Autonomous and Self-organized Artificial Intelligent Orchestrator for a Greener Industry 4.0

Deliverable

D4.1 Initial Reusability, Explainability, Trustworthiness Report

Actual submission date: 29/02/2024

Funded by
the European Union

Project Number:	101070181		
Project Acronym:	TALON		
Project Title:	Autonomous and Self-organized Artificial Intelligent Orchestrator for a Greener Industry 4.0		
Start date:	October 1st, 2022	**Duration:**	36 months
D4.1 Initial Reusability, Explainability, Trustworthiness Report			
Work Package:	WP4		
Lead partner:	SIDROCO Holdings Ltd (SID)		
Author(s):	Konstantinos Kyranou (SID)		
Reviewers:	EXOS, FACTOR		
Due date:	29/02/2024		
Deliverable Type:	R	**Dissemination Level:**	PU
Version number:	1.0		

Revision History

Version	Date	Author	Description
0.1	15/12/2023	SID	First release of the Table of Contents
0.2	15/01/2024	SID	Integrated provided contributions from partners
0.3	15/02/2024	SID	Finalised the initial draft
0.5	21/02/2024	FACTOR	Review of the contents
0.5	21/02/2024	EXOS	Review of the contesnts
1.0	29/02/2024	ENG	Final coordinator review before submission

Table of Contents

List of Figures

List of Tables

Definitions and acronyms

AI	*Artificial Intelligence*
ABAC	*Attribute-Based Access Control*
ADJP	*Adjective Phrase*
ADVP	*Adverb Phrase*
API	*Application Programming Interface*
AR	*Augmented Reality*
ARIEC	*Anonymous Repository of Incidents*
BLSTM	*Bidirectional Long Short-Term Memory*
CA	*Consortium Agreement*
CDN	*Content Delivery Network*
CS	*Construction Safety*
CNN	*Convolutional Neural Network*
CRF	*Conditional Random Fields*
DP	*Digital Policies*
DoA	*Description of Action*
DNN	*Deep Neural Network*
DT	*Decision Tree*
E2C	*Edge-to-cloud*
EC	*European Commission*
EU	*European Union*
ELMo	*Embeddings from Language Models*
FedAvg	*Federated Average*
FedProx	*Federated Proximal*
FedAdam	*Federated Adam*
FedAdagrad	*Federated Adagrad*
FedYogi	*Federated Yogi*
FL	*Federated Learning*
FSL	*Few-Shot Learning*
GA	*Grant Agreement*
GDPR	*General Data Protection Regulation*
IAM	*Identity and Access Management*
ImAM	*Image Anonymization Module*
IG	*Information Gain Attribute Evaluation*
IoT	*Internet of Things*
IOB	*Inside Outside and Beginning*
LOF	*Local Outlier Factor*
LSTM	*Long Short-Term Memory*
MAML	*Model-Agnostic Meta Learning*
mAP	*Mean Average Precision*
ML	*Machine Learning*
MSE	*Mean Squared Error*
MAE	*Mean Absolute Error*
NER	*Named Entity Recognition*
NIC	*Network Interface Controller*
NP	*Neural Processes*
NLP	*Natural Language Processing*
PPE	*Personal Protection Equipment*
PC	*Project Coordinator*
PCA	*Principal Component Analysis*
PEP	*Policy Enforcement Point*
PIP	*Policy Information Point*
PDP	*Policy Decision Point*
POS	*Part-of-Speech*
PRP	*Prepositional Phrase*
PII	*Personally Identifiable Information*

QoS	*Quality of Service*
RBAC	*Role Based Access Control*
RFE	*Recursive Feature Elimination*
RMSE	*Root Mean Squared Error*
RNN	*Recurrent Neural Network*
ROI	*Region of Interest*
SSO	*Single sign-on*
TC	*Technical Coordinator*
TCP	*Transmission Control Protocol*
TL	*Transfer Learning*
TR	*Trust Level*
UC	*Use Case*
VP	*Verb Phrase*
VR	*Virtual Reality*
WP	*Work Package*
XAI	*eXplainable Artificial Intelligence*
YOLO	*You Only Look Once*

Disclaimer

Executive Summary

The "Initial Reusability, Explainability, Trustworthiness Report" provides an in-depth analysis of TALON's initial technical developments. It meticulously outlines TALON's modules design principles, mechanisms, and services aimed at ensuring reusability, explainability, and trustworthiness. This document serves as a comprehensive report, delineating TALON's core attributes essential for navigating the complexities of Industry 4.0/5.0 initiatives with reliability and transparency.

In this document, all technical components developed in the first stage of TALON's development phase are reported. These modules fall under the three key objectives which constitute the holistic solution of TALON platform. In this work, various state-of-the-art AI and data operation methods and tools have been employed, to create the necessary technological landscape in which the UCs will build upon.

This work marks the initial phase of TALON's development journey, characterised by a detailed clarification of the technical processes progress. While it is important to note that not all modules have reached the same level of maturity, it's equally crucial to emphasise that each module is actively progressing forward. Meticulously outlined clear plans for research, development, and implementation for every module have been set up, ensuring a structured approach towards the project's objectives.

I Introduction

1.1 Purpose of the Deliverable

This deliverable, titled the "Initial Reusability, Explainability, Trustworthiness Report" has been compiled to present and elaborate the initial results related to the technical exercises of Work Package 4 (WP4). These preliminary outcomes specifically address the key concepts of reusability, explainability and trustworthiness. This report covers the following major pillars:

i. TALON's AI/ML cognition and Transfer Learning (TL) functionalities

ii. TALON's hybrid learning system which integrates both Few-Shot Learning (FSL) and Federated Learning (FL) methodologies

iii. TALON's diverse eXplainable Artificial Intelligence (XAI) procedures, including the conceptualisation and implementation of monitoring and visualisation mechanisms

The deliverable starts with a high-level description regarding the methods, modules and techniques utilised to meet the reusability, explainability and trustworthiness aspects of the TALON platform, paying particular attention to the green yet efficient characteristics of the implementation. In the upcoming sections, the various reusable data modules, explainable AI functionalities and trustworthy data and AI models are thoroughly explained, along with the upcoming scheduled and envisioned actions.

1.2 Relation with other Work Packages, Tasks and Deliverables

Task 4.1, Task 4.2 and Task 4.3 draw upon the insights provided in "Use Case, KPIs, Requirements, Specification, Slices & Technology Enablers Definition Report" from D2.1 and the "Installation & Demonstration Planning, Evaluation Methodology & KPIs Definition Report" from D5.1. Deliverable 2.1 is a major prerequisite for the proper and accurate writing of this deliverable, because it elucidates the technical requirements of the Use Cases (UCs) and offers an initial analysis of the challenges faced by those demonstrators, setting the correct landscape of the actual development implementation. Deliverable 5.1 acts as a resource for obtaining information about the KPIs definition, as well as the installation and evaluation of the various TALON components.

Deliverable 4.1 is utilised as input for Task 5.1 "Installation & Demonstration Planning, Evaluation Methodology & KPIs Definition" and Task 5.2 "TALON Platform Setup, Operation, Continuous Integration & Maintenance", which are mostly related to the design, identification, operation and preservation of the various modules of TALON platform. Moreover, D4.1 is used as a source for Task 5.3 "Automatic UATV Coordination", Task 5.4 "I5.0 Automation & Planning", Task 5.5 "AR/VR for Training & Maintenance" and Task 5.6 "Human Robot Collaboration" which essentially are the pilots of TALON project and leverage from the developed technologies of WP4.

1.3 Structure of the Document

Following this introduction, the document is structured in the following way:

- Section 2: In this section, a detailed approach is provided towards addressing the aspects of data and/or AI models reusability, explainability and trustworthiness. Particular attention is also drawn to the greener AI operations, which optimize the lifecycle of the data operations, while maintaining accuracy and robustness.
- Section 3: This section illustrates the reusable nature of TALON's AI-based predictive analytics functionalities and data operations.
- Section 4: In this part the various XAI modules are described

- Section 5: This section of the deliverable describes the data procedures and AI algorithms that consolidate the trustworthiness of the platform.
- Section 6: A brief presentation of forthcoming developments and activities.
- Conclusion: Summarises/finalises the document

2 TALON Methodology regarding Reusability, Explainability and Trustworthiness

2.1 Data and AI Models Reusability via Few-shot and Meta-learning & Data Curation and Enrichment

Applications of contemporary AI models often demand significant time and resource investments, accompanied by substantial development and maintenance expenses, particularly within the industrial domain [1]. On the other hand, industrial datasets exhibit heterogeneity, characterised by high velocity, veracity, and magnitude, thus complicating internal industrial data processes to a considerable extent and elevating overall operational costs. Such ecosystems also require continuous support of a variable number of resources or various operational devices and attestations for availability, autonomous management and functionality. Robustness, adaptability and flexibility are just some of the boxes that intelligent algorithms need to tick, taking into consideration the important input variables which is the data. In order to accommodate such complex and important tasks, data should be realistic, operational and provide added value to the business intelligence after the analytical processes.

One of the key remaining challenges in data exploration and reusability is effective data curation [2]. Both conventional and contemporary industrial environments struggle to harness data from heterogeneous devices to extract valuable business intelligence. Big, sparse, streaming, unstructured and legacy are just some of the data "categories" that pose significant difficulties for data processing and AI models that consume them. TALON's data curation engine provides a holistic solution for data cleaning, adjustment and homogenisation. Data consumers (i.e. TALON's AI/ML modules and dashboard) demand that specific requirements are met in order to effectively use data, and this is often a peculiar task, especially when dealing with industrial data, which originate from various sources. In that sense, TALON goes beyond the standard procedures, introducing also the human-in-the-loop aspect in the data curation engine, enabling stakeholders to choose the desired curation functionality among a pool of functionalities, all of which, address various data optimisation challenges.

TALON addresses the aforementioned challenges originating from next-generation industrial systems, introducing the reusability concept and realising optimal AI and data solutions. These solutions are oriented towards developing and implementing a knowledge repository, containing optimal deployment configurations or AI systems and serving as reusable models. This practice will speed up industrial enterprises efficiency, while simultaneously providing the ability to create, deploy, and employ optimal AI systems. Real-time, onsite edge computing is widely adopted nowadays and the numerous benefits come along with the respective challenges. TALON's third demonstrator creates the landscape for AI-powered deployments at the edge, facilitating real-time on-site augmented reality/virtual reality (AR/VR) assisted device maintenance and personnel training. TALON platform extends reusability for human-involved maintenance and tasks that may involve potential hazards. This approach not only reduces latency in communication but also alleviates computational burdens inherent in such systems. While the nature and scope of the data may vary depending on the UC scenario and the ML model task, reusability of digital assets will be exploited to achieve seamless incorporation of data and metadata into diverse knowledge sources.

Cost-efficient learning constitutes a major value indicator in industrial workspaces. Time, resources and budget are key metrics to evaluate the performance – cost balance across data operations and AI models efficacy. Data and AI models are utilised either for descriptive or predictive purposes in manufacturing, but the human factor remains the most important one. To that end, novel ML techniques that leverage sources that generate little data should be employed, to enable a smooth

and reliable human-robot collaboration and make the best out of the existing data sources. TALON introduces and develops state-of-the-art Few-Show ML techniques that can undertake a compact dataset comprising information with ground truth specific to the focus area. Unlike the direct feature mapping in generic Deep Neural Networks (DNNs), FSL places greater emphasis on the reusability of features. Moreover, this technique emerges as a viable alternative against the traditional data-demanding deep learning models.

Meta-learning, transfer-learning, and data augmentation are the most research-explored techniques regarding FSL. While each of these techniques is based on different underlying principles, they all share a common goal of dealing with data scarcity and enabling the training of models that are effective, adaptable, and robust in volatile and rapidly changing settings. In the context of TALON, these techniques are utilized to grant AI models reusability, cost and energy efficiency, and adaptability, properties that are highly desirable given the nature of the use cases, e.g., industrial settings, and the devices, e.g. hardware-constrained edge devices, in which these models will be deployed.

Meta-learning [1] or learning-to-learn is a learning paradigm that aims to enable rapid adaptation of ML models to novel tasks that contain only a minimal number of training samples. To achieve this, meta-learning models extract the common structure shared between these tasks, typically in the form of an underlying task distribution with specific yet unknown characteristics, and leverage it as prior knowledge to accelerate training in new tasks. Given the federated and decentralized nature of the TALON platform, a typical pattern in these settings is the emergence of small datasets scattered across different edge devices. In general, these datasets contain different samples, yet it is assumed that they share similarities since the edge devices are deployed in the same or similar environments. Consequently, each dataset can be treated as a separate task, and meta-learning can be applied to enable training across these tasks in an efficient and reusable manner.

Transfer learning [2] is another common approach when the objective is rapid adaptation in scenarios with limited training samples. More specifically, transfer learning approaches leverage acquired feature representations that are generalizable to enable rapid adaptation to downstream tasks with limited data. Obtaining these representations is typically achieved through pretraining in large sets of data, a task that is not always cost-efficient. However, the existence of numerous publicly available pretrained models, suitable for a plethora of tasks, promotes reusability since they can be trained once and then used in downstream tasks with little adaptation. As a result, for the effective deployment of transfer learning models in environments where cost efficiency and rapid adaptation are critical components, transfer learning can be a valuable tool in achieving these goals. The reusable feature extractor can be stored in an AI model repository and be shared across the edge devices, each dealing with a separate downstream task, promoting the idea of model reusability. At the same time, since the adaptation of the shared model to each task requires minimal extra training, it is highly suitable even for devices with limited computational power.

Finally, data augmentation [3] approaches have also been extensively used in the context of FSL. Since the main difficulty in the aforementioned scenarios arises from the scarcity of training data, a straightforward approach is to increase the volume of these data. In general, collecting additional real-world data is not always feasible due to constraints related to time and cost. In such cases, an efficient way is to augment the existing data by generating synthetic ones that can be used for AI model training. Overall, data-augmentation methods create novel samples by applying various transformations to the existing data. One of the main benefits of these transformations is that they are straightforward, lightweight, and easy to implement, imposing only a minimal computational overhead compared to the performance benefits they can provide. For instance, in the case of images, simple transformations such as cropping, flipping, and color jittering can be enough to augment the existing dataset and lead to models that demonstrate high performance. As a result,

data-augmentation techniques are a powerful tool that is leveraged in TALON to promote efficiency and adaptability.

2.2 Data and AI Models Explainability

AI stands as the primary driver of the industrial revolution, empowering intelligent machines to independently perform tasks such as independent monitoring, interpretation, diagnostics and various analytical operations [3]. In the industrial domain, ML models often handle sensitive tasks where human involvement dictates the imperative for model reliability. The challenge lies in ensuring the transparency of these models, particularly when they are employed in critical decision-making processes or safety-critical applications. TALON's primary objective is to develop ML models that not only exhibit high predictive accuracy, but also demonstrate reliability in alignment with the peculiarities of industrial settings, where the human factor necessitates trust in the model's outcomes.

One of the main objectives of this project is the reliability, transparency and accountability of the developed ML algorithms. AI-driven decisions are made more transparent as their interpretability enables domain experts to comprehend, question, and potentially enhance them. This is further highlighted in the case of TALON's demonstrators, where the human factor is directly or indirectly involved, either being qualified personnel/operators in the manufacturing premises or first responders in critical situations. TALON addresses the interpretability issues of autonomous and complex industrial AI systems going beyond the traditional ML model feature importance by introducing a novel four-layered system in one unified explainable AI (XAI) platform. The different XAI operations, which are specifically tailored for the TALON project, are divided into:

- Trust Level 1 (TrL1) which pertains to the raw data origination, examining the source reliability and attesting for their quality.
- Trust Level 2 (TrL2) which deals with the processing phase of the data and checks the adjustments and feature engineering part of the data. This is an important phase as it attests to the quality of the dataset and provides insights regarding potential imbalances and outliers that could affect the final ML outcome.
- Trust Level 3 (TrL3) which evaluates the AI model's robustness, by employing novel XAI algorithms to inspect the reliability of the various ML models.
- Trust Level 4 (TrL4) which attests to the robustness of the ML algorithms by examining the fidelity and consistency of the model's inference.

Overall, generic approaches of explainability focus mainly on interpretations provided to verify the performance of AI-based inference and prevent potential errors [1]. This usually translates technically to one or two explainable methods to evaluate the ML model and ultimately the use of visualization techniques to demonstrate the corresponding results. TALON's XAI platform goes beyond the ordinary explanations that are often used in ML models. The developed/envisioned work introduces the four aforementioned TrLs to evaluate and provide feedback to stakeholders regarding the whole process of the data life cycle. This is not only aligned with the contemporary industrial requirements, but also operates as a proactive measure of visualising and interpreting the whole process from its source until the final outcome. Visualisation, segmentation and metrification of transparency for data and AI algorithms, enable optimal description of AI models operations, evaluation of expected operational impact and promotes human-robot collaboration.

2.3 Data and AI Models Trustworthiness via Anonymisation, Federated Learning and Authorisation

In the realm of industrial ecosystems, the integration of complex systems is commonplace, with safety-critical operations constituting a prevalent facet. Within this domain, the imperative to safeguard the integrity of systems manifests through a dedicated emphasis on the trustworthiness attribute. Given that the seamless functionality of these systems is contingent upon the application of cutting-edge Artificial Intelligence (AI) methodologies, particular attention is directed towards ensuring the trustworthiness of both data and AI models.

In a more comprehensive sense, the attainment of trustworthiness is orchestrated through the implementation of systematic strategies. These include the adoption of data anonymization techniques, the incorporation of FL methodologies for the training of AI algorithms, and the judicious utilization of Authorization techniques. The latter not only bolsters the security mechanisms inherent to the targeted systems but also serves as a pivotal element in fortifying the overall dependability and safety of these intricate industrial ecosystems.

Anonymization constitutes the process of systematically eliminating or encrypting Personally Identifiable Information (PII) within datasets to safeguard the privacy of individuals. This procedure serves the purpose of minimizing the potential risk of re-identification, thereby assuring that the data employed for the training of models remains impervious to tracing back to specific individuals. Various sophisticated techniques, including but not limited to generalization [4], suppression [5], and noise injection [6], are crucial in effecting anonymization. These methodologies are meticulously applied to the data, sustaining a balance between obscuring sensitive information and retaining the requisite utility for subsequent deployment in Machine Learning (ML) algorithms.

Federated Learning [7] represents a transformative paradigm shift in ML, characterized by its decentralized architecture. Notably, FL prioritizes data localization, conducting collaborative training on individual devices responsible for generating data, thus diverging from the conventional centralized approach. The distinctive feature of FL lies in its robust preservation of data privacy. Sensitive information is deliberately confined to local devices, mitigating privacy concerns associated with centralized models. The exchange of information in FL is limited to model updates, transmitted in the form of aggregated gradients. This type of sharing mechanism enhances the global model iteratively while maintaining the privacy and integrity of individual datasets.

To fortify the security and privacy aspects inherent in the collaborative training process, FL employs sophisticated techniques. These include model aggregation, secure multi-party computation, and the integration of differential privacy measures. Collectively, these techniques contribute to the establishment of a secure and privacy-preserving environment, ensuring the integrity of data and model parameters throughout the FL ecosystem.

Authorization encompasses the delineation and enforcement of access controls, ensuring that exclusively authorized entities possess the privilege to access and manipulate data or models. Its primary objective is to thwart unauthorized access, modification, or utilization of sensitive information, thereby instating a protective layer of security for both the data and AI models. The implementation of authorization policies is often facilitated through established methodologies such as Role-Based Access Control (RBAC), Attribute-Based Access Control (ABAC), and other sophisticated access management frameworks [8] [9]. These frameworks serve as instrumental tools in the enforcement of access controls, ensuring the integrity and confidentiality of data and AI assets within a given system.

Within the context of the TALON project, a synergistic integration of these methods and strategies becomes imperative to uphold the trustworthiness of data and AI models slated for deployment across

diverse pilot use cases. Notably, the introduction of anonymization techniques during data collection and storage serves as an initial privacy safeguard. Simultaneously, robust authorization mechanisms should regulate access to anonymized data in accordance with predefined user roles and responsibilities.

In the collaborative paradigm of FL, where model training spans distributed datasets without direct raw data exchange, a combination of anonymization and authorization is essential. Authorization protocols play a pivotal role in ensuring that only authorized entities engage in the FL process, thereby fortifying data privacy.

Prior to the deployment of FL models, an imperative step involves verifying that the data used for training has undergone anonymization, thus reinforcing the protection of user privacy. This comprehensive approach, encompassing anonymization, authorization, and FL strategies, converges to establish a robust framework for maintaining the trustworthiness of both data and AI models within the TALON project.

Finally, the human factor assumes a crucial role in these procedures, increasing the effectiveness of the aforementioned methods and elevating the levels of trustworthiness and privacy through continuous monitoring and auditing. It is essential to consistently monitor and audit anonymization processes, FL protocols, and authorization policies to ensure sustained trustworthiness. Adaptations should be made in response to evolving privacy regulations and emerging security threats. This careful oversight aligns with the overarching objectives of the TALON project by prioritizing ongoing scrutiny and adapting to dynamic regulatory landscapes. This approach ensures that the project remains committed to fostering a trustworthy and ethically sound deployment of AI technologies.

2.4 Green & High-Performance AI Deployments/Networks

Interdisciplinary dataset application and less data-intensive AI/ML algorithms set the perfect stage for a greener and novel ecosystem. Even though the recent trend within the ML and deep learning field is the deployment of increasingly large AI models that are extensively trained on large volumes of data, there is also a growing skepticism towards this learning paradigm [10]. Specifically, in modern times where green energy, sustainability, and protection of the environment against threats such as climate change have become more crucial than ever, modern ML seems to follow the opposite direction by promoting AI models that consume very large amounts of energy during training and consequently lead to increased CO2 emissions that directly affect the environment. In recent years, the concepts of sustainability, green development, and energy efficiency have also penetrated the field of AI, leading to the development of Green AI methods that consume less energy but maintain similar performance compared to their corresponding energy-consuming counterparts.

Apart from its obvious environmental benefits, Green AI can also be crucial in expanding the application of ML methods to various real-world use cases. Using large-scale ML and deep learning models in various real-world applications can become difficult or even impossible due to computational complexity and energy efficiency considerations. On the other hand, learning in such settings can also be challenging when data is scarce, expensive, or time-consuming to acquire and computational resources are limited. At the same time, restricting the size of the training data can also be deliberate in the sense that model training is the most energy-consuming part of an AI model's lifecycle and, therefore, energy-efficient and sustainable AI solutions should be able to maintain high levels of performance and accuracy even when trained with limited data. Based on the aforementioned insights, it is evident that achieving these objectives regarding sustainability and energy efficiency requires moving away from the standard ML paradigm, where different models are trained from scratch for each given task, and towards reusable models that can adapt to novel tasks in unknown settings given only a few training samples. FSL, a new learning paradigm that focuses on learning across similar tasks with limited data by leveraging their shared underlying structure, has

recently emerged as a suitable candidate that complements large-scale ML by addressing its limitations in these data-scarce settings.

In the context of this project, FSL approaches can help develop energy-efficient and sustainable AI models that are scalable, reusable, and adaptable to novel settings. In particular, deployed models can leverage previous knowledge to enable quick adaptation, eliminating the need for extensive retraining when presented with a new task, and allowing for dynamic model reusability in challenging cases where data collection is difficult or costly. Also, rapid adaptation is relevant to new tasks and existing tasks deployed in volatile settings. Due to the ever-changing nature of many real-world application environments, such as TALON's use cases, the deployed models must be able to adjust to these changes with minimal retraining. This not only minimizes the downtime of these models when dealing with these changes but also their energy consumption by obviating the need for extensive retraining each time a change occurs. Finally, FSL can minimize overall computational needs during training, leading to energy savings and promoting a sustainable AI learning paradigm. Therefore, it is evident that FSL can play a significant role in increasing data efficiency and reducing the energy usage of TALON's architecture.

Apart from the efficient training of AI models, the use of FSL is also highly aligned with TALON's aim to provide cloud computing services closer to end users. More precisely, in TALON, the AI models used in each application are deployed on the edge nodes following a decentralized architecture and, subsequently, need to be trained on the edge. However, numerous edge devices are discerned by hardware and computational constraints, necessitating the deployment of lightweight models that require minimal training. As a result, FSL can be used to minimize the amount of data required and computational power needed to train these models on edge devices with hardware limitations.

Overall, creating models that can maintain high levels of performance when trained with limited data is crucial in terms of sustainability and adoption of AI in volatile real-world applications and environments. With that consideration, Green AI is a major consideration in this project, and, therefore, various approaches within the FSL paradigm are developed to address the pressing issues of data and energy efficiency.

3 AI Cognition Modules for Dynamic Reusability

3.1 Transfer, Few-shot and Hybrid Learning Framework

3.1.1 Internal Architecture and Key Technologies

Machine and deep learning have rapidly advanced in the last decade, thanks to the increased availability of computational resources like GPUs, massive datasets, and sophisticated novel algorithms like the Transformer architecture. These models have excelled in numerous tasks including image classification, natural language processing, and industrial applications, even surpassing human performance in some cases. However, training such AI models poses challenges given the vast amounts of data and the computational power needed [11]. At the same time, they suffer poor performance under scenarios of continual and incremental learning.

Various real-world applications entail learning in settings where data is scarce, expensive, or time-consuming to acquire and the availability of computational resources is limited. At the same, the need to adhere to imposed privacy constraints necessitates the use of algorithms that are capable of learning from only a handful of available data. Moreover, reducing the size of required datasets in training efficient and accurate AI models can play a pivotal role in energy-efficient and sustainable AI solutions. In particular, this can be achieved by transitioning away from the established paradigm of training different AI models from scratch and moving towards reusable models capable of rapidly adapting to varying tasks when deployed in novel settings. Consequently, these models demonstrate increased data efficiency and can help reduce the energy footprint of TALON's architecture.

Towards this end, modern learning paradigms including FSL [12] and TL [13] can play an important role. In FSL, the goal is to develop models that can achieve high levels of performance across different learning problems, even though only a limited number of training data is available for each given problem. This is usually achieved by utilizing some form of prior knowledge that is shared across these tasks, overcoming the limitations of conventional ML methods that are prone to overfitting under such data scarcity. As a result, FSL methods minimize the need for collection and preprocessing of large volumes of data and accelerate deployment and adaptation to new settings.

Based on the goal of FSL and the type of problems in which it is mostly applied, it is evident that its objective differs from that of traditional ML. Specifically, in standard ML the goal is to find a model f_θ that minimizes the prediction error within a specific problem whose data are generated by an unknown but specific data distribution $p(x, y)$. This can be formulated as:

Equation 1: Optimization objective of standard machine learning algorithms

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{(x,y)\sim p(x,y)}[\mathcal{L}(f_\theta(x), y)]$$

However, in FSL, the goal is to develop a model that is able to generalize across multiple tasks, even though each one of them comes with only a limited number of training samples. In general, it is assumed that there is an underlying common structure shared across these tasks that is typically modeled as an unknown task distribution $q(T)$ that generates these tasks. Subsequently, each task is associated with a different data-generating process, formed as a task-specific data distribution $p(x, y)$. Developing a model f_θ that is capable of generalizing across tasks, is equivalent to minimizing the prediction error across all tasks and all associated datasets, which can be formulated as:

Equation 2: Optimization objective of FSL algorithms

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{T\sim q(T)}[\mathbb{E}_{(x,y)\sim p(x,y)}\mathcal{L}(f_\theta(x), y)]$$

FSL Approaches

Given the fact that FSL covers a wide range of different learning problems where the main objective is to learn under data scarcity, it is logical that different approaches and paradigms have been proposed throughout the years to deal with it. These include traditional approaches such as meta-learning and TL as well as emerging ones like hybrid methods and Neural Processes (NP). An overview of the major categories used within the FSL paradigm is provided below.

Meta-Learning: The primary objective of meta-learning [14] is to facilitate quicker adaptation to new tasks by leveraging previously acquired knowledge. To accomplish this, meta-learning models extract the shared common structure from a set of tasks and use it as a basis that enables rapid adaptation to unfamiliar tasks with limited data. This can be viewed as a learning-to-learn approach because the meta-learning model strives to learn how to effectively generalize within each given task. Subsequently, based on the specific techniques employed, meta-learning algorithms can further be split into four different categories:

1) **Metric-Based:** These approaches heavily rely on the basic principles of embedding learning where the goal is to learn an appropriate representation space for the data. In the context of FSL, the goal is to learn an embedding space where each task's data can be represented in such a way that the model can easily generalize from them, despite their limited number. These approaches are typically characterized by two components: a feature extractor that is either shared across or specifically tailored for each learning task, and a classifier based on a metric that is either predefined or learned from the data. Metric-based meta-learning models have primarily been used in the field of computer vision and, consequently, they have relied on CNN architectures to embed the relative input data. Common models within this category are Siamese Networks [15], Prototypical Networks [16], Relation Networks [17], and Matching Networks [18].

2) **Model-Based:** A common theme in this type of approach is the development of neural network architectures that promote rapid learning in novel settings, thus reducing the volume of data needed to achieve competitive performance. This is typically achieved through the combination of modules whose intrinsic properties such as their internal state, or memory enhance the reusability and sustainability of the final model. One major subcategory of model-based methods is architecture-based models which leverage the internal state of Recurrent Neural Networks (RNNs) alongside a formulation of the learning objective as a Reinforcement Learning (RL) problem [19]. Other approaches have also been proposed including generic modular architectures that can concurrently be applied to different applications [20] and architectures that leverage modern sparse neural network structures [21]. On the other hand, the second major subcategory of model-based approaches is memory-based methods which include models equipped with external memory modules. These memory-augmented networks can either be used to store model parameters [22] or specific data sample representations [23], enabling the model to retain useful information and thus adapt faster to novel tasks.

3) **Optimization-Based:** One of the main disadvantages of commonly used neural network optimization methodologies is the lack of guarantees for fast convergence to an optimal solution. As a result, most of these algorithms fail under scenarios where data is limited, since

they are unable to produce models capable of generalizing to tasks of increased difficulty. Optimization-based models advocate for acquiring adaptable learning rules that enable fast adaptation to new tasks, alleviating the need for multiple iterations and enhancing model efficiency by reducing the data and time needed for training [24]. A common theme in these methods is casting the meta-learning problem as a bi-level optimization problem and applying techniques that are suitable for this problem formulation. Another typical approach within this paradigm is learning an appropriate set of initial parameters that can converge to each task's optimal parameters with only a few training iterations. For instance, in Model-Agnostic Meta-Learning (MAML) [25], an architecture-agnostic solution is proposed that extracts the common structure shared across different tasks in the form of initial model parameters by performing gradient descent on the aggregated errors of all training tasks. During testing, the final model is then able to adapt to novel tasks utilizing only a minimal number of training samples and converging within a few training iterations. Finally, the useful inductive bias of existing learning algorithms can be enhanced via meta-learning by developing models that learn how to control the updates of the optimization algorithm. For instance, this can be achieved by meta-learning the learning rate and direction for each training iteration, as in MetaSGD [26], or by imposing a form of preconditioning on the gradients used to update the model weights [27].

4) **Probabilistic Extensions:** While most of the traditional meta-learning approaches have achieved excellent results, their deterministic formulation does not allow uncertainty quantification, which is particularly crucial especially in cases where data is limited. Consequently, many of the aforementioned approaches have also been extended to their probabilistic versions where model parameters are modeled as random variables and the goal is to obtain their distributions. Towards that direction, various Bayesian meta-learning approaches [28] have been proposed, aiming to adapt expressive generative models to small datasets by leveraging inductive biases learned and shared across multiple similar tasks.

Data Augmentation: An intuitive and straightforward way to deal with the lack of data that characterizes FSL problems is to incorporate more data in the training procedure to mitigate any risks of overfitting. Data augmentation approaches aim to do exactly that by creating additional training samples based on the existing ones. For instance, in the computer vision domain, standard data augmentation techniques might include simple transformations such as cropping, flipping, rotation, and translation of images. Image erasing techniques, where specific regions of an image are removed to avoid developing models that overly rely on specific parts, are also quite popular, as well as image mixing approaches that entail generating novel images via appropriate mixing of existing images. Data augmentation can also be seen as a learning problem itself where the objective is to find a suitable set of transformations in an automated manner, as in AutoAugment [29]. An important benefit of these approaches is the flexibility and reusability that they provide since they alleviate the need for domain-specific expertise. Finally, the rapid advancement in Generative AI can play a catalytic role in the context of FSL by generating diverse and realistic data using modern architectures such as diffusion models [30].

Transfer Learning: Deep learning models can utilize TL to apply the knowledge extracted from a learning problem with a corresponding dataset to a novel problem with a different but similar dataset. The initial learning problem is referred to as the source domain where in general data are abundant while the learning problem where the model is applied is referred to as the target domain where data availability is limited. Leveraging the plethora of data in the source domain, the model is pretrained so that it acquires strong inductive biases, in the hope that they will be useful in the target domain, mitigating the need for extensive additional training. It is evident that in the context of FSL, TL constitutes a two-step training procedure: pretraining in the source domain and finetuning in the target

domain. To avoid overfitting due to the small number of training data in the downstream target tasks, various techniques are employed with the most commonly used ones based on freezing part of the model's parameters and finetuning only the rest of them using the novel task's data. An approach that is quite popular in practice is freezing the pretrained feature extractor's parameters and only finetuning the parameters of the classifier head within each task [31]. Subsequently, these approaches have also been extended to incorporate various techniques that boost performance including transductive finetuning [32], leveraging invariance and equivariance properties [33], and combining TL with meta-learning [34]. Overall, TL can be advantageous in real-world applications compared to meta-learning approaches due to its simplicity and efficiency, which enhance the reusability of the model and facilitate rapid deployment in unknown settings. However, it is worth noting that TL is heavily dependent upon the existence of large volumes of data in the source domain, which is not always practically feasible, while it may also suffer from poor performance in cases where there is a large gap in the underlying structures of the source and target domains.

Neural Processes: In recent years, a novel efficient solution for FSL problems has emerged, namely NPs. NPs are a family of probabilistic models that aim to bridge the gap between deep learning models and Gaussian Processes by leveraging the accuracy of the former and the modeling of prior knowledge of the latter. Consequently, NPs combine the best of both worlds by being data efficient and highly accurate at the same time. In fact, NPs are capable of making accurate predictions after observing only a few training data points and efficiently scaling to complex functions and large datasets. In the last few years, numerous variations of NPs have been proposed including Conditional NPs [35], that condition the output prediction on an aggregated representation of the training samples, Attentive NPs [36] that incorporate the use of attention modules to enable more precise retrieval of relevant information, Convolutional Conditional NPs [37] that introduce translation equivariance as an inductive bias and NPs based on Transformer architectures [38].

Hybrid Methods: While FSL is applied to scenarios where data is scarce, there is a silent assumption that this data scarcity holds for labeled data. However, there are numerous cases where even though labeled data is limited, unlabeled data is abundant. In recent years, there has been an increasing interest in the development of novel approaches that make use of this type of data leading to the development of novel hybrid approaches that incorporate different learning paradigms within FSL. A typical scenario in this case is the combination of meta-learning with different established learning approaches. Specifically, unsupervised meta-learning [39] has emerged as a promising direction where meta-learning is directly applied to unlabeled data to learn how to extract their underlying shared representation. This can also be used as a form of model pretraining, that can subsequently be used for rapid adaptation in downstream tasks with a few labeled examples, leading to models that combine meta-learning with semi-supervised [40] and self-supervised learning [41]. Finally, meta-learning has been utilized alongside multi-task learning to expand its functionality under multiple objective functions [42] or learn how to better adapt to novel tasks given their datasets' intrinsic similarities and disparities [43].

Applications: While deep learning research has recently focused on scaling models as well as the datasets used in training these models, this type of scaling also comes with drawbacks regarding the collection, storing, and processing of these data, acting contrary to the goals of efficiency, scalability and real-world ease of applicability. At the same time, the advent of the Internet of Things (IoT) has led to the existence of numerous similar datasets that might be small in size but can be used jointly to enable efficient training and increase the reusability of the final AI models. With these considerations in mind, there has been a proliferation in the different types of applications that FSL has been applied to spanning from computer vision [44], to natural language processing (NLP) [45], and robotics [46]. Additionally, FSL has also been extensively used in healthcare applications [47]

where data privacy regulations impose restrictions on data availability as well as in industrial settings to improve fault diagnosis in cases of data paucity [48].

In the realm of TALON and its corresponding use cases, FSL and TL approaches can play a crucial role in developing solutions that are effective, energy efficient, scalable, and reusable bridging the gap between performance and sustainability of AI models. In particular, deployed models can leverage previously acquired knowledge and use it to quickly adapt to novel settings, that are associated with novel training datasets, without the need for extensive retraining. This greatly enhances the robustness of deployment under various settings and enables the dynamic reusability of the deployed AI model by mitigating any need for training from scratch every time it is deployed for a new case. At the same time, it allows the deployment of AI models in cases where data collection might be costly or even impossible, paving the road for the application of AI in novel settings where data is scarce. Finally, one of the main challenges of modern AI models is their high computational needs during their training. Specifically, modern neural networks have reached the scale of millions and billions of parameters necessitating the use of powerful hardware such as GPUs that are particularly energy demanding. By leveraging FSL and TL techniques, models can be trained just once and then adapted to novel downstream settings and tasks with only minor modifications, leading to significant energy savings and promoting an AI learning paradigm that is sustainable and environmentally friendly.

The reusability of AI models is evident not only in cases where the model at hand has to be utilized in a new environment or learning problem but also when the problem conditions for an already deployed model may change. This is directly applicable in the deployment scenario related to UC3, where computer vision algorithms are applied in the context of object detection and scene analysis. Industrial environments are characterized by constant changes that may include the introduction of novel objects that were not included in the training dataset of the computer vision model or for which there might not exist enough training samples. By combining the inductive biases of FSL and TL techniques with the ability of continual learning via class-incremental learning methods, deployed AI models can be adapted to these novel settings requiring only a minimal retraining procedure. This largely minimizes the downtime of these models and allows them to be reused under these new conditions without extensive training, making them more efficient and robust to changes in the environment. Therefore, FSL and TL can assist in adding flexibility to traditional manufacturing systems by rapidly adapting to changing demands and promoting sustainability by curtailing energy consumption.

Finally, the capabilities of FSL and TL can be crucial in situations where AR is employed to instruct individuals on how to safely and efficiently operate machinery used in production processes, in a way that is more streamlined and user-friendly. Additionally, they can also be utilized in the realm of VR for the industrial maintenance of equipment. For instance, in this scenario, a maintenance technician or support specialist can offer technical support, troubleshooting, and maintenance services to clients or organizations from a remote location.

Overall, the aforementioned benefits of this few-shot, transfer, and hybrid learning framework are highly aligned with TALON architecture's objective to enable dynamic reusability of datasets, algorithms, and models while prioritizing energy efficiency. More precisely, this learning framework constitutes part of the AI Cognition Layer that is amenable to the training of AI models for each given task and enables runtime adaptations based on smart decisions.

s

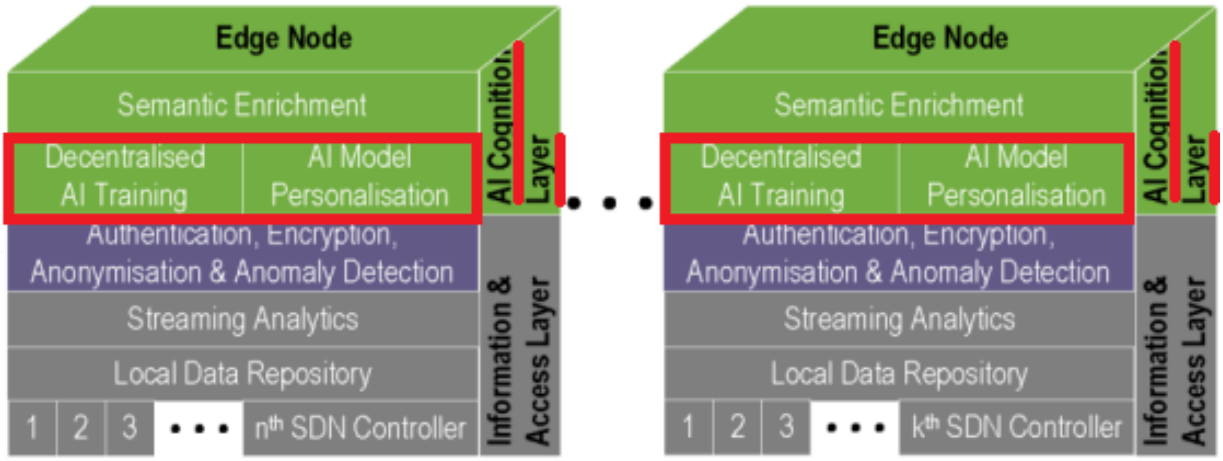


Figure 1: FSL and TL methods assist in enhancing AI model training in the AI Cognition Layer of each edge node

In agreement with TALON's goal to bring cloud computing services closer to the end users, the AI models to be used in each application are deployed on the edge nodes following a decentralized architecture based on FL principles. Since these AI models are deployed on the edge, training also takes place on the edge. Consequently, FSL and TL approaches can play a crucial role in minimizing both the required amount of data as well as the computational power needed to train these models on edge devices for which hardware restrictions may apply. This is evident in Figure 1 where FSL techniques can assist in making AI training procedures in the AI Cognition Layer, such as decentralized AI training, and AI model personalization more data and energy efficient, and the corresponding models more easily reusable and robust to changes in their environment. In particular, when presented with a new learning task associated with a dataset with limited samples, the AI models can leverage knowledge extracted from previous executions so that they maintain high performance in the novel tasks despite the limited number of available data. At the same time, in cases where deployed models need to be readjusted to changes in their environment (e.g., a previously unseen object that needs to be detected by a deployed computer vision model), this can be achieved with minimal retraining by combining previously acquired knowledge with a small number of new training samples. Therefore, these techniques are critical for the AI Cognition Layer's functionalities, as part of an architecture that supports high energy efficiency and dynamic scalability and reusability.

