate each process in the order illustrated in Figure 52.

X Stream to X API V2 Retriever

The X Stream module for tweet ingestion is crucial in the data processing pipeline, ensuring the initial reception and handling of tweet data from various sources in real-time as shown in Figure 53. The X API V2 Retriever captures this data efficiently, performing validation to maintain data integrity, format correctness, and completeness. It also carries out pre-processing tasks like data cleaning, normalization, and metadata extraction, ensuring consistency and organization. The processed tweet data is temporarily stored in buffers or queues to regulate flow and is then routed to downstream systems for further analysis or storage. The system continuously monitors data flow for anomalies, scales horizontally to handle increasing volumes, and maintains fault tolerance for uninterrupted operation, ensuring high performance, reliability, and accuracy for subsequent data processing.

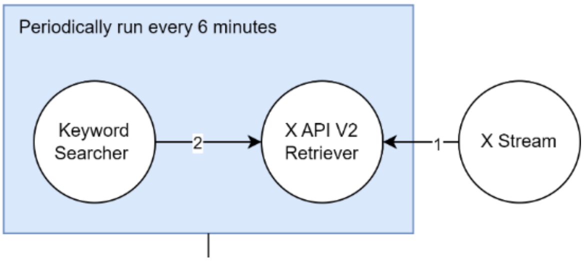


Figure 53: The X Stream, X API V2 Retriever, and Keyword Searcher

Keyword Searcher to X API V2 Retriever

The Keyword Searcher module, designed to run every 6 minutes, ensures the system remains updated with the latest data for timely insights. During each run, it performs comprehensive searches across various data sources to identify specific predetermined keywords related to wild forest fire. It scans the tweet database that accurately reflects the occurrence of the keywords. This data is then sent to the X API V2 Retriever for further processing, where it undergoes validation, cleaning, normalization, and metadata extraction as illustrated in Figure 53. The integration between the Keyword Searcher module and the X API V2 Retriever allows the system to handle large volumes of data efficiently. By running continuously, the module provides an up-to-date stream of relevant data, essential for real-time analysis and decision-making.

X API V2 Retriever to MSG Queue, Topic Pool Powered by RabbitMQ, and Tweet Topic

The X API V2 Retriever processes data from the X Stream and Keyword Searcher module, which provide real-time data and targeted keyword searches, respectively. Upon receiving data, the retriever validates it for integrity and completeness, performs pre-processing tasks like cleaning and normalization, and extracts metadata. The processed data is sent to the MSG Queue to manage flow and prevent bottlenecks. Additionally, the Topic Pool powered by RabbitMQ organizes topic-specific data, with the Tweet Topic (FF Queue) handling categorized tweets as displayed on Figure 54. This integration ensures efficient, scalable, and contextually relevant data processing for real-time analysis and decision-making.

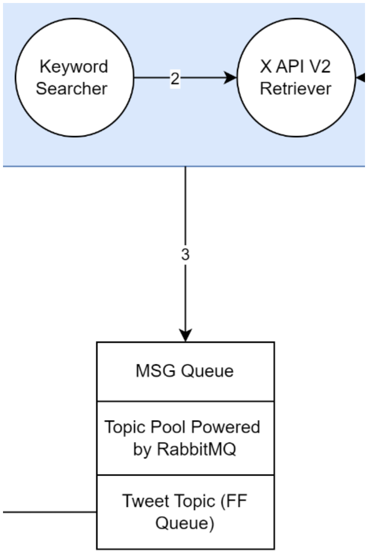


Figure 54: MSG Queue, Topic Pool, and Tweet Topic

MSG Queue, Topic Pool Powered by RabbitMQ, and Tweet Topic to [SUBSCRIBE] Streaming Tweet

The transition from Tweet Topic to [SUBSCRIBE] Streaming Tweet involves subscribing to specific tweet streams based on predetermined topics as displayed on Figure 55. Once tweets are categorized in the Tweet Topic (FF Queue), the system subscribes to these categorized streams for continuous monitoring and analysis. This subscription mechanism ensures that the system remains updated with the latest tweets related to each topic, enabling timely insights and responses.

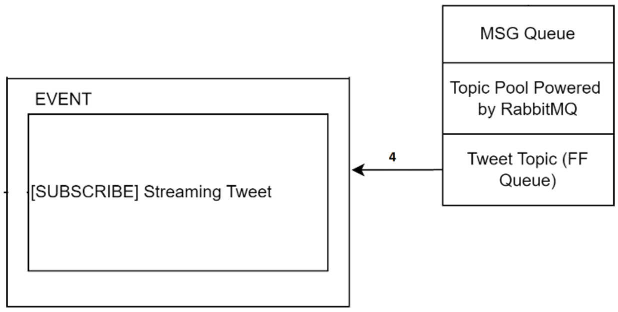


Figure 55: [Subscribe] Streaming Tweet

[SUBSCRIBE] Streaming Tweet to Language Detection Model

The Web Service Streaming Tweet component subscribes to the Language Detection Model to significantly enhance its processing capabilities as illustrated in Figure 56. This subscription process involves establishing a continuous and robust connection with the language detection model, facilitating seamless integration and real-time communication between the two components. This sophisticated setup ensures that each tweet processed by the Web Service Streaming Tweet component is instantly analyzed for language detection, providing immediate and accurate language identification. By leveraging the model, the system can efficiently handle a high volume of tweets, maintaining real-time processing speed and improving overall accuracy in language analysis, thereby enriching the quality and relevance of the data processed.

