D3.1 DETECTION MECHANISMS TO IDENTIFY DATA BIASES AND EXPLORATORY STUDIES ABOUT DIFFERENT DATA QUALITY TRADE-OFFS FOR AI-BASED SYSTEMS

Revision: v.1.0

Work package	WP3
Task(s)	Task 3.1 and Task 3.2
Due date	February 28, 2023
Submission date	February 28, 2023
Deliverable lead	University of Tartu (UT)
Version	1.0
Authors	Abdul-Rasheed Ottun, Mehrdad Asadi and Huber Flores (**University of Tartu**) Nikolay Tcholtchev, Michell Boerger (**Fraunhofer FOKUS**) Souniel Park, David Solans Noguero, Nicholas Kourtellis (**Telefonica**) Drasko Draskovic , Illija Tanaskovic, Sasa Klopanovic (**Mainflux**) Aaron Ding (**TUDelft**)
Reviewers	Bartlomiej Siniarski (**UCD Dublin**) Vinh Hoa La (**Montimage**)

Grant Agreement No.: 101021808
Call: H2020-SU-DS-2020
Topic: SU-DS02-2020
Type of Action: RIA

Abstract	The quality of data used for training AI models is of critical importance to guarantee their robust performance. Besides the large body of well-established methods for data fusion, preparation and augmentation, additional characteristics of the data can contribute towards making AI model trustworthy. Data fairness and transparency are key aspects to investigate when analyzing the features extracted from the data that are used by the model to support their autonomous decision making. Advancements in model training using distributed devices introduces a higher level of complexity when dissecting AI model logic. In this document, we investigate fairness and transparency methods to detect possible biases that are introduced as models are re-trained over time. Through rigorous benchmarks and studies that consider different applications scenarios, we explore the performance of these methods and data trade-offs that impact the overall model inference process. Our results suggest that it is possible to detect induced and non-induced changes over data used for training the models. This requires however augmenting current standard machine learning pipelines with components that analyze data quality throughout the pipeline from input data to model deployment.
Keywords	AI-based systems, Secure AI, Explainable AI, Accountable AI, Requirements analysis, Trustworthy AI for Cybersecurity, Accountability, Privacy Preservation, Resilience Engineering

Document Revision History

Version	Date	Description of change	List of contributor(s)
v0.1	05/07/2022	Initial Table of Content	Huber Flores
v0.2	10/02/2023	Integrated contributions from all involved partners	All listed contributors
v0.3	14/02/2023	Presentation of document to internal reviewer	Bartlomiej Siniarski
v0.4	14/02/2023	Presentation of document to internal reviewer	Vinh Hoa La
v0.5	25/02/2023	Addressed comments from internal reviewers	Abdul-Rasheed Ottun, Mehrdad Asadi, Huber Flores.
v1.0	26/02/2023	Prepared document for final submission	Abdul-Rasheed Ottun, Mehrdad Asadi, Huber Flores.

DISCLAIMER

COPYRIGHT NOTICE

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Project funded by the European Commission in the H2020 Programme		
Nature of the deliverable:	**R***	
Dissemination Level		
PU	Public, fully open, e.g., web	✓
CL	Classified, information as referred to in Commission Decision 2001/844/EC	
CO	Confidential to SPATIAL project and Commission Services	

* R: Document, report (excluding the periodic and final reports)

DEM: Demonstrator, pilot, prototype, plan designs

DEC: Websites, patents filing, press & media actions, videos, etc.

OTHER: Software, technical diagram, etc.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

EXECUTIVE SUMMARY

AI-based systems are transforming the Internet and the services that are consumed through it. Indeed, 97.2% of organizations aim to promote their services by utilizing big data and AI-based algorithms [1]. To this day, it is possible to encounter applications implementing AI-based functionality ranging from sophisticated advertisement recommendations to generative AI conversations using Chatbots, e.g., ChatGPT. While data is still increasing exponentially over the years, a key limitation that is preventing the adoption of AI functionality at scale is the quality of data used for training the AI models. Data quality is a critical aspect that not just defines the performance of AI models for decision making, but it also establishes whether AI models can be trustworthy or not. Making AI robust and trustworthy requires a clear understanding of the impact of data for AI model training.