3.1.2 Communication Interfaces

TALON architecture provides both centralized and decentralized AI model training functionalities leveraging a FL approach. In principle, AI models are deployed on the edge nodes of this decentralized approach and utilize FSL and TL techniques to minimize the number of available data needed for training as well as maximize the energy efficiency of the whole procedure. In this framework where FL is combined with FSL and TL, the goal is to minimize the amount of data that needs to be shared in each deployment scenario to each one of the involved nodes. In Figure 2, a high-level overview of the combined FL framework alongside the FSL techniques used is provided. It is evident that FSL is used on the edge nodes, yet there is direct communication with a central server that is responsible for providing the edge nodes with the necessary model artifacts and obtaining the metrics related to the outcome of the deployed models. This is also aligned with the AI Cognition Layer's functionalities that are present in both edge nodes as well as the central server.

On a higher level, the FSL and TL functionalities constitute part of the algorithms provided within the AI Cognition Layer, which are interconnected regarding the flow of information (e.g., data streams) and leverage AIOps for model delivery. As input, they receive the AI models that are stored in a centralized AI model repository as well as the curated small-size datasets associated with the application tasks at hand and they produce the updated models that have been readjusted using these datasets. At the same time, metrics associated with the performance of the model that are of interest to the relevant stakeholders as well as metrics that can be used as input during the resource optimization procedure that is executed by the AI Orchestration Layer are also produced. Next, the relevant components that are in direct communication with the FSL, TL, and hybrid learning framework are presented, as well as the overall flow of information across these components.

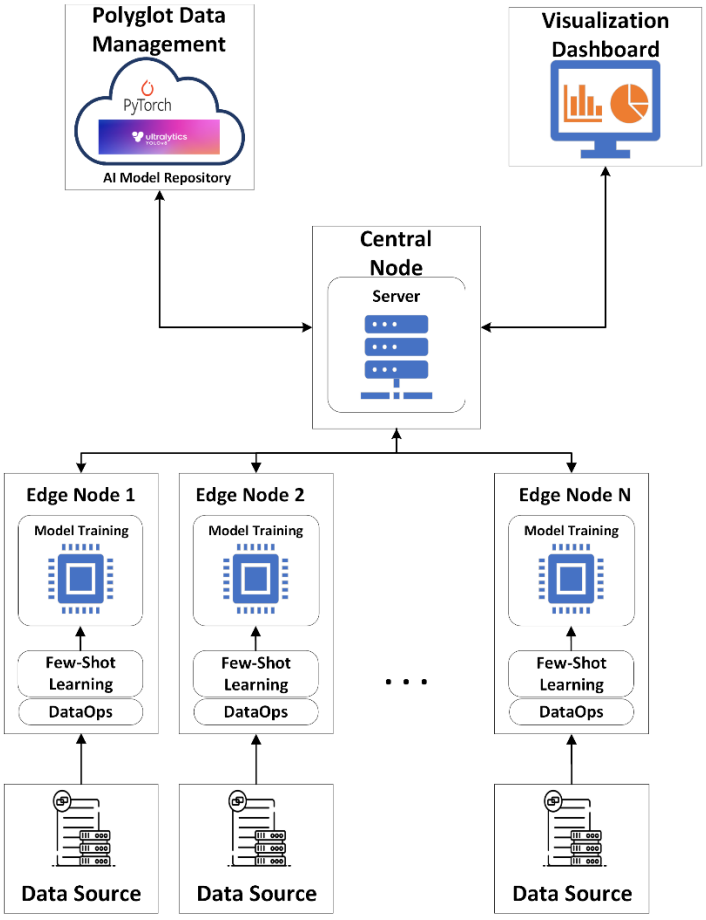


Figure 2: High-Level Overview of FSL's positioning within TALON's E2C architecture

Polyglot Data Management: This constitutes a centralized component responsible for storing the AI models to be used in each application. To optimize each AI model for the given tasks, the model is retrieved from the polyglot data management where it is stored and passed to the edge nodes.

AI Cognition & DataOps: The next step is to retrieve the data to be used in the optimization of the AI model. While data is commonly present in the edge nodes where the AI models are deployed, overall preprocessing is necessary to ensure its high quality that will enable the development of effective models efficiently. This data lifecycle constitutes of several subcomponents of the AI Cognition Layer including data ingestion of the raw data, data curation which is responsible for the veracity, timeliness, and transparency of the data, data encryption and anonymization that ensure alignment with imposed privacy constraints, and semantic enrichment that promotes data interoperability and reuse.

Visualization Dashboard: After optimization and execution of the AI models based on the FSL and TL framework, the outcome of the deployed final models is exposed to the visualization dashboard to provide useful insights to the relevant stakeholders. More specifically, this requires aggregation of the relevant information from the edge nodes to the central server. Subsequently, the user has access to a unified user interface where they have provision over the decentralized components of interest via analytics reports.

Overall, it is clear that while FSL, TL, and hybrid learning functionalities constitute part of the AI Cognition Layer, the interconnected nature of TALON's architecture necessitates the development of suitable communication interfaces via Application Programming Interfaces (APIs) and connection points which are currently in progress. It is also important that the development of these interfaces takes into consideration the use of appropriate authorization techniques that would ensure security and robustness under different deployment environments. This approach would help meet the needs of the different platform stakeholders and as a result, it is developed in accordance with the needs of TALON's use case demonstrators.

3.1.3 Scientific and Technical Results

As already mentioned, the main functionalities of the FSL, TL, and hybrid learning framework can find immediate application to the deployment scenario related to UC3. More precisely, it is essential that computer vision algorithms used in object detection tasks can detect novel objects when presented with only a minimal set of these objects' images. In the case of UC3, the models are deployed in an industrial setting where their goal is to detect objects related to the existence or not of protective equipment by the personnel as well as the existence of different hazards, such as fires that might pose a threat for these workers. Detection of these objects can be performed using modern object detection algorithms and, in this case, the YOLOv8 object detection model is utilized due to its efficiency and effectiveness.

Dataset: Having selected a suitable model for the task at hand, the next step is to define an appropriate dataset that will be used during model training and evaluation. Towards this end, two different datasets are examined, specifically one that contains images related to construction safety (CS) and a second one that contains images of protective equipment (PPE). To ensure that the proposed model is capable of learning under data scarcity, the number of images for each object is limited to 1, 2, 3, 5, 10, or 30 images, which corresponds to a scenario of 1-shot, 2-shot, 3-shot, 5-shot, 10-shot or 30-shot object detection. It is also worth noting that these curated datasets have the role of local data sources for each edge node, as they have been presented in Figure 2.

Training Process: In the realm of TALON, the success of the deployed models does not solely depend on their performance accuracy but also on their ability to be trained and deployed sustainably, promoting greener AI approaches. Towards this end, TL in the context of FSL is selected as the most appropriate training method, since it allows the development of accurate solutions in a data and energy-efficient manner. More precisely, a pretrained version of YOLOv8 is utilized, which has been trained in a large-scale object detection dataset and can be used off the shelf for object detection tasks. In the context of TALON, this pretrained model is saved in the AI Model Repository and is retrieved and passed to each edge node when needed. After retrieval of the pretrained model, the next step is finetuning the model on the local small-size dataset derived from the node's associated data source. One of the main challenges that arise when attempting to finetune an AI model in such a small dataset is overfitting, which could lead to model performance deterioration. To mitigate the risk of overfitting, a common approach is to freeze the network and finetune only a fraction of its parameters. The YOLOv8 model provides a natural way of splitting the network into smaller components since it consists of a feature extractor backbone and an object detection head that can further be split into different detector modules. As a result, three different finetuning approaches are

tested: finetuning the whole AI model, finetuning only the object detection head, or finetuning only the detector modules. Additionally, each of these models is trained under two different scenarios. In the first one, the model is evaluated on a held-out test after each training epoch to select the best-performing model, while in the second one, the model is trained without further evaluation. While the latter model may be susceptible to underfitting or overfitting if the number of training epochs is not carefully selected a priori, it is also true that the former is more energy and resource-demanding since it needs to be evaluated on a validation set of arbitrary size after each training iteration.

Evaluation Process: Based on both performance and sustainability-related factors, it is evident that model evaluation should be performed in two different axes. On one hand, high-level performance is crucial, especially in industrial settings and applications related to worker safety, and as a result, optimal models should demonstrate strong detection capabilities, measured using the mean average precision (mAP) metric. On the other hand, to ensure that the optimal models adhere to the principles of sustainable and Green AI, energy consumption during training is monitored. Energy consumption is also highly correlated with carbon dioxide emissions and as a result, minimizing the former also leads to the minimization of the latter. In general, it is expected that models that are more intensively trained achieve better performance. However, increasing the training time as well as the number of trainable model parameters results in greater energy consumption. It is clear that performance and sustainability are two conflicting factors and optimizing both of them at the same time is not always feasible. As a result, the goal is to find an appropriate trade-off between model performance and energy efficiency.

Experimental Results: Initially, the detection performance of the best finetuned models, based on the mAP metric, is examined for a different number of available training samples for each object class. The relevant results can be seen in Figure 3, where it is evident that increasing the number of shots boosts model performance. However, as for the finetuned strategy, it seems to be less relevant in most cases.

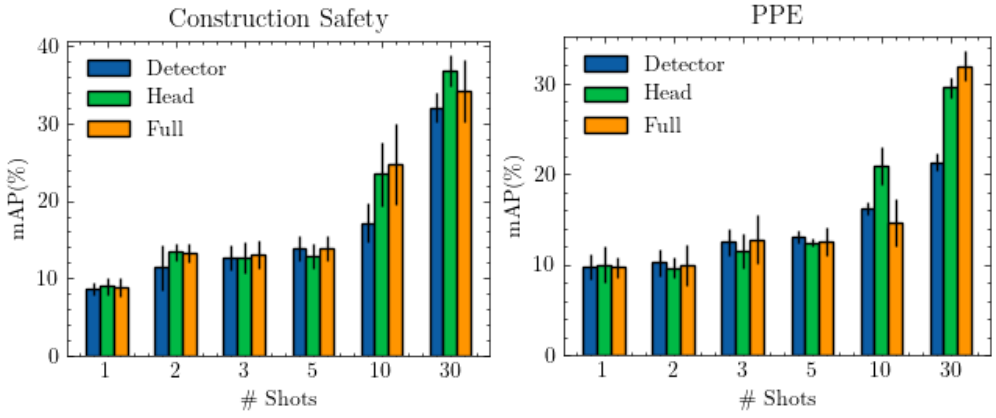


Figure 3: Performance of finetuned models for different number of shots

It is also worth examining whether searching for the best-performing model using a validation set in each training iteration also results in better performance during testing. Consequently, the performance between models that only differ in the use of the validation dataset is compared and the results are presented in Figure 4 for the CS dataset, and Figure 5 for the PPE dataset. Based on the resulting bar plots, it is evident that using a validation set to select the optimal model leads to a small performance boost in most cases, with the effect being more noticeable in the PPE dataset.

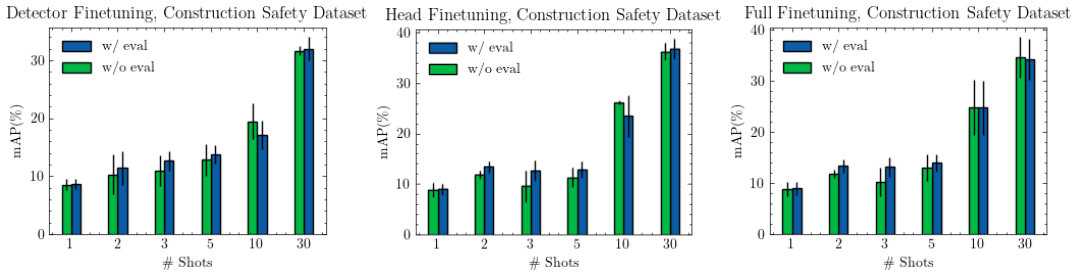


Figure 4: Performance influence of using an evaluation set in the CS dataset

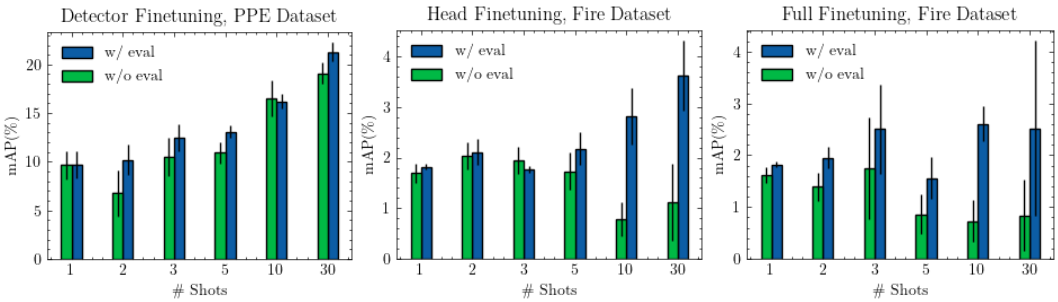
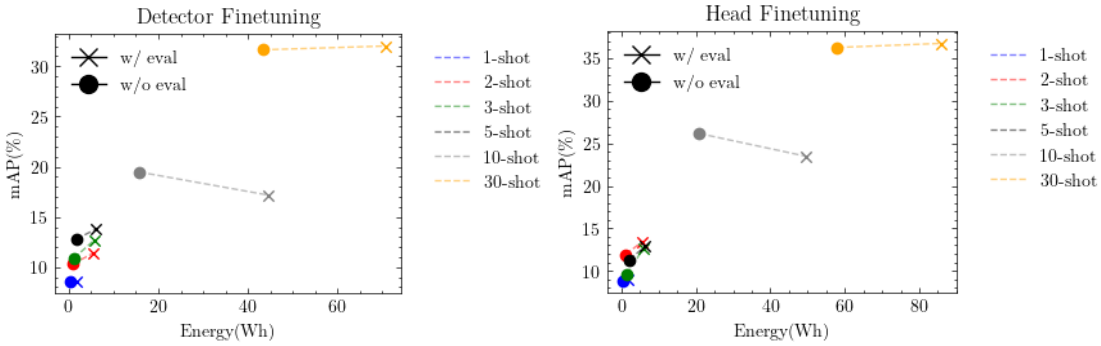


Figure 5: Performance influence of using an evaluation set in the PPE dataset

While performance is a critical component of the model evaluation process, it is important to be framed within the Green AI framework, and more specifically, energy consumption during training. Following the two axes framework presented above, in Figure 6 and Figure 7, scatterplots of the model performance with respect to the energy consumed during training are presented for each of the examined datasets. More specifically, in each scatterplot, models that have the same components finetuned yet differ in the number of shots used during training as well as the existence of a validation set during training are compared. The optimal models should demonstrate both high performance (higher mAP) as well as lower energy consumption and therefore should be closer to the upper left corner of the diagrams.



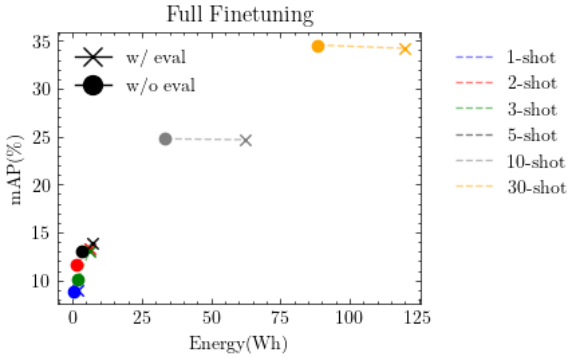


Figure 6: Model performance with respect to energy consumption in the CS dataset

Overall, it is evident, that there is a clear trade-off between performance and energy consumption since the best-performing models require more intensive training and thus consume more energy. As a result, further research is needed to determine solutions that can further push the boundaries of effective Green AI models. Yet, the fact that even the most energy-consuming models can achieve decent performance while remaining on the scale of less than 100Wh is very encouraging.

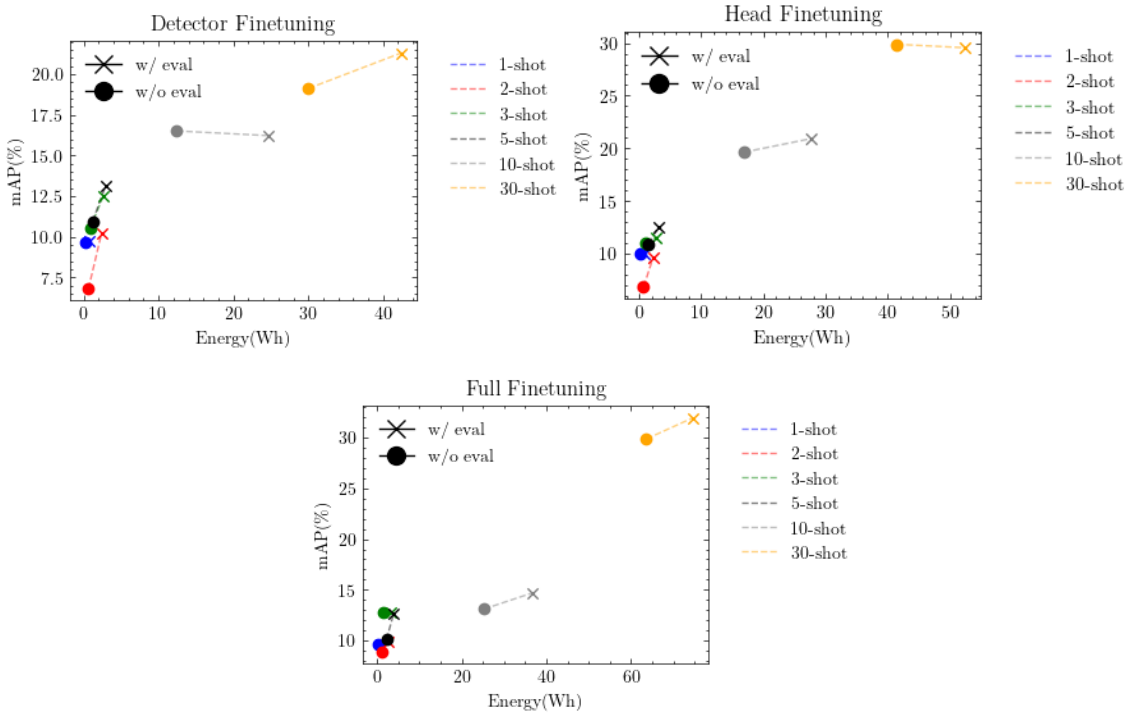


Figure 7: Model performance with respect to energy consumption in the PPE dataset

Finally, it would be interesting to put into perspective how different finetuning techniques affect both performance as well as energy consumption. Specifically, different finetuning approaches translate to different numbers of trainable parameters, with detector module finetuning having the least, head module finetuning being in the middle, and full model finetuning incorporating the most. It is expected that increasing the number of parameters would lead to greater energy consumption. However, there is not an obvious correlation between the number of parameters and model performance that can be assumed beforehand. In Figure 8 the performance of the best models that have occurred following the three different finetuning strategies, for a varying number of shots, in each dataset, is presented, once again elucidating the trade-off between model performance and power consumption. However, it is notable that when the number of shots is increased, finetuning the head module of the model

seems to constitute a good compromise between the two conflicting optimization objectives. As a result, further research is required towards implementing optimal strategies based on the partial finetuning of AI models.

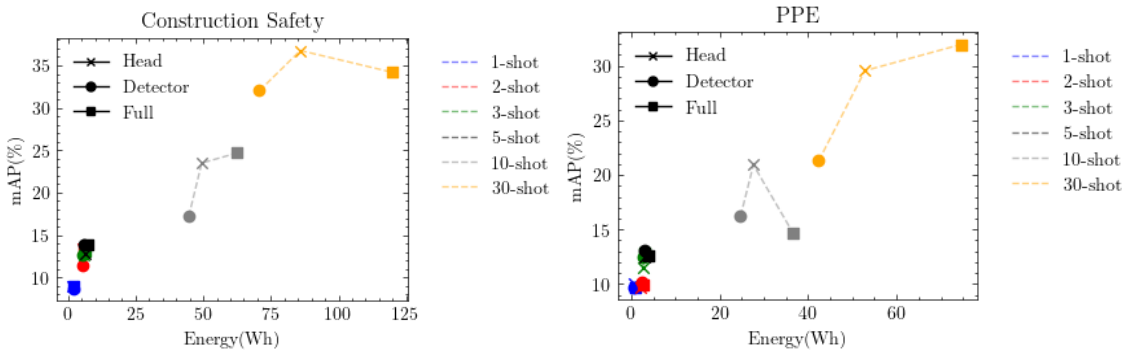


Figure 8: Model performance with respect to energy consumption for different finetuning strategies

3.1.4 Deployment Results

Based on TALON's architecture that supports a decentralized framework where AI functionalities are brought closer to the end-users, the FSL, TL, and hybrid learning functionalities are deployed on the edge nodes. This is done to facilitate their utilization within each given task without causing any additional burden due to data transfer, which would be suboptimal in terms of security and energy efficiency. Consequently, it is critical that these edge nodes where the FSL and TL framework are deployed can support its functionalities.

In general, FSL aims to produce AI models that are scalable and reusable, therefore it is logical that the framework that supports it is also lightweight and easily reproducible. Currently, the two main components the framework comprises are the training of the YOLOv8 model and its evaluation process. As for model training, this is done using Ultralytics' training framework in Python and leveraging the pretrained weights that are publicly available. Additionally, the fact that only a minimal number of samples is used during finetuning obviates the need for using energy and computationally intensive devices such as GPUs, making the framework applicable to cases where there are hardware constraints imposed. As for the evaluation procedure, the main dependency is the use of CodeCarbon, a lightweight Python library that enables the measurement of energy consumption as well as carbon dioxide emissions for various types of hardware.

In total, the deployment of the framework on edge nodes necessitates that it is as lightweight as possible so that it can even be deployed in environments where there might be hardware limitations. Currently, the FSL, TL, and hybrid learning framework is in a pre-deployment phase since it constitutes a heavily research-oriented task that requires further exploration into finding a novel optimal solution. However, the solutions included in this framework are developed with reusability, scalability, and optimal deployment considerations in mind.

3.2 Semantic Enrichment and Alignment

3.2.1 Internal Architecture and Key Technologies

Using semantic enrichment plays crucial roles in ensuring system's efficiency. Semantic enrichment can significantly enhance the quality of collected TALON data by adding context, meta-data, and intelligence to the initial data. Such metadata makes it possible for the end users to find, understand, and use it easily, regardless of their specific domain knowledge about the data. It can support intelligent analysis, decision-making, and policy formulation, all crucial for achieving the goal of TALON. In addition, in the context of AI/ML, the development of accurate and effective models is

assisted by metadata that can provide essential information about the structure of the data. For example, understanding the semantic relationship between the features, predictive maintenance which is the aim of TALON UC2: I5.0 Automation & Planning, can be improved, reducing waste, and extending equipment lifespan. In a nutshell, the semantic enrichment of TALON's data targets the data reusability and reduction of learning latency, which is one of the aims of TALON O-5: To enable reusability of datasets, algorithms, metrics and models.

To achieve the data enrichment, data pre-processing needs to be done first. Feature selection or feature extraction are key components of this phase, helping to reduce the dimensionality and focus on the most informative features within the dataset.

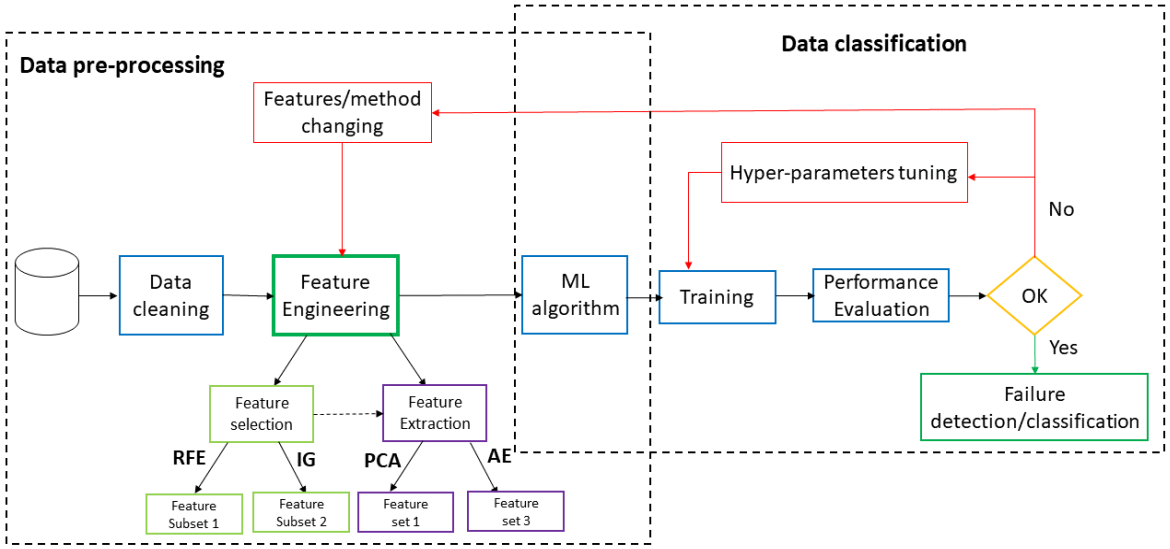


Figure 9: Structure of the experimental methodology

Figure 9 describes the methodology followed in this study, which includes two main processing steps: (i) data preprocessing and (ii) data classification. In this work, we focus on data pre-processing and especially feature engineering task in order to improve the classification performance.

Data preprocessing is a data mining technique that is used to make the data suitable for analysis and knowledge discovery [49]. It is a preliminary step that can be done through data cleaning & features engineering.

Data cleaning: As the dataset consists of different features with values on different scales, it needs to be scaled. This can be done by Min-Max normalization to perform feature scaling. It is a technique that scales every feature of our dataset between 0 and 1, the maximum value of that feature gets transformed to 1 and the minimum value gets transformed to 0. Correlation filtering is used to remove the redundant features and hence reduce the over-fitting of the classifiers as well as decrease the complexity of computation. It uses the correlation coefficient that indicates the linear correlation between two random features. The values of correlation cor range from −1 to 1. Usually, if $|cor| > 0.5$, this means that f_i and f_j have a strong correlation; if $|cor|$is close to 0, means that there is no linear correlation between the features. In fact, features are usually designed with their unique contributions, and removing any of them may affect the training accuracy to some degree [50]. That is why we have deleted just one redundant feature. In other words, we deleted every one of the two

features that have a correlation |cor| = 1, which means that they are totally redundant. Features Engineering: is the data of your data set to improve ML model training, leading to better performance and greater accuracy.

Features engineering is a crucial step in the ML model training. It aims to improve the model performance by crafting informative features. It involves both feature extraction and feature selection, and the choice between them depends on the nature of the problem.

A. **Features selection methods**: Try to pick a subset of features that "optimally" characterize the target variable. Features selection is the process of selecting the best features in a given initial set of features that yield a better classification performance, regression, and finding clusters efficiencies [49]. They can be distinguished into two broad categories, which are filters and wrappers. Wrapper Methods require a learning algorithm to use its performance as the evaluation criterion (e.g. classifier accuracy). One of the well-known wrapper methods is Recursive Feature Elimination (RFE) [51]. Starting from all the feature sets, RFE recursively removes features in order to maximize accuracy. Then it ranks the features based on the order of their elimination. On the other hand, filter Methods find the best features set by using some independent criteria (i.e. Information measures) before applying any classification algorithm. Information Gain Attribute Evaluation (IG) [52] is one of the most used filter methods [53]. The main concept of this approach is to rank subsets of attributes by calculating the IG entropy for each attribute in decreasing order.

B. **Feature extraction methods**: Feature extraction performs a transformation of the original variables to generate other features using the mapping function that preserves most of the relevant information. This transformation can be achieved by a linear or non-linear combination of original features. In this work, we will use the most frequently used feature extraction methods such as Principal Component Analysis (PCA) [54]. PCA has been widely unitized in feature extraction and data compression to reduce computational complexity, distractive noise, and the risk of over-fitting, as well as its calculation flexibility and reversibility [54]. The purpose of PCA is to extract new features called principal components (PCs) (less or equal to the initial features,) which are the linear combinations of the original attributes, orthogonal to each other, and capture the maximum amount of variation in the data [55].

All experiments employed Python as the programming language, Scikit-learn for ML models, and Pandas for data manipulation, as well as Matplotlib for results visualisation.

3.2.2 Communication Interfaces

For the communication interface, we will use REST API for features engineering tasks using the output from the data curation task as input to our API. This will be implemented and showcased accordingly in deliverable D4.3, which pertains to the finalised version of the technical developments of the TALON platform.

3.2.3 Scientific and Technical Results

Research results

The purpose of this experiment is to select relevant features from the initial dataset to constitute a reduced dataset. In this context, a comparative analysis of IG and Recursive Features Elimination (RFE) has been conducted.

Dataset description

MetroPT is a dataset for predictive maintenance, collected in 2022, which is an outcome of a Predictive Maintenance project with an urban metro public transportation service in Porto, Portugal

[56]. Its goal is to predict the failure in a metro system at least two hours in advance. Table 1 presents the dataset details.

Table 1: Dataset details

Number of features	21
Number of samples	10,773,588
Output	Failure
Output values	0: Normal, 1: two hours before failure, 2: failure state

As shown in Figure 10 increasing the number of features does not always improve the classification performance. Thus, Finding the significant input features to the model is an important task.

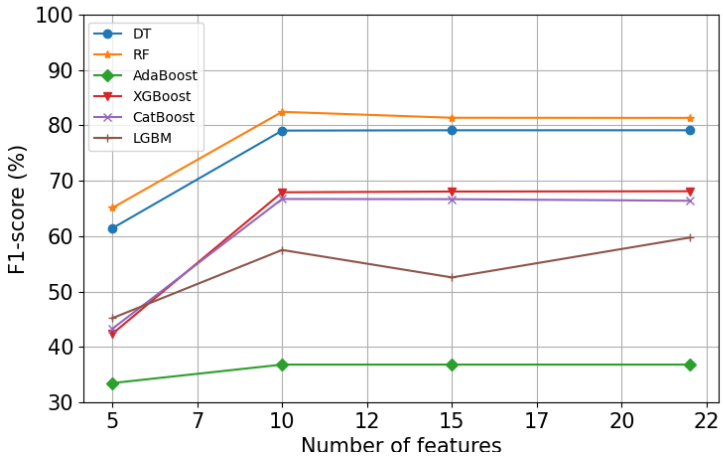


Figure 10: Classification performance with different features set

Failure Classification performance with Feature selection methods:

Table 2 presents the feature rank of the two feature selection methods: RFE and IG. To further clarify, rank 1 is correlated to the most relevant features based on the used features selection methods. Using these ranks we evaluated the classifiers with two feature selection methods on different reduced versions of the data. We tested feature subsets containing the top 5, 10, and 15 features. Also, we test the accuracy and efficiency of the classification with the entire data against the best-selected feature subsets selected by RFE or IG.

Table 2: Features ranking of our dataset using RFE & IG methods

Features Name	Description	Rank with RFE	Rank with IG
TP3	Pressure generated at the pneumatic panel.	1	11
Oil_temperature	Temperature of the oil	2	8
H1	Valve	3	10

Motor_current	Motor's current	4	7
Flowmeter	Airflow	5	1
Reservoirs	Pressure inside	6	2
TP2	Pressure on the compressor	7	12
DV_pressure	Pressure exerted	8	3
gpsLat	Latitude position	9	4
gpsLong	Longitude position	10	6
Towers	Signal	11	17
gpsSpeed	Speed (km/h)	12	9
DV_eletric	Electrical signal	13	5
MPG	Responsible for activating the intake valve	14	18
COMP	Electrical signal of the air intake valve on the compressor	15	16
LPS	Signal activated when the pressure is lower than 7 bars	16	13
Caudal impulses	Signal produced by the flowmeter	17	14
Pressure_switch	Signal activated when pressure is detected on the pilot control valve	18	19
Oil_level	The oil level on the compressor	19	15

As Decision Tree (DT)-based models are the simplest classifiers, we applied them to train and test the entire and the reduced dataset with feature selection methods. The obtained results are presented in Table 3, Table 4 and Table 5. These results indicate that the majority of DT-based models can perform better with the reduced features set, especially those of RFE, than with the entire dataset in terms of accuracy, precision, recall, and F1-score. Specifically, in contrast to the entire dataset and IG method, RFE with just 10 features gives the best failure classification results. A reason for these results is that not all features in our dataset help to separate classes during the classification task. Moreover, RFE provides a feature ranking based on their importance within the context of the selected model and this ranking can help to identify the most relevant features for the model.

Table 3: Classification performance with entire dataset

Model	Accuracy	Precision	Recall	F1-score
DT	98.14	78.37	79.91	79.12
RF	**98.75**	**95.23**	**73.96**	**81.36**
Adaboost	97.78	53.56	35.57	36.81
XGBoost	98.26	91.27	60.67	68.11
Catboost	98.28	91.61	57.89	66.39

LGBM	98.11	83.42	53.38	59.74

Table 4: Classification performance with the best features subset (top 10) using RFE

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
DT	98.14	78.39	79.77	79.06
RF	**98.79**	**95.33**	**75.32**	**82.45**
Adaboost	97.78	53.56	35.57	36.81
XGBoost	98.26	91.27	60.41	67.92
Catboost	98.29	91.81	58.22	66.71
LGBM	98.08	78.99	52.33	57.51

Table 5:Classification performance with the best features subset (top 15) using IG

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
DT	97.92	75.85	78.06	76.92
RF	**98.75**	**95.27**	**73.77**	**81.26**
Adaboost	97.78	53.56	35.57	36.81
XGBoost	98.26	91.26	60.54	67.97
Catboost	98.29	91.93	57.75	66.34
LGBM	98.13	86.09	55.28	61.80

In order to illustrate the impact of feature selection on ML models, Figure 11 presents the training and test time with the entire and reduced (with features selected by RFE) dataset of the DT model. It can be seen that the feature selection method especially RFE finds the most relevant features subset and in turn decreases the time required for training.

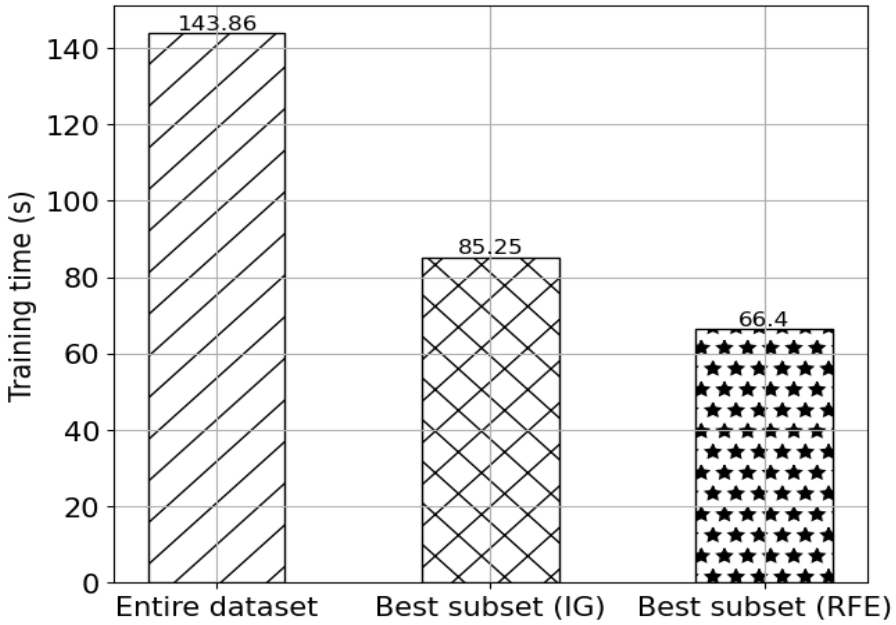


Figure 11: Training time comparison

Failure Classification performance with Feature extraction method (PCA)

Table 6 presents the results of the newly generated feature using PCA. To note here, we evaluated the performance of the different DT-based classifiers using different amounts of extracted features but we present only the best results which is the 19 extracted features. It can be seen that PCA outperforms RFE and IG with the majority of classifiers. However, with the PCA requiring 19 features, it does not lead to feature reduction in the same way as RFE. Using more features can overfit the model as well as increase the training and inference time. RFE provides a more interpretable feature subset, as it identifies which features are retained. However, one of PCA's limitations is that the information about how the initial features contribute is often lost. Thus, the choice between them depends on various factors, including the modeling objective, interpretability requirements, and the trade-off between dimensionality reduction and predictive accuracy.

Table 6: Classification performance with the extracted (19) features using PCA method

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
DT	98.5	81.65	82.3	81.98
RF	99.02	96.65	79.13	86
Adaboost	97.87	60.57	35.02	36.18
XGBoost	**98.33**	**94.71**	**60.78**	**96.36**
Catboost	98.32	92.84	55.31	64.62
LGBM	98.07	77.46	52.59	57.31

4. Future directions:

After identifying the optimal 10 features via RFE method, we are set to enrich the failure prediction dataset. This enrichment involves:

- Seasonal Feature Creation: Integrating time-based features to capture seasonal effects that may influence failures state.
- Weather-related feature integration: Adding external weather data such as temperature and humidity to consider environmental impacts on system performance.
- Cross-correlation features: Establishing features based on the cross-correlation analysis between some features, revealing hidden patterns and dependencies critical for predicting failure more accurately.

Also, we will possibly use manufacturing ontologies to further enhance the analysis by providing a semantic framework that identifies patterns over time and might signify a potential failure.

Regarding actual use case data, as one of the objectives of our future work, we will provide enrichment to a dataset related to UC 2: Automation & Planning. In particular, the FACT team will provide time-series data concerning different manufacturing components of the Nakamura 2 machine. Those data will be enriched so that they assist the prediction of component failures.

Also, we started to apply data enrichment for image data, related to fire detection, which is relevant to TALON UC1: UATVs coordination. In the particular UC, a set of drones fly in a cubical formation above a small area and they take images. These images are dispatched to Command & Control center for fire & smoke detection.

As the first step in the process, we generated a diverse set of images from the original collection of drone image data. This is achieved by using various transformations including: adjusting the orientation, changing the scale, and the images colour. These augmentations aim to create a robust dataset that captures the multifaceted nature of image fires. In future work we will try to make more transformations to the initial images. On the other hand, by leveraging the power of the generative AI, we plan to convert the visually enriched fire images into descriptive text. For example, based on the image analysis, the model will generate accurate and detailed textual descriptions, encapsulating the characteristics and the nuances of each fire scene.

3.2.4 Deployment Results

The data enrichment component's role is to augment and improve data quality by adding relevant information, which can significantly increase its usability for analytical purposes. Therefore, we will provide *user-friendly interface* for data enrichment components to ensure that the end-users can effectively interact with this service. A *user-friendly interface* can bridge the gap between powerful backend data processing capabilities and the end-users who leverage these capabilities to drive value from their data.