Language Detection Model to Fire Event Detection Model

Once the language of each tweet is detected, the data is forwarded to the Fire Event Detection model. This model classifies the tweet as either a fire event or a non-fire event. Tweets identified as non-fire events are disregarded, while those classified as fire events are passed to the Location Detection model for further analysis as shown in Figure 56.

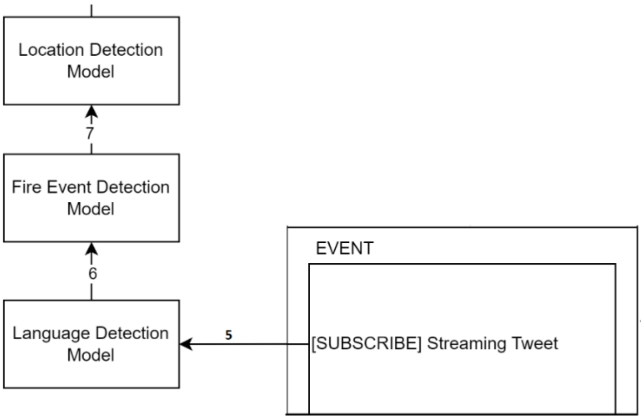


Figure 56: Classification Model

Fire Event Detection Model to Location Detection Model

After the tweets are classified into fire and non-fire event, the classified data is passed to the **Location Detection Model as illustrated in Figure 56**. The Location Detection model then determines the specific location of the reported fire, enabling a rapid and accurate response to potential fire incidents. This multi-step process ensures that only relevant tweets are acted upon, improving the efficiency and effectiveness of fire detection and response efforts. The Location Detection Model is built using NER approach. It employs advanced natural language processing techniques to meticulously scan the tweet content, extracting and tagging these entities with high precision.

Location Detection Model to Coordinate Intersect or Tag B-Geo to Geometry

Named entities tagged with B-Geo (geographical identifiers) undergo a critical intersection process with coordinates to create detailed geometrical representations of the data. This process involves mapping the identified geographical entities to their corresponding latitude and longitude coordinates, effectively transforming textual mentions of locations into precise spatial data as shown in Figure 57. By integrating these geographical tags with exact coordinates, the system generates accurate geometrical representations that can be visualized on maps or used in spatial analysis. This fusion of textual and spatial data enriches the dataset, enabling a comprehensive understanding of the geographical distribution and context of the tweets.

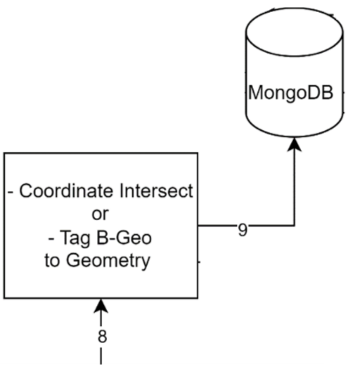


Figure 57: Coordinate Intersect and MongoDB

Coordinate Intersect or Tag B-Geo to Geometry to MongoDB

Storing geometrical data points in a MongoDB database is crucial for efficient retrieval and analysis of spatial information. Data points, including coordinates and shapes, are collected from sources like GPS devices and sensors, and organized into MongoDB documents with metadata such as timestamps as shown in Figure 57. These documents are stored in collections, with indexes on key fields to enhance search speed. MongoDB's replication and sharding features ensure high availability and scalability, distributing data across servers for redundancy and handling large datasets. The database supports efficient retrieval through geospatial queries and the aggregation framework, as well as full-text search. Stored data can be analyzed using geospatial analytics, machine learning models, and visualization tools to predict patterns, optimize routing, and create interactive maps.

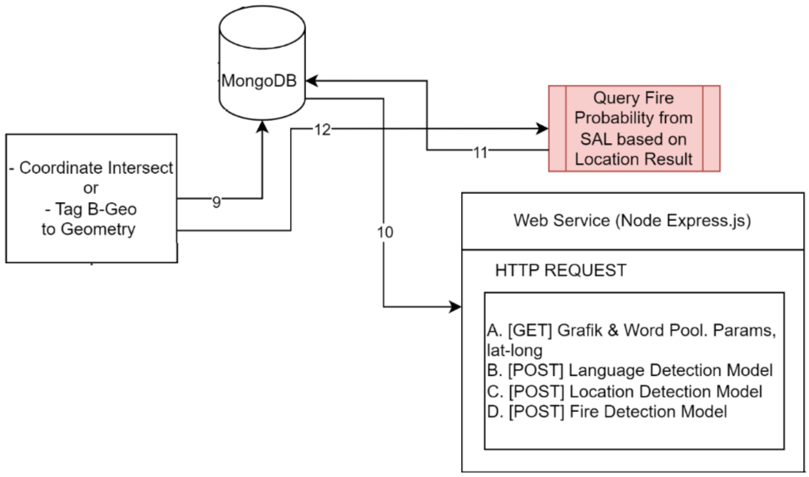


Figure 58: MongoDB to Web service and Query Fire Probability from SAL based on Location Result and Web Service

MongoDB to Web Service (Node Express.js)

The Node Express.js web service facilitates interaction with MongoDB data through well-defined API endpoints as shown in Figure 54. It enables efficient data retrieval and posting, ensuring seamless integration between the web application and the database. Key functionalities include the [GET] Grafik & Word Pool endpoint (/getGrafikWordPool), which retrieves graphical data and word pools based on latitude and longitude, and the [POST] Classification endpoint (/postClassification), which allows users to submit and store classification data. Additional endpoints include [POST] NER (/postNER) for named entity recognition data and [POST] Language Detector (/postLanguageDetector) for language detection data. The service involves defining routes, using middleware for request parsing and error handling, connecting to MongoDB with pooling and error management, and ensuring security through input validation and access control. Standardized responses with clear status codes and messages provide precise feedback to clients.