Existing methods to analyze the quality of data are tailored for classical centralized architectures [2]. Latest advancements in distributed machine learning however are providing the basis for training robust AI models. Indeed, federated learning architectures can easily aggregate contributions from different devices to train a global model [3]. Similar to this, collaborative processing can distribute the execution load of an AI model over multiple end-devices to support model inference at the edge of the network [3]. While the rise of distributed machine learning methods is facilitating the construction and deployment of AI models, there are several trade-offs that require to be analyzed and characterized to make these models also trustworthy.

In this document, we investigate several data trade-offs to achieve trustworthy AI as models are built in a distributed environment. To do this, we present different benchmarks and the analysis of four (4) different applications scenarios. Each application scenario is designed to evaluate the impact of a specific data quality aspect in model training. Our experiments consider different types of data, e.g., image and sensor data; and rely on biases caused by induced and non-induced changes to modify the quality of data to quantify its effect in model performance. To validate further the insights obtained in our experiments, models are also evaluated using traditional centralized approaches. In addition to this, our work also reflects on current state-of-the-art methods (including XAI for model analysis) and discusses the implications of our results towards achieving trustworthy AI.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

TABLE OF CONTENTS

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

LIST OF FIGURES

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

LIST OF TABLES

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

ABBREVIATIONS

Abbreviation	Definition
ADASYN	Adaptive Synthetic
ADB	Android Device Bridge
ADL	Activities of Daily Living
AGV	Autonomous Ground vehicles
AI	Artificial Intelligence
ANOVA	Analysis of Variance
AUC	Area under the ROC Curve
CDD	Conditional Demographic Disparity
CI	Class Imbalance
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSV	Comma-Separated Values
DCA	Difference in Conditional Acceptance
DCPI	Data Center Performing Index
DI	Disparate Impact
DCR	Difference in Conditional Rejection
DD	Demographic Disparity metric
DNN	Deep Neural networks
DPL	Difference in Proportion Labels
DPPL	Difference in Positive Proportions of True Labels
DT	Decision Tree
ETL	Extract, Transform and Load
FL	Federated learning
FLAAS	Federated Learning as a Service
FN	False Negative
FP	False Positive
FPR	False Positive rate
FT	Counterfactual Flip Test
GPU	Graphics processing unit
HAR	Human activity recognition
i.i.d	Independent and identically distributed
JM	Joint Models
JS	Joint samples
KL	The Kullback-Leibler divergence
KNN	K-Nearest Neighbors
LiDAR	Light Detection and Ranging
LIME	Local the Interpretable Model-agnostic Explanation
LOOCV	Leave-one-out cross-validation
LR	Logistic Regression
LRP	Layer-wise Relevance Propagation
MAE	Mean Absolute Error
MAR	Missing at Random
MCAR	Missing Completely at Random
ML	Machine Learning
MLP	Multilayer Perceptron
MNAR	Missing Not at Random
MSE	Mean Square Error

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

OS	Operating System
RAM	Random Access Memory
ReLU	Rectified Linear Unit
RF	Random Forest
RGB	red, green and blue
RMSE	Root Mean Square Error
RND	Research and development
ROC	receiver operating characteristic
SA	Single-App
SD	Standard Deviation
SGD	Stochastic gradient descent
SHAP	Shapley Addictive Explanations
SMOTE	Synthetic Sampling and Data Generation
SOTA	State-of-the-art
SPATIAL	Security and Privacy Accountable Technology Innovations, Algorithms, and machine Learning
SQL	Structured Query Language
SVM	Support Vector Machine
TE	Treatment Equality
TFF	TensorFlow Federated
TN	True Negative
TP	True Positive
TPR	True positive rate
UnimiB SHAR	Dataset for Human Activity Recognition Using Acceleration Data from Smartphones
XAI	Explainable Artificial Intelligence

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

1. BACKGROUND

Data quality has evolved into a key process whose goal is to evaluate the data-driven functionality of a wide range of computing systems [2]. The assessment of data quality is commonly performed as an iterative and continuous process over time. The main idea is that as more data is collected, it is transformed into knowledge that reinforces and improves the functionality of systems and applications. Data quality has been studied in multiple computing domains, such as multimedia [4], Internet of Things [5], databases [6, 7], information systems and cyber-physical systems [8] as representative examples. Thus, a wide range of methods and approaches have also been developed to improve the quality of data, e.g., interpolation and data augmentation. Multiple definitions that support the idea of data quality have been established, but each definition is somehow dependent on the application domain. For instance, in the context of transactional systems [9], data quality is defined as the *core dimensions: accuracy, completeness, consistency, currency, security, and reliability"* [10]. Likewise, in pervasive computing systems, data quality also is defined as the ability of sensor hardware to collect high resolution samples at a suitable frequency rate, e.g., accelerometer [11]. Despite the application domain, AI models can exploit this data to automate the decision making functionality of computing systems [12]. Typically, models learn over time as more data is collected. Modern system architectures and applications are designed to collect and process large amounts of big data, which then is used to train AI models.