3.3 Data Curation

3.3.1 Internal Architecture and Key Technologies

The data curation module has been designed to address data curation problems inside the TALON Architecture by applying a series of algorithms to TALON datasets. Considering its architecture (Figure 12), the data curation module is composed of three main components:

- The APIs backend provides the API needed to find the right curation algorithm to apply, based on the inputs from the XAI module. Moreover, they allow to start a curation task and monitor its progress.
- The Curation engine applies the algorithms to the datasets themself. It consumes messages from a Kafka queue and writes back results to the proper database. MongoDB is used for

tabular datasets and time series, while Minio is used for filesystem-structured datasets (e.g., image datasets).

- The status watcher, whose role is to check statuses of curation jobs and update a DB consequently.

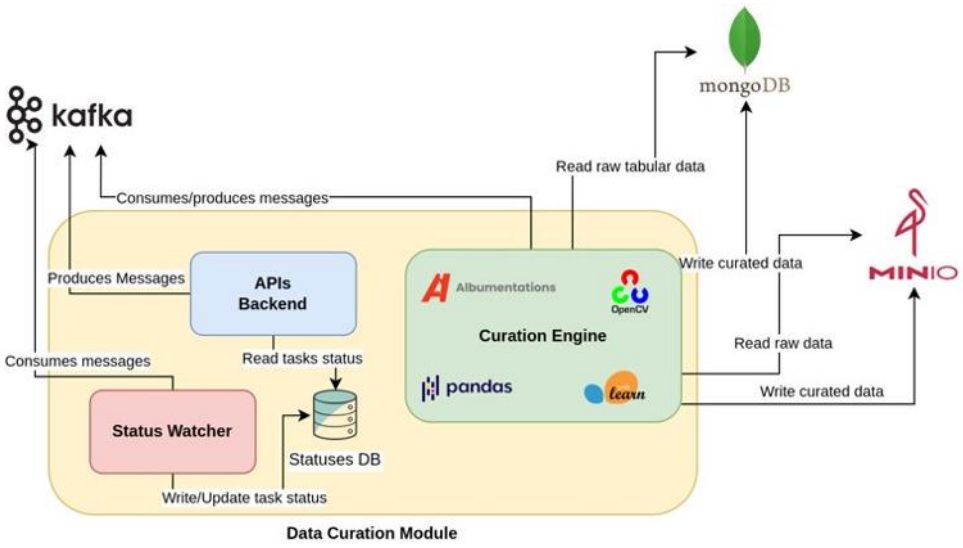


Figure 12: Data Curation Architecture

The core of the curation engine is its catalogue of algorithms, organized in a hierarchical tree-like structure. The main goal is to allow tree-like navigation based on a starting point – suggested by XAI module – and user inputs. User inputs assist in fine-tuning the use of the available algorithms in order to optimize data preprocessing.

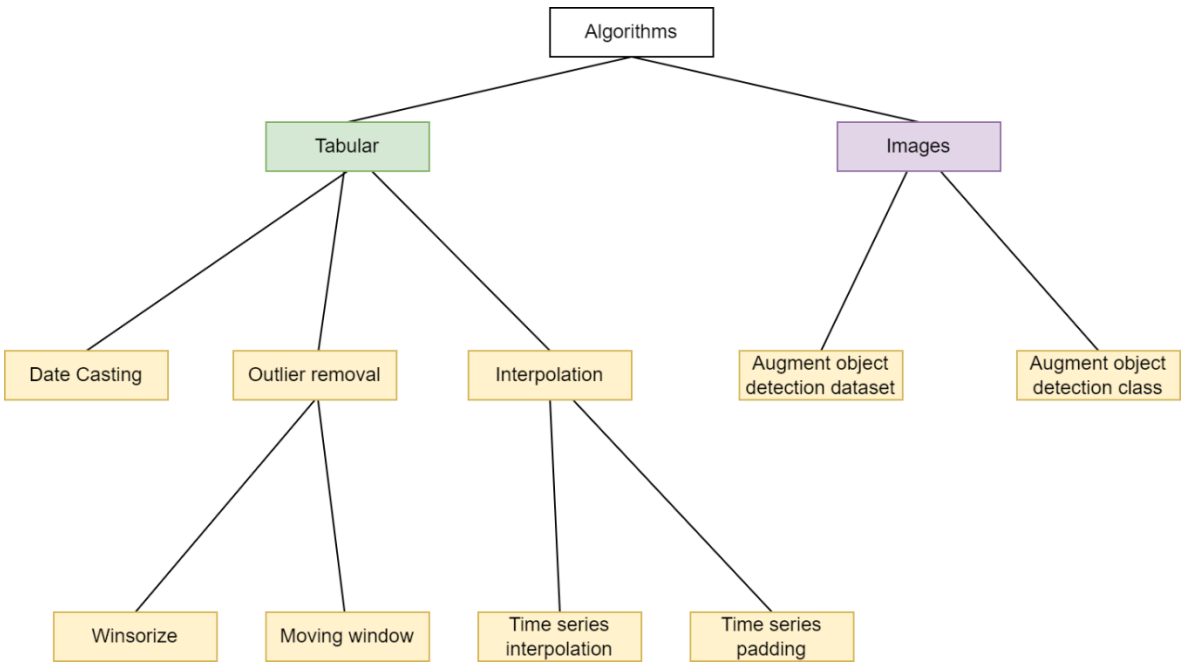


Figure 13: Hierarchical structure of Data Curation Core

Developed algorithms include:

- Applicable on **tabular dataset**:
    - Date casting algorithm: this can be used to preprocess dates based on an initial and targeted format
    - **Winsorize**: this is an outlier removal algorithm that works by limiting extreme values, after statistically analyzing the signal itself (implemented with Scipy[1] function *scipy.stats.mstats.winsorize*[2])
    - **Moving window**: another outlier removal algorithm. it utilizes Z-score calculated with a rolling window and a threshold in order to remove outliers. The Z-score test is a commonly used statistical method for identifying outliers in time series data. The approach we followed utilizes the normalized data and considers any data points above or below a certain threshold as outliers.
    - **Time series interpolation**: this is implemented by wrapping *Pandas*[3] function *pandas.DataFrame.interpolate*[4]. This way it is possible for the user to make use of any of the numerous options available in pandas itself, e.g., linear interpolation, nearest, polynomial interpolation etc.
- Applicable on **object detection datasets**:
    - **Data augmentation**: applies image augmentation transforms[5] (through Albumentation[6] library) to an entire folder or subfolder maintaining labeling information (or accordingly transform them based when geometric transformation is applied).
    - **Balance classes**: same as above, but it allows to specify a specific class to augment, while ignoring the others, as well as a percentage (or a specific number) of augmentations to generate.

3.3.2 Communication Interfaces

The main link between the Curation Module and the rest of TALON architecture is represented by its APIs (Figure 14). There are four APIs available:

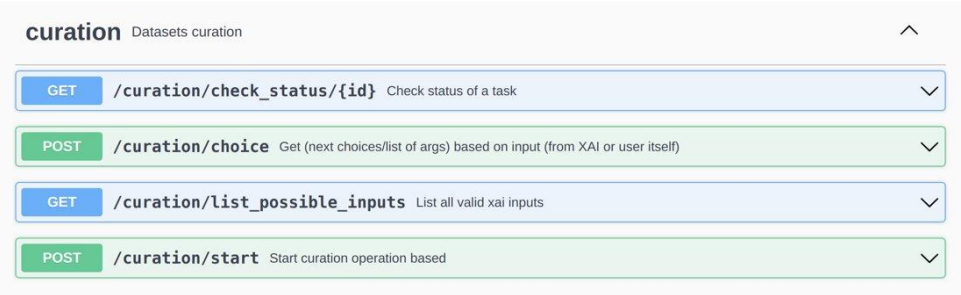


Figure 14: Curation Module Swagger

[1] https://docs.scipy.org/doc/
[2] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.mstats.winsorize.html
[3] https://pandas.pydata.org/docs
[4] https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.interpolate.html
[5] https://albumentations.ai/docs/api_reference/augmentations/transforms/
[6] https://albumentations.ai/docs

The first one is used to list all of the possible valid inputs. The curation module expects indeed an input from the XAI module, containing information on possible data imperfections (e.g., incomplete time series, unhealthy image datasets, presence of outliers, unbalanced datasets etc.). In order to check that an input is formatted as expected, this first API (Figure 15) can be used. This also gives back a useful description for each possible input keyword, as in the example below (note that the example includes only a subset of the final available algorithms).

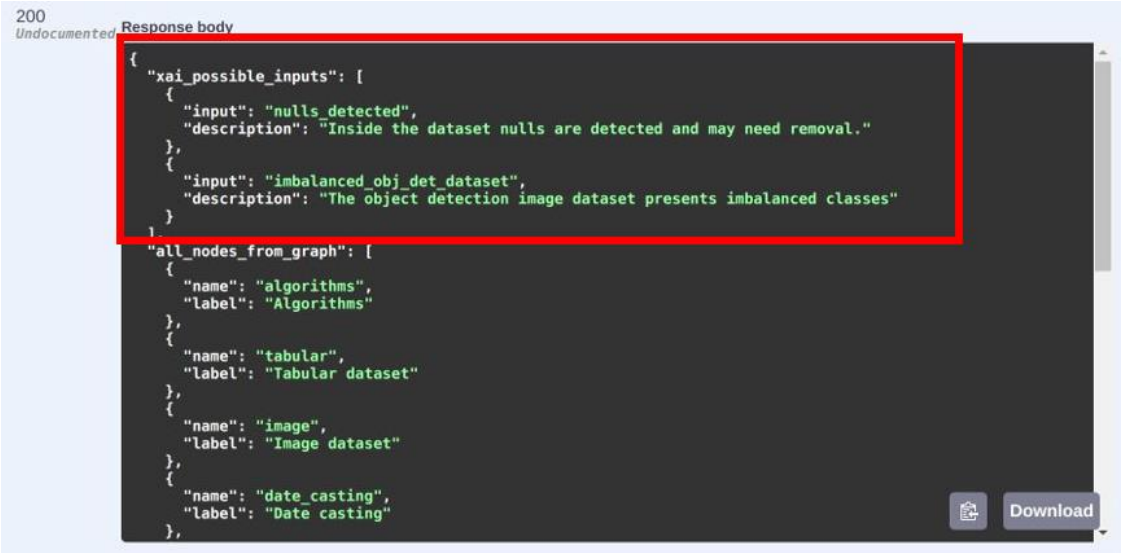


Figure 15: Input Validation API

The second API is used to navigate the algorithms graph. Given a certain input, next choices are returned, helping the user configure the algorithm. Once the algorithm configuration is ready, this can be started using a third API endpoint.

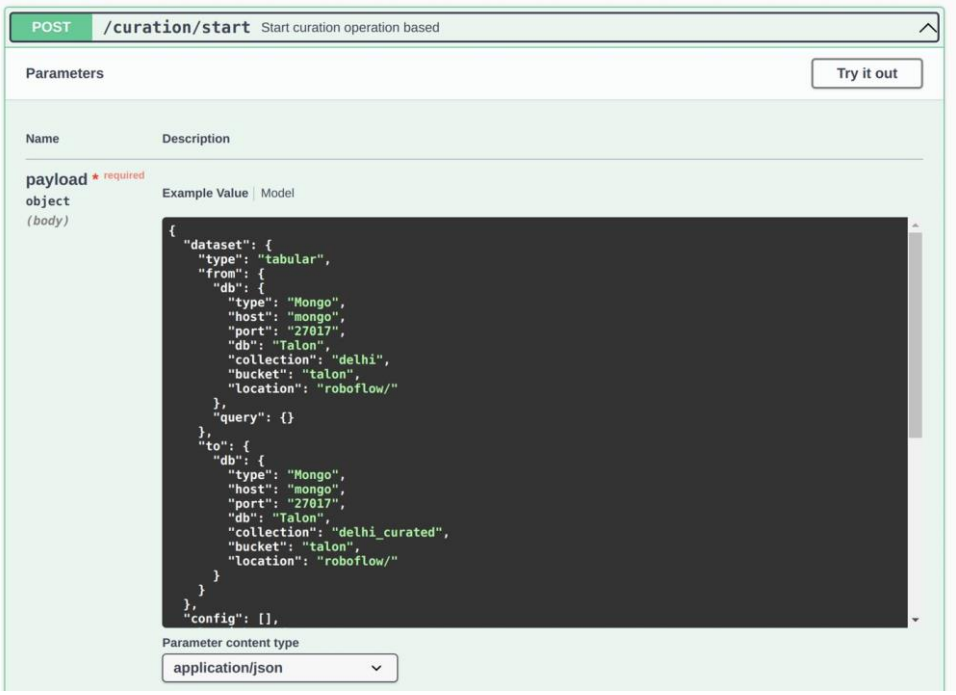


Figure 16: API endpoint for starting a curation job

A final API endpoint can be used to monitor the advancement of a started task as well as its errors in case there are any.

3.3.3 Scientific and Technical Results

Here the results from two algorithms will be presented. The first example shows the application of the class augmentation algorithm on a single image (for demo purposes). From one image four new ones are generated, applying a series of transforms associated with a probability. For an extensive list of transforms check https://albumentations.ai/docs/. Thanks to a series of configurations, it is possible to augment classes, by a certain number of images. Performing an augmentation on a dataset that is going to be used to train an object detection model helps in increasing robustness of the model and making it properly generalized, avoiding overfitting.

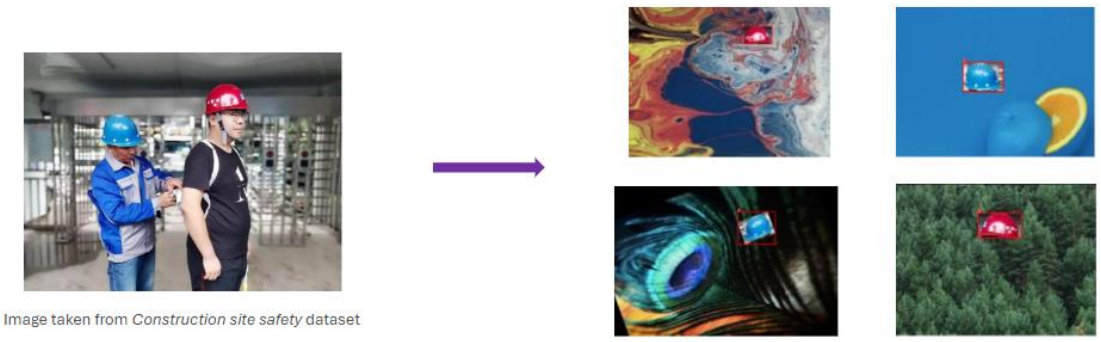


Figure 17: Example of class augmentation

The second example shows how a curation algorithm can be used for recovering missing values in a time series. More specifically the plots below show the application of a linear regressor curation algorithm to replace null values. The Delhi climate dataset has been modified, removing values that have been restored using curation module. As before a series of parameters can be used by the advanced user to fine tune the algorithm; otherwise default values can be used to simplify the process and still get results.

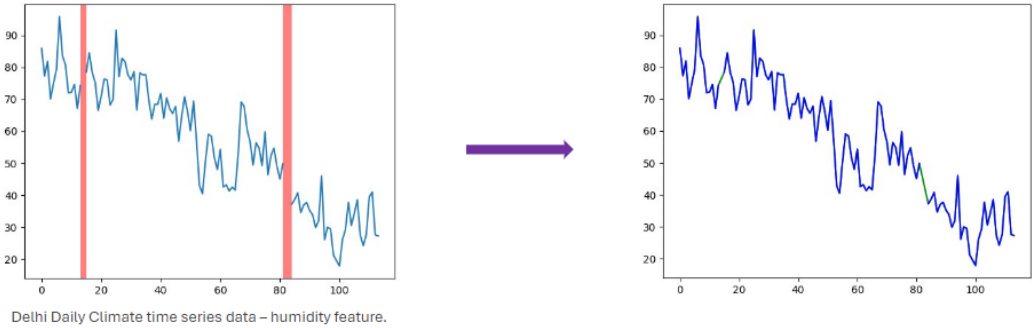


Figure 18: Linear interpolation on time series

As can be seen from the plot on the left, some values are missing (columns in red). On the right plot those values are restored using curation algorithms. In particular a linear interpolation algorithm has been used.

3.3.4 Deployment Results

The curation module is composed of containerized microservices, easily deployable on any cloud environment. Here are its requirements:

- An instance of MongoDB
- An instance of MinIO
- An instance of Apache Kafka

MinIO, an S3-compatible storage system, proves invaluable for image data storage and seamlessly integrates with ML libraries. ML frameworks like TensorFlow and PyTorch natively support S3, treating it as a filesystem. This simplifies data access, enabling smooth implementation of ML pipelines, especially models' training.

MinIO's scalability ensures efficient handling of growing datasets, vital for training ML models. Parallel processing capabilities accelerate feature extraction and model training, while S3's consistent organization facilitates standardized data curation and batch data modification.

Since the curation module elaborates tabular data, a second storage is needed. While Minio can be used to store csv files, Mongo proves to be a better choice for tabular data, especially for its data querying capabilities. MongoDB's scalability ensures efficient management of growing datasets, a crucial aspect when dealing with extensive tabular data. The ability to handle complex queries and indexes facilitates the quick retrieval and processing of information, streamlining data curation algorithms applied to tabular datasets.

Furthermore, MongoDB's support for dynamic schemas accommodates evolving data requirements during the curation process. This flexibility is beneficial when adapting to changes in data structure or introducing new attributes. The database's indexing capabilities enhance the performance of data retrieval, a key factor in optimizing data curation algorithms for tabular data.

Since the curation engine work is regulated by a jobs' queue, a message broker both capable of allowing communication between microservices and storing queues' information is needed. Inside the curation module architecture, Apache Kafka is used for this purpose. It serves indeed as a powerful and efficient tool within the microservices architecture, handling communication and job queues for the various data curation tasks. The rest of the server needs to transmit that a curation job must start. For this purpose, a KAFKA queue is used, maintaining scalability and reliability, providing a distributed, fault-tolerant messaging system. Moreover, the asynchronous nature of kafka messaging helps in making the system more reliable.

3.4 Self-healing and Self-correcting

The scope of the Self-healing and Self-correcting tool will be to monitor, identify and suggest solutions to the network users and nodes solutions in order to mitigate the different problems that are identified. The Self-healing and Self-correcting has a twofold purpose, one is to identify the existence of a network problem and classify it properly and at the same time analyse the problem and suggest a solution in order to reduce the effect of the problem, to prevent it from happening in the future. The application of the Self-healing and Self-correcting mechanism will increase the resilience of the network and by extension the overall performance of the network and the facility. At the same time this solution will increase the efficiency of the system as the network nodes will cooperate seamlessly. More specifically, TALON's self-healing is of vital importance in order to make sure that the network

communications are working seamlessly. This is important in order for the rest of TALON modules to work properly.

The Self-healing and Self-correcting tool will consist of an AI method that will be able to detect problems on the network and next another method will suggest a potential solution for healing and correcting the issue. In order to train the AI model to detect the network problems, there is first the need for identifying the most common network problems. Below the three most common problems in a computer network are analysed, which are latency, CPU usage and data package size to transfer [7].

Latency in computer networks refers to the delay or time taken for data to travel from the source to the destination. Excessive latency can lead to various problems that impact the overall performance and user experience within a network. One significant issue is the degradation of real-time communication applications. In applications such as voice and video calls, or equipment communication, low latency is crucial for maintaining smooth and seamless interactions. High latency can result in delays, lag, and disruptions, leading to a poor user experience. This is particularly critical in scenarios where quick and timely responses are essential, such as an industrial environment for time sensitive applications[8].

Another problem associated with latency is its impact on data transfer rates. High latency can lead to slower data transfer speeds, as the time taken for data packets to travel between devices increases. This is especially problematic in large file transfers, streaming services, and cloud-based applications. Users may experience delays in accessing or retrieving data, leading to frustration and decreased productivity. Additionally, latency issues can exacerbate network congestion problems, as devices may have to wait longer to send or receive data, causing bottlenecks and reducing overall network efficiency[9].

Latency also poses challenges in distributed computing and remote server environments. In cloud computing, for example, applications and data are often hosted on servers located at a considerable distance from end-users. High latency in communication between the user and the cloud servers can result in slow response times and decreased performance for web applications. This can impact businesses relying on cloud services for critical operations, affecting productivity and user satisfaction. Minimizing latency is crucial for optimizing the performance of computer networks and ensuring a responsive and efficient digital experience.

High CPU usage in computer networks can lead to several problems that affect overall system performance and user experience. One primary concern is resource contention, where multiple applications or processes compete for the limited processing power of the CPU. When the CPU is heavily utilized, it may struggle to efficiently execute tasks from various applications simultaneously. This can result in slower response times, delayed data processing, and a general slowdown in system performance. Users may experience lag, unresponsiveness, and degraded performance, particularly in multitasking scenarios.

Another issue associated with high CPU usage is increased power consumption and heat generation. Modern CPUs are designed to operate within specific thermal and power limits. When the CPU is consistently operating at high usage levels, it can lead to elevated power consumption and heat production. This not only impacts the energy efficiency of the system but can also result in thermal

[7] *Characterizing software self-healing systems, Computer Network Security: Fourth International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security*
[8] *Network heal thyself [computer network management], Engineering & Technology*
[9] *AI-assisted Computer Network Operations testbed for Nature-Inspired Cyber Security based adaptive defense simulation and analysis, Future Generation Computer Systems*

throttling, where the CPU reduces its clock speed to prevent overheating. In the context of computer networks, especially in data centers where multiple servers are deployed, managing CPU usage is crucial to maintaining efficient energy consumption and preventing hardware failures due to overheating.

In networked environments, high CPU usage can also affect the responsiveness of network-related tasks. For example, routers, switches, and other networking equipment rely on the CPU for packet processing, routing decisions, and other critical functions. If the CPU is overwhelmed by excessive usage, it may struggle to handle network traffic efficiently. This can lead to network congestion, dropped packets, and increased latency, negatively impacting the overall network performance. Proper monitoring and management of CPU usage are essential to ensure the reliable and efficient operation of computer networks.

Variable data send rates in computer networks can introduce several challenges that impact the reliability and efficiency of data transmission. One significant issue is network congestion. When data is sent at variable rates, it can lead to uneven distribution of network traffic, causing congestion at certain points within the network. This congestion can result in packet loss, increased latency, and decreased overall network performance. Variable send rates can make it difficult for network administrators to predict and manage traffic patterns effectively, leading to suboptimal utilization of network resources.

Another problem associated with variable data send rates is the potential for inefficient use of available bandwidth. In scenarios where data is sent sporadically or in bursts, there may be periods of underutilisation followed by sudden spikes in network activity. This uneven utilisation can make it challenging to allocate resources effectively, leading to wasted bandwidth during periods of low activity and insufficient capacity during peak times. Proper network planning and traffic shaping mechanisms are crucial to address these issues, ensuring a more consistent and efficient use of available network resources.

Variable data send rates can also impact the Quality of Service (QoS) in real-time applications. Applications such as video streaming, online gaming, and VoIP communication require a consistent and predictable data flow to maintain a satisfactory user experience. Fluctuations in data send rates can result in jitter, buffering, and disruptions in real-time applications. To mitigate these issues, network administrators may need to implement QoS mechanisms that prioritize certain types of traffic or allocate sufficient bandwidth for critical applications. By addressing the challenges associated with variable data send rates, networks can provide a more stable and reliable communication environment for users and applications.

The size of data being transmitted in computer networks can introduce several challenges that affect network performance and efficiency. One significant problem is increased network congestion. Larger data sizes require more bandwidth for transmission, and when multiple devices are concurrently transferring large files, it can lead to congestion and slower data transfer speeds. This congestion not only affects the efficiency of data transmission but can also lead to delays, packet loss, and increased latency. Network administrators must carefully manage and allocate bandwidth to accommodate varying data sizes and prevent congestion issues.

Another issue associated with large data sizes is the potential for resource exhaustion. Network devices such as routers, switches, and firewalls have finite resources, including memory and processing capacity. When handling large volumes of data, these devices may become overwhelmed, leading to performance degradation and, in extreme cases, network outages. Efficient data handling mechanisms, such as data compression and decompression, can be employed to reduce the impact of large data sizes on network devices. Additionally, proper network design and hardware scaling are

crucial to ensure that the infrastructure can handle the increasing demands of data size within the network.

The size of data can also impact data storage and backup processes within the network. Large datasets require more storage space, leading to increased storage costs and potential challenges in data backup and recovery. Managing large amounts of data efficiently becomes crucial to avoid storage bottlenecks and ensure that backup processes are completed within acceptable timeframes. Network administrators need to implement effective data management strategies, including data deduplication and compression, to optimize storage utilization and streamline backup operations. By addressing the problems associated with data size, organizations can maintain a well-functioning and responsive network infrastructure.

3.4.1 Internal Architecture and Key Technologies

In this section the overall architecture of the Self-healing and Self-correcting tool will be presented and analysed. The Self-healing and Self-correcting tool will collect data from the network and also the operators of the TALON system in order to collect data related to user experience. Figure 19 presents the overall architecture of the Self-healing and Self-correcting tool. The collected data will be analysed by a LLM model and will be able to detect that there is a problem on the network based on a knowledge base of network problems dedicated to which decision is made on the particular type of failure and at the same time a suggestion on what should be the solution for mitigating this failure.

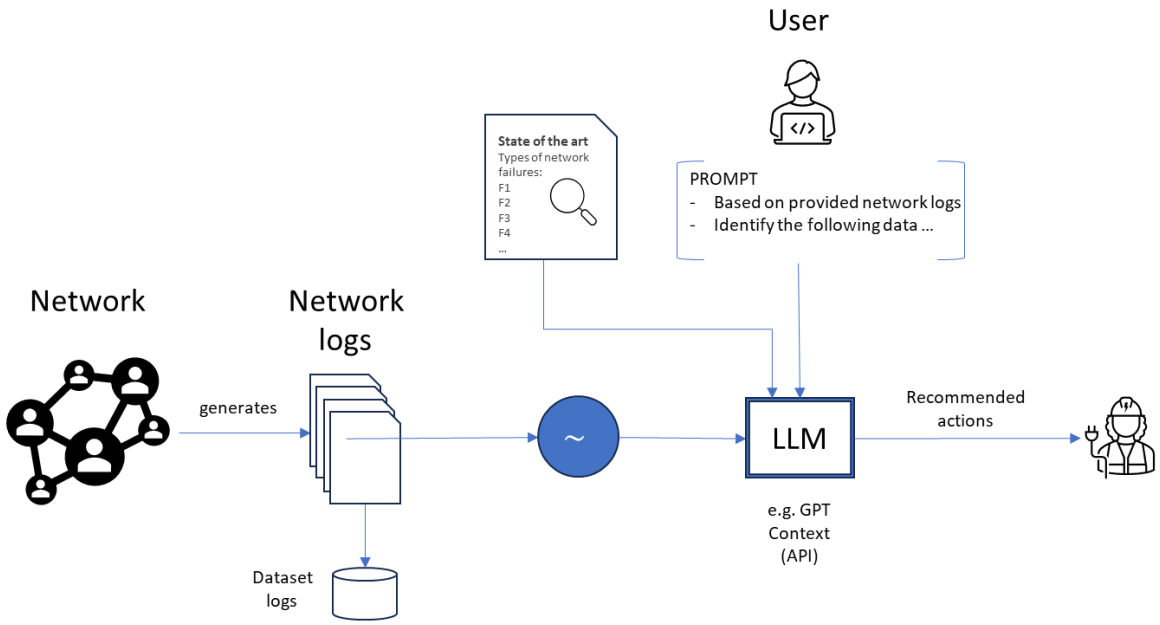


Figure 19: Self-healing and Self-correcting architecture

In order to address and improve high latency or CPU usage in a computer network, several potential corrections can be considered. These corrections involve a combination of hardware upgrades, network optimizations, and software adjustments to enhance overall performance. Firstly, upgrading hardware components can significantly contribute to reducing latency and CPU usage. This may involve investing in more powerful processors, additional RAM, or faster network interface cards. These upgrades can provide the necessary resources to handle increased network traffic and computational demands, resulting in a more responsive and efficient network. Secondly, optimizing

network architecture and configuration settings can play a crucial role in mitigating latency issues. Properly configuring routers, switches, and other network devices can help streamline data flow and minimize unnecessary delays. Implementing QoS policies can also prioritise critical traffic, ensuring that latency-sensitive applications receive the necessary bandwidth and resources.

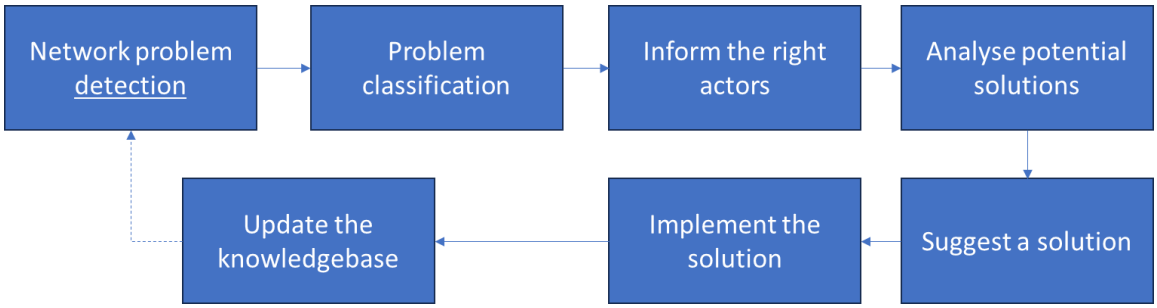


Figure 20: Workflow diagram for the self-healing module

Additionally, the use of Content Delivery Networks (CDNs) can be employed to distribute content strategically across multiple servers, reducing the distance between users and the content they are accessing. This can lead to faster load times and lower latency, especially for widely accessed resources such as web pages or streaming media. Furthermore, implementing efficient protocols and algorithms can contribute to a reduction in latency. For example, using Transmission Control Protocol (TCP) variants designed for low-latency environments, or employing advanced routing protocols, can enhance the efficiency of data transmission. Additionally, the adoption of compression techniques can reduce the volume of data transferred over the network, leading to lower CPU usage and faster response times. Lastly, regularly monitoring and analysing network performance can help identify and address latency issues proactively. Utilizing network monitoring tools can provide real-time insights into traffic patterns, bottlenecks, and potential sources of latency. With this information, administrators can make informed decisions to optimize the network and pre-emptively address any emerging performance issues. In summary, a holistic approach involving hardware upgrades, network optimization, protocol improvements, and proactive monitoring can collectively contribute to a more responsive and efficient computer network with reduced latency and CPU usage.

3.4.2 Communication Interfaces

The Self-healing and Self-correcting tool will communicate with the rest of the TALON components through specifically designed APIs, which are under development at the present time.

3.4.3 Scientific and Technical Results

At the current state there are no tangible results as the task is in design phase where the available data are analysed and the overall architecture is developed in alignment with the GA and the rest of the TALON components. The figures below illustrate the different data types that are available. More specifically, Figure 21 illustrates the CPU usage of the TALON toy application. In network communication, CPU usage is intricately tied to tasks such as packet processing, encryption/decryption, and protocol stack operations. The volume and complexity of network traffic, along with the efficiency of the network interface controller (NIC) and protocol stack implementation, contribute to the CPU workload. Additionally, interrupt handling, multithreading, and the presence of security software impact CPU usage. Optimizing network settings, considering hardware offloading options, and ensuring efficient buffer and queue management are key strategies to mitigate CPU load during network communication.

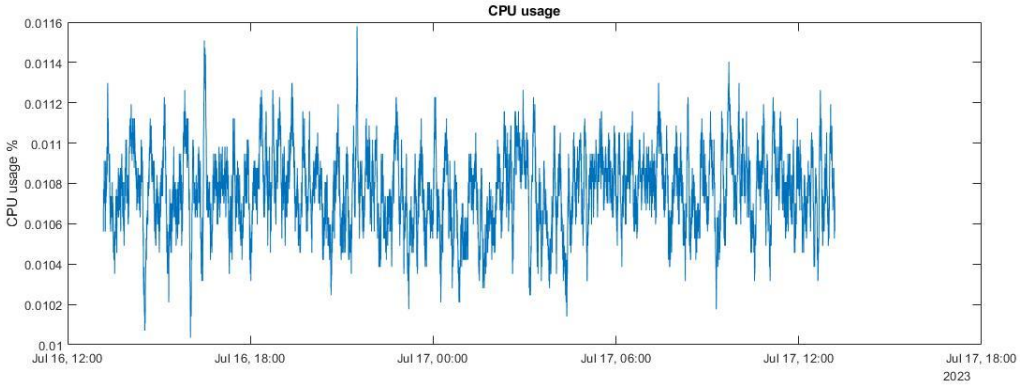


Figure 21: CPU usage for toy application provided by UBITECH

Figure 22 illustrates the send and receive data rates for the TALON toy application. In general, it can be observed that the data streams are steady and predictable, but there are several cases that the data flow is spiking, causing potential disturbance to the entire network as a big portion of the network bandwidth is consumed by only one application causing potential issues to the network.

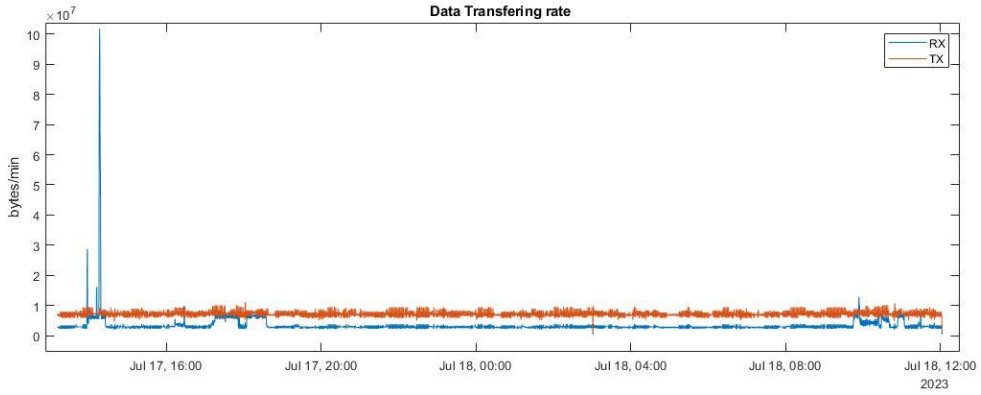


Figure 22: Send receive volume for toy application provided by UBITECH

Figure 23 illustrates the latency on the network communications. The average latency is nearly steady at 4.23 ms but there are several occasions that the latency surpasses 6ms. This is a result of intensive data communications during those times. This latency could create a serious problem with the overall response time of the TALON platform.

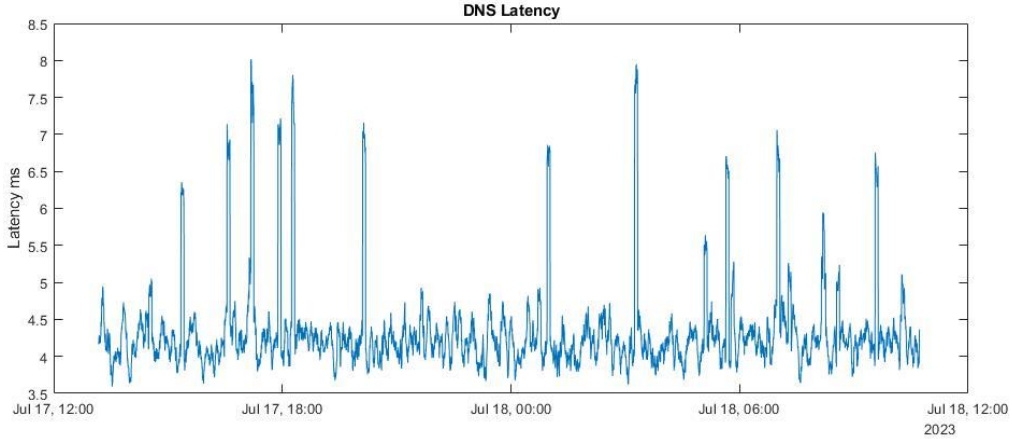


Figure 23: Latency for toy application provided by UBITECH

3.4.4 Deployment Results

Currently there are no deployment results, the future steps are to further analyse the TALON data and start the implementation of the events prediction, which will be the different events that can cause problems to the network. Once the prediction of those event is achieved, their occurrence will be correlated with the different activities happening in the network and therefore we will be able to know the root cause of the problem. Deployment phase and integration within the TALON platform will initiate immediately after, and the corresponding results will be showcased in the following D4.3.

4 Explainable AI Framework

Explainable AI (XAI) plays a critical role in the TALON project, aiming at revolutionising and increasing the transparency of the next-generation industrial systems that incorporate advanced technologies like AI, edge-to-cloud (E2C) computing, blockchain, and visualization. XAI is essential for attaining a high degree of explainability, trustworthiness, and transparency. It renders the operations, outcomes, and decisions of AI models to be transparent and comprehensive to human users, a fundamental requirement within industrial systems where AI decisions have a considerable impact. Understanding AI's decision-making processes significantly bolsters trust and reliability, a crucial aspect, particularly within industrial applications [57].

In TALON, XAI is pivotal in advancing intelligence to the network's edge, validating, and ensuring data sources reliability, understandability of AI decisions, and overall transparency throughout the full TALON system pipeline and operations. In particular, it is comprehensively structured across different Trust Levels (TrLs), each covering different phases of TALON's system pipeline, from analysis and validation of raw data through the explainability of predictions from AI models. These TrLs are categorized into two broad areas: explanations of data (TrL1 and TrL2) and explanations of model results (TrL3 and TrL4), dealing with two primary data types – images and time-series data [58].

4.1 Explanations of Data

4.1.1 Internal Architecture and Key Technologies

The first element of TALON's XAI framework is TrL1. It is focused on analysing, validating, and providing insights into the reliability of data sources and the overall quality of the dataset, covering both image and time-series data, utilizing different techniques. Starting from image data, ensuring consistency within a set of images is important due to its direct and indirect impact on other TALON components. Inconsistencies within images can introduce errors, inaccuracies, and uncertainties that compromise the integrity of the dataset. Inconsistencies can arise from various factors including image acquisition processes, lighting conditions, camera settings, and image processing techniques. Detecting and addressing these inconsistencies not only enhances the accuracy and trustworthiness of the dataset but also ensures the robustness and reliability of subsequent analysis, models, and applications built upon it, while by analyzing and checking raw image data for inconsistencies such as formatting errors, corrupted files, and images that are outside the shape thresholds, misleading interpretations and results are significantly reduced. Regarding time-series data analysis, the examination of the reliability of the dataset entails an analysis of interruptions in the incoming traffic within the data stream. More precisely, interruptions (null values), often resulting from data recording errors, sensor failures, transmission issues, or cyberattacks, can significantly distort the integrity of the dataset, and compromise the accuracy of analyses and predictions. The detection of null values is critical to maintaining the fidelity of time series data, as these gaps can disrupt temporal patterns and influence the results of certain statistical analysis. By continuously detecting and mitigating null values and interruptions, users uphold the reliability and credibility of the time-series data analyses, ensuring the accuracy and robustness of insights derived from pattern recognition and trends.

As already described, TrL1 receives raw data, either image or time-series data, and applies a set of techniques in order to provide the desired insights. Regarding image data, the algorithm receives the path to the directory of the stored images, applies the corresponding algorithm to check the images for inconsistencies regarding their shape, format, and size, and outputs the results. Regarding time-series data, the core process follows the same pattern, meaning that the algorithm receives raw time-stamped data, and outputs the instances in which values are missing. In both data types, the output is two-fold; a visualization plot and a JSON-formatted output that can be communicated and consumed by other components. Regarding the JSON-formatted output, TrL1 can provide the index

where null values appear, while for image data, the file names of the images and the number of inconsistencies per inconsistency type are provided. Concerning the visualization plots, a highly interactive plot illustrating the finding of the null detection algorithm in time-series data is generated in order for users to easily identify the time frames and stamps that data points are missing while regarding the data reliability of image data, two interactive plots are generated providing the proportion of consistent data against the inconsistent data, and a more thorough representation of the breakdown of the inconsistencies appearing in the dataset.