Query Fire Probability from SAL Based on Location Result to MongoDB

The process involves querying the fire probability from the SILVANUS Storage Abstraction Layer (SAL) based on location results stored in MongoDB as shown in Figure 54. By integrating location-specific data retrieved from MongoDB, the system can accurately assess and compute the probability of fire occurrences. This interaction leverages the comprehensive data storage capabilities of MongoDB, ensuring that the location results are utilized effectively to enhance the situational awareness and fire risk prediction provided by SAL. Consequently, this process enables more precise and data-driven decision-making for fire detection and prevention strategies.

Coordinate Intersect or Tag B-Geo to Geometry to Query Fire Probability from SAL Based on Location Result

The coordinates extracted from the tweet are passed to the query system, which retrieves the corresponding fire probability links stored in the Situational Awareness Layer (SAL) as shown in Figure 58. This fire probability data is provided by the Silvanus Data Fusion App, ensuring accurate and up-to-date information. By linking tweet coordinates with fire probability data from SAL, the system enhances its ability to predict and respond to potential fire incidents, utilizing the sophisticated data integration capabilities of the Silvanus Data Fusion App for improved situational awareness and decision-making.

MongoDB to "Update Threshold via UI"

Updating threshold values stored in MongoDB via a user interface (UI) is essential for effective system management. The process begins with a user-friendly UI that displays current threshold values fetched from MongoDB, allowing users to review and adjust settings through intuitive input fields, sliders, or selection options. Users make changes using these mechanisms, and the UI validates the new values to ensure they are acceptable. Upon submission, the backend system processes the request, updates the MongoDB documents, and confirms the success of the update to the UI. The UI then refreshes to show the new values. The backend also performs validation to ensure the new values meet criteria, handling errors gracefully and providing user guidance. This method offers several advantages, including ease of use, real-time updates, accuracy through input validation, and improved efficiency in managing and controlling system parameters, ultimately enhancing monitoring, control, and user satisfaction.

MongoDB to Get All Users with Job

Retrieving job-related information from MongoDB involves several steps to ensure accurate and effective data use. The process begins by establishing a secure connection to MongoDB using appropriate credentials and a client library. After connecting, the specific database containing user information is selected. Next, the relevant collection, such as "users" or "employee_records," is identified, and a query is formulated to retrieve documents with job-related details, such as job titles and departments. Executing this query fetches the required documents, which are then processed for formatting, aggregation, or additional filtering. Error handling addresses any issues like connection failures or query errors, while data security measures ensure compliance with privacy regulations. The processed data can be used for generating reports, updating records, or performing analytics, such as payroll processing or organizational trend analysis.

Calculate Threshold Based on Job to MongoDB

Calculating and updating threshold values in MongoDB based on job data involves several key steps. Initially, relevant job data, such as job titles and performance metrics, is retrieved from MongoDB by executing a query. This data is then analyzed using methods like statistical calculations, trend analysis, and, in some cases, predictive modeling to determine appropriate threshold values. Once the new threshold is calculated, it is updated in MongoDB by connecting to the database, selecting the correct collection, and executing an update query. Validation checks ensure the update is applied correctly, and subsequent verification confirms the accuracy of the new value. Finally, documenting and communicating the updated threshold values is essential for transparency and effective system management.

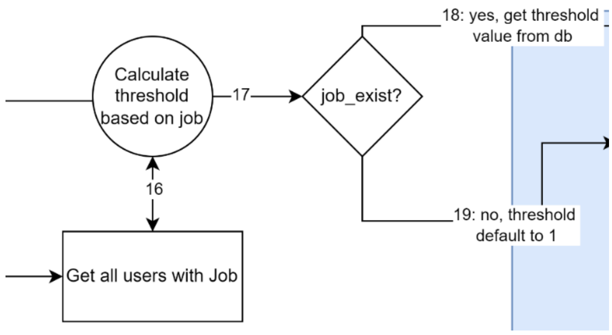


Figure 59: Calculate threshold based on job, get all used with Job, and Job_Exist

Calculate Threshold Based on Job to Get All Users with Job

Calculating and updating threshold values in MongoDB based on job data involves retrieving relevant job attributes from the database, such as job titles and performance metrics, by executing a query as shown in Figure 59. The retrieved data is analyzed using statistical methods, trend analysis, and, if needed, predictive modeling to determine new threshold values. Once calculated, the new threshold is updated in MongoDB by connecting to the database, selecting the appropriate collection, and executing an update query. Validation checks ensure the update is correct and meets required constraints. After updating, the new threshold value is verified for accuracy, and testing may be conducted to ensure it functions properly. Finally, documenting and communicating the changes ensures transparency and informs stakeholders, supporting effective system management and decision-making.