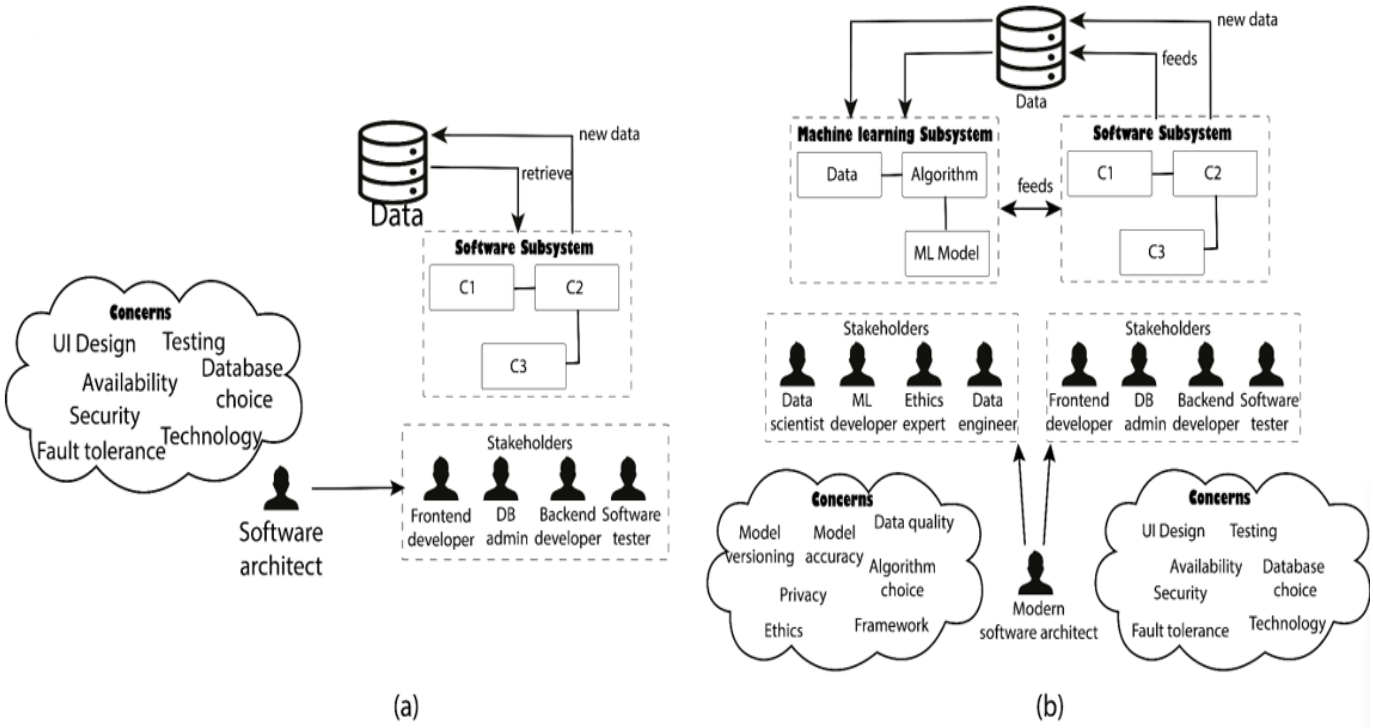


Figure 1: System architectures [130], a) Classical client-server architecture; b) Modern architecture with AI-based functionality

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

To illustrate this, let us consider Figure 1 (a), which shows a classical system architecture (client/server). In this architecture, stored data in a server is just fetched by clients, such that it can be presented to users. For instance, a catalog of books can be explored by users. Unlike these, in an AI-based architecture (Figure 1 (b)), interactions of users with the system are stored, such that that information is used later to improve further the experience of users. For instance, a catalog that recommends books to users based on their input keywords. AI models can also be built with external data sources to provide extra functionalities to users. For instance, the AI model to recommend books can be built with data from other bookstores to improve the robustness and quality of recommendations. In both architectures, we can observe that there is a need for new stakeholders managing different data aspects, e.g., data scientists and privacy experts, among others. We can also observe that the complexity and expertise required to build an AI-based architecture is higher when compared with its classical counterpart. Data is a key component that requires a wide range of methods to analyze its quality from different perspectives.