Following TrL1, TALON's XAI TrL2 has been developed. The main goal of TrL2 is to apply a set of techniques in order to detect imbalances in the dataset for image data and detect and report outliers in time-series data. Regarding the former, detecting imbalances in image data is crucial for ensuring the efficacy and fairness of ML models. Imbalances occur when certain classes or categories within the image dataset are disproportionately represented compared to others. This imbalance can lead to biased model training and false predictive outcomes, as the model may prioritize the majority class while overlooking the minority ones. In practical terms, this means that the model's ability to accurately recognize and classify images from underrepresented classes may be severely compromised. Moreover, imbalances can hinder the model's generalization capabilities, causing it to perform poorly on unseen data or in real-world scenarios where class distributions differ from those in the training dataset. By detecting and addressing imbalances in image data, ML models and other AI-related components of TALON can foster more equitable model training and improve the model's ability to accurately classify and interpret images across all classes, thereby enhancing the overall performance and reliability of the ML system. Bringing the focus on time-series data, outliers can have a significant impact on the accuracy and reliability of insights and predictive AI models. These anomalies are data points that deviate substantially from the majority of observations within the dataset. More precisely, outliers can influence the performance of predictive models by introducing noise and bias into the training process. ML algorithms may assign undue significance to outliers, resulting in models that fail to generalize effectively to new data or exhibit poor predictive performance. Therefore, identifying and appropriately handling outliers in time-series data is essential for preserving the integrity and reliability of analysis and predictions, enabling more accurate insights, and facilitating the development of robust predictive models.

From a technical perspective, TrL2 receives pre-processed data as input. This pre-processed data is the output of other TALON components within the AI Cognition Layer which receives as input both the findings of TrL1 and human decisions (Human-in-the-loop) and outputs the aforementioned pre-processed dataset. Then, TrL2 analyses the data using corresponding libraries and produces results based on the data type in a similar two-fold way as TrL1. Regarding image data, an interactive bar plot showing the number of instances per class is showcased along with a sample set of images for a better understanding of the dataset and a JSON-formatted output which includes the number of instances per class. Finally, concerning time-series data, TrL2 applies the Local Outlier Factor (LOF)[10] algorithm in order to find any outliers appearing in the dataset across all columns. LOF Identifies anomalies by assessing the relative density of an observation in relation to its neighbouring data points. Here is a brief overview of the mathematical formulation of LOF:

$$reachability - distance_k(A, B) = max\{k - distance(B), d(A, B)\}$$

The reachability distance from object A to object B is the actual distance between the two objects, but it must be at least the distance of B's k-nearest neighbour. Objects within the k-nearest neighbours

[10] https://scikit-learn.org/stable/auto_examples/neighbors/plot_lof_outlier_detection.html

of B are treated as equally distant. This approach aims to minimize statistical fluctuations between all points A near B, with higher values of k leading to a smoother effect.

The local reachability density for an entity A is formally established as follows:

$$lrd_k(A) := 1/(\frac{\sum_{B \in N_k(A)} reachability - distance_k(A,B)}{|N_k(A)|})$$

which is the reciprocal of the average distance from the neighbours of object A. Then, the local reachability densities are contrasted with those of the neighbouring points using:

$$LOF_k(A) := \frac{\sum_{B \in N_k(A)} \frac{lrd_k(B)}{lrd_k(A)}}{|N_k(A)|} = \frac{\sum_{B \in N_k(A)} lrd_k(B)}{|N_k(A)| \cdot lrd_k(A)}$$

which is calculated by dividing the average local reachability density of the neighbours by the local reachability density of the object itself.

4.1.2 Communication Interfaces

In this section, the focus shifts towards an overview of the API tailored to support TrL1 and TrL2 data analysis, visualization dashboards, and JSON-formatted insights within the TALON framework. Serving as a pivotal component, this API facilitates seamless communication between various elements of TALON, allowing for the generation, retrieval, and transmission of structured JSON insights ready for consumption by other components of the framework. This streamlined communication pathway enhances collaboration, expedites decision-making processes, and augments the overall efficiency of TALON's data analytics capabilities. The API structure is illustrated in the following figure.

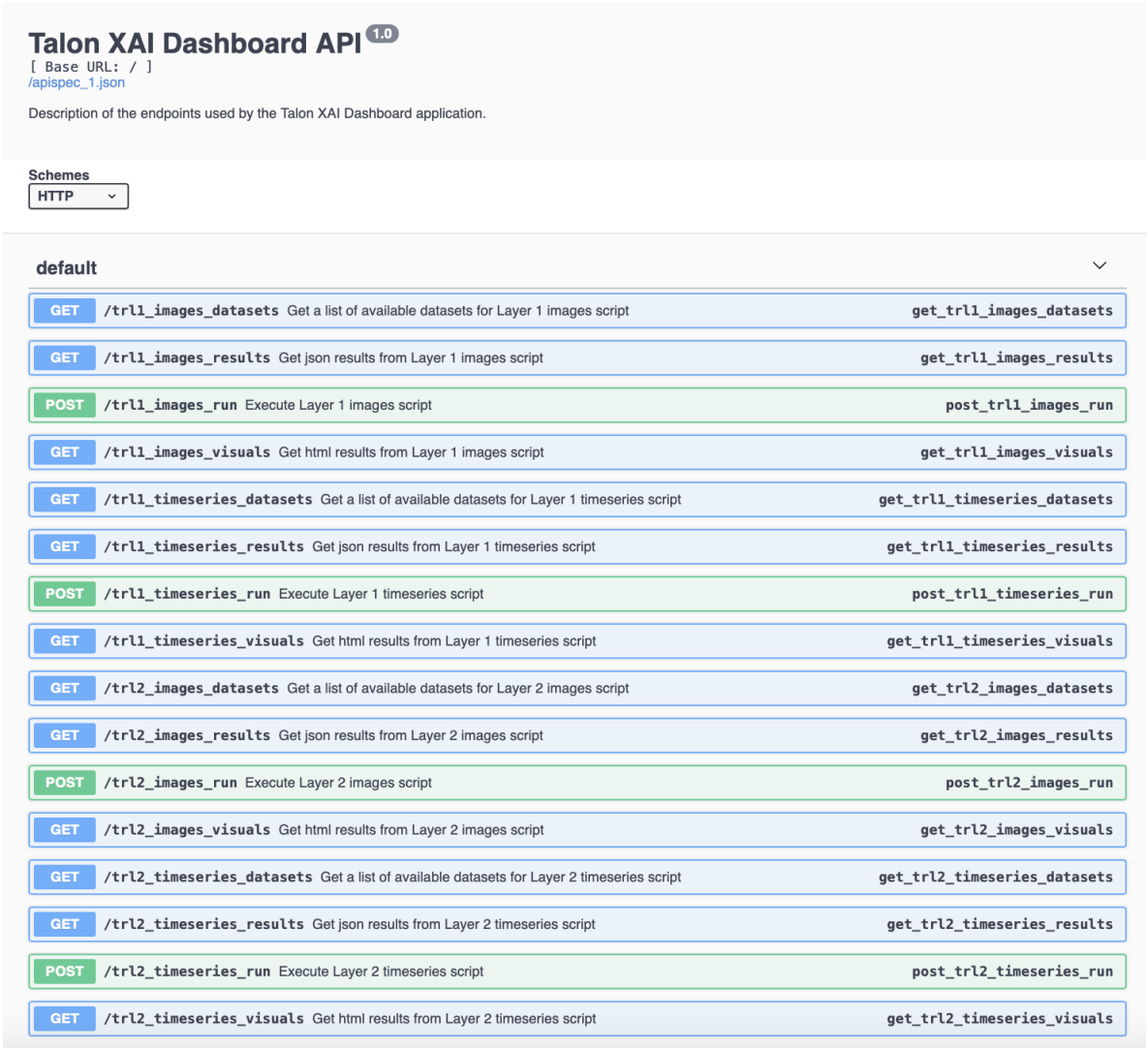


Figure 24: Talon XAI Dashboard API Schema

As it is shown in the API scheme, for every TrL and for every data type (image, time-series), a set of requests exists. In particular, a POST request executes the corresponding script, while three (3) different GET requests are focused on providing the visualization of the insights produced by the scripts, the JSON-formatted outputs that can be reached from other TALON components, and another GET request that returns the available datasets for every level and every data type that is mainly used for visualization purposes. This structured approach ensures systematic data access and processing, facilitating seamless integration with TALON's broader ecosystem and enhancing visualization capabilities.

4.1.3 Scientific and Technical Results

To showcase the effectiveness of TrL1 and TrL2 methodologies, a demonstration is presented using publicly available datasets. This demonstration highlights their capacity to improve data reliability, enhance model performance, and promote fair data representations in machine learning systems. This demonstration aims to emphasize the importance of these algorithms in improving data-driven decision-making processes and bolstering the reliability and trustworthiness of AI systems.

The first application domain of TALON's XAI module lies within the explanations and explorations of image data. Aiming to validate the developed techniques of TrL1 and TrL2 in the context of image data, two (2) different, but directly linked to TALON use cases, public datasets were selected. More precisely, the first dataset is called "Construction Safety" (available via Roboflow) which provides a set of annotated images featuring individuals working in construction environments. The objective of this dataset is to detect PPE, and especially, five (5) different classes (objects):

1. Person
2. Vest
3. Helmet
4. No vest
5. No helmet.

Aiming for a better understanding of the dataset, a sample of the images included is presented next:

Figure 25: PPE dataset examples

The results associated with the aforementioned dataset are illustrated in the following figures, in which the scripts for the detection of certain image inconsistencies, and imbalances were executed:

Figure 26: PPE Image Inconsistencies & Inconsistence breakdown GUI

Regarding TrL1, as shown in the previous figures, in 67.4% of the images no inconsistencies were detected, while in the remaining 32.6% the majority of the inconsistencies are located in the shape of the images, with formatting issues appearing in a small proportion, and no corrupted images were identified.

Regarding TrL2, which is concerned with detecting dataset imbalances, the results are summarized in the following plot:

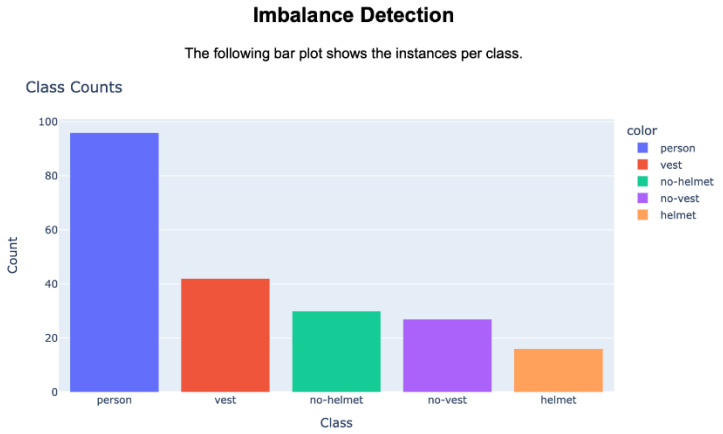


Figure 27: PPE Imbalance detection

As shown in the figure, the “person” class is the most represented class in the dataset, while class “helmet” has the smallest number of appearances. This indicates that an AI algorithm may struggle to identify the least represented class and add bias to the detecting model.

The second dataset that TrL1 and TrL2 were evaluated with, is called “Fire and Smoke Detection Dataset”. In particular, this dataset includes almost 400 images and the objective is to detect different objects (classes):

1. Fire
2. Vest
3. Helmet
4. Boots
5. Protective Glasses
6. Gloves
7. Mask.

A sample of images of the dataset is presented next:

Regarding TrL1, as shown in the following figures of Figure 28, the results indicated a greater percentage of “healthy” images, reporting 96,5%. The remaining 3,5% is split among the three inconsistency categories: shape, format, and corrupted files.

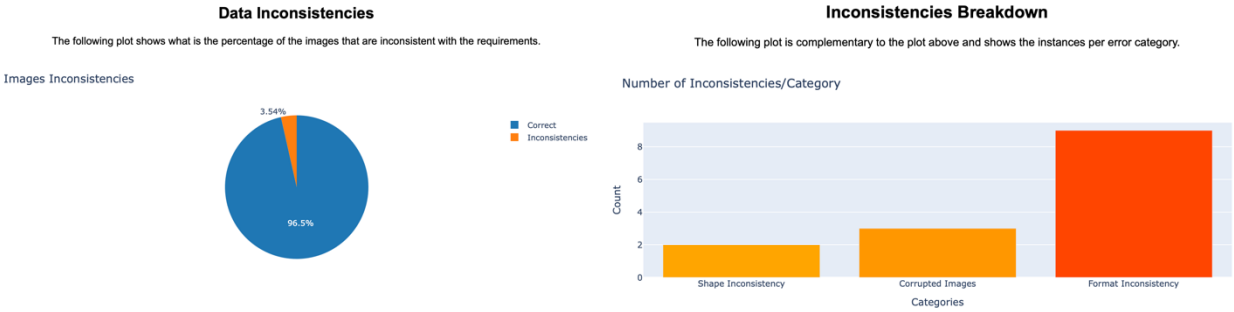


Figure 28: Fire and Smoke Detection Image Inconsistencies & Inconsistence breakdown GUI

Concerning TrL2, the class distribution shows that the class “Fire” is the most represented class, while the least represented class is “Mask”.

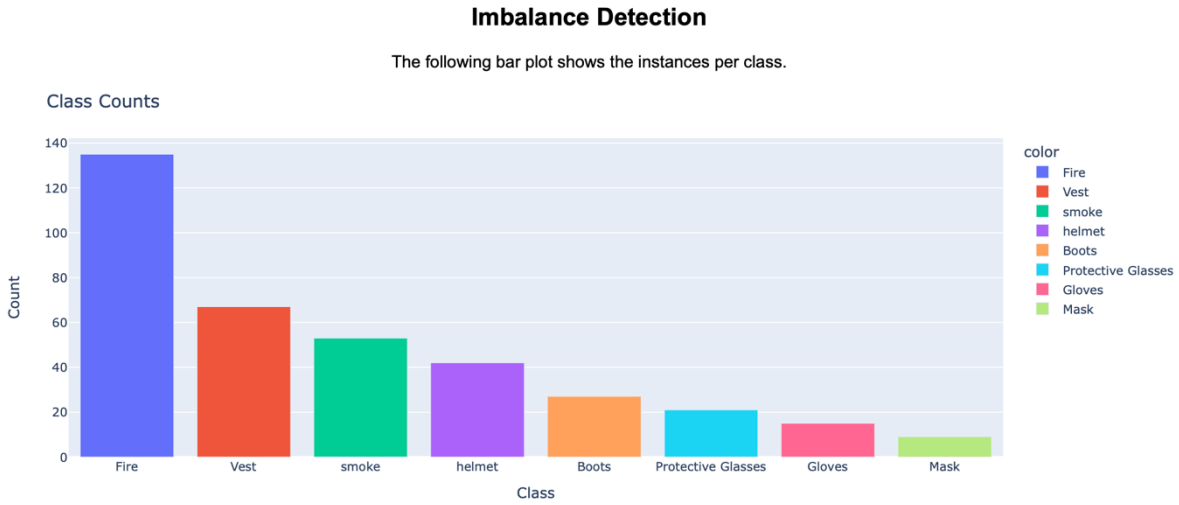


Figure 29: Fire and Smoke Detection Imbalance detection

TrL1 and TrL2 applicability is also put to the test on time-series data. In a similar manner, and with the goal remaining to be the validation of TrL1 and TrL2, two different time-series data were utilized. The first dataset encompasses data collected from January 1st, 2013, to April 24th, 2017, in Delhi, India. The dataset includes four key features: mean temperature, humidity, wind speed, and mean pressure.

Regarding TrL1, the script aims to identify any disruptions in the captured traffic in order to validate the integrity and reliability of the data source. As shown in the following figures (Figure 30, Figure 31), the feature “humidity” and “mean_temp” present certain disruptions in the dataset, indicating that several steps need to be followed in order for the dataset to be able to serve as input to AI models that possibly cannot handle null values, while the features “mean_pressure” and “wind_speed” does not show any empty values in the time reported time window.

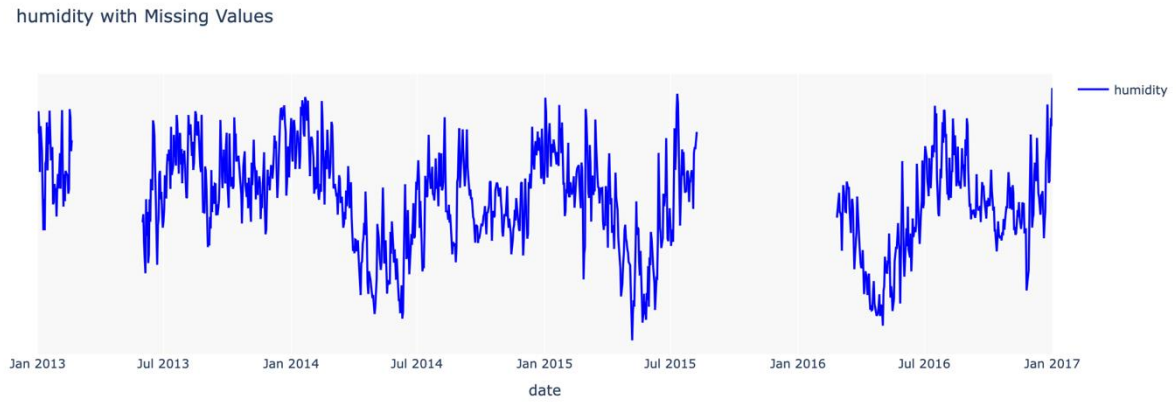


Figure 30: Humidity with Missing values

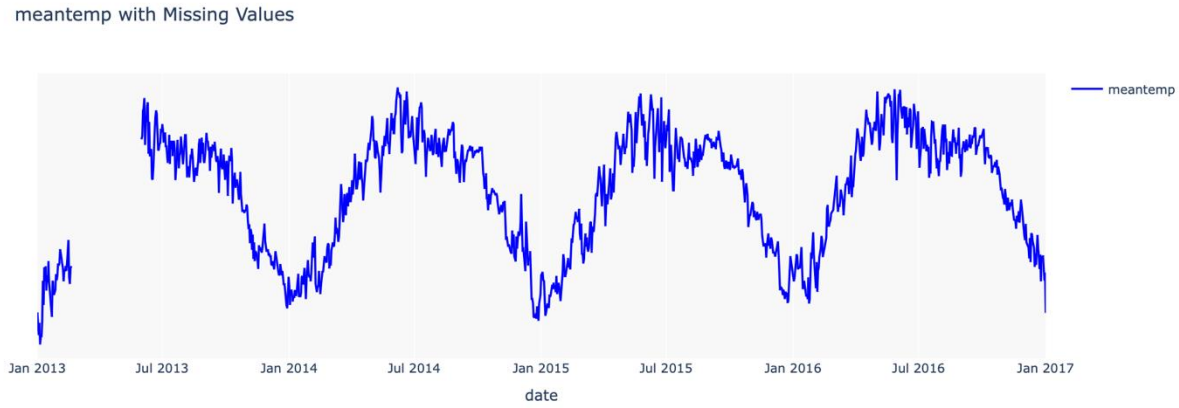


Figure 31: Mean temperature with Missing values

Regarding TrL2, the Local Outlier Factor (LOF) algorithm was applied to the features and the results showed that certain points may be outliers that can add bias to the AI models, reducing their performance in real-world scenarios (Figure 32, Figure 33).

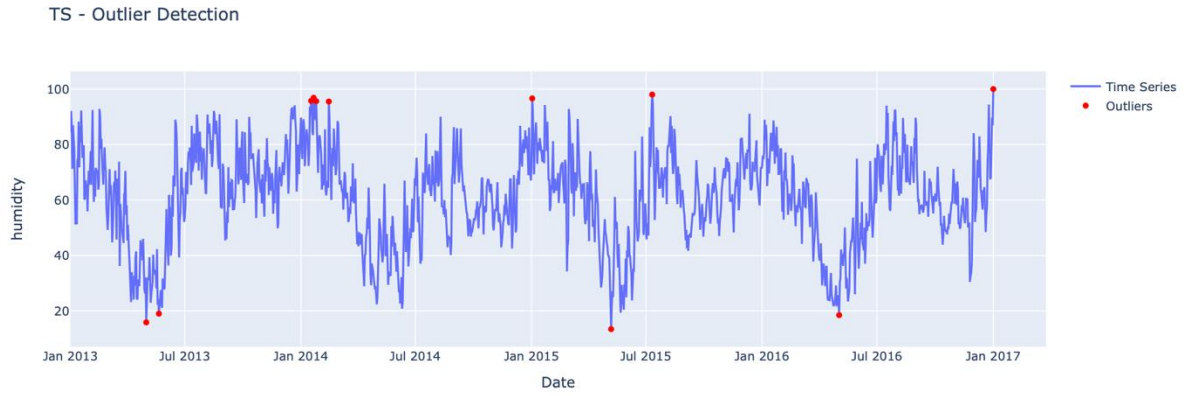


Figure 32: Humidity outlier detection

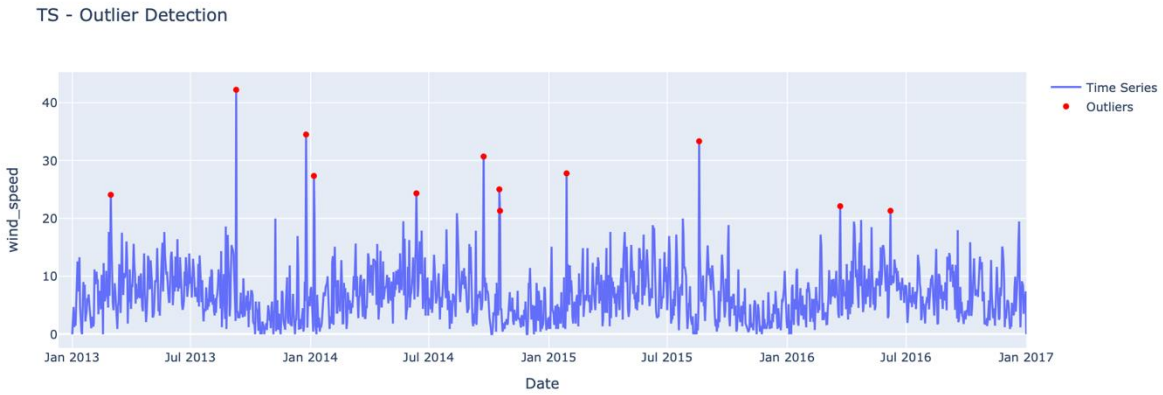


Figure 33: Wind speed outlier detection

At this point, it is important to note that the reported outliers and their corresponding indexes are transmitted in another component of the TALON framework, which is responsible for making the decisions and performing the corresponding actions regarding their handling.

The second time-series dataset is the popular Microsoft Stocks dataset that provides features such as opening, closing, low, and high stock prices, and the sales volume of the stock. The dataset contains daily data from 2015 to 2022, excluding weekends when the stock market is closed.

Regarding TrL1, as shown in the following figures (Figure 34, Figure 35), certain consecutive dates are missing from the low, and high stock prices.

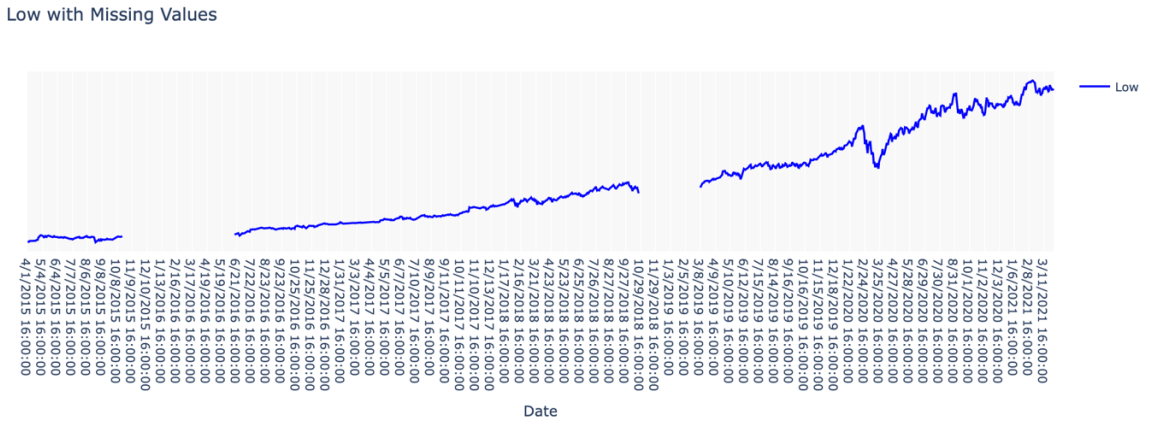


Figure 34: Missing values on "Low Stock Price" feature

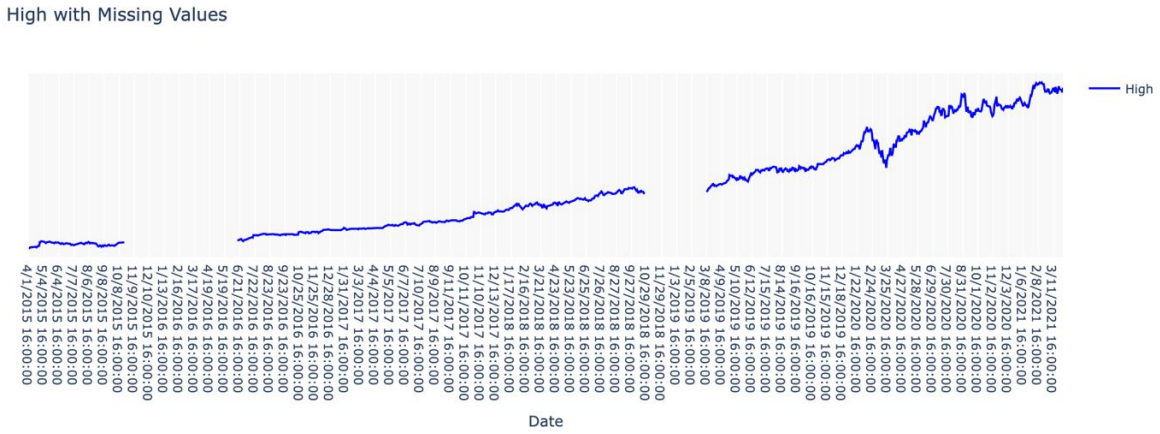


Figure 35: Missing values on "High Stock Price" feature

Based on the figure above, an interesting insight is that the disruptions in the data are in the same time window in three out of the four features where null values were detected. This may indicate that a specific device was offline or malfunctioned during the data capturing, or a cyberattack may have been conducted against the network, resulting in loss of connectivity.

Regarding TrL2, as shown in the following figures, several samples were marked as “outliers”. However, due to the nature of the dataset, the marked points may not be outliers. As already mentioned, other TALON components are responsible for the decision and the action regarding the outliers, while TrL2 only reports its findings.

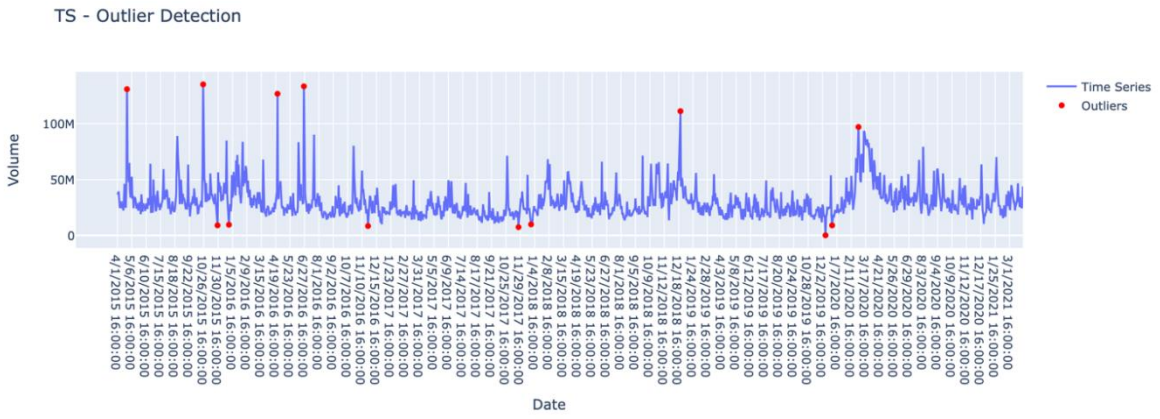


Figure 36: LOF on "Stock Sales Volume" feature

4.1.4 Deployment Results

The early deployment process of TrL1 and TrL2 has been performed utilising the schemes explained in Section 5.5 Authentication and Authorisation. The early deployment of those technical aspects is depicted in TALON’s unified dashboard, realized with the aforementioned APIs and authentication mechanisms.

4.2 Explanations of AI Model Results

4.2.1 Internal Architecture and Key Technologies

While TRL4 is under the phase of requirements specification, there is significant progress when it comes to TRL3 for images. Within the TALON project's TRL3 phase, the integration of Eigen-CAM (Eigen Class Activation Mapping) stands out as a revolutionary step in AI model explainability, particularly for object detection tasks. Eigen-CAM, a powerful technique for making Convolutional Neural Networks (CNNs) interpretable, is pivotal in elucidating the decision-making process of AI, thereby enhancing trust and transparency framework.

Eigen-CAM, a pivotal tool within the TALON project, revolutionizes the interpretability of AI decisions in CNNs through its innovative use of PCA. This technique, distinct from gradient-based methods, employs principal components to generate insightful visualization heatmaps. These heatmaps pinpoint the most critical regions in an input image, that significantly influence the network's decision-making process. Rooted in the principles of linear algebra, specifically eigenvalues and eigenvectors, Eigen-CAM provides an accessible and more straightforward path to visual explanation [59].

Moreover, regarding TrL3 and TrL4 significant development effort has been performed with respect to time series analysis. For this purpose, SHAP[11] (SHapley Additive exPlanations), Lime-for-Time[12] and LIME[13] (Local Interpretable Model-agnostic Explanations) methods have been designed and developed to enhance the interpretability of time series forecasting models. Based on cooperative game theory, SHAP determines the corresponding importance of each characteristic in a forecast by evaluating how much it contributes to the discrepancy between the actual and expected predictions. On the other hand, LIME uses perturbations to the input information and observations of the changes in the model's output to provide a local, interpretable approximation to explain how a specific prediction is generated. The SHAP and LIME techniques are two novel ways to improve the transparency and reliability of AI models in time series research. These techniques greatly aid in the creation of reliable and understandable AI systems by providing interpretable justifications for model predictions. This unlocks the door towards more informed decision-making across a range of time series analytic applications.

4.2.2 Communication Interfaces

The integration of Eigen-CAM within TALON's architecture under Trl3 will exemplify the project's commitment to interoperable design. The specific module will communicate with other TALON services through well-established inter-process communication protocols. Its API, conforming to Swagger specifications, will be integrated into the TALON's dashboard ensuring the seamless interaction with other components of the TALON ecosystem, like computing nodes and blockchain modules. Regarding the SHAP/Lime analysis features, the communication interfaces are to be developed in the upcoming months.

4.2.3 Scientific and Technical Results

The practical applications of Eigen-CAM within TALON have been tested in crucial areas like CS, fire detection and the identification of personal protective equipment. For instance, in the context of CS, Eigen-CAM has effectively identified and highlighted features in images pertinent to safety measures providing the appropriate attention map. Similarly, in fire detection and the recognition of personal protective equipment, the tool has demonstrated its capability to focus on and reveal the essential

[11] https://shap.readthedocs.io/
[12] https://github.com/emanuel-metzenthin/Lime-For-Time
[13] https://github.com/marcotcr/lime

elements within images. In Figure *37*, Figure *38* and Figure *39* the capability of Eigen-CAM to identify the areas responsible for the decision of the object detection model is vividly illustrated. In the broader framework of the TALON project Eigen-CAM's ability to visually underscore the most relevant features recognized by the CNN layers substantially aids in model debugging and training. This visualization not only enhances the understanding of how different network layers interpret various features in the input data but also aids in identifying and rectifying biases within the training dataset.

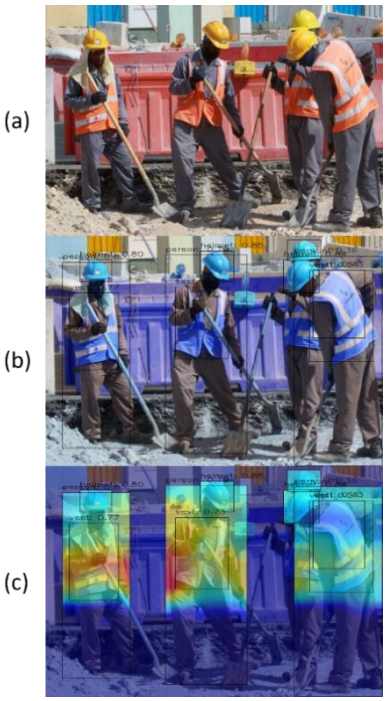


Figure 37: (a) Initial image for Construction Safety Measures (b) Detected Safety Objects (c) Eigen-CAM Safety Objects Analysis

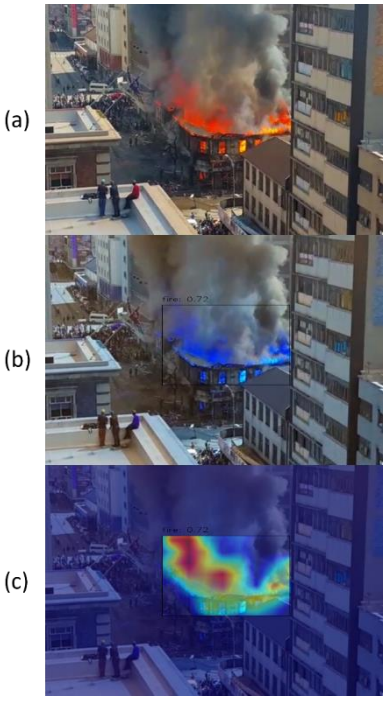


Figure 38: (a) Initial image for Fire event (b) Detected Fire Objects (c) Eigen-CAM Fire Objects Analysis

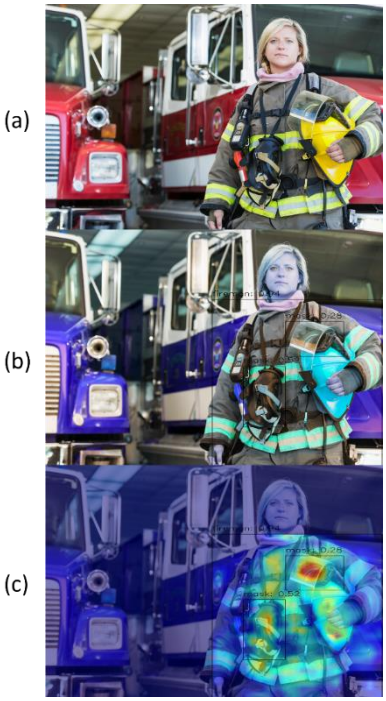


Figure 39: (a) Initial image for Personal Protective Equipment (b) Detected Protective Equipment Objects (c) Eigen-CAM Protective Equipment Objects Analysis

Another category of explanations produced in TALON relates to time-series explanations. For this purpose, we modified the Lime-for-Time[14] to explain loss-of-signal (LOS) events in optical communications systems. This category of time-series explanations is linked with the Scenario 2 of Use Case 2 (cf. D2.1-Use Case, KPIs, Requirements, Specification, Slices & Technology Enablers Definition Report). We assume fiber optic communications in the manufacturing domain to link different services of the shop floor. Ensuring the reliability and performance of these communications is essential for hosting and serving the services and applications within the factory. Optical fiber links may have various types of faults over time, and it is crucial to identify and classify these faults accurately. To address this need, we developed a classifier for fiber fault classification to identify different types of faults in optical fiber link. Explanations of the faults of the optical fiber link enable to interpret the root cause, perform trouble shooting and better maintain the backbone optical network. In the figure below we present different categories of faults in optical fiber link.

[14] *https://github.com/emanuel-metzenthin/Lime-For-Time*

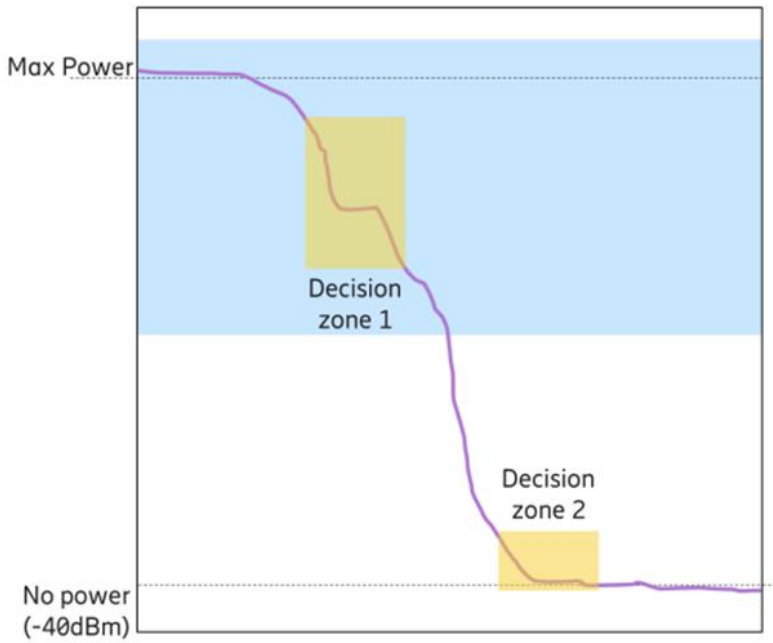


Figure 40: Sample time-series showing the optical received power transient. The example shows how some peculiarities in the transient shape may be associated with a specific root cause thus providing explainability of the classification

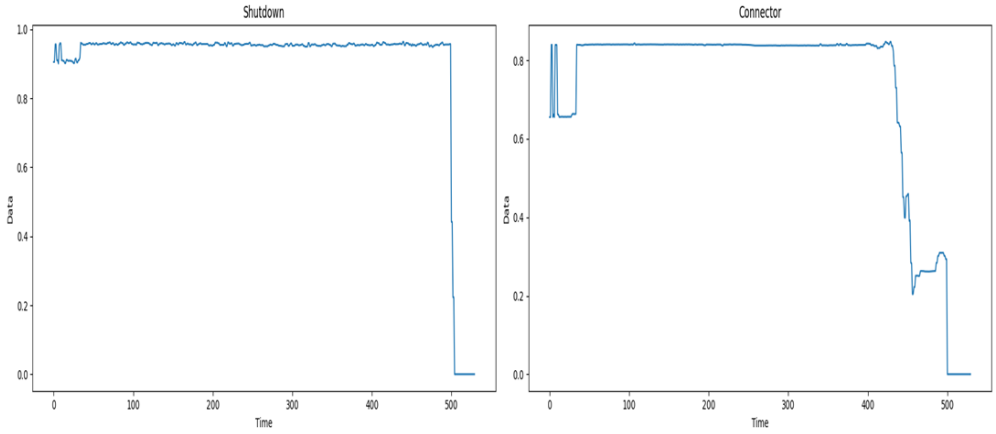


Figure 41: Different Categories of Optical Link Faults

The dataset used contains time-series of variable length with deltas in the optical power received by the optical transceiver because of an optical link problem. When a fault occurs the received optical power experience a transient behavior according to the type of fault. In our case, we have four categories of faults that may occur, namely SFPConnector fault, Connector, Shutdown, and Stress.