Calculate Threshold Based on Job to Job Exist

The system performs a crucial validation step by checking if a job exists based on the calculated threshold. This involves comparing the job data against predefined threshold criteria to determine whether the job meets the necessary conditions for further action. The calculated threshold serves as a benchmark, representing the minimum or optimal parameters required for the job to be considered valid or actionable. During this validation process, the system meticulously examines various job attributes, such as performance metrics, completion status, or resource utilization, against the threshold values as illustrated in Figure 59. If the job meets or exceeds these thresholds, it is flagged as existing and eligible for subsequent processing or analysis. Conversely, if the job falls below the threshold, it may be excluded from further consideration, ensuring that only relevant and significant jobs are prioritized. This step is essential for maintaining the efficiency and accuracy of the system, as it filters out inconsequential or incomplete jobs, thereby optimizing resource allocation and focusing on jobs that meet the established standards. By leveraging calculated thresholds, the system ensures that all processed jobs align with the desired performance and quality metrics, supporting robust decision-making and operational excellence.

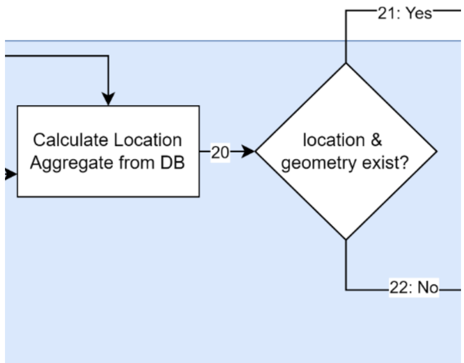


Figure 60: Calculate Location Aggregate from DB and Location and geometry exist

If Yes, Get Threshold Value from DB and Send to Calculate Location Aggregate from DB

If the job exists, retrieves the threshold value from the database and sends it to the module responsible for calculating the location aggregate.

If No, Threshold Defaults to 1 and Send to Calculate Location Aggregate from DB

If the job does not exist, defaults the threshold value to 1 and sends it to the location aggregate calculation module.

Calculate Location Aggregate from DB to Location & Geometry Exist

The system undertakes a comprehensive process to calculate location aggregates from the database and verify the existence of location and geometry data as displayed in Figure 60. This process begins with querying the database to retrieve relevant geographical information, which includes location coordinates and associated geometrical shapes such as polygons, circles, and lines. Once the data is gathered, the system performs aggregation calculations to compile summary statistics or collective metrics that represent the spatial distribution and characteristics of the locations. These aggregates might include metrics like the average density of points within a region, the total area covered by certain geometries, or the frequency of specific location tags. Concurrently, the system checks for the presence and completeness of location and geometry data, ensuring that all necessary spatial information is available and accurately recorded. This verification step is critical, as it identifies any gaps or inconsistencies in the data that could affect subsequent analysis or decision-making processes. By meticulously calculating location aggregates and confirming the existence of robust location and geometry data, the system enhances the reliability and depth of spatial analysis, facilitating informed insights and strategic planning based on comprehensive geographical information.

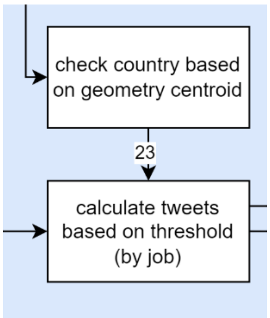


Figure 61: Check country based on geometry centroid and calculate tweets based on threshold

Location & Geometry Exist Yes: Check Country Based on Geometry Centroid

If location and geometry data exist, check the country based on the geometry centroid.

Location & Geometry Exist No: Calculate Tweets Based on Threshold (by Job)

If location and geometry data exist, calculates the number of tweets based on the job-specific threshold.

Check Country Based on Geometry Centroid to Calculate Tweets Based on Threshold (by Job)

The system leverages the country information obtained from the geometry centroid to perform a detailed analysis of tweet volumes in relation to job-specific thresholds as illustrated in Figure 61. This process begins by identifying the centroid of the geographical shapes associated with the tweets, which provides a precise central point for each defined area. Using this centroid, the system determines the corresponding country for each location, thereby contextualizing the tweet data within specific national boundaries. With the country information in hand, the system then calculates the number of tweets originating from each country, comparing these volumes against the job-specific thresholds that have been established. These thresholds serve as benchmarks for determining the significance or relevance of the tweet activity in relation to the job at hand. By cross-referencing the tweet counts with the predefined thresholds, the system can accurately assess whether the volume of tweets meets, exceeds, or falls short of the necessary criteria. This detailed approach ensures that only the most pertinent tweet data is considered for further analysis or action, optimizing the relevance and effectiveness of the information being processed. The integration of country-specific tweet volumes with job-specific thresholds enhances the system's ability to make informed decisions based on geographically contextualized social media activity.