Feeding AI models with data is not a simple task. Data requires heavy analysis, manipulations and transformations before it can be used as model inputs. Quantifying and characterizing the quality of data is fundamental to produce models with high prediction and estimation capabilities. This also is important to link model predictions with actual data inputs used to train the models, and for helping in the dissecting of their internal logic. Accomplishing a robust assessment of the influence of data quality on AI models is difficult. Indeed, the quality of data plays a role in the fair, trustworthy and transparent behavior of models that is just extracted by analyzing characteristics of data in depth, such as distribution, correlations and biases.

In the following section, we reflect on the evolution of data quality. We highlight key challenges and issues in using big data for AI models; and review different techniques that can be used to minimize unbalanced behavior in the inference process of models. In addition to this, through rigorous experiments that consider several applications, we demonstrate how induced and non-induced changes in data influence the performance of AI models. We also show that it is possible to detect those changes using a variety of different methods. Lastly, we analyze how data quality is important not just for improving model performance, but also for dissecting the logic of models using XAI methods.

1.1. A BRIEF HISTORY OF DATA QUALITY

Data quality research emerged by focusing mostly on operational transactional data [6]. Relational (SQL) databases rely on methods to enforce integrity, consistency and deduplication of information [13]. Data of transactions are subject to ACID principles (Atomicity, Consistency, Isolation and Durability) to ensure safety as data is modified. As data evolved to support more strategic decision making of organizations, data warehouses were introduced. These warehouses comprehend data from multiple databases sources. Data warehouses are built through a set of tools that Extract, Transform and Load (ETL), such that it is possible to combine multiple individual databases into one. Databases and data warehouses pose multiple challenges to achieve data quality. Multiple data management processes are designed to overcome them. For instance, data acquisition, curation, cleaning, transformation, linking, integration and validation are some of them.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

As the adoption of transaction systems increased, tools and methods to easily handle and analyze data quality have been developed. However, as data incrementally becomes larger, the data management problem has exacerbated even further data quality concerns. Indeed, the management of large data requires other types of solutions to fulfill service demands, in terms of storage and data retrieval. Non-relational (non-SQL) emerged to address these scalability and performance problems. Unfortunately, data has become ubiquitous and is emerging from a range of different sources, e.g. the Web, social networks, IoT devices and autonomous systems. Thus, data provenance (veracity) has also become a quality aspect to consider when using the data. Data with a high degree of confidence can power reliable decision making, while data with a low degree of confidence should be avoided. Data provenance is thus of particular importance when certifying the usage of applications in situations where critical concerns can arise from using them.

Data-driven approaches based on real-time analytics are started to be adopted by organizations to support advanced decision making throughout all their processes. Machine and deep learning (simply AI algorithms) are the foundation to achieve this. Different types of prescriptive, predictive and diagnoses analysis can be powered by AI algorithms [14]. While the poor quality of data also hampers analytics, AI also takes into consideration additional data concerns for building robust models. AI considerations for data can be categorized into three categories: 1) Model representation, 2) Model performance metrics and 3) Model optimization. Model representation deals with the most suitable algorithm that adjusts to the data, e.g., linear. Likewise, model performance metrics quantify how good the model is given a specific criterion, e.g., recall, accuracy, loss, among others. As different models can be built based on the same data, model optimization deals with how to search for the best model given the space of possibilities. As data can have different impact in each consideration, building a robust AI model becomes a complex task.

1.2. DATA QUALITY ISSUES IN BIG DATA

Big data introduces data management challenges that have been related to the ***Vs*** paradigm (Volume, Velocity and Variety) [15]. Volume relates to the amount of data, whereas Velocity refers to the speed at which the data is gathered. Likewise, Variety depicts the different representations that data can take. As big data has evolved, other Vs have been introduced, such as Veracity and Value. The former refers to the documentation of quality and uncertainty, whereas the latter focuses on the support for decision making provided to business organizations.