We aligned the dataset with fault category labels and trained the time-series with two conventional machine learning classifiers (e.g., KNN and Random Forest), as depicted in Figure 42 below:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

rf = RandomForestClassifier()
knn = KNeighborsClassifier()

rf.fit(X_train, y_train)
knn.fit(X_train, y_train)
```

Figure 42: Fault Detection Classifiers

The accuracy of the classifiers is reported in the table below:

Table 7: Fault Detection Classifiers Accuracy

Models	**Accuracy**	**Precision**	**F1**
Random Forest	59.7%	61.6%	54.7%
AdaBoost	37.6%	40.5%	36.9%
KNN	57.7%	58.6%	54%
LGBM	**60%**	**63.7%**	**55.4%**

By modifying the Lime-for-Time library to fit the task of optical faults explanation, we experimented over different time windows to better explain the cause of the fault correlated with the received optical power (i.e., the observed and measured event). In this version of D4.1, we report our early experimental results regarding time-series explanations. We have focused on defining the pre-processing pipeline (i.e., time-series normalization, data imputation with zeros when needed, etc.), the training of the multi-class classifiers and the incorporation of time-series explanations. We highlight in dark green the part of the time-series mostly contributing to the fault / event. The figures below depict the part of the time-series which contributes to the fault class of SFPConnector and Connector, respectively. In Figure, the explanation score is 0.7 and in Figure 44 the explanation score is 0.65.

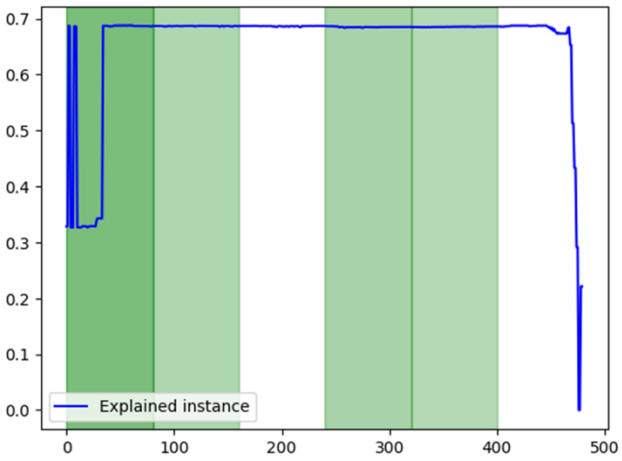


Figure 43: Explanation on SFPConnector Fault Category

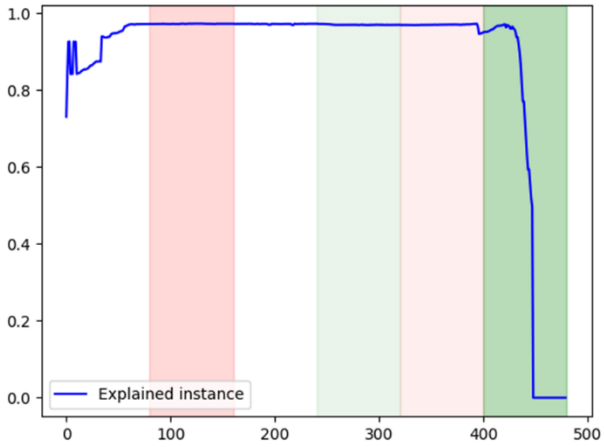


Figure 44: Explanation on Connector Fault Category

Apart from Lime-for-Time another category of explanations that can be successfully applied to timeseries explanations is SHAP[15]. In TALON we use this type of explanations to provide valuable insights into the type of power loss that has been identified by the Machine Learning model for the Scenario 2 of Use Case 2 (cf. D2.1-Use Case, KPIs, Requirements, Specification, Slices & Technology Enablers Definition Report). As stated in the problem definition, there are 4 distinct events of signal transition loss. Connector which is caused by the disconnection of one of the connectors in the passive optical network, SFPConnector which is caused by the unplugging of the connector in the SFP module on the transmitting side, Shutdown which is caused by the SFP module being powered down on the transmitting side, and finally Stress which is caused when the fiber patch in the passive network is manually stressed and broken. Hence, the expectation is not only to produce a Machine Learning model with good prediction accuracy on the type of failure event that triggered the power loss but at the same time provide reasoning and justification behind its prediction to consolidate the classification for a domain expert. For that, the timeseries gets broken down in fixed size time windows, starting from the first timestamp when the loss is first detected and going back in time. This enables the model to learn on short term, medium term as well as long term power information and

[15] https://github.com/shap/shap

increase both its predictive accuracy as well as explainability. Furthermore, an extra layer of feature engineering is applied so that the power per time window is categorized as low, medium or high depending on its value. This mapping allows for a much better explainability of the model's prediction, although it has an impact on the model's accuracy.

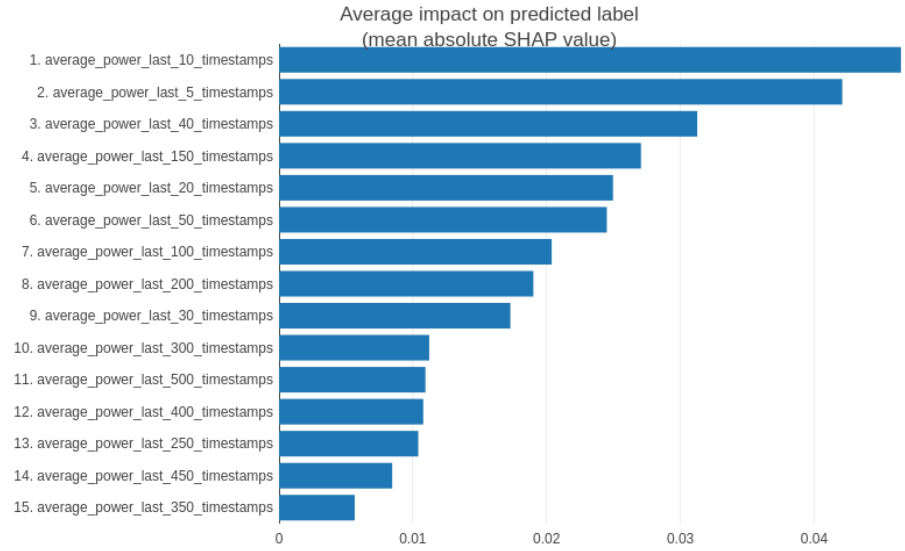


Figure 45: Feature importance calculated by the mean absolute SHAP values

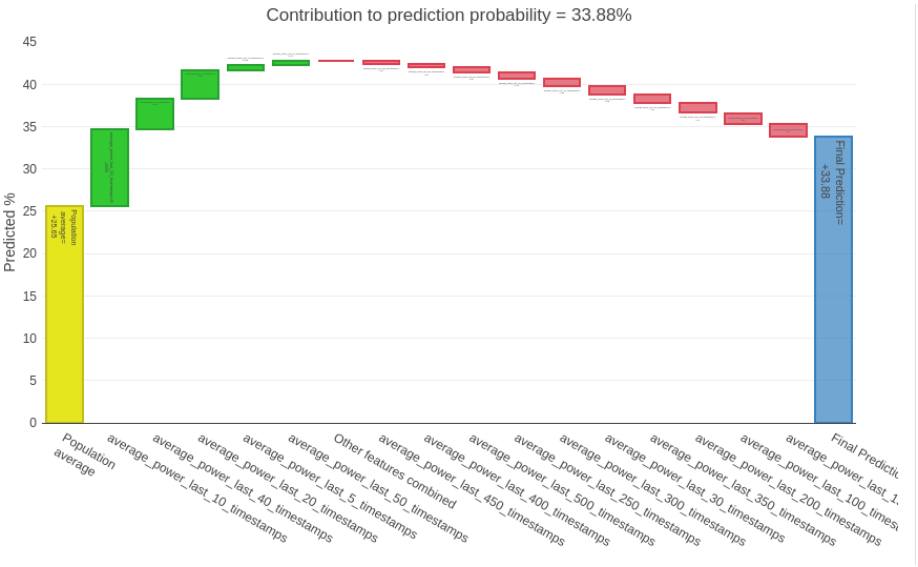


Figure 46: Features contribution towards the model's individual prediction. Features with green are positive contributors, while red features contribute negatively

4.2.4 Deployment Results

Since TrL3 and TrL4 are still in research and development phase, the deployment is scheduled to occur within the upcoming months.

5 Data and AI Models Trustworthiness

5.1 Textual, Tabular & Numerical Anonymisation

5.1.1 Textual, Tabular and Numerical Anonymization

In the premise of the Data and AI Models Trustworthiness, TALON develops a framework for anonymizing a variety of input data that are going to be used by the TALON subsystems. This system will leverage this wide spectrum of data to perform analytics, to train advanced AI models, for decision support, process evaluation and other deployment-related operations, also in line with the plethora of the TALON usage scenarios, as signified by the TALON use cases. In the variety of shapes and sizes that this data may come, a big portion refers to textual, tabular or numerical information and samples. This can be obtained by different heterogeneous sources, such as, reports, logs, sensor and telemetry data, deployed edge devices, and so on, and represent private, personal or otherwise sensitive information, under the umbrella of each end-user's ownership. TALON considering these preconditions and respecting EU and International laws, regulations, legislation and standards, develops an integrated and adaptable Textual, Tabular and Numerical Anonymization component, as defined in the TALON Architecture, to enable the system and its dependencies to handle, curate and process such data, respecting the users' and data safety, security, privacy and integrity.

5.1.2 Internal Architecture and Key Technologies

This section introduces how the anonymization internal architecture, methodology and business logic operate, with respect to scientific and technological techniques used.

5.1.2.1 Data Anonymisation Process & Key Technologies

To address data privacy concerns, TALON develops a data anonymization methodology based on benchmark and state-of-the-art technologies leveraged in the field. This solution aims to establish a standardized approach for anonymizing PII in both structured and unstructured data. It uses rules, NLP and ML to anonymize common entities like names, emails, and phone numbers. Additionally, k-anonymity strategies are considered for this purpose.

Anonymization involves removing sensitive information from documents, and especially in free-form text, to avoid privacy breaches and to provide an overall disguise for all intents and purposes. The developed anonymization dependencies employ various techniques to detect sensitive data, including removal, categorization, and pseudonymization. In removal, sensitive references are replaced with placeholders like "*" or "blank". Categorization involves substituting a label that indicates the type of sensitive information. Pseudonymization replaces private data with similar non-identifiable data, applicable in certain data types and applications. An example of these strategies is illustrated in Table 8.

Table 8: Example of anonymisation process

Methods	Original Data	Transformed Data
Removal	John Smith works at HSBC Bank	<REF> works at <REF>
Categorization	John Smith works at HSBC Bank	<NAME> works at <LOCATION>
Pseudonymization	John Smith works at HSBC Bank	Peter Green works at NatWest Bank

The anonymization process typically follows Named Entity Recognition NER) principles, as sensitive information often directly references real-world entities like people and organizations [60]. For

instance, NER discerns between different types of entities, such as identifying 'John Smith' as a person and 'HSBC Bank' as an organization. This categorization is crucial for the initial phase of anonymization, where entities are identified and classified before neutralization [61]. However, the process faces challenges due to the subjective nature of anonymization and the scarcity of domain-specific annotated training data. Consequently, NLP-based anonymization, a popular method within the NER task, often operates on a case-by-case basis, allowing for rapid ML adaptation to specific requirements.

In respect to Secure Information Sharing there is a notable need for anonymization in domains like clinical and health-related documents to comply with privacy laws, including the General Data Protection Regulation (GDPR). There are a lot of examples of anonymization techniques employed, in an effort to realise the aforementioned practice. For example, [62] introduces a text anonymization system suitable for both structured and freeform English texts, evaluating various methods. This system employs NLP and AI for PII detection and implements k-anonymity for effective anonymization. It is integrated into the Cloud-based Anonymous Repository of Incidents (ARIEC), aiding energy operators in critical infrastructures. ARIEC anonymizes data on cybercrimes, cyberattack incidents, and major software updates, allowing European energy operators to share cybersecurity incident details without compromising private assets. This ensures open access to technical details of attacks, enabling preventive measures.

The field of data anonymization, particularly in handling sensitive data across various sectors like healthcare and education, is witnessing significant advancements. The adoption of deep learning models has notably enhanced the effectiveness of text anonymization and privacy protection strategies. In [63] a clinical de-identification method using contextualized word representations combined with sub-document level analysis has been proposed. This technique effectively recognizes and labels sensitive information in clinical texts, benefiting from the deeper semantics and interrelationships captured within the larger text context.

In the NLP domain, [62] work focuses on privacy preservation. They offer a comprehensive review, presenting a taxonomy categorizing techniques for data security and privacy-enhancing technology. Their study delves into the theoretical aspects, privacy risks, available datasets, and metrics for privacy evaluation, acknowledging challenges like dataset size, biases, processing costs, and the balance between privacy and utility. Bidirectional Long Short-Term Memory (BLSTM) and Conditional Random Field (CRF) [64] have also been proposed for multivariable constraints for anonymization purposes. This model showcases high accuracy in utilizing demographic data to identify at-risk students, aiding educational institutions in making informed decisions.

5.1.2.2 Business Logic

A standard anonymization system typically comprises three main components: i) a pre-processing module for tasks like text normalization and feature extraction, ii) one or more parallel or sequential NER classifiers, and an anonymization module that replaces identified NERs in the text. The system's architecture is detailed in the following figure.

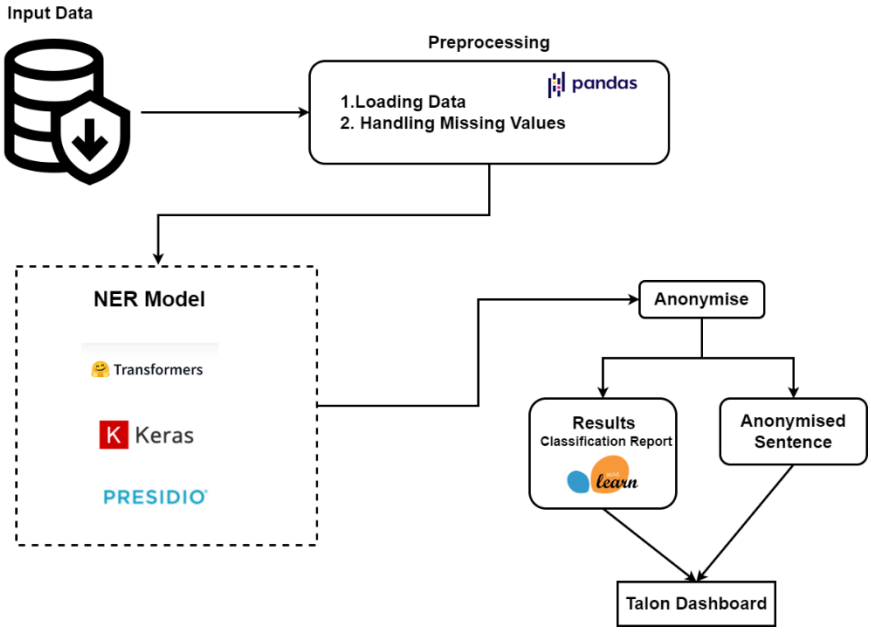


Figure 47: Anonymisation System Architecture

Data loading & Preprocessing

The anonymization process starts with loading JSON input data into a dictionary and flattening it. Each record undergoes text normalization, including i) stemming, ii) lemmatization, and iii) tokenization. Stemming simplifies words to root forms, lemmatization ensures valid roots, and tokenization breaks text into tokens for tasks like document classification and translation. The output, formatted as a dictionary or data frame, facilitates text analysis. Additionally, text normalization involves a) eliminating extra whitespaces, b) removing punctuation, c) converting to lowercase, d) removing accents, e) discarding special characters, f) eliminating emojis, g) resolving contractions, h) converting word numerals, i) correcting spelling, and j) removing stop words. Anonymization transforms sentences into indices and pads them for consistency. This pre-processing is vital for numerical conversion of text data, aiding ML models. Techniques such as Bag of Words or Count Vectorizer are crucial for feature extraction, identifying aspects like text type, capitalization, title case, and statistical properties.

Rule Based Anonymisation Method

An approach to identifying sensitive data that requires anonymization involves the use of rules. For example, emails and phone numbers adhere to specific formats that can be recognized through rules enabling the representation of any kind of email or phone number. Table 9 displays the creation of two rules for this operation, which assist in identifying emails and phone numbers. The system employs a rule-based strategy to identify phone numbers and emails. A supplementary manual process was implemented to enable users to selectively designate specific fields or keys within the JSON data structure for automatic anonymization.

Table 9: Rule based filtering example

Token Type	Detection Rule
Phone numbers	r'(3. [-]??3. [-]??4. —3. *3. [-]??4. —3. [-]??4.)'
Email addresses	r'[̈-]+@[̈-]+'

With this manual method, certain fields can be specified, and if they are, their values can be made anonymous. Furthermore, the user must supply not only the key/field name but also the kind of data (such as email, name, or address) in order to accommodate both removal and categorization for the k-anonymity methods. In this instance, no pre-processing is done and all of the text contents in this field are anonymized.

NLP Based Anonymisation Methods

The NLP-based method for text anonymization assumes that the text has been preprocessed, which includes segmenting and tokenizing the text. The procedure commences by dividing the unprocessed text into sentences using a sentence segment, and subsequently breaking down each sentence into individual words using a tokenizer. Subsequently, part-of-speech tags are assigned to every sentence, facilitating the identification of named entities. This step entails identifying potentially intriguing entities within the text. After identifying the intriguing entities, their connections are established. The process of entity detection mainly depends on the Chunking technique, which entails dividing and categorizing sequences of multiple tokens. The smaller boxes in Figure 48 represent the word-level tokenization and part-of-speech tagging, while the larger non-overlapping boxes (chunks) in the source text depict the higher-level chunking.

During the process of chunking, phrases are extracted from unstructured text with the goal of identifying the constituents, such as noun groups and verb groups. The formation of phrases involves five essential categories: *i) Noun Phrase (NP), ii) Verb Phrase (VP), iii) Adjective Phrase (ADJP), iv) Adverb Phrase (ADVP), and v) Prepositional Phrase (PRP).*

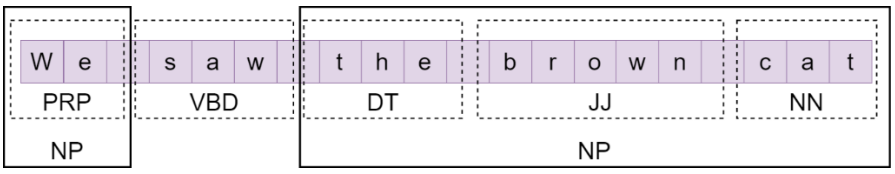


Figure 48: Segmentation Example

The chunker, who is in charge of this process, is trained using the 'conll2000' corpus [65], which employs inside, outside, and beginning (IOB) tags to indicate chunks. The IOB format is a standard tagging format used for tokenization in chunking operations. It indicates the beginning and end positions of the chunk, as well as its types. Consequently, an IOB tagger can be trained on these IOB tags using a Naive Bayes Classifier. The sample using IOB format is displayed in Table 10. During the tagging process, each word is assigned a part-of-speech (POS) tag. A Unigram Tagger, belonging to the Lookup Taggers category, is employed. This tagger utilizes a frequency-based approach to identify and retain the most commonly occurring words, along with their associated relevant tags, by considering the context of individual words. Once the segmentation, labeling at the Token and Chunk levels, and tagging process are completed, the attention turns to NER. NER entails the identification of distinct categories of individuals, including organizations, individuals, and dates. The design of a NER system encompasses multiple approaches, one of which involves utilizing a lookup table containing a list of names. Nevertheless, this method frequently proves ineffective due to its tendency to incorrectly identify locations, dates, and individuals, as well as its potential to misinterpret entities such as 'ON' and 'READING'. DATE and PERSON entities present similar challenges, as the word 'May' can have multiple interpretations, referring to either a date or a person. Similarly, 'Christian Dior' can be understood as either a person or an organization.

Table 10: IOB format Sample

Word	POS	Tag

John	NNP	I-PER
is	VBZ	O
traveling	VBZ	O
to	IN	O
Paris	NNP	I-LOC
in	IN	O
France	NNP	I-LOC

As mentioned, a certain number of tags are used to aid the anonymizer in identifying and recognize the meaning of each word, with respect to its location and role in the sentence. The following Table 11 enumerates the meaning of the utilized tags.

Table 11: Tag Descriptions

Tag	Description	Tag	Description
O	Other	I-art	Inside Artifact
B-geo	Beginning of Geographical Entity	I-per	Inside Person
B-gpe	Beginning of Geopolitical Entity	I-gpe	Inside Geopolitical Entity
B-per	Beginning of Person	I-tim	Inside Time
I-geo	Inside Geographical Entity	B-nat	Beginning of Natural Phenomenon
B-org	Beginning of Organisation	B-eve	Beginning of Event
I-org	Inside Organisation	I-eve	Inside Event
B-tim	Beginning of Time	I-nat	Inside Natural Phenomenon
B-art	Beginning of Artifact		

5.1.2.3 Machine Learning and Deep Learning Anonymisation Method

The method employs three advanced ML and DL techniques, namely CRF, Long Short-Term Memory (LSTM), and Embeddings from Language Models (ELMo), to improve text analysis and named entity recognition [66] [67].

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 75)                0
_________________________________________________________________
embedding_1 (Embedding)      (None, 75, 40)            117040
_________________________________________________________________
bidirectional_1 (Bidirection (None, 75, 100)           36400
_________________________________________________________________
time_distributed_1 (TimeDist (None, 75, 50)            5050
_________________________________________________________________
crf_1 (CRF)                  (None, 75, 18)            1278
=================================================================
Total params: 159,768
Trainable params: 159,768
Non-trainable params: 0
_________________________________________________________________
```

Figure 49: Summary of the Parameters of the Neural Network

CRF considers word sequences as key features, focusing on sequences rather than isolated words. It uses a formula where 'Y' is the hidden state, such as part of speech, and 'X' represents observed variables like entities or word sequences. The formula (Equation 3) comprises a normalization part, yielding output probabilities, and a logistic regression component with weighted features. Feature weights $\theta_k f_k$ are determined by maximum likelihood estimation.

Equation 3: Probability Density Function

$$p(y|x) = \frac{1}{Z(x)} \prod_{t=1}^{T} exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t) \right\}$$

The second model, based on a Recurrent Neural Network (RNN) architecture, aims to output a tag 'Y' for a given token 'X'. Its structure includes an embedding layer, bi-directional LSTM layers, a time-distributed dense layer, and a CRF for final output generation. The embedding layer sets maximum sequence lengths and converts tokens into vectors. The bidirectional LSTM processes inputs from the embedding layer, and the time-distributed layer, based on an RNN Architecture, enables dense operations at each timestep.

The third incorporated model, ELMo, uses bidirectional LSTM-trained vectors, differing from traditional representations by assigning each token a function of the entire input sentence [68]. ELMo representations, derived from all LSTM layers, improve performance compared to using only LSTM layers. This model captures context-dependent word meanings in higher-level LSTM states and syntax-related aspects in lower levels. It starts with a 2-layer bidirectional LSTM with a residual connection, using character embeddings for token representation, which then passes through a 2-layer network before the LSTM layer.

In the frame of the TALON project's tabular, textual and numerical anonymization, the abovementioned methods were developed and utilized and evaluated. The deployment and results produced for the described methodology are outlined below.

5.1.3 Communication Interfaces

The component developed to perform the tabular, textual and numerical anonymisation is deployed in the TALON Cloud-edge continuum service ecosystem (Figure 50) and communicates on-demand information to the requesting components.

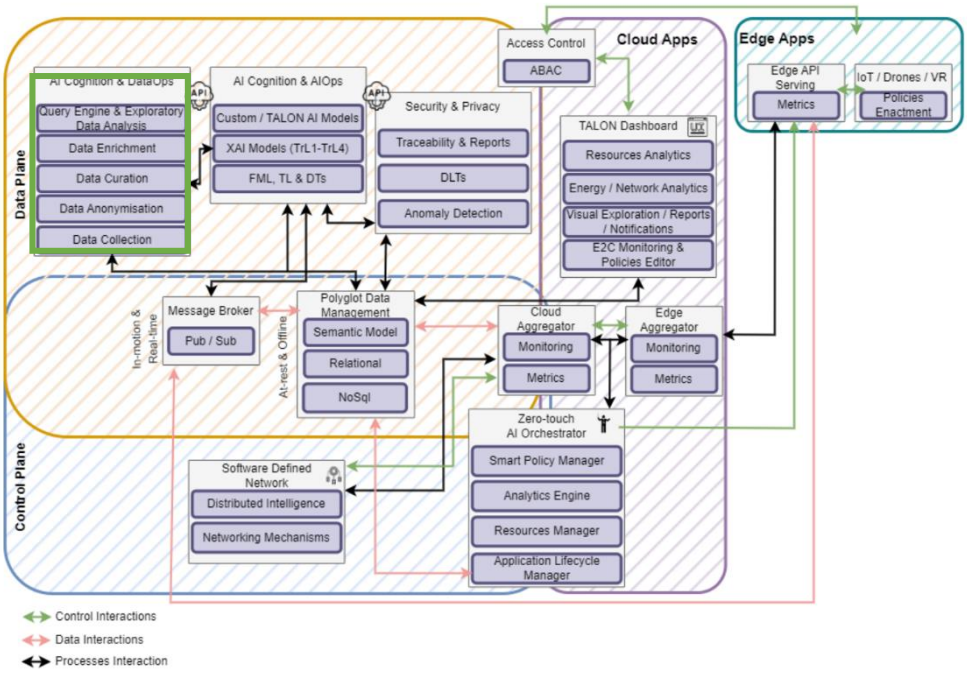


Figure 50: Textual and Numerical Anonymisation module position in TALON's Architecture

In particular the component deploys a self-contained micro-service in the TALON infrastructure. The micro-service encapsulates the functionality to perform anonymization by utilizing the aforementioned methods. The component exposes a set of interfaces and external APIs, providing collaborating components to utilize the available anonymization methods. A summary of the APIs is provided in Figure 51 below:

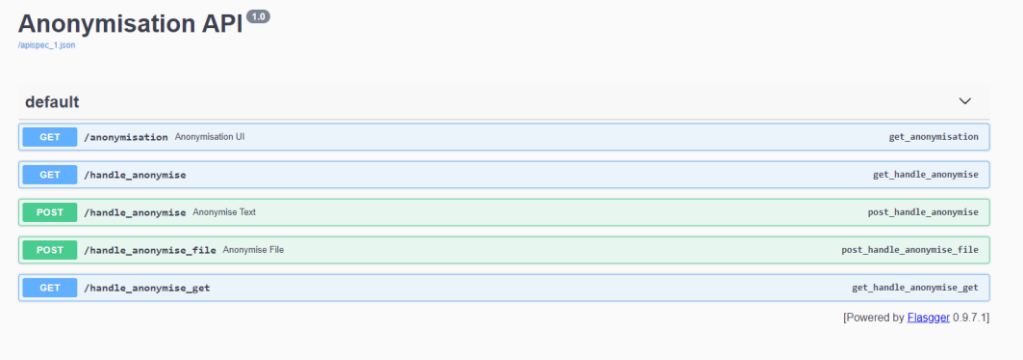


Figure 51: Anonymisation Swagger

The anonymization component provides API access for:

- An integrated tabular, textual and numerical anonymization GUI interface
- A function to handle free-form text
- A function to handle tabular and numerical data

The service performs any process on the provided input (data to be anonymized) and returns the anonymized entity back to the requesting origin.

As mentioned, the Anonymization function is deployed as a micro-service in the respective deployment. The service communicates with other entities via an API Gateway operating as a proxy. Figure 52 shows this interaction.

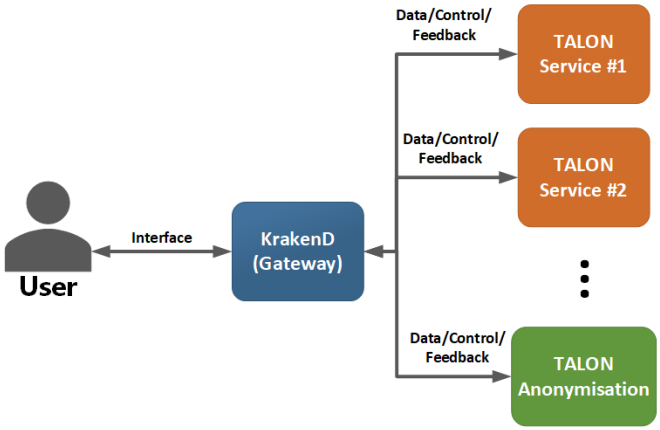


Figure 52: Deployment Interaction

5.1.4 Scientific and Technical Results

In order to validate the developed methodology and establish its integral operation into the TALON Cloud-edge ecosystem, a series of evaluations were performed on the employed anonymization techniques. These evaluations are especially aimed at quantifying the PII detection and identification

of NERs in tabular, textual and numerical information. The assessment of the devised anonymization methods [69] was conducted using publicly accessible NER datasets.

In particular to evaluate the implemented methods the conll2000 dataset was utilized. A total of 47,959 sentences were used, with an average sentence length ranging from 20 to 30 words. A sample of the dataset can be seen in Table 10. Notably, the longest sentence in the dataset consists of 46 words. Figure 53 illustrates the sequential process of text anonymization, showcasing how each word in the sentence is individually separated and assigned a POS and Tag.

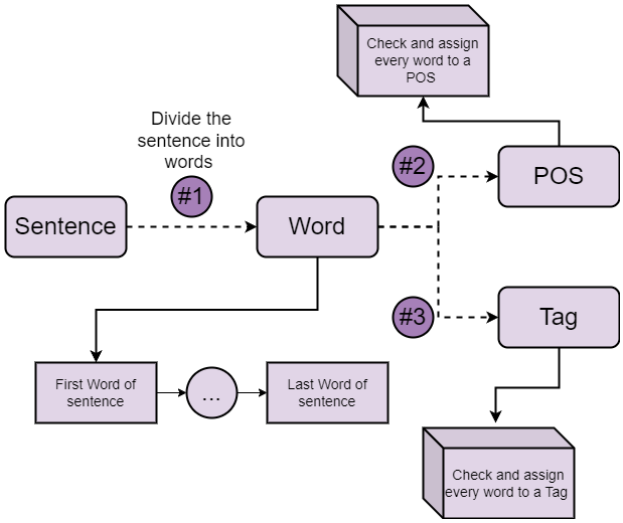


Figure 53: POS and Tag Assignment

Furthermore, to successfully validate the results of the aforementioned methodology, a series of quantitative metrics were utilised to outline the effectiveness of the leveraged techniques. Specifically, the following metrics were used:

Table 12: Quantitative Metrics

Evaluation Metrics	**Description**	**Formula**
Precision	Measure of correctly identified positive cases among all positive predictions.	$Precision = \frac{TP}{TP + FP}$
Recall	Ratio of correctly identified positive cases to all actual positives.	$Recall = \frac{TP}{TP + FN}$
F1-Score	Harmonic mean of precision and recall, balancing both metrics.	$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$
TP = True Positives, FP = False Positives, FN = False Negatives		

Table 10 shows the results of the detectors in the operation of identifying the entities for anonymization, that the anonymization service will effectively anonymizer or redact from the overall information. The NER-ELMo model achieves an F1-score, accuracy, and recall of 0.82. This demonstrates the model's ability to achieve a harmonious equilibrium between recall and accuracy,

showcasing well-balanced performance in accurately detecting and categorizing objects. However, the NER-LSTM model surpasses the competition by achieving an impressive F1-Score of 0.96, along with accuracy, recall, and F1-score values of 0.97. The results indicate that the NER-LSTM model successfully achieves high performance in accurately identifying and categorizing entities. The NER-CRF model achieves an accuracy of 0.95, recall of 0.96, and F1-score of 0.95. These metrics indicate that the model demonstrates strong performance in classifying and identifying items.

Table 13 - Quantitative Results

Model	Precision	Recall	F1-Score
NER-ELMo	0.82	0.82	0.82
NER-LSTM+CRF	**0.97**	**0.96**	**0.96**
NER-CRF	0.95	0.96	0.95

In general, Table 13 indicates that all three models demonstrate satisfactory performance for NER. The NER-LSTM model exhibits superior accuracy, recall, and F1-Score, indicating its enhanced capability to accurately detect and classify entities. The aforementioned models were encapsulated into the tabular, textual and numerical anonymization module.

5.1.5 Deployment Results

The deployment of the Anonymisation module is quite straightforward. The module is deployed as a service in the TALON Cloud-edge ecosystem, as highlighted in the communication interface section. The module is installed on a service server and is connected to TALON's API gateway and proxy. To install this module, a number of prerequisite dependencies are needed. The system has been developed in Python and thus requires a number of Python libraries to operate. The key needed libraries are listed below.

- `Flask`
- `Flask-SQLAlchemy`
- `flask-cors`
- `flasgger`
- `faker`
- `pandas`
- `numpy`
- `termcolor`
- `tqdm`
- `psutil`
- `keras`
- `tensorflow`
- `pytorch`

The service is containerized using Docker, providing modularity and ease of deployment. The base system for the docker image is python:3.8-slim-buster, which encapsulates all the needed and basic system dependencies to run the service. Furthermore, the service exposes the aforementioned APIs that are leveraged by a user interface. This interface is incorporated into the TALON dashboard utility, offering on-demand textual, tabular and numerical anonymization. The interface can be seen below in figure Figure 54:

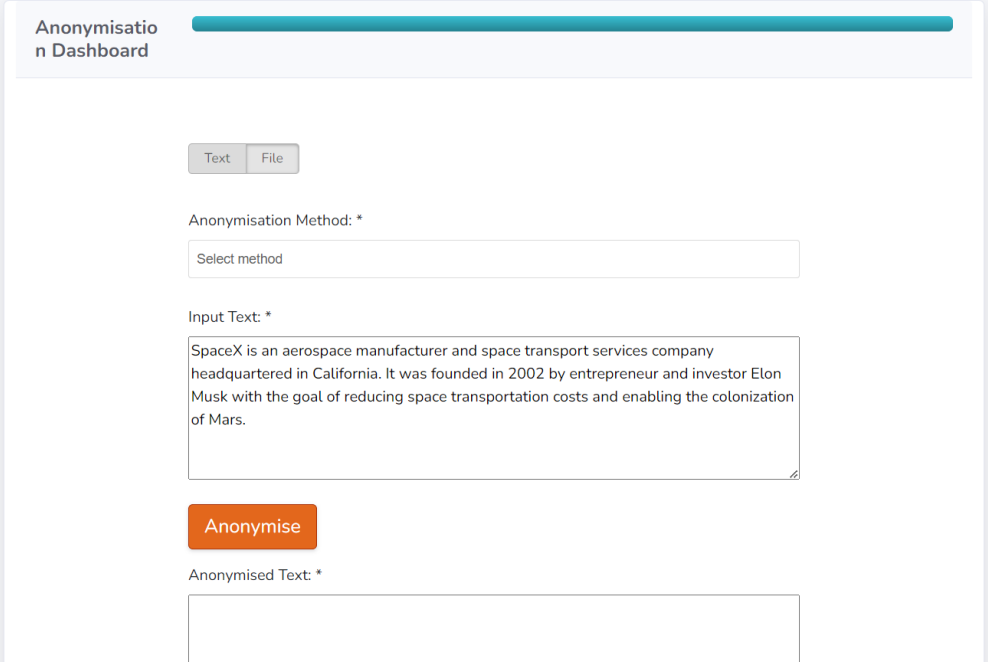


Figure 54: Anonymisation Interface

Albeit the provision of a visual interface, the developed service is capable of operating without user input, but by also connecting to other services of the TALON architecture. By utilizing the provided APIs, any service in need of textual, tabular and numerical anonymization can leverage the operation of the Anonymization module.

5.2 Image Anonymisation

5.2.1 Internal Architecture and Key Technologies

Protecting the privacy of individuals captured in images is crucial, especially when using them for AI development and deployment. This challenge is particularly relevant within the TALON project, where images play a role in pilots like "AR/VR for training and maintenance" (UC3 Scenario 2) and "Human Robot Collaboration" (UC4). That's the reason why a robust Image Anonymization solution is needed for the TALON framework. This ensures compliance with privacy regulations, safeguards sensitive information, and allows TALON to leverage visual data ethically and responsibly. Ultimately, image anonymization plays a key role in achieving our goals of trust, transparency, and responsible AI development within TALON.

The Image Anonymization Module (ImAM) is one of the Access & Security layer's mechanisms that are designed to protect the confidentiality, integrity, and availability of data and systems.

It provides a comprehensive solution for obfuscating faces in image and video files. It comprises an HTTP REST API server for file ingestion, an anonymizer module for the actual anonymization process, an output module for deploying the anonymized files and a database (MongoDB) as a secure repository for the processed original files.

In the figure below, the architecture of the Image Anonymization Module is shown, reporting also the open-source libraries used for realization.

It also presents the Auditor, whose scope is to act as a cron-job and to delete, every day, the original image/video files present in DB, if any, whose dates are older than a configured retention-period. This job permits the TALON framework to be compliant with Article 5(1)(e) of the GDPR which stated:

(e) storage limitation: Personal data shall be kept in a form which permits the identification of data subjects for no longer than is necessary for the purposes for which the personal data are collected or subsequently processed.

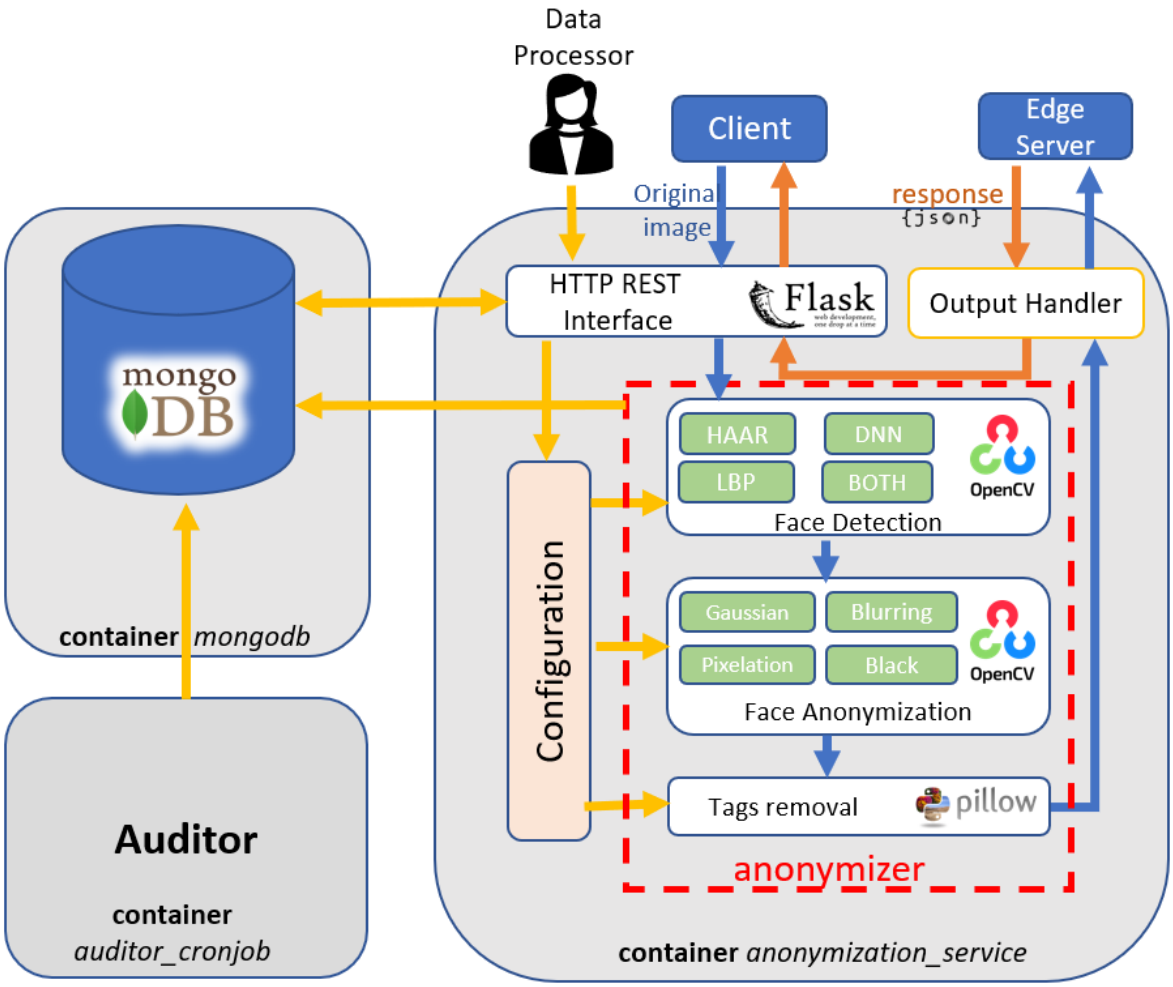


Figure 55: Image Anonymization Module architecture

ImAM utilizes a Partial Anonymization paradigm to effectively conceal faces in image and video content while maintaining the overall context and visual integrity:

1. Perform face detection
2. Extract face Region of Interest (ROI)
3. Face anonymization
4. Store blurred face in original image.

This approach involves selectively obscuring or removing identifiable information, such as faces, without compromising the overall visual experience. In the case of Partial Anonymization, faces are typically blurred, pixelated, or masked to render them unrecognizable.

ImAM offers a flexible configuration mechanism enabling the selection of various face detection techniques, applicable to both image and video files, they are part of OpenCV (Open-Source Computer Vision Library), a free and open-source software library used for real-time computer vision and image processing:

- DNN: Leverages advanced neural network architectures, particularly CNNs, to achieve superior face detection and anonymization capabilities. Images are processed through trained networks to effectively replace or obscure facial features.
- Haarcascades: Employs a cascade of classifiers based on ML principles to efficiently detect faces. This technique efficiently identifies faces by applying a series of classifiers, making it suitable for subsequent anonymization procedures.
- LBP (Local Binary Pattern): Employs texture analysis for face detection in image and video analysis. It analyzes local patterns to identify faces and can subsequently be applied for anonymization, modifying facial features while preserving texture characteristics.
- Both: This option involves a sequential approach in which the DNN detection technique is initially applied, and on the output result the Haarcascades technique is applied. This combination aims to enhance face detection accuracy, particularly for image files.