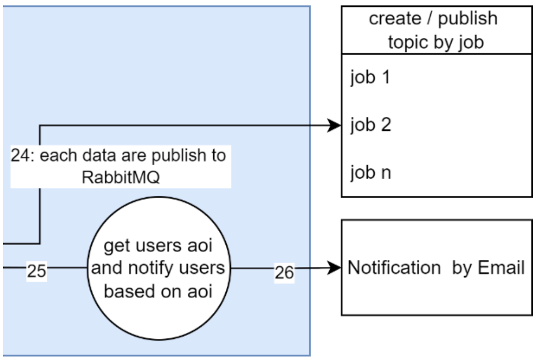


Figure 62: Each data is published to RabbitMQ and Get users aoi, notify users based on aoi, create/ publish topic by job and Notification by Email

Calculate Tweets Based on Threshold (by Job) to Publish Data to RabbitMQ and Create/Publish Topic by Job

Publishes each calculated tweet dataset to RabbitMQ and creates/publishes topics categorized by job.

Calculate Tweets Based on Threshold (by Job) to Get Users AOI and Notify Users Based on AOI

Retrieves users' Areas of Interest (AOI) and sends notifications to users based on their AOI.

Get Users AOI and Notify Users Based on AOI to Notification by Email

The system effectively manages user engagement by sending targeted email notifications based on their specified Areas of Interest (AOI) as shown in Figure 62. This process begins by analyzing each user's profile to identify their unique AOI, which could include topics such as environmental changes, regional events, or industry-specific updates. Utilizing this personalized information, the system continuously monitors relevant data streams and identifies content that aligns with the users' interests. When pertinent information is detected, the system generates tailored email notifications, ensuring that the content is highly relevant and valuable to each recipient. These emails are meticulously crafted to include key insights, updates, and actionable information, enhancing the user's experience and engagement. By delivering notifications that are directly aligned with users' AOI, the system not only fosters a more personalized interaction but also increases the likelihood of user engagement and satisfaction. This targeted approach ensures that users receive timely and pertinent updates, keeping them informed and connected to the topics they care about most.

8 Conclusions

This deliverable showcases the final version of the Semantic Information Fusion Framework, a significant milestone in the SILVANUS project. By integrating a wide range of data sources, from environmental sensors to social media feeds, the framework strengthens wildfire management across prevention, detection, and restoration phases. A key outcome is the development of the SemKB, which efficiently organizes and dynamically updates real-time data, providing essential decision support for managing wildfire incidents. The framework's capabilities are further enhanced through innovations like the enrichment of semantic space with intensity information and the integration of AI-driven technologies for biodiversity assessment. Additionally, the development of a multilingual forest fire alert system ensures that crucial information is communicated effectively across diverse linguistic regions, highlighting the project's commitment to inclusivity and real-time responsiveness.

Importantly, during the final year of SILVANUS, pilots are being conducted to validate key components such as the SemKB and the Integrated Data Insights (UP9h). These pilots will test the functionality, real-time data integration, and practical applications of the framework, ensuring that the system operates effectively in real-world scenarios. The insights gained from these pilots will contribute to refining the system further and establishing its efficacy in wildfire management.

9 References

[1] S. M. Marotta, V. Masucci, S. Kontogiannis and K. Avgerinakis, "From Unified Ontology to Knowledge Base: Data Fusion for Enhanced Wildfire Management.," in *Proceedings of the 8th World Conference on Smart Trends in System, Security, and Sustainability (WorldS4 2024).*, London, 2024.

[2] M. Iacovou, S. Kontogiannis and K. Avgerinakis, "Ontology Data Insights and Alerts for Wildfire Protection.," in *Proceedings of the 13th EETN Conference on Artificial Intelligence (SETN 2024).*, Piraeus, 2024.

[3] U. Nations., "What is climate change?," 2023. [Online]. Available: https://www.un.org/en/climatechange/what-is-climate-change.

[4] G. A. Angueira, A. Schuyler Nunn and C. E. Thompson, "23 Predictions for 2023. Fix Solutions Lab.," 13 December 2022. [Online]. Available: https://grist.org/fix/policy/23-predictions-for-2023-climate-justice-forecast-trends/.

[5] K. Latham, "Wildfires: The changing face of the Mediterranean landscape," 2023.

[6] T. Mackintosh, S. Seddon and A. Rhoden-Paul, "Rhodes fires: Thousands forced to flee 'disaster film' wildfires.," 24 July 2023. [Online]. Available: https://www.bbc.com/news/uk-66286741 .

[7] N. Salahjeh and R. Razek, "At least 106 people were killed in Maui's wildfires, officials say.," 16 August 2023. [Online]. Available: https://edition.cnn.com/2023/08/15/us/hawaii-maui-wildfires-death-toll-tuesday/.

[8] A. Apostolakis, S. Girtsou, G. Giannopoulos, N. S. Bartsotas and C. Kontoes, "Estimating next day's forest fire risk via a complete machine learning methodology," *Remote Sensing,* vol. 14, no. 5, 2022.

[9] I. Prapas, S. Kondylatos, I. Papoutsis, G. Camps-Valls, M. Ronco, M. Á. Fernández-Torres and N. Carvalhais, "Deep learning methods for daily wildfire danger forecasting.," 2021.

[10] H. Boström and e. al., "On the Definition of Information Fusion as a Field of Research.," 20077. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-1256 .

[11] P. Godec, M. Pančur and e. a. N. Ilenič, "Democratized image analytics by visual programming through integration of deep models and small-scale machine learning.," *Nature Communications,* vol. 10, no. 4551, 2019.

[12] J. Devlin, M.-W. Chang, K. Lee and Toutanova, "K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.," 2018.

[13] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks.," 2019.