A data value chain is typically adopted as a metaphor that defines the information flow within information systems running in organizations. The chain defines the different processes that data undertakes before it generates value or insights in the form of decision support. Figure 2 shows the data value chain as described within [16]. Achieving data quality in big data manifests in the form of having a balance and harmonization of features that can be used to build models. As a result, additional “Vs” has been added to the paradigm over time, e.g., veracity [17]. Indeed, extracting data representations from big data that can be used as input for building AI models is important. Adding too many features can overfit the model, and not adding enough can provide low model performance. Extracting features from data requires applying methods to achieve a certain level of data quality to train the model. These methods include missing data

SPATIAL project is funded by the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 101021808.

imputation, outlier detection, deduplication removal, aggregation, integration and correlation of data features.

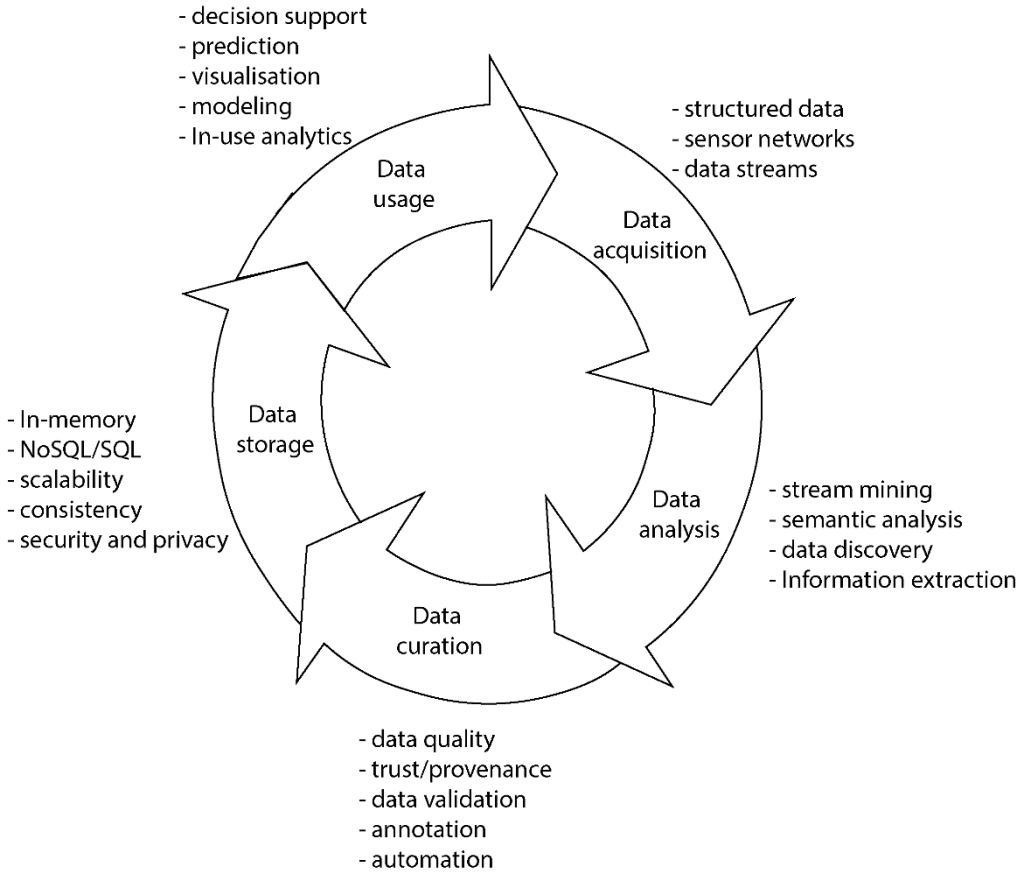


Figure 2: Data value chain

1.2.1 Dealing with missing data

The removal of outliers and unreliable data is also part of the data preparation process [18]. Data comprises observable and non-observable types. Thus, data cleaning can be a complex task to perform. Missing (non-observable) data is a major concern when preparing data for AI models. Missing data is classified into three categories, 1) Missing completely at Random (MCAR), 2) Missing at Random (MAR) and Missing Not at Random (MNAR) [19]. MCAR implies that there is no specific pattern to the missing data. This means that observable data is independent of non-observable. Similarly, MAR captures the situation where data is missing independently of the non-observed data. Likewise, MNAR implies that missing data is related to the non-observable data. In this case, the observed data can easily be characterized as biased data. Indeed, missing and incomplete data can lead to biased results because the dataset is not completed or ignored [20].