Each of these face detection techniques possesses unique strengths and limitations, and their effectiveness may vary depending on the specific use case and application requirements. During the integration process, the most suitable technique tailored to the TALON project's needs will be selected, accompanied by parameter fine-tuning for the chosen method.

In addition to face detection, ImAM provides a range of anonymization methods for face obfuscation, coming from OpenCV library:

- Gaussian blurring: A widely used image processing technique involving a Gaussian kernel to smooth an image. In the context of face obfuscation, Gaussian blurring effectively obscures facial features by blurring the pixels in the image.
- Blurring: A simple yet effective method for face obfuscation. It involves applying a filter to the image that averages the brightness of pixels in a small area.
- Pixelization: Divides the image into a grid of small squares and assigns a single color to each square. This technique effectively obscures faces by reducing the image resolution and making it difficult to distinguish facial features.
- Masking: Employs a black mask to cover the detected face. This method provides the most comprehensive obfuscation of facial features.
- Detection Only: No face obfuscation is performed; only the edges of the detected face are highlighted in green. This mode is intended for testing purposes only

The four anonymization techniques effectively conceal facial features, and the choice of the most appropriate method depends on the specific application. Gaussian blurring is well-suited for images with intricate details, as it helps preserve the overall appearance of the image while obscuring faces. Blurring is a versatile technique suitable for various image types. Pixelization is ideal for images requiring complete obfuscation, as it effectively obscures facial features. Masking provides the most comprehensive obfuscation of faces.

In the next figures are shown the various options described above:

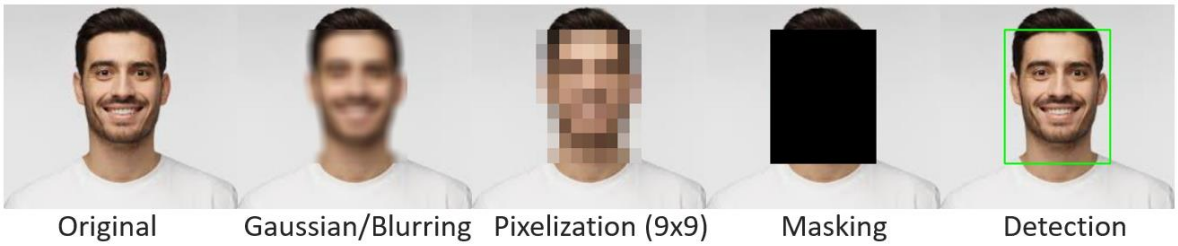


Figure 56: Comparison of available face obfuscation methods

These face obfuscation techniques have gained widespread adoption for privacy protection in situations where facial recognition or identification poses a risk. They provide varying levels of anonymity while maintaining the overall structure of the face.

In summary, the Image Anonymization Module for the TALON project offers a comprehensive and robust solution for anonymizing faces in images and video files. It provides flexibility in face detection techniques and anonymization methods, providing users with a range of customization options to suit their specific requirements. The ImAM ensures that sensitive facial information is effectively concealed while preserving the visual content of images and videos.

The ImAM also provides a range of customization options. These options allow users to tailor the anonymization process to their specific requirements.

One way to customize the ImAM is to modify the configuration file, which is named configuration.ini. This file contains a number of parameters that can be used to control the behavior of the ImAM:

- face_detection_technique: the face detection technique applied to the image files ("dnn", "haarcascades", "both", "lbp").
- video_detection_technique: the face detection technique applied to the video files ("dnn", "haarcascades", "lbp")
- dnn_path / dnn_video_path: specify how much sensible has to be the dnn applied. Note: more sensible values mean higher probability to find faces and also higher false positive.
- haarcascades_scalefactor: specify the sensibility of the haarcascades applied.
- haarcascades_minneighbors: specify how much neighborhood is required to pass it as a face rectangle during sliding window approach. Note: lower values imply greater face detection but also greater false positive.
- haarcascades_minsize: specify the smallest window that haarcascades can detect.
- masking_image: the obfuscation technique to apply to image files to the detected faces ("gaussian", "blurring", "pixelization", "masking", "detection")
- masking_video: the obfuscation technique to apply to video files to the detected faces ("gaussian", "blurring", "pixelization", "masking", "detection")
- pixel_depth: specify how many pixel (pixel_depth x pixel_depth) will be used to obfuscate the detected faces when masking_image/ masking_video is set to pixelization.
- blur_depth: specify the depth of the blurring applied in case of gaussian/blurring.

Another setting in the configuration.ini permits to remove EXIF tags from files:

- exiftag_removal: if yes, the EXIF tags will be removed in the generated output file, using the Pillow library, a fork of PIL (Python Imaging Library).

Exif tags are hidden data embedded in image files, containing information like the camera model, date and time of capture, and location details. When anonymizing images, these tags can reveal personal information about the individuals in the photo also if faces are obfuscated.

For instance, location tags can expose where a person has been, while camera information can link them to specific devices. To protect privacy, removing Exif tags is crucial during image anonymization. By eliminating these embedded details, you ensure the privacy of individuals in the images.

Other important settings in configuration.ini is the possibility of retaining the original image/video files to permit the ImAM to perform pseudonymization. For this purpose, ImAM delivers a protected Mongo database to store the original files. MongoDB is a NoSQL database, flexible, scalable and easy to implement.

Parameters for MongoDB are:

- pseudonymization: if yes, the Identity and Access Management system will retain the original image files within the MongoDB database. This precautionary measure is due to the potential need to retrieve the original data for legal or compliance purposes in the future. If pseudonymization is set to “no”, the original images are not retained, and the mongo’s parameters can be omitted.
- mongo_url : the URL where MONGO DB is located.
- mongo_port = the MONGO DB port.
- mongo_database = the database to use (and to create if it is not already created).
- mongo_collection = the collection to use (and to create if it is not already created).
- mongo_user = the mongo user.
- mongo password = the mongo user password.

Through the configuration.ini file it is also possible to configure:

- anonymization_tag: it is a string that will be added at the beginning of the output file name, can be left empty.
- date_tag: if set to “*yes”* the date and time (format YYYYMMDD_HHMMSS) will be added just after the anonymization tag, if present, in the output file name.
- port_flask: specify the flask port to be used.
- destination_url = specify the *url* of the output file’s destination (edge server).
- input_url: specify the *url* of the input file’s server (client).
- test_phase: if “yes” the generated output files are not sent to the destination url but kept locally for testing.
- dockerization: specify if the Image Anonymization function has been delivered via containers.

5.2.2 Communication Interfaces

The ImAM facilitates interaction through a set of well-defined HTTP REST APIs, designed with Flask (a complete and easy to use micro web framework), suited to the TALON use cases that need to anonymize image data. These APIs allow authorized users to perform the following actions:

- **Image Upload and Processing**: Upload images for subsequent processing, including face detection and anonymization. Users can specify their desired options for these operations.
    - POST/upload
- **Module Configuration**: Pertaining to the Image Anonymization Module, configurable parameters are provided to accommodate the diverse image scenarios encountered within TALON's UC3 and UC4 use cases. These parameters encompass, but are not limited to:
    - Face Detection and Anonymization Techniques: The module supports the selection of appropriate techniques to effectively address the varying requirements of different image scenarios.
    - Image Anonymization Module Tuning: Specific adjustments can be made to optimize the module's performance for each use case.
    - TALON Server API: The destination API for processed images can be configured.
    - Output File Formatting: Prefix label and timestamps can be incorporated into the naming of processed files.

  Configuration of the module is facilitated through the following two APIs:
    - PUT/update_config: This API enables the modification of configuration parameters.
    - GET/retrieve_config: This API facilitates the retrieval of the current configuration settings.

  By leveraging these configurable options and APIs, the Image Anonymization Module can be tailored to meet the specific needs of each use case, ensuring optimal performance and adherence to privacy guidelines.
- **GDPR Compliance**: The TALON framework incorporates a set of APIs that facilitate compliance with the GDPR, particularly concerning the "Right to erasure" ("Right to be forgotten", Article 17 of the GDPR). This right empowers individuals to request the erasure of their personal data from databases.

  The following APIs enable the identification, retrieval for visual check, and subsequent deletion of images in accordance with the *right to be forgotten*:

  1. **Identification**:
    - GET/retrieve_image_info_by_name: This API allows for the retrieval of information pertaining to an image based on its filename.
    - GET/retrieve_image_info_by_date: This API enables the retrieval of images captured on a specific date.

  2. **Retrieval**:
    - GET/download_image_id: This API facilitates the download of an image based on its unique identifier.
    - GET/download_image_name: This API allows for the download of an image by specifying its filename.

  3. **Deletion**:
    - POST/delete_image: This API enables the permanent deletion of an image from the retention database.

By leveraging these APIs, TALON ensures that images can be efficiently located, verified, and subsequently erased from the system, empowering individuals to exercise their right to be forgotten and fostering GDPR compliance.

There are nine HTTP REST APIs defined for Image Anonymization Module (Figure 57):

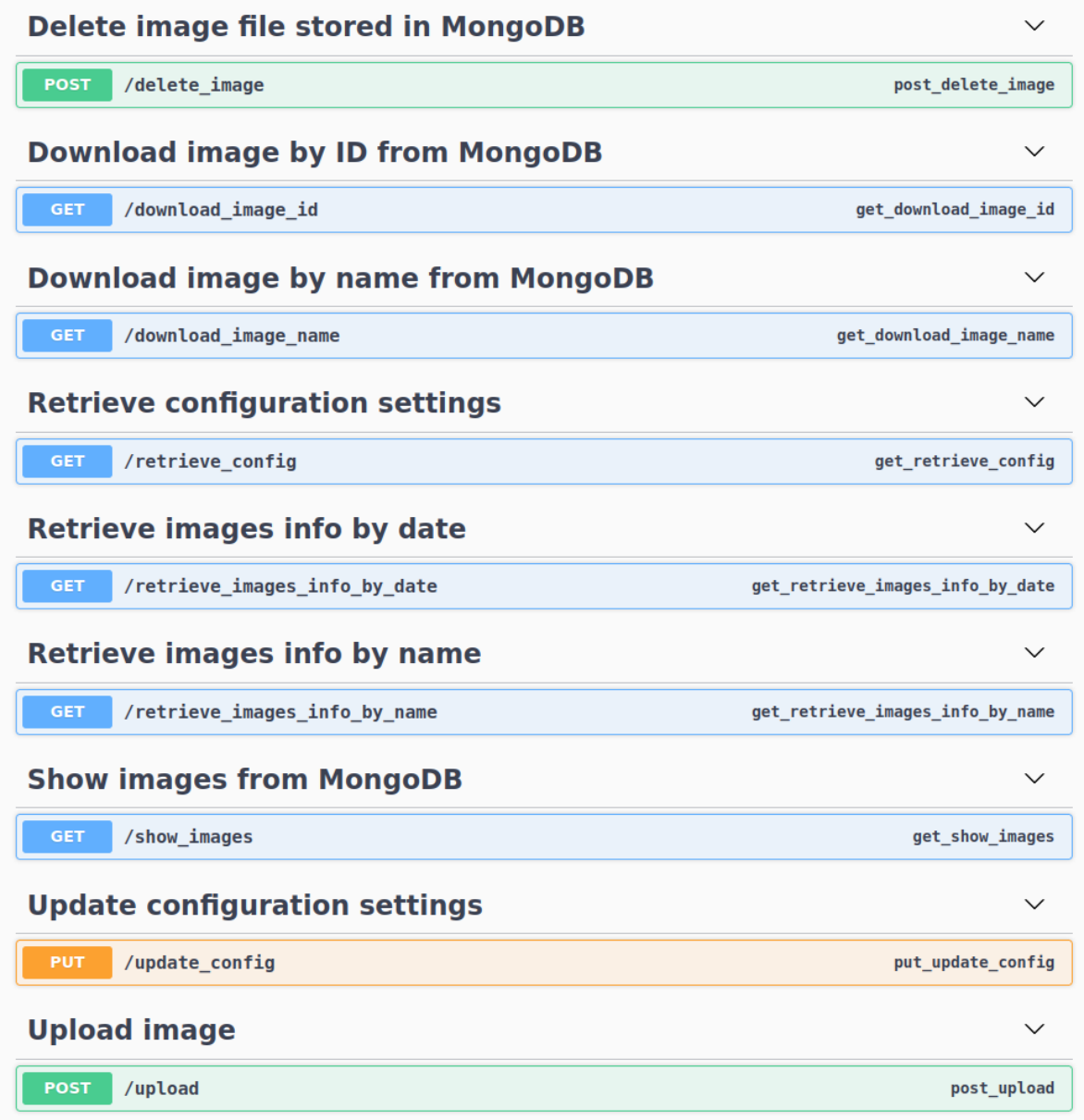


Figure 57: Image Anonymisation Swagger

5.2.2.1 API for image upload and processing

A single HTTP REST API is provided to ingest, and elaborate, the file. This API will be utilized by services (clients) that require to anonymize images before making them available among Talon's partners.

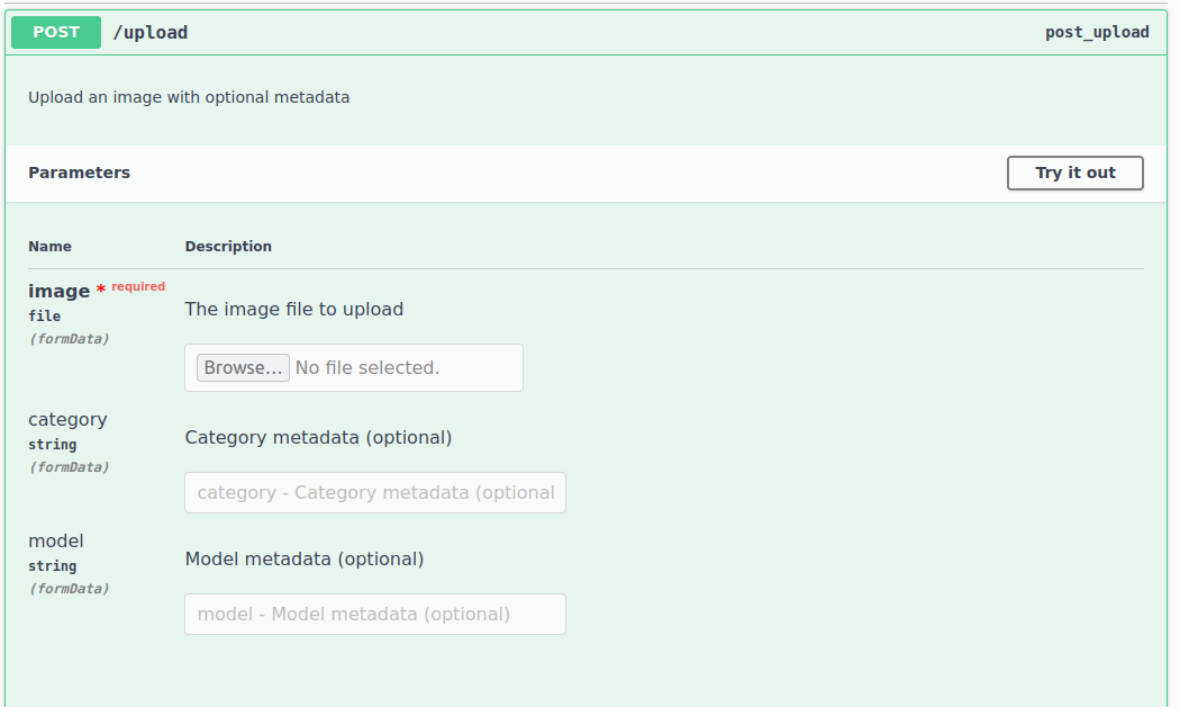


Figure 58: API /upload

It receives in input the *file* and optionally the metadata (*category* and *model*), see below examples:

curl -X POST -F image=@//directory/Volti.jpg -F category='face01' -F model='euro01' http://127.0.0.1:5000/upload

The following file type are supported:

- jpg / jpeg
- png
- gif
- mp4
- 3gp
- avi
- mov
- mkv
- wmv

Below are the possible responses:

Figure 59: API /upload: responses

When the upload and elaboration succeed the code 200 with {category, model} received from the Edge Server are reported back to the Client.

In case of an error response, the proper code is returned with an explanatory message:

Table 14: Error Description and Response

Error Number	Error Description	Message
400	Bad Request, possible message	“No file provided in the request”
415	Unsupported Media Type, possible message	“File not supported”
500	Internal Server Error	“'NoneType' object has no attribute 'shape'”
503	Server Unavailable (Mongo DB Connection Error)	“Connection failure”

5.2.2.2 APIs for configuration handling

Two HTTP REST APIs have been implemented to manage configuration settings: one for uploading new configurations and another for retrieving the current configuration.

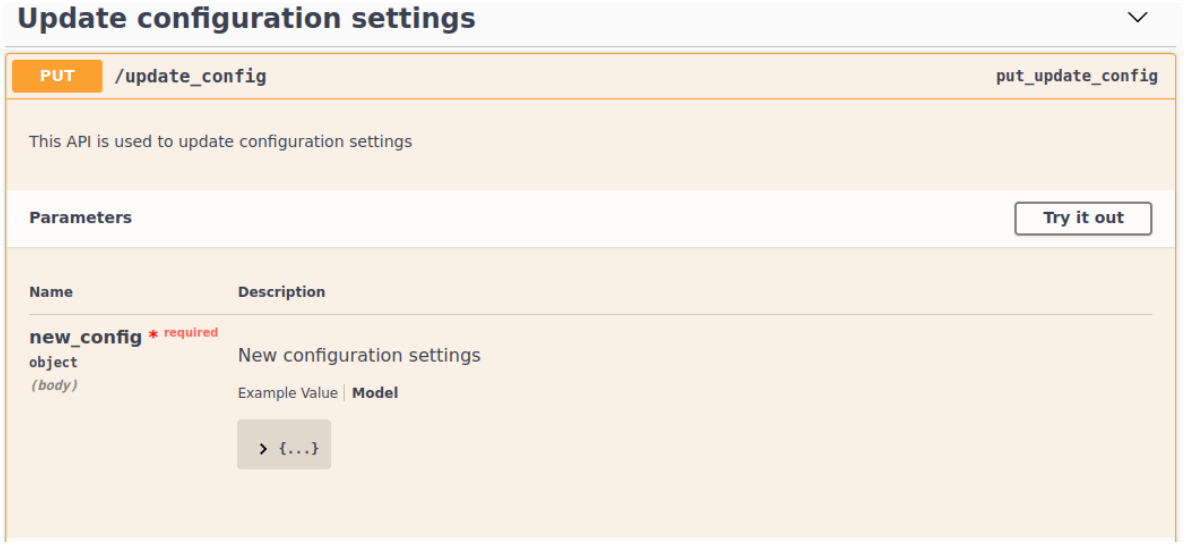


Figure 60: API / update_config

The API *update_config* receives in input a *json* file with the parameters to be updated:

curl -X PUT -H "Content-Type: application/json" -d @/directory//new_configuration.json http://localhost:5000/update_config

Follow an example of json file:

{"anonymization_tag":"ANONYM","face_detection_tecnique":"DNn","blur_depth":"2","date_tag":"no", masking_image":"masking","masking_video":"pixelization","pixel_depth":"9 "}

Below are the possible responses:

Figure 61: API /update_config: responses

When the ImAM configuration update succeeds the code 200 is reported with the message:

- 200: Configuration settings successfully, message:
    - Configuration.ini successfully updated

In case of an error, the proper code is returned with an explanatory message:

- 400: Bad Request, possible messages:
    - " Failed to decode JSON object"

In the below figure are reported the admitted configuration values.

```
{
    "settings":
    {
        "pseudonymization": ["no","yes"],
        "mongo_url": "any",
        "mongo_port": "int",
        "mongo_database": "any",
        "mongo_collection": "any",
        "mongo_user": "any",
        "mongo_password": "any",
        "masking_image": ["gaussian","blurring","pixelization","masking","detection"],
        "masking_video": ["gaussian","blurring","pixelization","masking","detection"],
        "pixel_depth": "int",
        "blur_depth": "int",
        "face_detection_technique": ["dnn","haarcascades","both", "lbp"],
        "video_face_detection_technique": ["dnn","haarcascades", "lbp"],
        "haarcascades_path": "any",
        "haarcascades_scaleFactor" : "float",
        "haarcascades_minNeighbors": "int",
        "haarcascades_minSize": "int",
        "dnn_path": "any",
        "dnn_depth": "float",
        "dnn_video_depth": "float",
        "file_input_dir": "any",
        "file_output_dir": "any",
        "file_backup_dir": "any",
        "file_not_supported_dir": "any",
        "anonymization_tag": "any",
        "date_tag": ["no","yes"],
        "port_flask": "int",
        "destination_url": "http",
        "input_url": "http",
        "test_phase": ["no","yes"],
        "exiftag_removal": ["no","yes"]
    }
}
```


Figure 62: Admitted configuration values

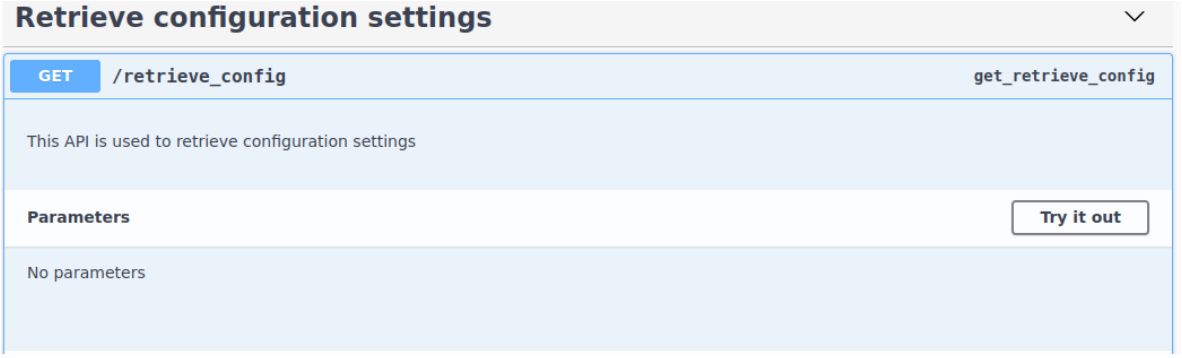


Figure 63: API /retrieve_config

To retrieve the current configuration, no parameters are needed in input.

An example of request and answer message:

curl -X GET http://localhost:5000/retrieve_config

{ "anonymization_tag": "ANONYM",
"blur_depth": "15",
"date_tag": "yes",
"destination_url": "http://127.0.0.1:5001/upload_server",
"dnn_depth": "0.14",
"dnn_path": "/home/teiginc/TALON/",
"dnn_video_depth": "0.2",
"dockerization": "yes",
"exiftag_removal": "no",
"face_detection_technique": "both",
"file_backup_dir": "/home/teiginc/TALON/backup_file_dir/",
"file_input_dir": "/home/teiginc/TALON/input_file_dir/",
"file_notsupported_dir": "/home/teiginc/TALON/notsupported_file_dir/",
"file_output_dir": "/home/teiginc/TALON/output_file_dir/",
"haarcascades_minneighbors": "3",
"haarcascades_minsize": "20",
"haarcascades_path": "/home/teiginc/TALON/",
"haarcascades_scalefactor": "1.15",
"input_url": "http://127.0.0.1:5001/upload_client",
"masking_image": "masking",
"masking_video": "masking",
"mongo_collection": "pilot3",
"mongo_database": "talon",
"mongo_password": "root",
"mongo_port": "27017",
"mongo_url": "localhost",
"mongo_user": "root",
"pixel_depth": "9",
"port_flask": "5000",
"pseudonymization": "yes",
"video_face_detection_technique": "dnn"}

Responses — Response content type: application/json

Code	Description
200	Configuration settings retrieved successfully Example Value | **Model**

```
{
  anonymization_tag        string
                           Anonymization tag to be inserted as prefix in the anonymized output file name, if empty no tag will be used
  blur_depth               integer
                           Blur depth
  date_tag                 string
                           Enable date tag, the tag YYYYMMDD_HHMMSS is added, after anonymization tag, in the anonymized output file name
                           Enum:
                           > Array [ 2 ]
  destination_url          string
                           Destination URL
  dnn_depth                float
                           DNN depth
  dnn_path                 string
                           DNN path
  dnn_video_depth          float
                           DNN video depth
  exiftag_removal          string
                           Enable exif tag removal to the anonymized output file
                           Enum:
                           > Array [ 2 ]
  face_detection_technique string
                           Face detection technique
                           Enum:
                           > Array [ 4 ]
  file_backup_dir          string
                           File backup directory
  file_input_dir           string
                           File input directory
  file_not_supported_dir   string
                           File not supported directory
  file_output_dir          string
                           File output directory
  haarcascades_minNeighbors integer
                           Haarcascades minNeighbors, how much neighborhood is required to pass it as a face rectangle
  haarcascades_minSize     integer
                           Haarcascades minSize, the smallest window that haarcascades can detect
  haarcascades_path        string
                           Haarcascades path
```

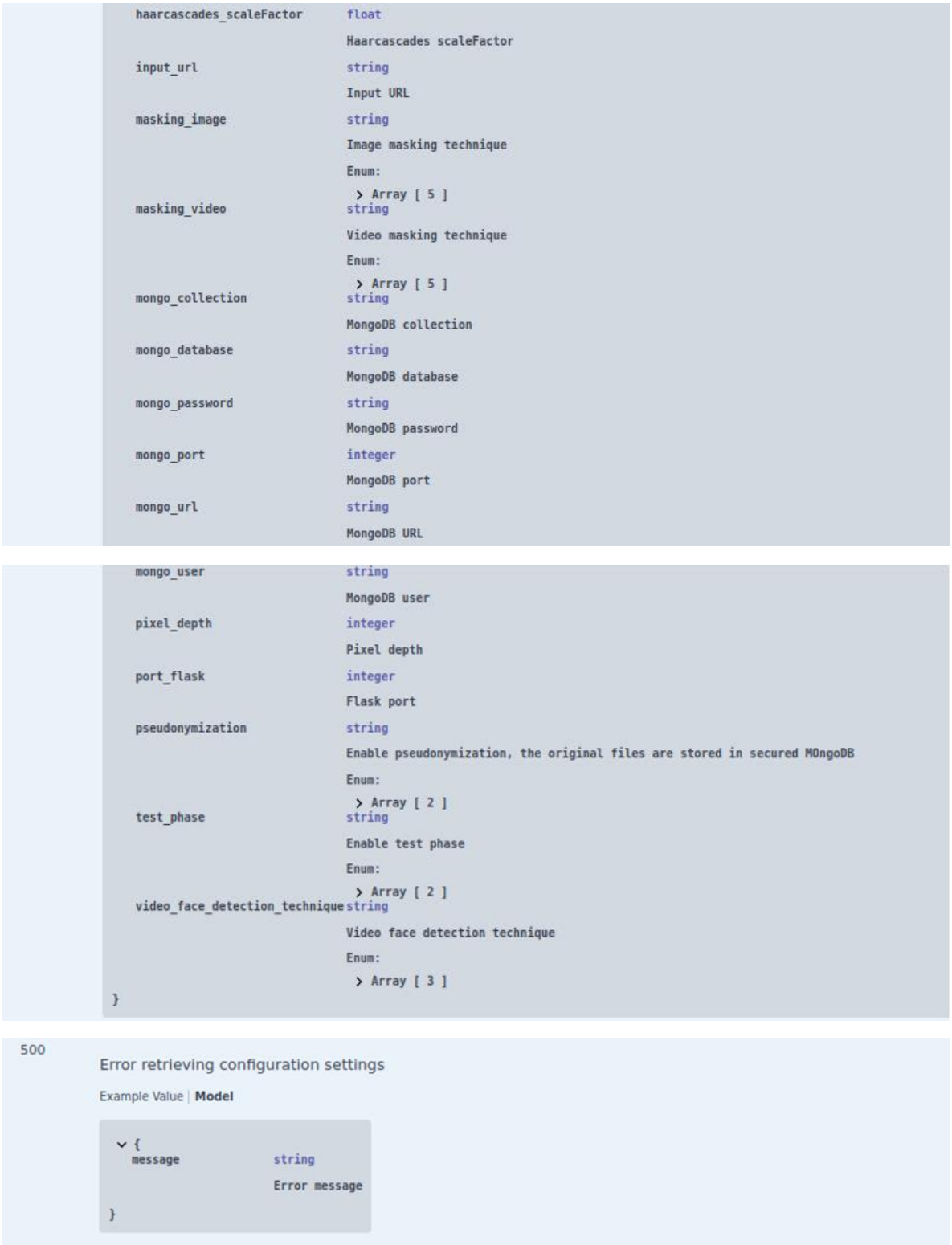


Figure 64: API /retrieve_config: responses

5.2.2.3 APIs for GDPR Compliance

The following HTTP REST APIs are used to retrieve information of record stored in MongoDB (the original images), to download the files stored and to delete them from MongoDB.

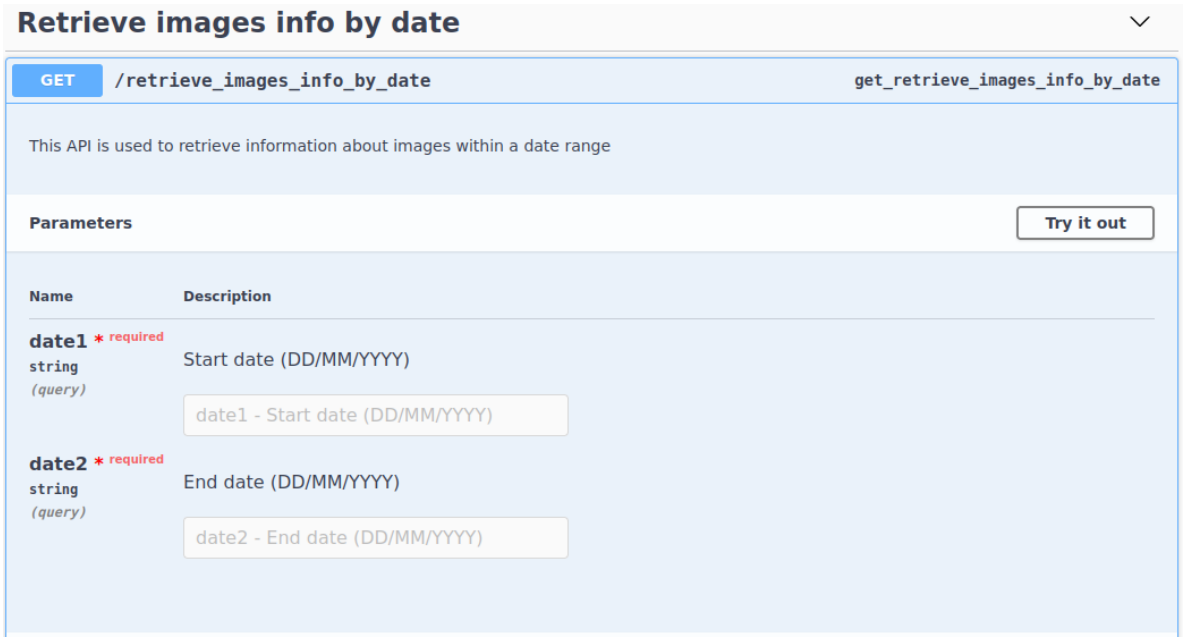


Figure 65: API /retrieve_images_info_by_date

This API is used to retrieve information of all the record with the stored date included in the request's time window (date1<->date2).

An example:

curl -X GET "http://localhost:5000/retrieve_images_info_by_date?date1=01/01/2022&date2=31/12/2023"

Example of result:

[{
"_id": "65bd7807f6f82b8f2746efc7",
"date": "02/02/2024",
"name": "Images_112.png",
"type": "image"
},
{
"_id": "65bd780df6f82b8f2746efc9",
"date": "02/02/2024",
"name": "Images_248.png",
"type": "image"
}]

Below are the possible responses:

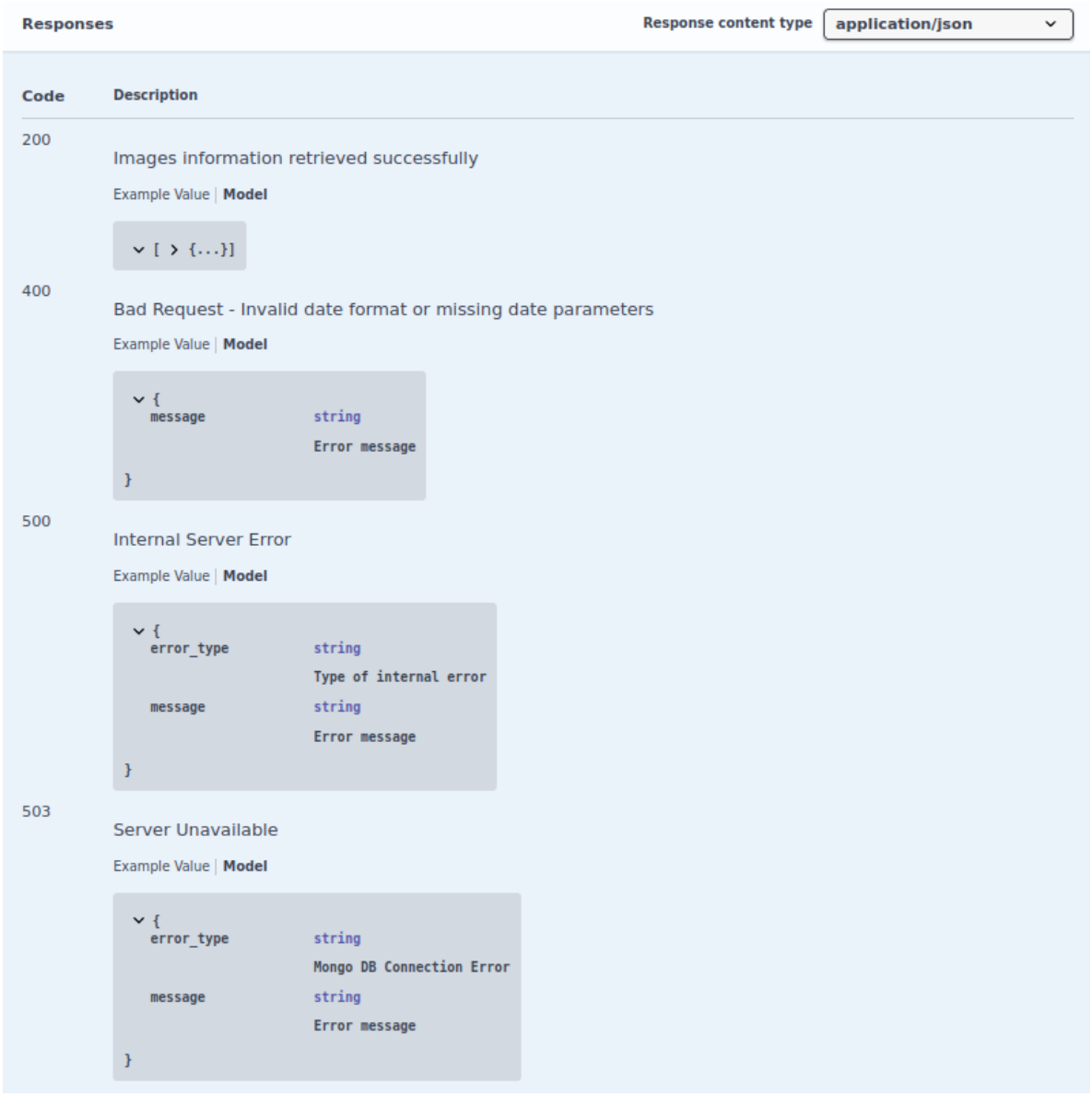


Figure 66: API /retrieve_images_info_by_date: responses

In case of an error reponse, the proper code is returned with an explanatory message:

- 400: Bad Request, possible messages:
  - “Invalid date format. Please use DD/MM/YYYY format”
  - “Both date1 and date2 parameters are required”
- 500: Internal Server Error, with
  - Error_type: InternalServerError
  - Message: “Internal Error”
- 503: Server Unavailable
  - Error_type: MongoDBConnectionError
  - Message: localhost:27017: [Errno 111] Connection refused, Timeout: 30s, Topology Description: <TopologyDescription id: 65be66b157ec880490ad2d13, topology_type:

Unknown, servers: [<ServerDescription ('localhost', 27017) server_type: Unknown, rtt: None, error=AutoReconnect('localhost:27017: [Errno 111] Connection refused')>]

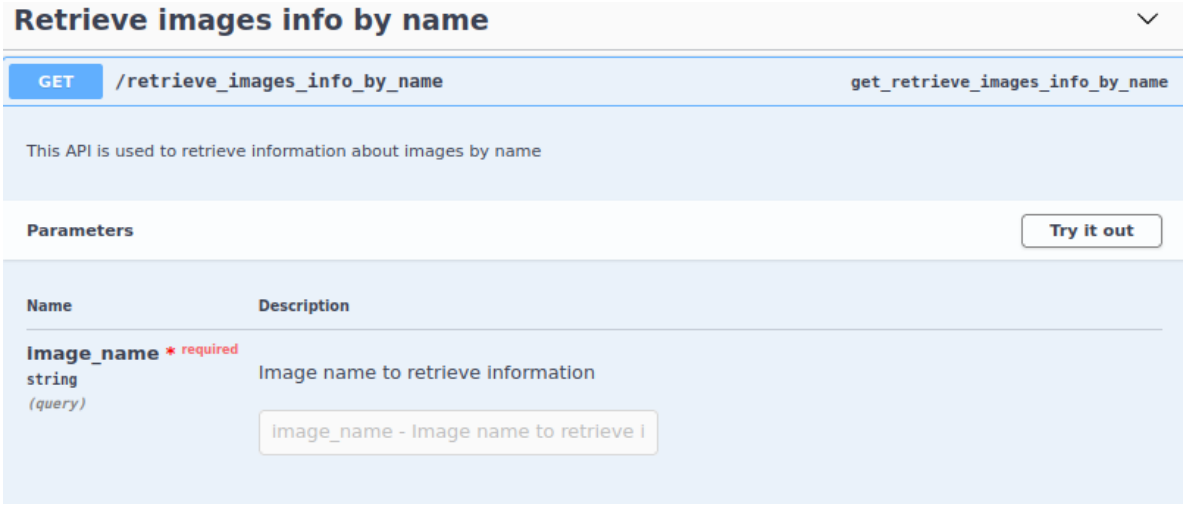


Figure 67: API /retrieve_images_info_by_name

Another possibility to retrieve records from MongoDB is to search them for file name (image_name):

curl -X GET "http://localhost:5000/retrieve_images_info_by_name?image_name= Images_248.png "

Example result:

[{
"_id": "65bd780df6f82b8f2746efc9",
"date": "02/02/2024",
"name": "Images_248.png",
"type": "image"
}]

Below are the possible responses:

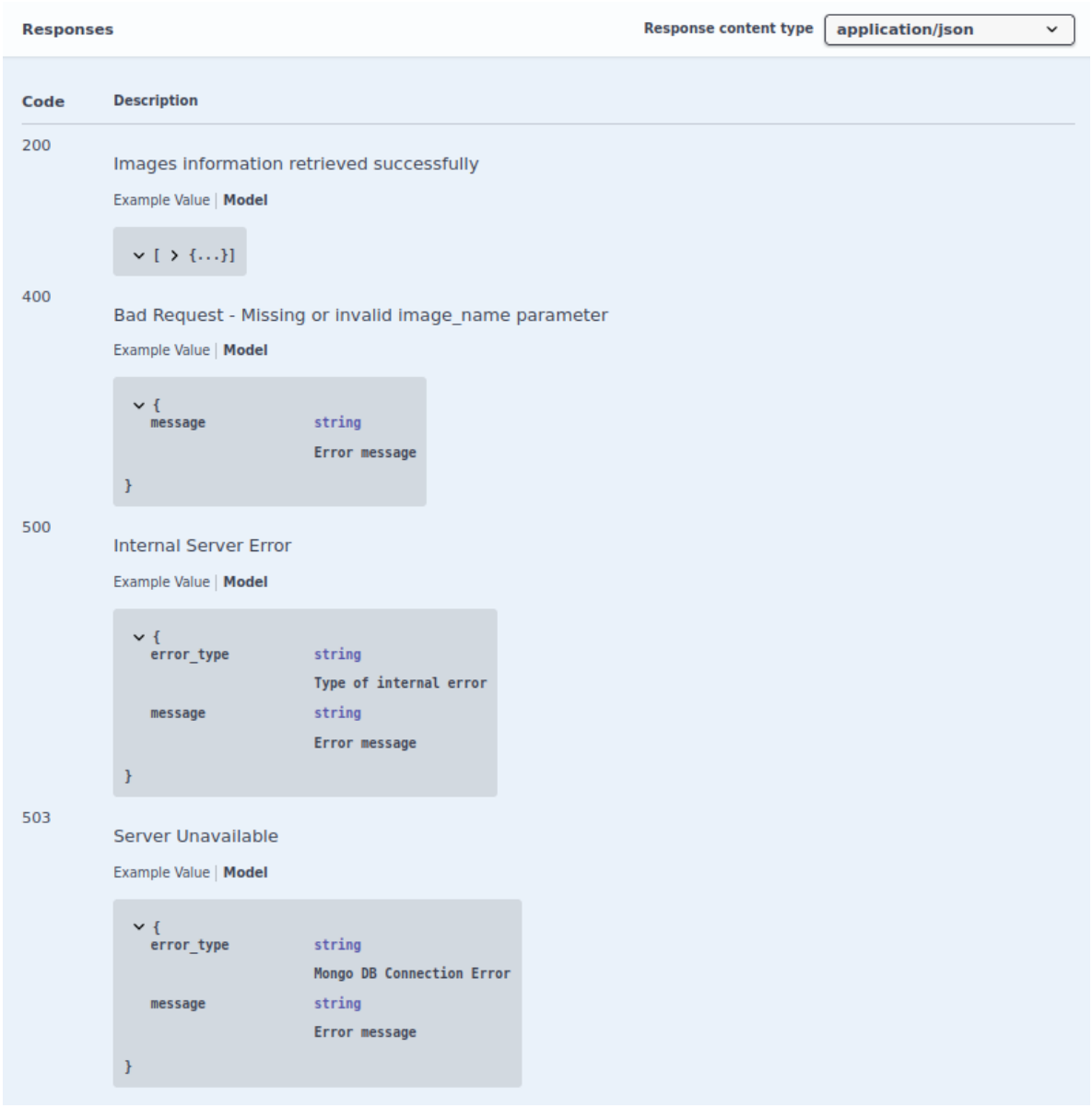


Figure 68: API /retrieve_images_info_by_name: responses

In case of an error response, the proper code is returned with an explanatory message:

- 400: Bad Request, possible message:
  - “image_name parameter is required”
- 500: Internal Server Error, with
  - Error_type: InternalServerError
  - Message: “Internal Error”
- 503: Server Unavailable
  - Error_type: MongoDBConnectionError

Message: localhost:27017: [Errno 111] Connection refused, Timeout: 30s, Topology Description: <TopologyDescription id: 65be66b157ec880490ad2d13, topology_type: Unknown, servers:

[<ServerDescription ('localhost', 27017) server_type: Unknown, rtt: None, error=AutoReconnect('localhost:27017: [Errno 111] Connection refused')>]

The next two HTTP REST APIs permit to download the file stored in MongoDB.