[14] A. Wierzbicka, "Semantic Primitives. Athenäum," 1972.

[15] C. Goddard and A. Wierzbicka, "Meaning and Universal Grammar: Theory and Empirical Findings. Amsterdam: John Benjamins.," 2002.

[16] C. Goddard and A. Wierzbicka, "Words and Meanings: Lexical Semantics Across Domains, Languages, and Cultures. Oxford University Press.," 2013.

[17] P. Wittek, S. Darányi and Y. Liu, "A Vector Field Approach to Lexical Semantics. Presented at the 8th International Conference on Quantum Interaction, Filzbach, Switzerland. June 30 - July 3.," 2014. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:hb:diva.

[18] P. Wittek, S. Darányi, E. Kontopoulos, T. Moysiadis and I. Kompatsiaris, "Monitoring Term Drift Based on Semantic Consistency in an Evolving Vector Field. Proceedings of IJCNN-15. Presented at the International Joint Conference on Neural Networks.," 2015. [Online].

[19] S. Darányi, P. Wittek, K. Konstantinidis, S. Papadopoulos and E. Kontopoulos, "A Physical Metaphor to Study Semantic Drift. Proceedings of SuCCESS-16, 1st International Workshop on Semantic Change & Evolving Semantics, 1695.," 2016.

[20] L.-L. Ma, C. Ma, H.-F. Zhang and B.-H. Wang, "Identifying influential spreaders in complex networks based on gravity formula, Physica A: Statistical Mechanics and its Applications," vol. 451, no. 205-212, 2016.

[21] J. Liu, Q. Zhang, L. Fu, X. Wang and S. Lu, "Evolving knowledge graphs. In IEEE INFOCOM 2019-IEEE Conference on Computer Communications, 2260-2268.," 2019.

[22] P.-L. Yang et al, "AIGCrank: A new adaptive algorithm for identifying a set of influential spreaders in complex networks based on gravity centrality. Chinese Phys. B 32 058901.," 2023.

[23] X. Wu, T. Lin and M. Y. Yang, " Identifying influential risk spreaders in cryptocurrency networks based on the novel gravity strength centrality model. Applied Economics Letters, 1–10.," 2024.

[24] L. Lv, T. Zhang, P. Hu and D. Bardou et al, "An improved gravity centrality for finding important nodes in multi-layer networks based on multi-PageRank. Expert Systems with Applications Vol. 238, Part D, 122171,," 2024.

[25] F. Armaselu et al, "LL(O)D and NLP Perspectives on Semantic Change for Humanities Research. Semantic Web Journal.," 2021. [Online]. Available: https://www.semantic-web-journal.net/system/files/swj2848.pdf.

[26] F. Hinder, V. Vaquet and B. Hammer, " One or two things we know about concept drift -- a survey on monitoring in evolving environments. Part A: detecting concept drift. Frontiers in Artificial Intelligence, 7, 1330257.," 2024.

[27] D. M. Blei and J. D. Lafferty, "Dynamic topic models. Proceedings of the ICML. ICML'06. pp. 113–120.," 2006.

[28] M. Grootendorst, "BERTopic: Neural topic modeling with a class-based TF-IDF procedure.," 2022.

[29] R. Campello, D. Moulavi and J. Sander, "Density-Based Clustering Based on Hierarchical Density Estimates In: Advances in Knowledge Discovery and Data Mining, Springer. 160-172.," 2013.

[30] M. L. and J. Healy, "Accelerated Hierarchical Density Based Clustering. In 2017 IEEE International Conference on Data Mining Workshops (ICDMW), 33-42.," 2017.

[31] C. A. Crawford, "Topic Flow in Social Media Conversations. The University of Tulsa.," 2017. [Online]. Available: https://www.proquest.com/openview/60c8692891b0ee53d6a7ddfda991e9c3/1?pq-origsite=gscholar&cbl=18750.

[32] E. Sanandres, C. Madariaga and R. Abello, "Topic modeling of Twitter conversations. JADT'18," pp. 684-691, 2018.

[33] G. D. Melo and M. Bansal, "Good, Great, Excellent: Global Inference of Semantic Intensities. Transactions of the Association for Computational Linguistics 1(July), . ACL.," pp. 279-290, 2013.

[34] A. Soler and M. Apidianaki, "BERT Knows Punta Cana is not just beautiful, it's gorgeous," 2020.

[35] C. Shivade, M.-C. Marneffe, E. Fosler-Lussier and A. Lai et al, "Corpus-based discovery of semantic intensity scales. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies," pp. 483-493, 2015.

[36] O. Bakhshandeh and J. F. Allen, "From Adjective Glosses to Attribute Concepts: Learning Different Aspects that an adjective can describe.," *Proceedings of the 11th ACL International Conference on Computational Semantics. ,* 15-17 April 2015.

[37] C. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI".

[38] A. M. Jacobs, "Sentiment Analysis for Words and Fiction Characters From the Perspective of Computational (Neuro-)Poetics. Frontiers in Robotics and AI 6.," 2019.

[39] J.-K. Kim, "Adjusting Word Embeddings with Semantic Intensity Orders. In Proceedings of the 1st Workshop on Representation Learning for {NLP}, Berlin," pp. 62-69, 2016.

[40] M. Kasri, M. Birjali and A. Beni-Hsanne, "Word2Sent: A new learning sentiment-embedding model with low dimension for sentence level sentiment classification.," *Concurrency Computat Pract Exper.,* 2021.