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Methods to deal with missing data have been extensively investigated. Deletion is the simplest method to overcome this problem. In this method, the missing values are usually discarded from the dataset before further analyzing it. Unfortunately, this approach may lead to bias in the analysis if the data is not randomly distributed. Two types of deletion called List-wise and pairwise can be applied for missing data. The former approach is considered a default choice in statistical software packages [21] and requires removing every data point that includes one or more missing value. The latter approach goes against information loss that may be caused by list-wise deletion. This is done by the elimination of values when there is a certain data point required to validate the existence of missed value [22]. Both deletion approaches can produce low bias results in the case of MCAR and MAR data [22].

Imputation methods are also adopted as good practice methods to deal with missing data. Imputation methods handle missing values by replacing them with some predicted values. The missing values are commonly predicted by considering non-missing data set evaluation [23]. Simple and regression imputation methods can easily overcome this issue. In a simple imputation process, the missing values are individually filled with quantitative and qualitative analysis of all non-missing values [24]. This analysis encompasses different methods such as the mean, mode, and median of the available non-missing dataset. Although the simplicity of applying these methods has made them popular in many cases [25], they can also cause problems of unrealistic results on a high-dimensional dataset [26]. In the case of regression imputation, it is a well-known statistical technique for handling missing data. Regression imputation consists of two phases. In the first phase, a regression model is developed considering every non-missing valued dataset. After that, in the second phase, the model will be used for missed values imputation based on the regression model prediction [15]. The performance of imputation methods is measured in terms of Mean Absolute Error (MAE), Mean Square Error (MSE) and Root Mean Squared Error (RMSE). Other imputation methods to deal with missing data include hot-deck imputation and expectation imputation [27]. Machine learning methods to deal with imputation also have been designed based on classical algorithms, such as K-nearest Neighbour (KNN), Support Vector Machine (SVM), Decision trees and clustering.

1.2.2 Dealing with data deduplication, heterogeneity and aggregation

Removing duplicated data is important to reduce ambiguity in the inference process of AI models. The removal of duplication is linked to characteristics of data heterogeneity, which can be manifested as structured, un-structure and semi-structured. Structured data is stored in well-defined schemes, such as relational databases. Unstructured data depicts situations in which data is stored in using different formats and it is retrieved from different sources. Linking methods to identify all possible duplicates and merge them into one have been developed to overcome this problem. Solutions to this issue also include the use of the Bloom filter, which requires that a hash function is assigned to evaluate the membership of data.

Conflict resolution methods can also be applied to merge multiple data representations into one (a form of linking) [28]. Entropy methods to reconcile multiple data replicas have been adopted, such that inconsistencies in data are minimized [29]. Data currency analysis identifies the latest value among multiple values of an entity. Some of these values may be obsolete. Thus, by identifying the latest value, it is possible to use it instead of the older versions [30]. Data currency can solve data issues regarding error detection and repair [31]. Data currency repairing can also be utilized to integrate data. Solutions can be divided into two types, semantic-based methods and statistical-based ones. The former focuses on finding errors and fixes based on expert domain knowledge, functional and conditional dependencies and even time dependencies [32,

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

33]. Likewise, statistical-based methods obtain the most possible correct strategy to repair data based on data laws, derived from probabilistic datasets [34].

1.3. DATA QUALITY AND MACHINE LEARNING

Current practices for building AI models suggest that the more data, the more robust models will be produced. This is a partially true statement as this is highly dependent on the applications and characteristics of the collected data and its quality. For instance, an AI model to estimate the number of walking steps can be easily produced with a small number of samples from different users carrying a low-cost (accelerometer) sensor [35]. However, applications such as autonomous cars, even though they produce a large amount of data, e.g., LiDAR, do not have enough data to capture all the different driving situations that can be encountered by the car [36]. In these cases, the spatial and temporal characteristics of the data are said to be statistically abundant but sparse. Thus, data can be large and sparse at the same time, producing models that are under performing require heavy processing and could be expensive to build. Similarly, data can be small but abundant to produce robust models with good performance and low processing requirements. This suggests that raw data is not always in a suitable form for models to learn. Thus, extraction of variables/features is required a priori to build the model. AI models require high quality data to operate effectively. Here, the quality of the data is linked directly to the ability of the data to cover all the different situations that the system can face. This quality is quantified in terms of specialized metrics rather than data sizes. In the following, we review common practices and methods to assess and improve quality of data before model training and validation.