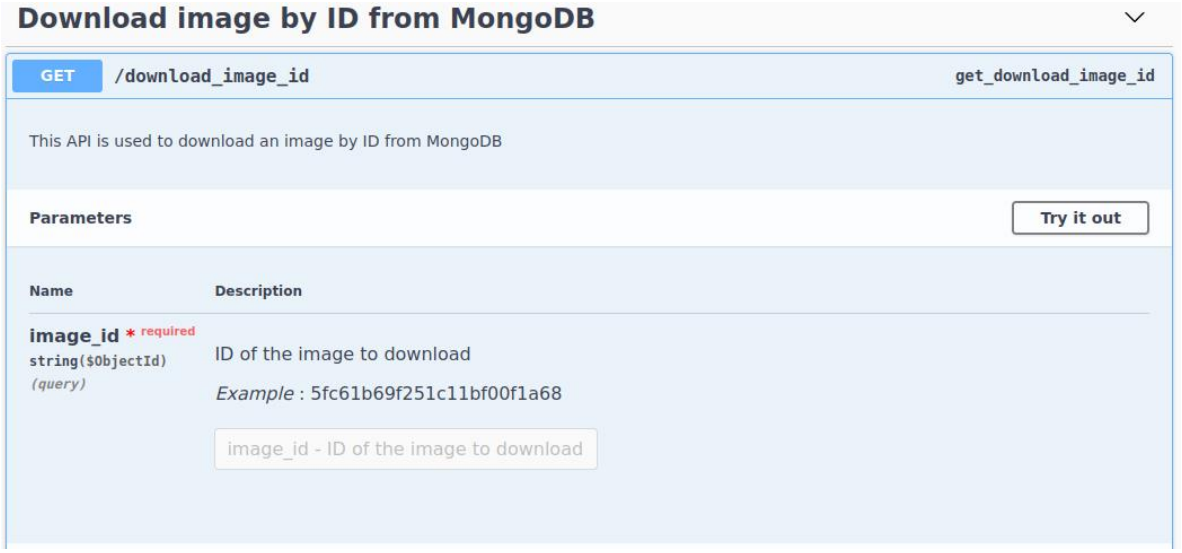


Figure 69: API /download_image_id

The first API uses the unique record id:

curl -X GET http://localhost:5000/download_image_id?image_id=65378a40dcaee0af62b60c1a

When the record is found the 200 response code is given with the message:

- "Record downloaded in TALON/download_image_path directory"

The file will be saved under $HOME/TALON/download_image_path/ directory of the pod anonym_service, to retrieve it the Data Processor operator needs to :

1. connect to the pod:
   kubectl exec -it <anonym_service -POD-NAME> -- /bin/bash
2. enter in the download directory:
   cd TALON/TEST/download_image_path
3. inspect the directory:
   ls -latr
4. copy the file into the desired directory into the worker node hosting the pod:
   kubectl cp <anonym_service -POD-NAME>::$HOME/TALON/TEST/download_image_path/<file_to_copy> /path/for/download//<file_to_copy>

Below are the possible responses:

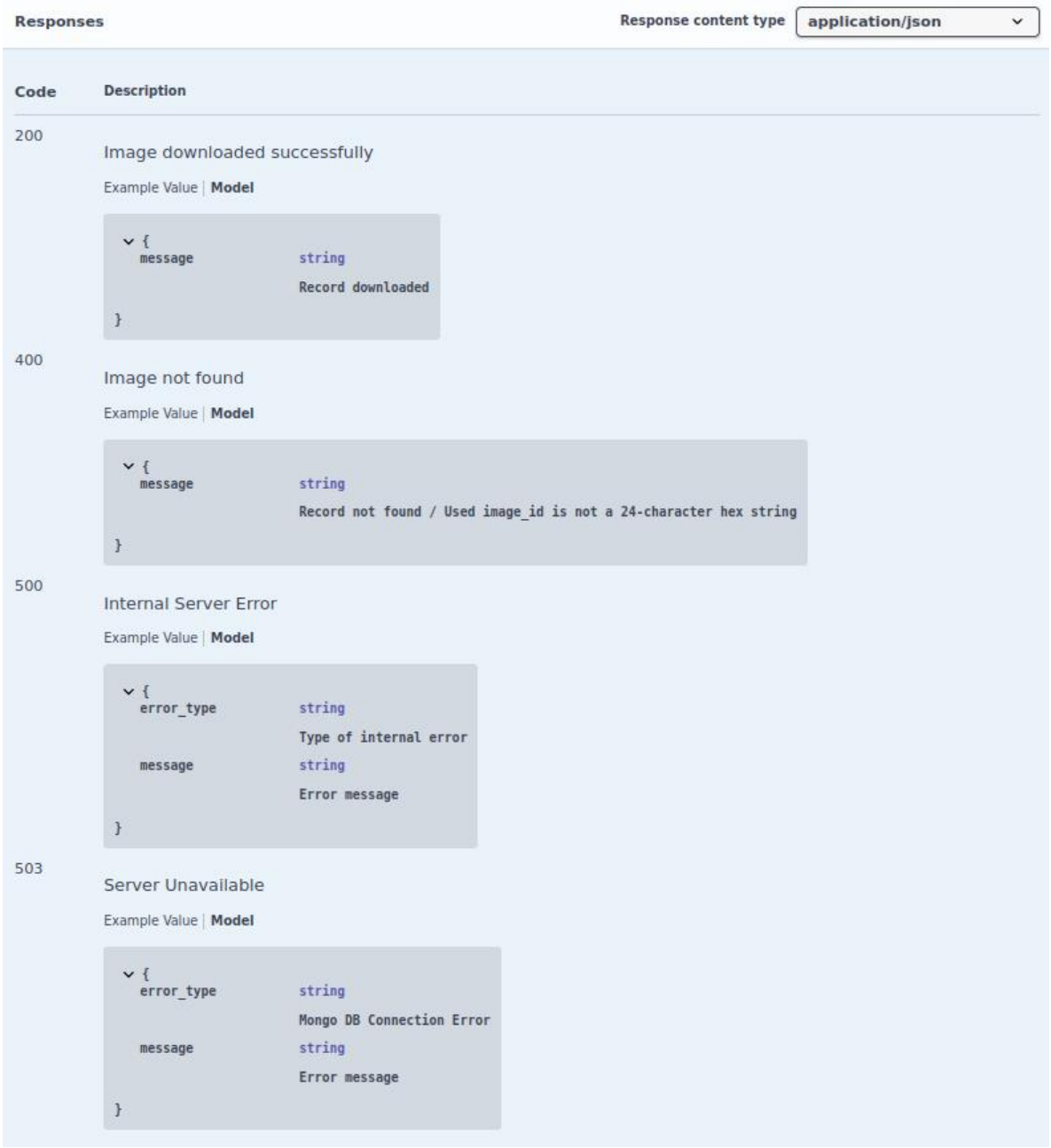


Figure 70: API /download_image_id: responses

In case of an error response, the proper code is returned with an explanatory message:

- 400: Bad Request, possible messages:
  - “Record not found”
  - “Used image_id is not a 24-character hex string”
- 500: Internal Server Error, with
  - Error_type: InternalServerError
  - Message: “Internal Error”
- 503: Server Unavailable

- Error_type: MongoDBConnectionError
- Message: localhost:27017: [Errno 111] Connection refused, Timeout: 30s, Topology Description: <TopologyDescription id: 65be66b157ec880490ad2d13, topology_type: Unknown, servers: [<ServerDescription ('localhost', 27017) server_type: Unknown, rtt: None, error=AutoReconnect('localhost:27017: [Errno 111] Connection refused')>]

The second API for downloading the files from MongoDB uses the name of the file:

Figure 71: API /download_image_name

curl -X GET "http://localhost:5000/download_image_name?image_name=Images_112.png"

When the record is found the 200 response code is given with:

- Message: “Record downloaded”
- Image_path: “/TALON/download_image_path/<file_name>”

The file will be saved under $HOME/TALON/download_image_path/ directory of the pod anonym_service, to retrieve it the Data Processor operator needs to :

1. connect to the pod :
   kubectl exec -it <anonym_service -POD-NAME> -- /bin/bash
2. enter in the download directory:
   cd TALON/download_image_path
3. inspect the directory:
   ls -latr
4. copy the file into the desired directory into the worker node hosting the pod:
   kubectl cp <anonym_service -POD-NAME>:$HOME/TALON/download_image_path/<file_name> /path/for/download//<file_name>

Below are the possible responses:

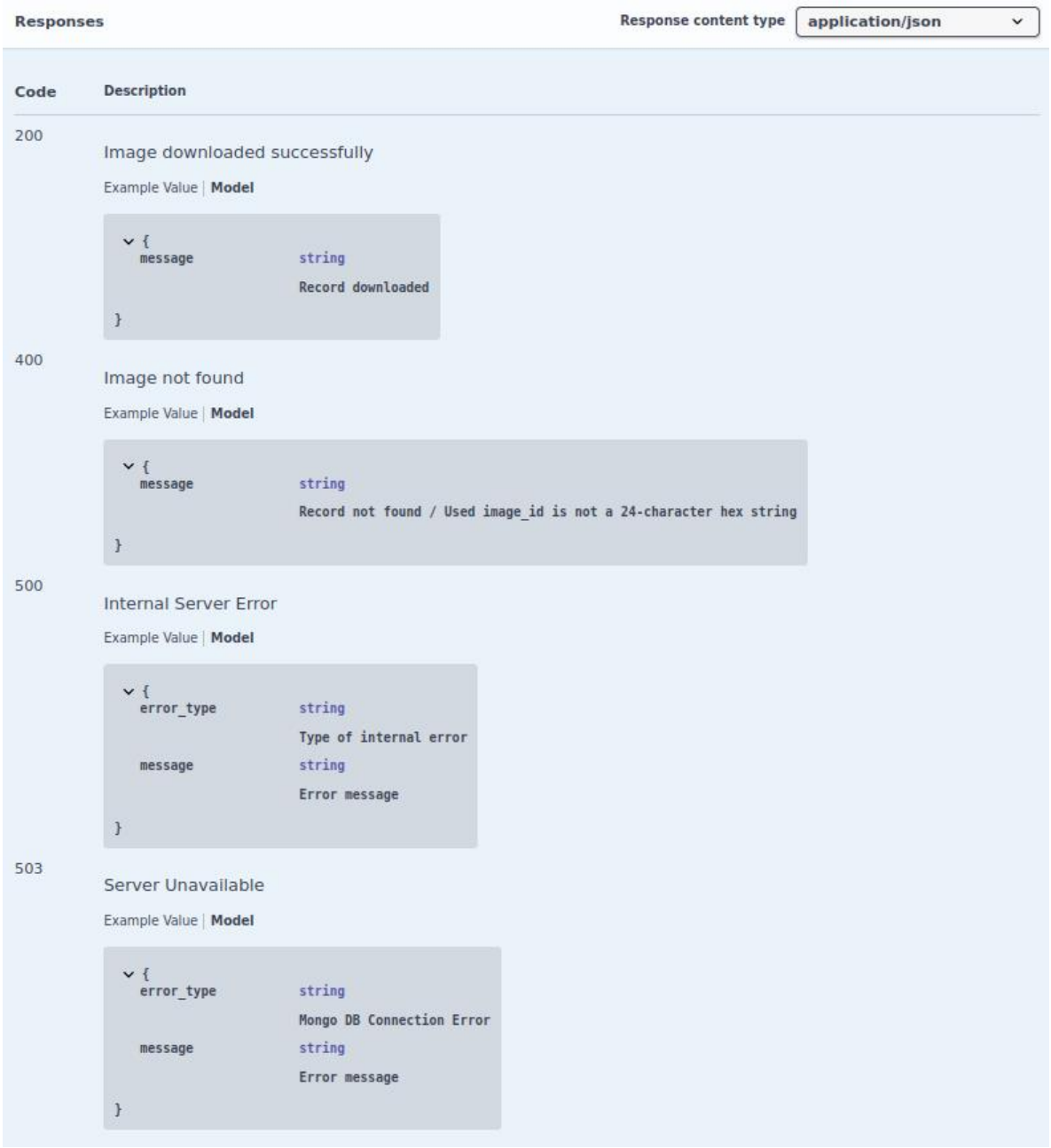


Figure 72: API /download_image_name: responses

In case of an error response, the proper code is returned with an explanatory message:

- 400: Bad Request, possible messages:
  - “Not image found with the given name”
- 500: Internal Server Error, with
  - Error_type: InternalServerError
  - Message: “Internal Error”
- 503: Server Unavailable
  - Error_type: MongoDBConnectionError

- Message: localhost:27017: [Errno 111] Connection refused, Timeout: 30s, Topology Description: <TopologyDescription id: 65be66b157ec880490ad2d13, topology_type: Unknown, servers: [<ServerDescription ('localhost', 27017) server_type: Unknown, rtt: None, error=AutoReconnect('localhost:27017: [Errno 111] Connection refused')>]

The last API is used to delete files stored in MongoDB.

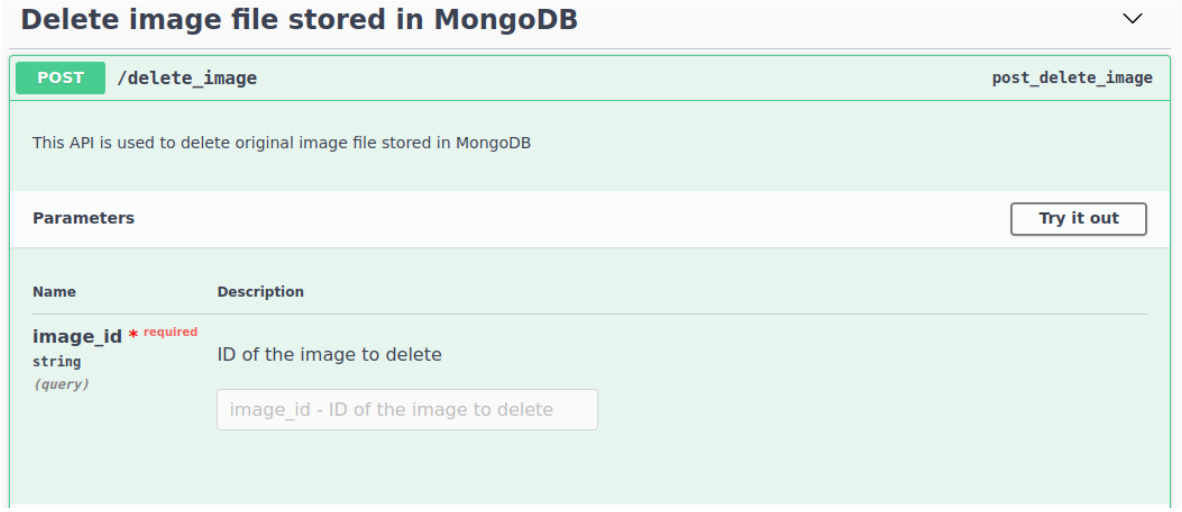


Figure 73: API /delete_image

In this case the unique mongo _id has to be known:

curl -X POST http://localhost:5000/delete_image?image_id=65378a40dcaee0af62b60c1a

When the record is found it is deleted and the 200 response code is given with:

- Message: “Record deleted”
- Image_path: “TALON/ download_image_path”

Below are the possible responses:

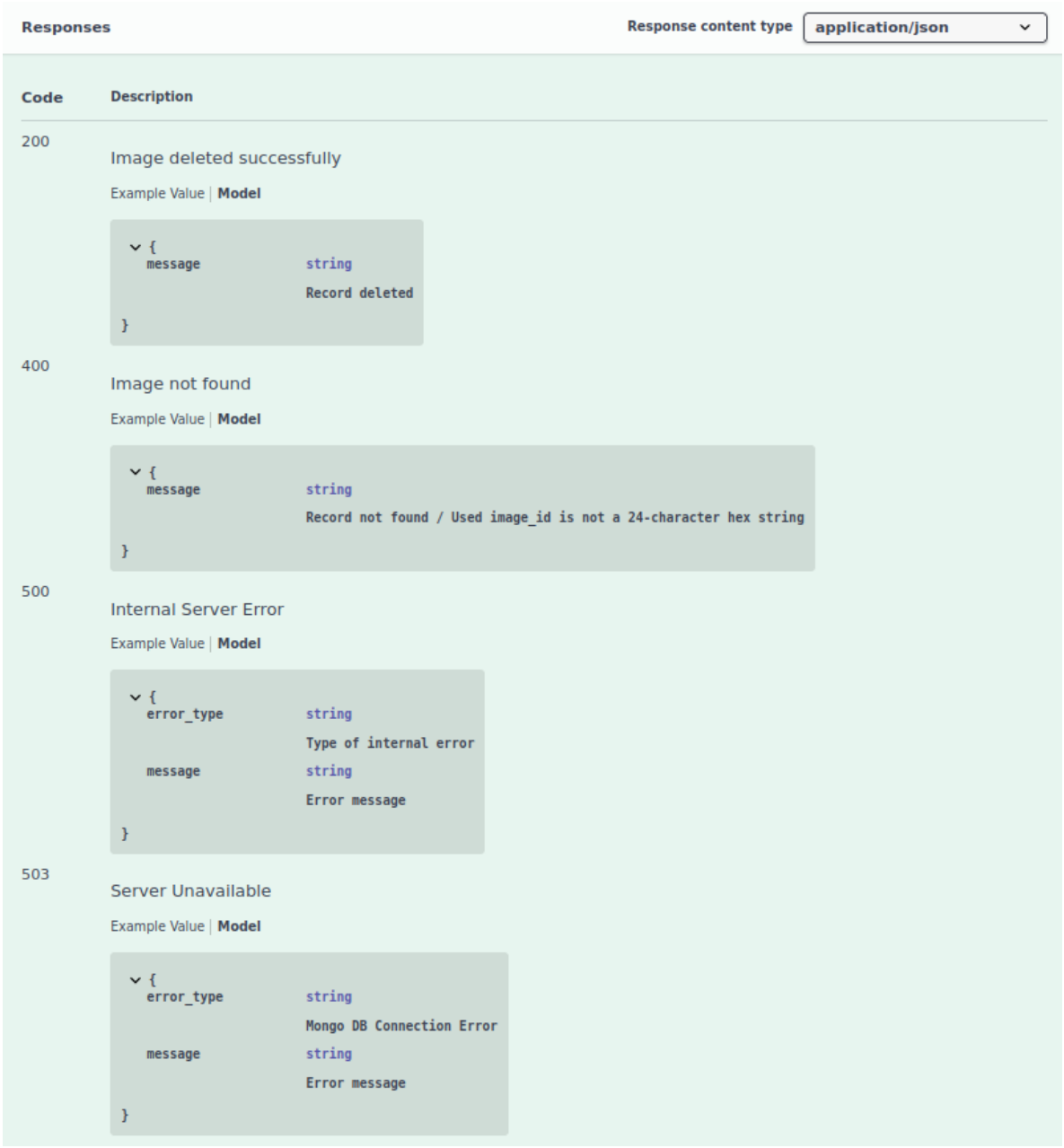


Figure 74: API /delete_image: responses

In case of errors response, the proper code is returned with an explanatory message:

- 400: Bad Request, possible messages:
  - “Record not found”
  - “Used image_id is not a 24-character hex string”
- 500: Internal Server Error, with
  - Error_type: InternalServerError
  - Message: “'<image_id used>' is not a valid ObjectId, it must be a 12-byte input or a 24-character hex string”
- 503: Server Unavailable

- Error_type: MongoDBConnectionError
- Message: localhost:27017: [Errno 111] Connection refused, Timeout: 30s, Topology Description: <TopologyDescription id: 65be66b157ec880490ad2d13, topology_type: Unknown, servers: [<ServerDescription ('localhost', 27017) server_type: Unknown, rtt: None, error=AutoReconnect('localhost:27017: [Errno 111] Connection refused')>]

5.2.3 Scientific and Technical Results

The Talon's UC3 (Scenario 2) and UC4 uses images that could contain human faces. The images are relevant for AI models and need to be anonymized before exchanging them among partners for further elaboration.

To train the ImAM module two different public datasets have been used, "Humans", where most of the images are portraits with varying angles, which should well suit the UC3 Scenario 2 and "Firefighters at Work", a more complex dataset where images with one or more firefighters are present in real scenes, used to train the ImAM in relation to UC4.

During the training has been observed that DNN algorithm performs better than the Haarcascades and LBP methods when the dataset used is "Humans". By increasing the complexity and introducing the "Firefighters at Work" dataset, it has been observed that the DNN algorithm still remains the most effective face detection technique; however, the Haarcascades algorithm can detect faces that are not identified by the DNN algorithm. This observation led to the introduction of the possibility of applying "Both" the DNN and Haarcascades methods, applying the DNN method first allows for the detection of most faces, while the subsequent application of the Haarcascades method to the DNN's results allows for the identification of any missing faces, see Figure 75, Figure 76 and Figure 77.

The "Both" techniques presented satisfactory results.

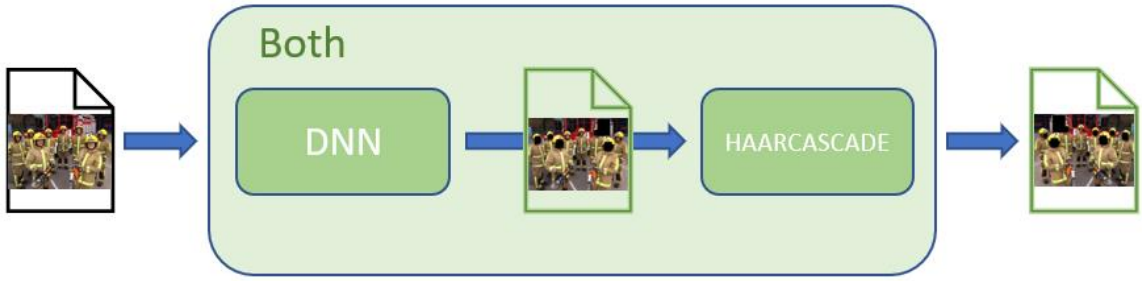


Figure 75: Both technique

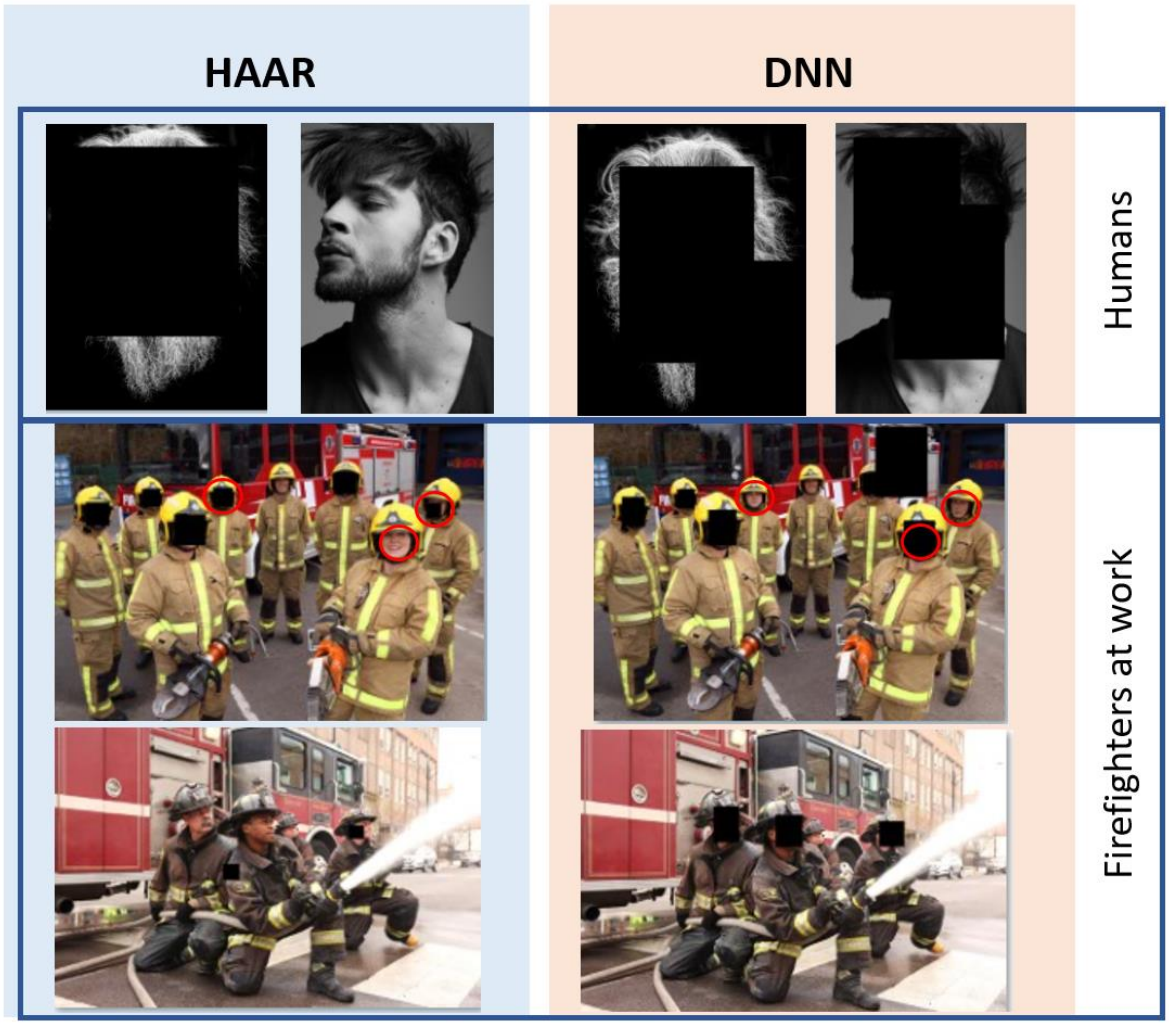


Figure 76: Comparison between Haarcascades and DNN techniques for “Humans” and “Firefighters at work” datasets

In the figure above it is possible to appreciate how DNN performs well with faces (extracted from “Humans” dataset), instead Haarcascades perform very well in case of front faces but has problems with lateral views or grimaces. Always in the previous figure it is possible to check the performances in complex scenarios (“Firefighters at work” dataset). In general, as already explained, the performance of DNN is better, but in some situations Haarcascades can also perform well, as seen in the red circles.

In the next figure are shown the results for LBP and the Both techniques. The LBP performances not well neither in complex faces nor complex scenarios. Instead, both, as a sum of DNN and Haarcascades, is the better choice, compare the red circles in Figure 76 and Figure 77 for Haarcascades, DNN and Both.

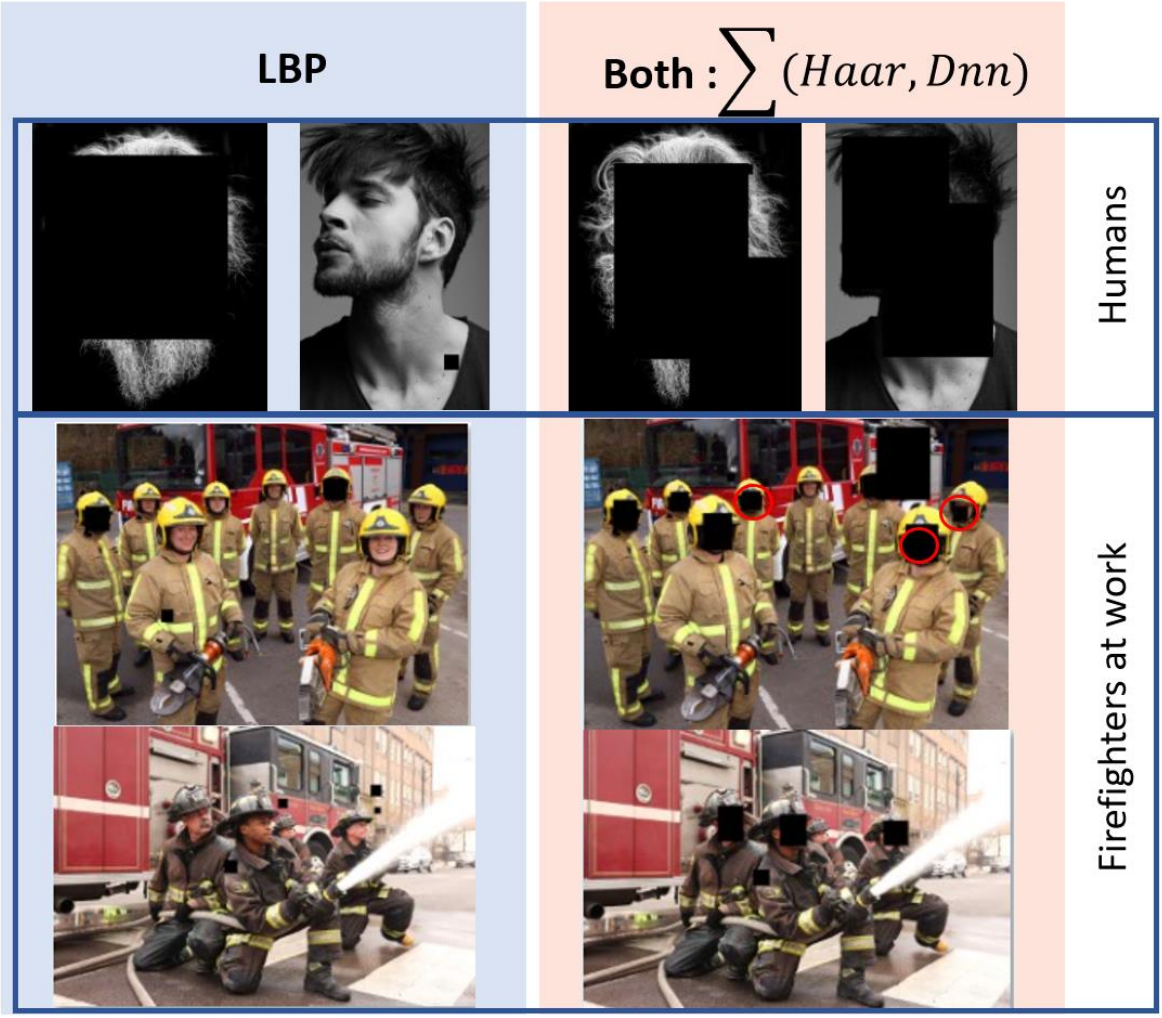


Figure 77: Comparison between LBP and Both techniques for “Humans” and “Firefighters at work” datasets

It was also observed that these methods are sensitive to parameter tuning, as small variations can lead to both an increase in the probability of face detection success and an increase in false positives. This finding highlights the need for specific parameter tuning on the TALON project images to achieve a perfect balance between success rate and false positive detection, see DNN and Haarcascades parameters in Figure 64.

5.2.4 Deployment results

The ImAM can be deployed in a Kubernetes cluster via Deployments yaml files. The 3 pods will act as:

- anonymization_service: The core component responsible for face detection and anonymization operations and also for the HTTP REST API server.
- mongodb: A database that stores and manages the original images, in case of a pseudonymization decision.
- cronjob: A scheduling tool employed to automatically remove original images stored in MongoDB after exceeding a preconfigured retention period.

Hardware requirements for the server/VM hosting the ImAM:

- CPU: Multicore CPU with at least 8 cores
- RAM: At least 32GB
- GPU (Optional): A dedicated GPU can enhance processing performance (e.g., Nvidia RTX 3080)

5.3 Federated Learning Framework

5.3.1 Internal Architecture and Key Technologies

FL, a distributed ML technique, enables the training of ML models across various devices or servers while keeping the data localized, thereby addressing significant challenges in privacy, data security and use of bandwidth. At the core of FL is the principle of model aggregation without the need to exchange local data samples. This approach not only preserves data privacy but also minimizes the risks associated with data centralization and transmission. The technical architecture of FL in TALON leverages several sophisticated aggregation techniques to enhance model performance and ensure data privacy.

In the realm of FL, several advanced aggregation and optimization techniques have been developed to address the unique challenges posed by distributed data sources. Among these, Federated Averaging (FedAvg), Federated Proximal (FedProx), Federated Adam (FedAdam), Federated Adagrad (FedAdagrad) and Federated Yogi (FedYogi) are noteworthy for their contributions to enhancing the performance and efficiency of FL models.

- Federated Averaging (FedAvg): FedAvg is a foundational technique in FL that aggregates model updates from multiple devices to update a global model. The FedAvg algorithm computes the weighted average of local model updates based on the number of data points on each device, effectively combining the local updates to form a new global model. The mathematical expression for FedAvg is as follows:

$$w_{global}^{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_k^{t+1}$$

  where w_k^{t+1} represents the global model weights at each iteration, K is the total number of devices and n_k is the total number of data points on device K [70].

- Federated Proximal (FedProx): FedProx extends FedAvg by adding a proximal term to the local optimization problem, which helps to address issues related to system heterogeneity and data distribution skewness. The proximal term ensures that the local updates do not diverge significantly from the global model. The FedProx can be written as:

$$w_k^{t+1} = arg\min_{w} \left(L_k(w) + \frac{\mu}{2} |w - w_{global}^t|^2 \right)$$

  where $L_k(w)$ is the local loss function of device *k*, μ is a hyperparameter controlling the strength of the proximal term and w_{global}^t is the global model weight [71].

- Federated Adam (FedAdam): FedAdam, as conceptualized within the framework of federated optimization techniques, adapts the Adam optimizer for federated settings. FedAdam is able to combine the benefits of adaptive gradient methods with FL for the adjustment of the learning rates based on the estimation of the first and second moments of gradients. The FedAdam can be formulated as:

$$m_{t+1} = \beta_1 m_t + (1 - \beta_1) g_t$$

$$v_{t+1} = \beta_2 v_t + (1 - \beta_2) g_t^2$$

$$\widehat{m_{t+1}} = \frac{m_{t+1}}{1-\beta_1^t}$$

$$\widehat{v_{t+1}} = \frac{v_{t+1}}{1-\beta_2^t}$$

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\widehat{v_{t+1}}} + \epsilon}\widehat{m_{t+1}}$$

where g_t is the gradient for time *t*, m_t and v_t are the estimations for first and second moments of the gradients, β are the decay rates and ϵ protecting the division by zero [72].

- Federated Adagrad (FedAdagrad): FedAdagrad adapts the Adagrad optimizer, adjusting the learning rate for each parameter based on historical squared gradients. The FedAdagrad rule is given by:

$$g_{t+1} = g_t + \nabla L_k(w_t)^2$$

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{g_{t+1}} + \epsilon}\nabla L_k(w_t)$$

where g_{t+1} accumulates the squared gradients and $\nabla L_k(w_t)$ is the gradient of the loss function on device *k* [72].

- Federated Yogi (FedYogi): FedYogi designed to address the aggressive increase in the learning rate that can occur in FedAdagrad due to the accumulation of squared gradients. FedYogi updates are similar to FedAdam but with a crucial adjustment to the second moment estimate [3]:

$$v_{t+1} = v_t - (1-\beta_2)\text{sign}(v_t - g_t^2)g_t^2$$

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{v_{t+1}} + \epsilon}m_{t+1}$$

The TALON project's FL framework is ingeniously designed to integrate with the cutting-edge concepts of E2C computing, blockchain technology and digital twins serving the project's ambition to revolutionize industrial systems by delivering unparalleled performance, explainability, trustworthiness and transparency. In the context of TALON, FL is deployed within a hybrid learning system capable of learning to optimize prediction efficiency while strictly adhering to privacy requirements.

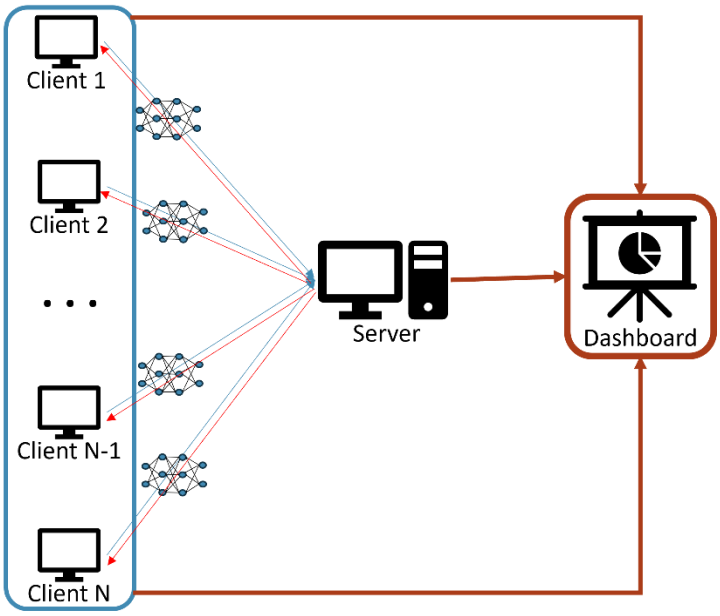


Figure 78: TALON's Federated Learning Concept

Figure 78 presents a schematic depiction of the FL architecture as it would be applied in the TALON project. At the periphery, we have a multitude of mobile devices that stand as proxies for edge devices within the network, such as sensors, drones or other IoT devices. These are the sources of local data and the initial training ground for the individual learning models of the TALON project.

Each mobile device undertakes the computation of a local model based on its unique dataset. The red arrows emanating from the devices symbolize the transmission of these locally computed model updates—consisting of gradients or parameter changes—toward a central server. This central server acts as an aggregation node, collecting all the local updates for further processing. The central server's role is crucial as it synthesizes the incoming updates into an enhanced global model, indicative of the arrow pointing outward from the server. This global model encapsulates the learned features and patterns from across the network's distributed datasets.