[41] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *Computer Networks and ISDN Systems. 30 (1–7),,* p. 107–117, 1998.

[42] D. F. Gleich, "PageRank Beyond the Web," *SIAM Review,* p. 321–363, 2015.

[43] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts.," *In Proceedings of the 2004 conference on empirical methods in natural language processing,* pp. 404-411, 2004.

[44] H. Kim and R. Anderson, "Temporal Node Centrality in Complex Networks," 2011.

[45] M. de Boer, R. Bakker and M. Burghoorn, "Creating Dynamically Evolving Ontologies: A Use Case from the Labour Market Domain," *Proceedings of the AAAI 2023 Spring Symposium on Challenges Requiring the Combination of Machine Learning and Knowledge Engineering (AAAI-MAKE 2023),* 2023.

[46] K. Cagle, "A Knowledge Graph Isn’t a Picture – It’s a Movie," 2024.

[47] V. Veitch and D. M. Roy, "The class of random graphs arising from exchangeable random measures," 2015.

[48] L. van der Maaten and G. Hinton, "Visualizing Data Using t-SNE," *Journal of Machine Learning Research,* p. 2579–2605, 2008.

[49] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability,* vol. 1, p. 281–297, 1967.

[50] P. J. Rousseeuw, "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis," *Computational and Applied Mathematics,* p. 53–65, 1987.

[51] D. Asteriou and S. G. Hall, "Vector Autoregressive (VAR) Models and Causality Tests," *Applied Econometrics,* p. 319–333, 2011.

[52] F. Alam, U. Qazi, M. Imran and F. Ofli, "HumAID: Human-Annotated Disaster Incidents Data from Twitter," *ICWSM 2021,* 2021.

[53] Z. Harris, "Distributional structure," *Word 10 (23),* pp. 146-162, 1954.

[54] M. Laclavík, Š. Dlugolinský, M. Šeleng, M. Ciglan, M. Tomašek, M. Kvassay and L. Hluchý, "Lightweight semantic approach for enterprise search and interoperability," *CEUR Workshop Proceedings: INVIT 2012,* pp. 35-42, 2012.

[55] M. Laclavík, M. Šeleng, Š. Dlugolinský, E. Gatial and L. Hluchý, "Tools for email based recommendation in enterprise," *ENTERprise Information Systems: CENTERIS,* 2010.

[56] M. Laclavík, M. Šeleng, M. Ciglan and L. Hluchý, "Ontea: Platform for pattern based automated semantic annotation," *Computing and informatics,,* vol. 28, pp. 555-579, 2009.

[57] L. Obrst, "Ontologies for semantically interoperable systems," *Proceedings of the twelfth international conference on Information and knowledge management (CIKM '03),* pp. 366-369, 2003.

[58] W. Gräther, I. Matranga, V. Savarino, K. Furdik and M. Tomášek, "Improving Collaboration Between Large and Small Enterprises Using Networked Services," *16th Working Conference on Virtual Enterprises (PROVE),* 2015.

[59] A. Setyanto et al, "Ecological Impact Assessment Framework for areas affected by Natural Disasters," *ACM Int. Conf. Proceeding Ser.,* p. 155–161, 2022.

[60] H. Tao, "Smoke Recognition in Satellite Imagery via an Attention Pyramid Network With Bidirectional Multi-Level Multi-Granularity Feature Aggregation and Gated Fusion," *IEEE Internet Things J,* vol. 11, p. 14047–14057, 2023.

[61] B. Haq et al, "Tech-Driven Forest Conservation: Combating Deforestation With Internet of Things, Artificial Intelligence, and Remote Sensing," *IEEE Internet Things J,* p. 1, 2024.

[62] Z. Mohammed, C. Hanae and S. Larbi, "Comparative study on machine learning algorithms for early fire forest detection system using geodata," *Int. J. Electr. Comput. Eng.,* vol. 10, p. 5507–5513, 2020.

[63] J. Balling, J. Verbesselt, V. De Sy, M. Herold and J. Reiche, "Exploring archetypes of tropical fire-related forest disturbances based on dense optical and radar satellite data and active fire alerts," *Forests,* vol. 12, 2021.

[64] K. Yazici and A. Taskin, "A comparative Bayesian optimization-based machine learning and artificial neural networks approach for burned area prediction in forest fires: an application in Turkey," *Nat. Hazards,* vol. 119, p. 1883–1912, 2023.

[65] G. N. Rao et al, "Neural fuzzy system design in forest fire detection," *Microsyst. Technol.,* vol. 30, p. 455–467, 2023.

[66] V. Fernández-García, M. Franquesa and C. A. Kull, "Madagascar's burned area from Sentinel-2 imagery (2016–2022): Four times higher than from lower resolution sensors," *Sci. Total Environ.,* vol. 914, 2024.

[67] H. Xu et al, "Detecting forest fire omission error based on data fusion at subpixel scale," *Int. J. Appl. Earth Obs. Geoinf,* vol. 128, p. 103737, 2023.

[68] E. Kurbanov et al, "Temporal and Spatial Analyses of Forest Burnt Area in the Middle Volga Region Based on Satellite Imagery and Climatic Factors," *Climate,* vol. 12, 2024.