1.3.1 CONSTRUCTING AI MODELS IN A NUTSHELL

The process of building an AI model from data can be summarized in Figure 3. From the figure, it is possible to observe a standard pipeline for AI model learning and deployment over time [3].

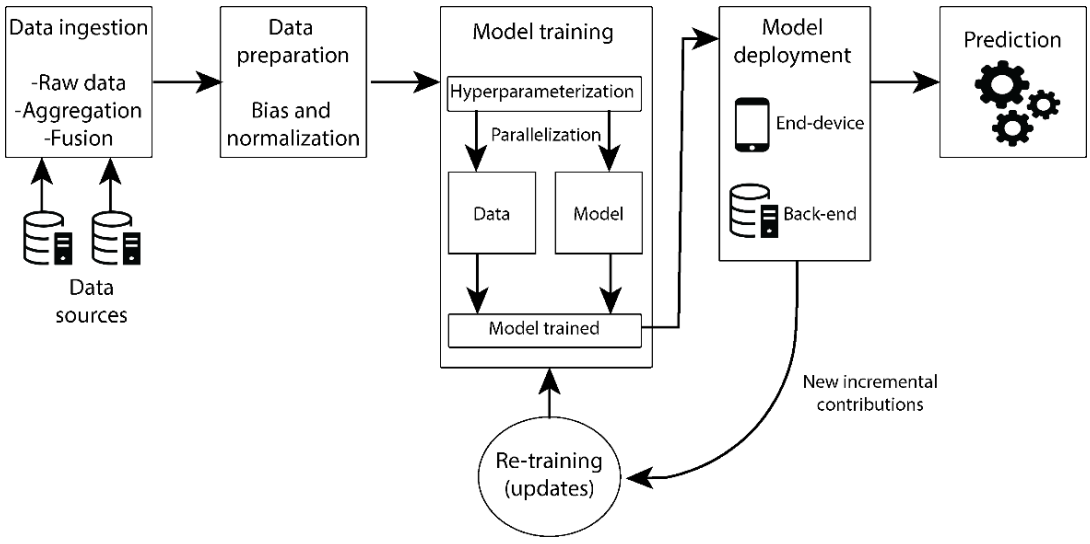


Figure 3: Standard pipeline to build AI models based on machine learning

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

(a) Data collection (data ingestion): Data is acquired from different sources and vendors; it also can be acquired via crowdsourcing/crowdsensing methods [37]. Enough data must be collected, such that it is possible to have available data for training and validation of the model. Data can be collected automatically by sensors, or by using participatory methods that request end users for data contributions. For instance, a user may be asked by a 3D map service to take a picture of the street in exchange for compensation [38].

(b1.1) Data preparation: Since raw data cannot be fed into AI models directly, several data preparations (or data pre-processing) steps are required to transform it into a suitable input for AI algorithms. Data preparation is the first stage in the ML task, and it consists 80% of the time in the data science pipeline [12]. Data scientists are usually faced with datasets that may include missing or invalid data, resulting in low accuracy and performance during the learning process. Hence, data preparation, or data pre-processing, cleans the data and prepares it for learning. Data preparation encompasses several stages: discovery, cleaning, and labelling are the most common.

(b1.2) Data cleaning and discovery: Several methods to clean data are available for this step as discussed in the previous section. In real scenarios, however there is a lack of sufficient data for model construction. Methods to discover data to improve AI models' performance issues have been investigated. Commonly, data discovery methods are categorized into attribute/tuple level and table level discovery [18]. The basic idea of tuple-level data discovery is finding an appropriate external data source with similar table properties and overlapping schemes compared to the in-hand dataset, such that it is possible to fill the missing data with those aggregated contributions [39,40]. Similarly, table-level data discovery methods provide interfaces that allow the users and data scientists to explore the data lake using keywords and extract the related datasets [22,41]. Analysis of co-founding factors that relate different variables of datasets can be also applied to augment datasets with additional data.