Essential to the TALON project, is the incorporation of a dashboard for visualizing the progression of the FL process. This dashboard provides a user interface for monitoring of various metrics that gauge the performance and evaluation of the learning models. Through this dashboard, stakeholders can observe the efficiency of the model training, the accuracy of the global model and other pertinent metrics that are critical for evaluating the success and effectiveness of the FL system.

Key technologies such as Flower are harnessed within the TALON FL framework to ensure secure and efficient implementation. Flower, facilitates the building of FL systems with a focus on adaptability and scalability across various computing environments [73]. By leveraging these technologies, TALON's FL framework promises to significantly enhance data and energy efficiency, reduce latency and provide a scalable solution across different industrial sectors.

5.3.2 Communication Interfaces

Within the TALON project's FL framework, the orchestration of communications between nodes and the central server is facilitated through Flower with its bespoke suite of APIs and communication protocols. As nodes engage with their local datasets and proceed with model training, the updates and interactions with the central server are conducted through the APIs provided by Flower. This framework ensures that the transmission of model updates, which are essential for the iterative learning process, is seamless and standardized. Flower's communication system is specifically

tailored to manage the nuances of FL, such as handling asynchronous updates from nodes and facilitating the aggregation of model parameters.

The central server, which serves as the fulcrum for aggregating the model updates received from the nodes, also relies on Flower's communication system to manage this process efficiently. The aggregation logic, which is central to FL, is encapsulated within the server, ensuring that the most suitable aggregation technique is applied to the incoming updates. The decision regarding the aggregation technique, while not yet finalized, will be determined by several factors like efficiency and performance.

In addition to the node-server interactions, the TALON project leverages a dedicated API to facilitate communication between the nodes and the server with the Dashboard. This API is the conduit through which performance, evaluation, and resource consumption metrics are transmitted to the Dashboard. The Dashboard, in turn, provides stakeholders with a visualization of these metrics, offering insights into the FL process's efficacy and efficiency. The use of Flower and its APIs ensures that the communication within the TALON project adheres to the highest standards of efficiency and security. This choice aligns with the project's commitment to creating a FL environment that is not only robust and scalable but also transparent and user-friendly.

5.3.3 Scientific and Technical Results

A demonstration demo has been implemented to serve as an example of the TALON platform's FL capabilities. The demo comprises of object detection time series forecasting instances.

For object detection the advanced YOLOv8 algorithm was used, a cutting-edge iteration of the You Only Look Once (YOLO) family. YOLOv8 is renowned for its real-time object detection capabilities, employing a single neural network to the full image, dividing the image into regions and predicting bounding boxes and probabilities for each region. These probabilities are then thresholded to detect the presence of objects [74]. A demonstration dataset showcasing CS, sourced from Roboflow Universe[16], was utilised. Figure 79 demonstrates same indicative samples of the dataset.

Figure 79: Roboflow Construction Safety dataset samples

Within this demo, the deployment involved three clients, each processing image data for object detection tasks. The performance of each node was measured in terms of recall and mean Average Precision (mAP) *[75]* after 10 rounds of training. The server aggregates these metrics to provide an average that reflects the collective performance of the federated system as shown in Table 15.

[16] https://universe.roboflow.com/roboflow-100/construction-safety-gsnvb

Table 15: FL Performance metrics for Object Detection

	Recall	mAP
Node 1	94.61%	88.42%
Node 2	74.23%	78.31%
Node 3	66.64%	71.38%
Average	78.49%	79.52%

Similarly, for time series prediction, a model based on LSTM networks was deployed. LSTMs are a type of RNN capable of learning long-term dependencies, particularly adept for time series prediction due to their ability to remember information for prolonged periods *[76]*. The LSTM model underwent training using simulated jet engine data provided by NASA's CMAPSS[17]. Performance, after 10 rounds of training, was quantified using Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) - standard metrics for assessing the accuracy of prediction models *[77]*. Table 16 presents the output of the training.

Table 16: FL Performance metrics for Time Series prediction

	MSE	RMSE	MAE
Node 1	73.84%	85.93%	73.02%
Node 2	78.00%	88.20%	73.37%
Average	76.00%	87.00%	73.00%

The empirical results from these demonstrations are not merely an exposition of the predictive prowess of YOLOv8 or LSTM models within the TALON platform. More crucially, they underscore the platform's capability to facilitate FL. This distinction is vital, as the aim is not to engineer the most superior FL model but to illustrate the platform's inherent ability to conduct federated training effectively. The success of these demonstrations validates the TALON platform's readiness to handle complex tasks like object detection and time series prediction in a distributed learning environment, where data privacy, system efficiency and collaborative learning are paramount.

5.3.4 Deployment Results

Federated learning is a key module being utilized in various UCs. The deployment specifications are yet to be defined, and thus the deployment phase of this module is planned to occur later in the project's life cycle.

5.4 Digital Twins

5.4.1 Internal Architecture and Key Technologies

The internal architectural design of the system will be based on the utilization of deep learning and time series analysis approaches for the generation of synthetic machine sensor data that faithfully replicates the intricate patterns and temporal dynamics observed in the real CNC machine dataset.

[17] *https://data.nasa.gov/Aerospace/CMAPSS-Jet-Engine-Simulated-Data/ff5v-kuh6/about_data*

The candidate approaches that are under consideration to carry out the task are Recurrent neural networks (RNNs), Generative adversarial networks (GANs) and Dynamic Time Warping Barycenter Averaging (DTWBA).

GANs offer a powerful framework for synthesizing machine sensor data through adversarial training between a generator and a discriminator network. The generator network learns to produce synthetic sensor data samples that closely resemble authentic sensor readings, while the discriminator network distinguishes between genuine and synthetic data. Through adversarial optimization, GANs facilitate the generation of sensor data with intricate statistical properties and high-fidelity representations of the underlying data distribution. By capturing the complex relationships and latent structures present in the CNC machine dataset, GANs enable the generation of synthetic sensor data that closely mirrors the statistical characteristics and temporal dynamics of the original data [7].

On the other hand, VAEs provide a probabilistic framework for learning latent representations of the CNC machine sensor data, facilitating the generation of synthetic sensor readings through sampling from the learned latent space. VAEs capture the underlying structure and variability in sensor data streams by modelling the data distribution in a continuous latent space. By encoding sensor data into a low-dimensional latent representation and decoding it back to the original space, VAEs enable the generation of diverse and realistic sensor data samples. Through principled probabilistic modeling, VAEs capture the inherent uncertainty and variability in sensor data, facilitating the generation of synthetic data that preserves the statistical properties and temporal dynamics observed in the real CNC machine dataset [8].

Finally, DTWBA generates new synthetic data by aligning multiple time series to a reference series using dynamic time warping (DTW) and then averaging corresponding data points to create a barycenter representation. This barycenter captures common temporal patterns and dynamics shared among the input time series, allowing for the generation of synthetic data points along this trajectory. DTWBA produces synthetic time series data that preserves the underlying temporal structure and statistical properties observed in the original dataset, making it a valuable tool for data augmentation and synthetic data generation [9].

Once the experimentation with the methodologies has been carried out, the most appropriate approach for the internal architecture of the system will be decided. The chosen methodology should effectively capture the statistical properties, temporal dynamics, and complex relationships present in real CNC manufacturing data, thereby facilitating various downstream applications such as anomaly detection, predictive maintenance, and system optimization in CNC machining processes.

5.4.2 Communication Interfaces

Seamless interaction with the data generation system and the storage of synthetic CNC sensor data will be facilitated through a robust communication interface embedded within the system architecture. The interface will adhere to the standardized protocol of a RESTful API, accompanied by a Swagger documentation explaining the provided API functionalities. The REST API endpoints will enable efficient retrieval and transmission of data between the user and the system repository. Additionally, the system will allow the user to dynamically adjust parameters such as sampling frequency, temporal resolution, and noise characteristics to tailor the synthetic data generation process according to specific requirements. Finally, Swagger will provide comprehensive documentation detailing the available endpoints, request/response formats, and supported operations, ensuring clarity and ease of integration for the TALON solution providers.

5.4.3 Scientific and Technical Results

The implementation of the synthetic time-series data generation module was based on the TimeGAN architecture, a generative model designed specifically for generating synthetic time series data. The

main architectural differences between TimeGANs and traditional GANs lie in how they handle temporal data and capture temporal dependencies. TimeGANs are tailored for generating time series data by incorporating recurrent neural network (RNN) architectures to capture temporal characteristics of the input. Unlike traditional GANs, which primarily handle static data like images, TimeGANs generate sequences of data points ensuring coherence over time, while the discriminator is adapted to process sequential data to assess temporal consistency. This specialized architecture and training procedure makes TimeGANs particularly suited for generating realistic time series data, extending the capabilities of traditional GANs [10].

The TimeGANs model was trained on CNC machine sensor data originating from FACTOR's Nakamura2 machine. The dataset consists of 2,453,281 samples from 14 different sensors, including temperature and current data. From the entirety of the dataset, a subset of 10,000 was used to train the model over 1000 epochs. Before proceeding to the model training, the data first underwent a minor preprocessing. The preprocessing included the scaling of the data inside the range of [0,1] using a MinMaxScaler and the fragmentation of the samples in 24 time-step long sequences. This sequence length also corresponds to the length of the synthetic data generated by the model.

To assess the fidelity of the synthetic data in replicating the behavior observed in the original dataset, we employed principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE), both reduced to two components, to visualize the overlap of the data in a two-dimensional space. By projecting the high-dimensional data onto a 2D plane, PCA and t-SNE enable us to discern patterns and similarities between the original and synthetic datasets, thus evaluating how well the synthetic data captures the underlying structure and variability present in the original dataset. The result of this qualitative assessment is presented in the following figure.

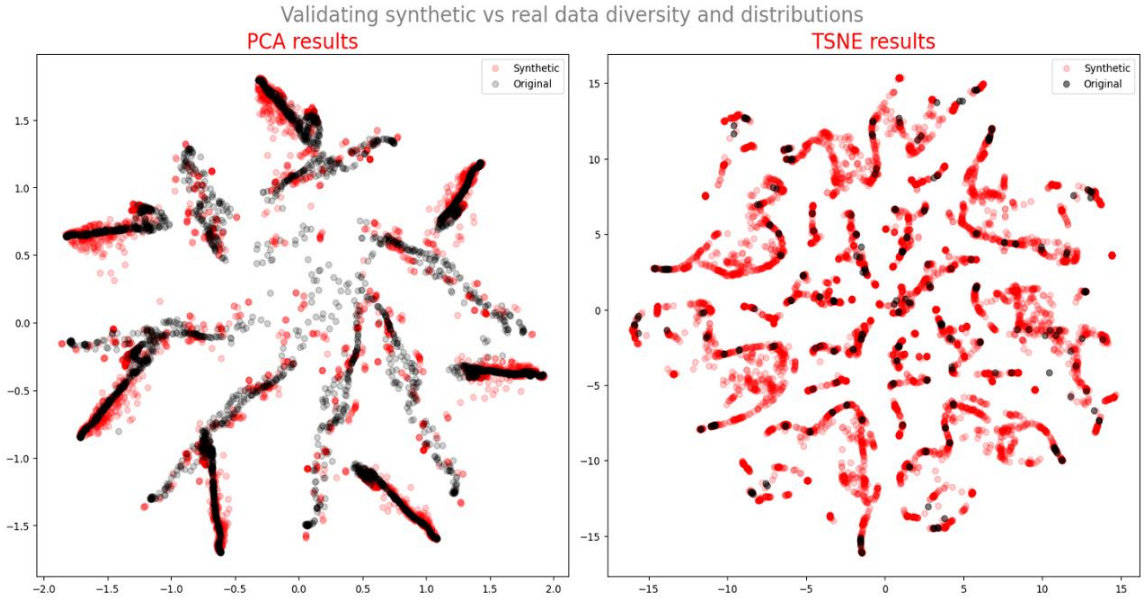


Figure 80: Synthetic vs real data validations

By observing the data points of the real and synthetic data, we can conclude that there is a notable overlap between them, suggesting that the generated data captures faithfully the structural characteristics of the real data.

Additionally, to further solidify the quality of the generative model, the similarity between the distributions of the real and synthetic datasets was assessed, utilizing the Kullback-Leibler (KL) divergence metric. This statistical measure quantifies the difference between two probability distributions, providing insights into the degree of similarity or dissimilarity between them. By

employing the KL divergence metric, we were able to assess the extent to which the synthetic dataset accurately reflects the distributional characteristics of the real dataset, thus facilitating a quantitative evaluation of the fidelity of the synthetic data generation process. Given that the scale of the data, which are normalized in the range of [0,1], a KL divergence of 0.0438 suggests that there is relatively little information lost when approximating one distribution with the other, indicating a certain degree of similarity between the two distributions.

Finally, the last evaluation methodologies employed, to assess the utility of the synthetic sensor data, are Train Real Test Real (TRTR) and Train Synthetic Test Real (TSTR). In TRTR, the model is trained on real data and then tested on real data, providing a benchmark for performance against authentic information. Conversely, TSTR involves training the model on synthetic data and then evaluating its performance on real data, allowing for an assessment of how well the synthetic data mimics real-world patterns. The utilized model was a regressor consisting of a single-layer GRU followed by a Dense output layer with a sigmoid activation function. After training two separate models, one on the real and the other on the synthetic dataset, the testing was conducted using real data. Based on the following results, it is clear that the synthetic data were capable of sufficiently replicating the original data patterns and achieving a comparable training performance.

Table 17: Comparative Analysis of performance

	R2 score	MAE	MRLE
Real	0.709582	0.018577	0.001434
Synthetic	0.622268	0.033815	0.002878

5.4.4 Deployment Results

The integration of the synthetic data generation solution into the TALON Cloud-edge ecosystem, will be facilitated by using the Docker containerization technology. The containers will encapsulate the entire synthetic data generation system, including the machine learning models, communication interfaces, and documentation, into portable and reproducible units. This approach will ensure the consistency and compatibility across diverse computing environments, enabling seamless deployment and integration within the TALON Cloud-edge ecosystem.

The containerization process will be streamlined using a Docker Compose YAML file. The YAML file will define the configuration for each component of the synthetic data generation system, simplifying the deployment and orchestration of the individual containers comprising the overall system.

5.5 Authentication and Authorisation

5.5.1 Internal Architecture and Key Technologies

Under the scope of TALON, we have designed and deployed the early prototype of a secure approach to authentication and authorisation of users and E2C services via IAM mechanisms. This approach eliminates the computational overhead between user access, applications and processes. To meet this objective, we have modified Keycloak1 which is an open-source software product allowing for single sign-on (SSO) with Identity and Access Management. As depicted in Figure XX, the authentication and authorisation services of TALON both support different user roles and usage of services by modifying the Keycloak framework by defining all the policies for the users, the environment, and the underlying resources.

The IAM mechanisms enable to secure the TALON web/REST APIs and also help to keep track of user roles, records and activity. The IAM mechanisms of TALON have several functional “points” that serve as the service node for retrieval and management of the policy, along with some logical

components for handling the context or workflow of policy and its assessment (valid or not valid access token to proceed or not). Figure 81 shows the main functional points, which are:

- Policy Information Point (PIP)
- Policy Decision Point (PDP)
- Policy Enforcement Point (PEP)

When these services are in an environment together, they function in harmony to provide access control decisions to users and services.

The PDP makes the access decisions by evaluating the applicable Digital Policies (DP) that are stored onto the Authoritative User Roles Store. The DP contains access control rules. User and (data and services) usage roles are the fundamental elements of DP, i.e. the building blocks of DP rules, which are then enforced by the PEP. The PDP implements the decision procedures according to the digital policy related to specific users and services.

Moreover, Keycloak offers a programming interface to support a set of endpoints for secure resource management, permission management and API for Policies (i.e., Policy API). With the resource management endpoint, resource servers can manage their resources remotely and enable policy enforcers to query the server for the resources that need protection. With the permission management endpoint, resource servers can create permission tokens. Keycloak also provides endpoints to manage the state of permissions and query permissions. Lastly, the Policy API allows resource servers to manage permissions for their users. In addition to the resource and permission APIs, Keycloak provides a Policy API from which permissions can be set to resources by resource servers on behalf of their users. An important requirement for this API is that only resource servers are allowed to access its endpoints using a special OAuth231 access token called PAT (i.e., protection API token).

The PEP is the logical entity that enforces policies for authorization and policy decisions in response to a request from a user or service / API wanting to access a protected environment; it executes the appropriate access decisions, as determined by the PDP, which shall either allow or deny user access to the requested protected object.

The PIP serves as the retrieval source of attributes, or the data required for policy evaluation to provide the information needed by the PDP to make the decisions. The Administration Point provides a user interface for creating, testing, and debugging rules and policies, and storing these policies in the appropriate repository.

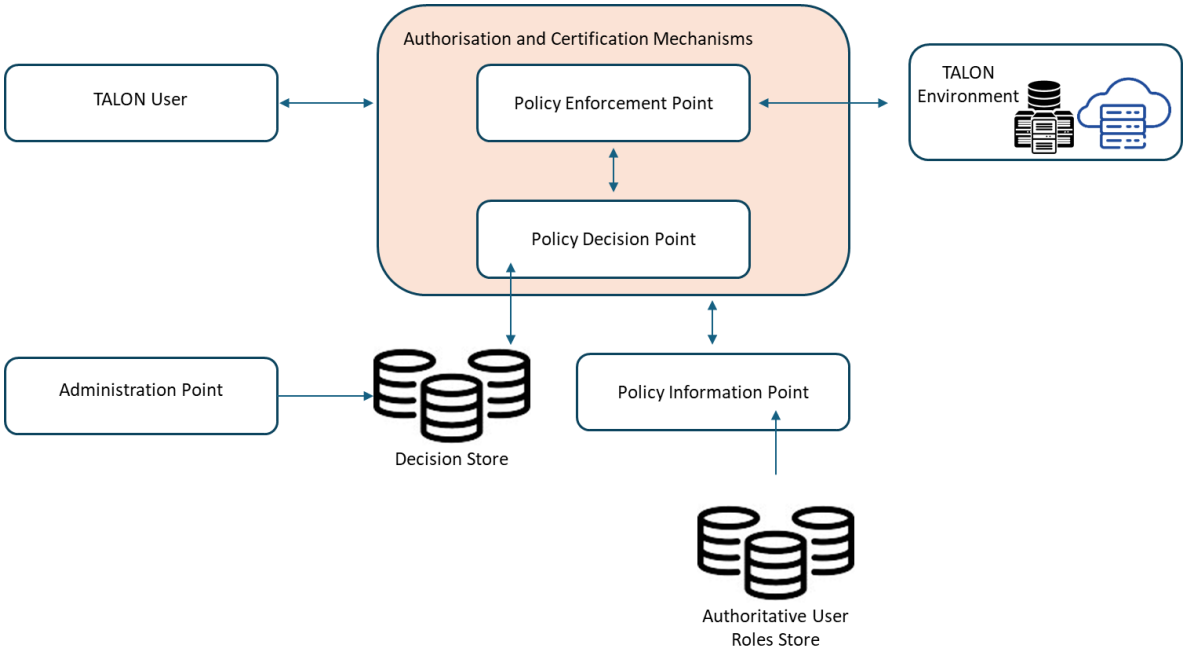


Figure 81: Authentication and Authorisation Mechanism

5.5.2 Communication Interfaces

The Authentication and Authorisation mechanism exposes a client-server API to handle POST and GET methods. The POST method triggers the authentication mechanism and enables end user successfully login to the platform, while the GET methods both handle end user requests and API calls via secure tokens, as follows:

- http://localhost:8180/POST/user_credentials
- http://localhost:8180/GET/user_credentials
- http://localhost:8180/GET/token

5.5.3 Scientific and Technical Results

Since the authentication and authorisation mechanisms are not related to a research task, we have not conducted an experimental evaluation. We only performed a benchmarking and an integration test ensuring that the TALON APIs interact securely. Also, we have tested and validated different user roles with different access rights to the services provided by the MINDS, ENG, and UBI technical partners.
The first prototype of the authentication and authorisation mechanisms for the TALON project has been deployed, integrated and evaluated with the components provided by MINDS, ENG, and UBI in the frame of WP3 and WP4 technical tasks. The TALON platform has a unified dashboard that provides a graphical user interface to the backend TALON services. Both the TALON dashboard and the backend APIs have been secured via authentication and authorisation mechanisms to protect all the interaction points within the TALON infrastructure. The landing page prompting the end user to provide her credentials is depicted in the below figure.

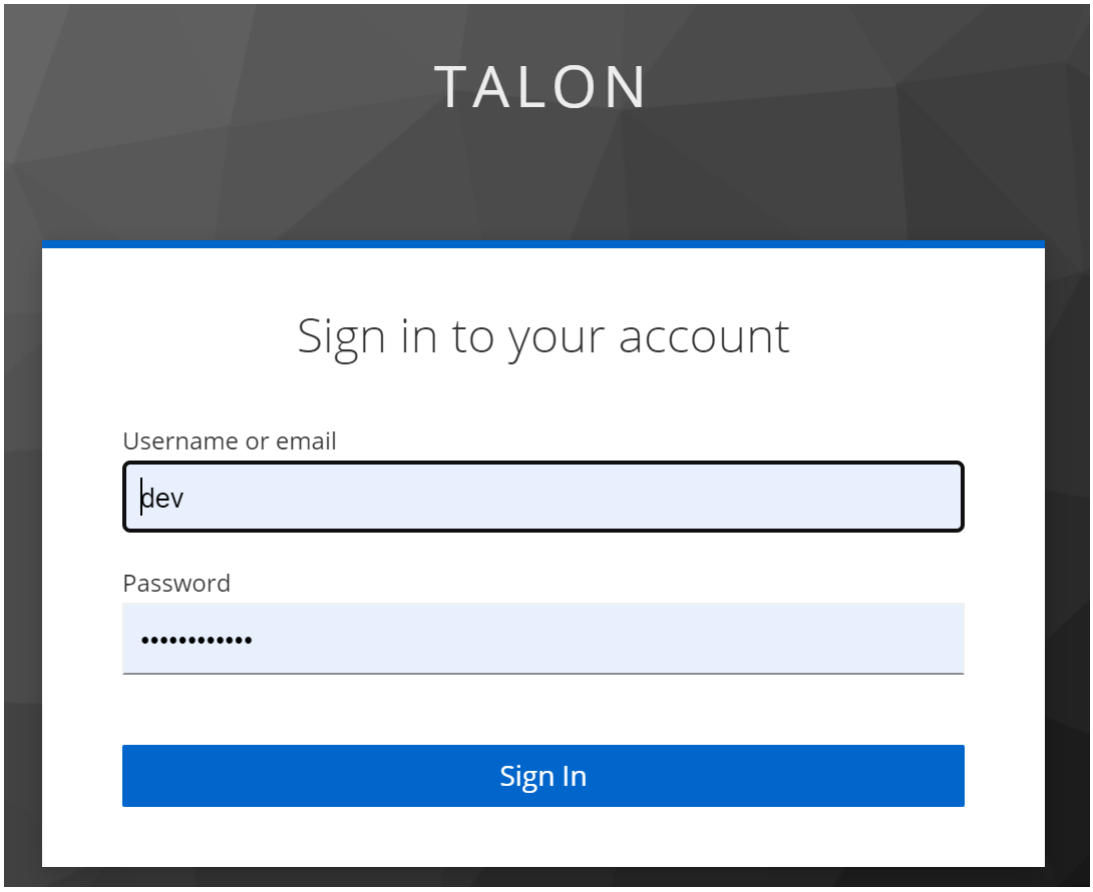


Figure 82: TALON Central Authentication and Authorisation Service

5.5.4 Deployment results

The early deployment of the Authentication and Authorisation mechanism has been completed in the support of a single user role (until M17) acting as the administrator of the TALON platform. We have also created a landing page where the end user can submit her credentials. This mechanism, the definition of the user roles and the underlying Roles Store system have been deployed and integrated with all the TALON APIs. The outcome for this is that the end user can transparently navigate within the TALON Platform and the unified dashboard via a single sign-on functionality.

6 Conclusion & Future Outlook

In the preceding sections, we elucidated a comprehensive methodology devised for the development and deployment of AI algorithms characterised by computational and energy efficiency, alongside advanced data processing techniques. These cutting-edge technologies are specifically designed to prioritise the conservation and augmentation of essential properties, including but not limited to reusability, explainability, and trustworthiness.

This deliverable initiates with a thorough contextualisation, focusing on characterizing the properties of reusability, explainability, and trustworthiness. Each property is allocated a dedicated section aiming to provide an overview of methodologies and detailed insights into implementations chosen for the TALON project. These implementations fall into two categories: model-based and data-based approaches. In model-based strategies, state-of-the-art algorithms like Transfer Learning, Few-shot Learning, Federated Learning, and Hybrid Learning are utilized, tailored to meet the specific requirements of each property and enhance reusability. In data-based approaches, diverse preprocessing and feature engineering methods are employed to align input data with AI algorithm specificities, potentially uncovering valuable knowledge and concurrently boosting explainability.

Ultimately, system-level approaches play a pivotal role in improving the holistic robustness of the integrated combination of the aforementioned technologies. Notably, entities focused on authentication and authorisation, coupled with Digital Twin technologies, are instrumental in preserving optimal levels of data and model trustworthiness. This becomes particularly crucial in addressing the safety-critical and continuously evolving nature of contemporary application domains heavily reliant on AI advancements.

In conclusion, it is noteworthy that all modules within the project have commenced their respective operational phases. While certain modules have progressed to the deployment stage, others remain in active phases of research and development. It is anticipated that all modules will be finalised and integrated into the comprehensive framework outlined in the forthcoming D4.3 report. This report will serve as a comprehensive repository, encapsulating the full spectrum of TALON's technical modules in their ultimate and completed form.

7 References

[1] *R. Subha, A. Haldorai and A. Ramu, "Artificial Intelligence Model for Software Reusability Prediction System," vol. 35, no. 3, pp. 2639-2654, 2022.*

[2] *H. Wang and K. Webster, "AI for Data Discovery and Reuse," 2019. [Online].*

[3] *I. Ahmed, G. Jeon and F. Piccialli, "From Artificial Intelligence to Explainable Artificial Intelligence in Industry 4.0: A Survey on What, How, and Where,"* IEEE Transactions on Industrial Informatics, *vol. 18, no. 8, pp. 5031-5042, 2022.*

[4] *K. S, Y. S and R. V. P, "An evaluation on big data generalization using k-Anonymity algorithm on cloud," in* IEEE 9th International Conference on Intelligent Systems and Control (ISCO)*, Coimbatore, India, 2015.*

[5] *Y. Xu, T. Ma, M. Tang and W. Tian, "A Survey of Privacy Preserving Data Publishing using Generalization and Suppression,"* Applied Mathematics & Information Sciences, *vol. 8, pp. 1103-1116, 2014.*

[6] *S. Ye, F. Wu, P. R. and H. Chen, "Noise Injection for Search Privacy Protection," in* International Conference on Computational Science and Engineering*, Vancouver, BC, Canada, 2009.*

[7] *H. B. McMahan, E. Moore, D. Ramage, S. Hampson and A. y. A. Blaise , "Federated Learning of Deep Networks using Model Averaging,"* CoRR, *vol. 602.05629, 2016.*

[8] *I. Ahmed , K. Tahir , M. Tahir , M. H. Habaebi , S. L. Lau and A. Ahad , "Machine Learning for Authentication and Authorization in IoT: Taxonomy, Challenges and Future Research Direction,"* Security and Privacy in the Internet of Things (IoT), *vol. 21, no. 15, 2021.*

[9] *S. Sreelakshmi Vattaparambil, S. Olov and B. Ulf, "Survey on Delegated and Self-Contained Authorization Techniques in CPS and IoT,"* IEEE Access, *vol. 9, pp. 98169-98184, 2021.*

[10] *G. Marcus, "Deep Learning: A Critical Appraisal,"* CoRR, *2018.*

[11] *G. Marcus, "Deep Learning: A Critical Appraisal,"* CoRR, *vol. 1801.00631, 2018.*

[12] *Y. Wang, Q. Yao, J. T. Kwok and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning,"* ACM Computing Surveys, *vol. 53, no. 3, 2020.*

[13] *F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong and H. Qing, "A Comprehensive Survey on Transfer Learning,"* Proceedings of the IEEE, *vol. 109, no. 1, pp. 43-76, 2021.*

[14] *T. Hospedales, A. Antoniou, P. Micaelli and A. Storkey, "Meta-Learning in Neural Networks: A Survey,"* IEEE Transactions on Pattern Analysis and Machine Intelligence, *vol. 44, no. 9, pp. 5149-5169, 2022.*

[15] *G. Koch, R. Zemel and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in* ICML deep learning workshop*, 2015.*

[16] *J. Snell, K. Swersky and R. Zemel, "Prototypical Networks for Few-shot Learning," in* Advances in Neural Information Processing Systems*, 2017.*

[17] *F. Sung, Y. Yang, L. Zhang, T. Xiang, T. H. Philip and T. M. Hospedales, "Learning to Compare: Relation Network for Few-Shot Learning,"* CoRR, *vol. 1711.06025, pp. 1199-1208, 2017.*

[18] *O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu and D. Wierstra, "Matching Networks for One Shot Learning," in* Advances in Neural Information Processing Systems (NIPS)*, 2016.*

[19] *Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever and A. Pieter, "RL2: Fast Reinforcement Learning via Slow Reinforcement Learning,"* CoRR, *2016.*

[20] *N. Mishra, M. Rohaninejad, X. Chen and P. Abbeel, "Meta-Learning with Temporal Convolutions,"* CoRR, *vol. 1707.03141, 2017.*

[21] *D. Chijiwa, S. Yamaguchi, A. Kumagai and Y. Ida, "Meta-ticket: Finding optimal subnetworks for few-shot learning within randomly initialized neural networks," in* Advances in Neural Information Processing Systems, *2022.*

[22] *T. Munkhdalai and H. Yu, "Meta Networks," in* roceedings of the 34th International Conference on Machine Learning, *2017.*

[23] *D. Santoro, S. Bartunov, M. Botvinick, D. Wierstra and T. Lillicrap, "Meta-Learning with Memory-Augmented Neural Networks," in* Proceedings of The 33rd International Conference on Machine Learning, *New York, USA, 2016.*

[24] *S. Ravi and H. Larochelle, "Optimization as a Model for Few-Shot Learning," in* International Conference on Learning Representations (ICRL), *2017.*

[25] *C. Finn, P. Abbeel and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in* Proceedings of the 34th International Conference on Machine Learning, *2017.*

[26] *Z. Li, F. Zhou, F. Chen and H. Li, "Meta-SGD: Learning to Learn Quickly for Few Shot Learning,"* CoRR, *vol. 1707.09835, 2017.*

[27] *S. Kang, D. Hwang, M. Eo, T. Kim and W. Rhee, "Meta-Learning With a Geometry-Adaptive Preconditioner," in* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), *2023.*

[28] *E. Grant , C. Finn , S. Levine , T. Darrell and T. L. Griffiths, "Recasting Gradient-Based Meta-Learning as Hierarchical Bayes,"* CoRR, *vol. 1801.08930, 2018.*

[29] *E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan and Q. V. Le, "AutoAugment: Learning Augmentation Strategies From Data," in* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), *2019.*

[30] *C.-M. Feng, K. Yu, Y. Liu, S. Khan and W. Zuo, "Diverse Data Augmentation with Diffusions for Effective Test-time Prompt Tuning," in* Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), *2023.*

[31] *W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang and J.-B. Huang, "A Closer Look at Few-shot Classification,"* CoRR, *vol. 1904.04232, 2019.*

[32] *G. S. Dhillon, P. Chaudhari, A. Ravichandran and S. Soatto, "A Baseline for Few-Shot Image Classification,"* CoRR, *vol. 909.02729, 2019.*

[33] *M. N. Rizve, S. Khan, F. S. Khan and M. Shah, "Exploring Complementary Strengths of Invariant and Equivariant Representations for Few-Shot Learning," in* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), *2021.*

[34] *B. Demirel, O. B. Baran and . R. G. Cinbis, "Meta-Tuning Loss Functions and Data Augmentation for Few-Shot Object Detection," in* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), *2023.*

[35] *M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. Rezende and S. M. A. Eslami, "Conditional Neural Processes," in* Proceedings of the 35th International Conference on Machine Learning, *2018.*

[36] *H. Kim, A. Mnih, J. Schwarz, M. Garnelo, A. Eslami, D. Rosenbaum, O. Vinyals and Y. W. Teh, "Attentive Neural Processes,"* CoRR, *vol. 1901.05761, 2019.*

[37] J. Gordon, W. P. Bruinsma, A. Y. K. Foong, J. Requeima, Y. Dubois and R. E. Turner, "Convolutional Conditional Neural Processes," in International Conference on Learning Representations (ICLR), 2020.

[38] H. Nguyen and A. Grover, "ransformer Neural Processes: Uncertainty-Aware Meta Learning Via Sequence Modeling," in Proceedings of the 39th International Conference on Machine, Baltimore, Maryland, USA, 2022.

[39] K. Hsu , S. Levine and C. Finn, "Unsupervised Learning via Meta-Learning," CoRR, vol. 1810.02334, 2018.

[40] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle and . R. S. Zemel, "Meta-Learning for Semi-Supervised Few-Shot Classification," CoRR, vol. 1803.00676, 2018.

[41] L. Dong Bok, L. Seanie, J. Ko, K. Kawaguchi, J. Lee and S. J. Hwang, "Self-Supervised Dataset Distillation for Transfer Learning," preprint, vol. 2310.06511, 2023.

[42] F. Ye, B. Lin, Z. Yue, P. Guo, Q. Xiao and Y. Zhang, "Multi-Objective Meta Learning," in Advances in Neural Information Processing Systems, 2021.

[43] M. Abdollahzadeh, T. Malekzadeh and . N.-M. (. Cheung, "Revisit Multimodal Meta-Learning through the Lens of Multi-Task Learning," in Advances in Neural Information Processing Systems, 2021.

[44] G. Zhang, Z. Luo, K. Cui, S. Lu and E. P. Xing, "Meta-DETR: Image-level few-shot detection with inter-class correlation exploitation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 11, 2023.

[45] X. Garcia, Y. Bansal, C. Cherry, G. Foster, M. Krikun, M. Johnson and O. Firat , "The Unreasonable Effectiveness of Few-shot Learning for Machine Translation," in Proceedings of the 40th International Conference on Machine Learning, 2023.

[46] J. Wang, J. Zhang, H. Jiang, J. Zhang, L. Wang and C. Zhang, "Offline Meta Reinforcement Learning with In-Distribution Online Adaptation," preprint, vol. 2305.19529, 2023.

[47] X. He, C. Tan, B. Liu, L. Si, W. Yao, L. Zhao, D. Liu, Q. Zhangli, Q. Chang, K. Li and D. N. Metaxas, "Dealing With Heterogeneous 3D MR Knee Images: A Federated Few-Shot Learning Method With Dual Knowledge Distillation," preprint , vol. 2303.14357, 2023.

[48] J. Chen, J. Tang and W. Li, "Industrial Edge Intelligence: Federated-Meta Learning Framework for Few-Shot Fault Diagnosis," in IEEE Transactions on Network Science and Engineering, 2023.

[49] O. Aouedi, K. Piamrat and B. Parrein, "Performance evaluation of feature selection and tree-based algorithms for traffic classification," in 2021 IEEE International Conference on Communications Workshops (ICC Workshops), Montreal, QC, Canada, 2021.

[50] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in 31st Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 2017.

[51] G. H. John, R. Kohavi and K. Pfleger, "Irrelevant Features and the Subset Selection Problem," in Machine Learning Proceedings 1994, San Francisco (CA), Morgan Kaufmann, 1994, pp. 121-129.

[52] J. Novakovic, "Using information gain attribute evaluation to classify sonar targets," in 17th Telecommunications forum TELFOR, Serbia, Belgrade, 2009.

[53] F. Salo, A. B. Nassif and A. Essex, "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection," Computer Networks, vol. 148, pp. 164-175, 2019.

[54] P. Nskh, N. M. Varma and R. R. Naik, "Principle component analysis based intrusion detection system using support vector machine," in 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), *Bangalore, India, 2016.*

[55] G. P. Zhang, "Neural networks for classification: a survey," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), *2000.*

[56] B. Veloso, R. P. Ribeiro, J. Gama and P. M. Pereira, "The MetroPT dataset for predictive maintenance," Scientific Data, *vol. 9, no. 1, p. 764, 2022.*

[57] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," IEEE Access, *vol. 6, pp. 52138-52160, 2018.*

[58] B. Goodman and S. Flaxman, "European Union Regulations on Algorithmic Decision-Making and a 'Right to Explanation'," AIMag, *vol. 38, no. 3, pp. 50-57, 2017.*

[59] M. B. Muhammad and M. Yeasin, "Eigen-CAM: Class Activation Map using Principal Components," in 2020 International Joint Conference on Neural Networks (IJCNN), *Glasgow, UK, 2020.*

[60] M. Baigang and F. Yi, "A review: development of named entity recognition (NER) technology for aeronautical information intelligence," vol. 56, no. 2, pp. 1515-1542, 2023.

[61] R. Catelli, F. Gargiulo, V. Casola, G. D. Pietro, H. Fujita and M. Esposito, "Crosslingual named entity recognition for clinical de-identification applied to a COVID-19 Italian data set," vol. 97, p. 106779, 2020.

[62] S. Sousa and R. Kern, "How to keep text private? A systematic review of deep learning methods for privacy-preserving natural language processing," Artificial Intelligence Review, *vol. 56, no. 2, pp. 1427-1492, 2023.*

[63] R. Catelli, V. Casola, G. D. Pietro, H. Fujita and M. Esposito, "Combining contextualized word representation and sub-document level analysis through Bi-LSTM+CRF architecture for clinical de-identification," vol. 213, p. 106649, 2021.

[64] D. Uliyan, A. S. Aljaloud, A. Alkhalil, H. S. Al Amer and M. A. E. A. Mohamed, "Deep Learning Model to Predict Students Retention Using BLSTM and CRF," IEEE Access, *vol. 9, pp. 135550-135558, 2021.*

[65] E. F. Tjong Kim Sang and S. Buchholz, "Introduction to the CoNLL-2000 Shared Task: Chunking," CoRR, *vol. 0009008, 2000.*

[66] N. Lavanya, M. Vedavani, H. Sabbani and R. Perugu, "Analysis of deep learning algorithms used for text," Journal For Innovative Development in Pharmaceutical and Technical Science (JIDPTS), *vol. 6, no. 6, 2023.*

[67] D. Biesner, R. Ramamurthy, R. Stenzel, M. Lübbering, L. Hillebrand, A. Ladi, M. Pielka, R. Loitz, C. Bauckhage and R. Sifa, "Anonymization of German financial documents using neural network-based language models with contextual word representations," International Journal of Data Science and Analytics, *vol. 13, no. 2, pp. 151-161, 2022.*

[68] I. Siniosoglou, P. Sarigiannidis, Y. Spyridis, A. Khadka, G. Efstathopoulos and T. Lagkas, "Synthetic Traffic Signs Dataset for Traffic Sign Detection & Recognition In Distributed Smart Systems," 17th International Conference on Distributed Computing in Sensor Systems (DCOSS), *pp. 302-308, 2021.*

[69] V. Ayala-Rivera, P. McDonagh, T. Cerqueus and L. Murphy, "A Systematic Comparison and Evaluation of k-Anonymization Algorithms for Practitioners," Transactions on Data Privacy, *vol. 7, no. 3, pp. 337-370, 2014.*

[70] A. Singh and J. Zhu, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, *2017.*

[71] T. Li , A. K. Sahu , A. Talwalkar and V. Smith , "Federated Learning: Challenges, Methods, and Future Directions," IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50-60, 2020.

[72] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Kone, S. Kumar and H. B. McMahan, "Adaptive Federated Optimization," CoRR, vol. 2003.00295, 2020.

[73] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet and N. D. Lane, "Flower: {A} Friendly Federated Learning Research Framework," CoRR, vol. 2007.14390, 2020.

[74] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[75] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," International Journal of Computer Vision, vol. 88, no. 2, pp. 303-338, 2010.

[76] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[77] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," International Journal of Forecasting, vol. 22, no. 4, 2006.

[78] A. Holzinger, C. Biemann, C. S. Pattichis and D. B. Kell, "What do we need to build explainable AI systems for the medical domain?," CoRR, vol. 1712.09923, 2019.

[79] M. T. Ribeiro, S. Singh and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," in Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016.

[80] W. Samek, T. Wiegand and K. R. Muller, "Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models," 2017.

Funded by
the European Union

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101070181