[69] D. Lee et al, "Single-Temporal Sentinel-2 for Analyzing Burned Area Detection Methods: A Study of 14 Cases in Republic of Korea Considering Land Cover," *Remote Sens.,* vol. 16, 2024.

[70] S. T. Saikumar and P. Sriramya, "Iot enabled forest fire detection and altering the authorities," *Int. J. Recent Technol. Eng.,* vol. 7, p. 429–433, 2019.

[71] S. Sendra, L. García, J. Lloret, I. Bosch and R. Vega-Rodríguez, "LoRaWAN network for fire monitoring in rural environments," *Electron,* vol. 9, 2020.

[72] J. Selle J, A. V. Krishna, M. S. Harish and K. B. Reddy, "An IoT based Alert System with Gas Sensors in a WSN Framework for Evasion of Forest Fire," *2nd Int. Conf. Adv. Comput. Innov. Technol. Eng. ICACITE 2022,* p. 1516–1521, 2022.

[73] A. Saida, C. Sreedhar, Samreen, A. Mohamma, K. Jamal and M. Ghalwan, "LORA Based Forest Fire Monitoring System," *E3S Web Conf,* vol. 430, p. 1–12, 2023.

[74] J. Alvarez et al, "Forest Fire Localization: From Reinforcement Learning Exploration to a Dynamic Drone Control," *J. Intell. Robot. Syst. Theory Appl,* vol. 109, 2023.

[75] A. Al-Dahoud, M. Fezari, A. A. Alkhatib, M. N. Soltani and A. Al-Dahoud, "“Forest Fire Detection System based on Low-Cost Wireless Sensor Network and Internet of Things," *WSEAS Trans. Environ. Dev.,* vol. 19, p. 506–513, 2023.

[76] R. Vikram and D. Sinha, "FogFire: fog assisted IoT enabled forest fire management," *Evol. Intell.,* vol. 16, p. 329–350, 2023.

[77] S. Manoj and C. Valliyammai, "Drone network for early warning of forest fire and dynamic fire quenching plan generation," *Eurasip J. Wirel. Commun. Netw,* vol. 2023, 2023.

[78] M. Krishnamoorthy, M. Asif, P. P. Kumar, R. S. S. Nuvvula, B. Khan and I. Colak, "A Design and Development of the Smart Forest Alert Monitoring System Using IoT," *J. Sensors,* vol. 2023, 2023.

[79] A. I. M. Anuar, R. Mohamad, A. M. Markom and R. Concepcion, "Real-time forest fire detection, monitoring, and alert system using Arduino," *Indones. J. Electr. Eng. Comput. Sci.,* vol. 33, pp. 942-950, 2024.

[80] S. K. Jayasingh, S. Swain, K. J. Patra and D. Gountia, "An Experimental Approach to Detect Forest Fire Using Machine Learning Mathematical Models and IoT," *SN Comput. Sci.,* vol. 5, p. 1–11, 2024.

[81] M. Hasan, M. A. Orgun and R. Schwitter, "Real-time event detection from the Twitter data stream using the TwitterNews+ Framework," *Inf. Process. Manag.,* vol. 56, p. 1146–1165, 2019.

[82] Z. Saeed et al, "What’s Happening Around the World? A Survey and Framework on Event Detection Techniques on Twitter," vol. 17, 2019.

[83] T. Papadimos et al, "Real-time Alert Framework for Fire Incidents Using Multimodal Event Detection on Social Media Streams," *Proc. Int. ISCRAM Conf.,* Vols. 2022-May, p. 623–635, 2022.

[84] A. Bozas et al, "From Research to Applications: What Can We Extract with Social Media Sensing?," *Springer Nature Singapore,* vol. 5, 2024.

[85] J. S. Almeida, C. Huang, F. G. Nogueira, S. Bhatia and V. H. C. De Albuquerque, "EdgeFireSmoke: A Novel Lightweight CNN Model for Real-Time Video Fire-Smoke Detection," *IEEE Trans. Ind. Informatics,* vol. 18, p. 7889–7898, 2022.

[86] K. Akyol, "Robust stacking-based ensemble learning model for forest fire detection," *Int. J. Environ. Sci. Technol.,* vol. 20, p. 13245–13258, 2023.

[87] M. S. Mondal et al, "Automating Fire Detection and Suppression with Computer Vision: A Multi-Layered Filtering Approach to Enhanced Fire Safety and Rapid Response," *Fire Technol.,* vol. 59, p. 1555–1583, 2023.

[88] R. Vikram and D. Sinha, "A multimodal framework for Forest fire detection and monitoring," *Multimed. Tools Appl.,* vol. 82, p. 9819–9842, 2023.

[89] P. Verma and R. Bakthula, "Empowering fire and smoke detection in smart monitoring through deep learning fusion," *Int. J. Inf. Technol.,* vol. 16, p. 345–352, 2024.

[90] K. Akyol, "A comprehensive comparison study of traditional classifiers and deep neural networks for forest fire detection," *Cluster Comput.,* vol. 27, p. 1201–1215, 2024.

[91] K. K. Paidipati, C. Kurangi, U. J, A. S. K. Reddy, G. Kadiravan and N. H. Shah, "Wireless sensor network assisted automated forest fire detection using deep learning and computer vision model," *Multimed. Tools Appl.,* vol. 83, p. 26733–26750, 2024.