(b1.3) Data labelling: After that is passively prepared, the next step is to assign correct labels, which can be used by AI models to perform correct predictions. The most common method to perform labelling of data with high confidence is human inspection, e.g., Captcha [42]. Other methods for crowdsourcing labelling tasks also have been adopted to improve the veracity and provenance aspects of data, e.g., Mechanical Turk [43]. While these methods can grant data with high confidence of AI usage, it can be biased depending on the human annotator group that performed the task. As a result, autonomous solutions also have been investigated [44]

(c) Model training: Once proper data input is in place for building the model, the training process is configured. The training process can be parallelized over the underlying computing resources based on the data or model partitions [45]. After the model is trained, it also must be tested using data. Cross-validation is a technique that can be used to achieve this. Different variations of cross-validation methods are used based on the amount of data available. Among them, leave-one-out cross-validation (LOOCV) and k-fold cross validation can be mentioned. LOOCV splits the dataset of size n into two parts (1 and n-1). One-part (1) is utilized to test the model, whereas the (n-1) parts are considered to build the model. This process is iteratively performed n-1 times, where a different data item is used over time for the evaluation. In parallel to this, k-fold cross-validation can also be applied for model evaluation. K-fold involves randomly dividing the data into folds of equal sizes. The first fold is selected for evaluation, while the rest k-1 is used for model training. This process is also repeated k-1 times, and each time, the testing fold is changed. In addition, bootstrapping can be also used for model validation, and it is typically used in situations where the dataset is small.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

(d) Model deployment, inference and incremental training: Once the model is trained and the performance evaluated, it can be deployed within systems and applications. In classical architectures, models are re-trained and deployed with them as more data is collected. In newer paradigms, such as federated learning (explained in detail in further sections), the model is re-trained by an aggregator, which then propagates a copy of the model to all the contributors.

1.3.2 TRUSTWORTHY AI MODELS

According to EU regulations [46, 47], trustworthy AI covers multiple dimensions, including scalability, (uncertainty) robustness, reliability, timeliness, safety, security, and trustworthiness (explainability, interpretability, transparency). Data quality can contribute towards improving the trustworthiness of AI models. This implies, however, analyzing additional characteristics which could provide extra insight on anomalies of AI models as updates and new data contributions are added over time. Data fairness and transparency are important aspects to ensure when building robust and trustworthy AI models. Indeed, the inference process of trustworthy AI models has clear links between its input data and output predictions. In this context, fairness is analyzed by applying methods to detect possible biases in the data. Biases can be induced or non-induced. Induced biases emerge from situations where the data is hampered as it is collected and prepared for storage. For instance, situational data that depends on the context. Likewise, induced biases arise from manipulation of data performed on purpose, such that the inference process of AI model can be controlled. Induced biases over data are the result of performing adversarial attacks on the data, such as data poisoning. For instance, a steganography method can be applied on purpose over data of IoT devices, such that the device consumes more energy when using the data for training (similar to sponge attacks [48]).

In the case of transparency, methods are used to analyze how data used for training is linked to the inference outputs of the model. Explainable AI (XAI) can be applied in this matter. XAI can help to understand the behavior of AI models and the factors affecting them. For example, XAI methods can be used to evaluate the quality of the data used for training [49]. Indeed, important features extracted from the data can be linked to model outputs, such that it is possible to understand the effect of data quality over the model inference process. In addition, data quality also plays a significant role in deriving explanations through XAI methods.

All in all, making AI models trustworthy through the analysis of data quality is not a (single) process that is applied once at the ingestion step (see Figure 3). Indeed, models are trained and updated over time, and methods to ensure fairness and transparency should be applied transversally through all the steps involved in building AI models. As models are trained in a distributed manner, the data analysis of each contributor induces a higher level of complexity for building AI models.

1.4. DATA QUALITY AND AI DECISION MAKING

As the inference process of AI models can be unbalanced or biased depending on the data used to build it. Several methods are available to analyze how data can govern the decision making of AI. In the following, we discuss the most common methods to achieve this.

1.4.1. Data biases

Imbalance data: Machine Learning (ML) as a concept represents the idea of detection of the pattern for issues that could not be solved analytically. The core of this idea and the cause of the popularization of ML in research and development (RND) community lies in improved

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

computability, new learning algorithms and large-scale data that became available in the last decade. Arguably, the most important factor in these accomplishments lies in large and high-quality datasets which contain information and could lead to significant conclusions.

Besides small datasets, one other problem that can jeopardize the quality of learning is an insufficient distribution of data inside the set. This distribution is called skewed data and it means that set does not contain enough information about all described phenomena. Common example is with healthcare datasets for specific diseases where the information about positive patients (sick patients) takes a small share regarding the whole population. This problem is called imbalanced data and it means that distribution of samples between classes inside the set is not equal. In real-world problem, imbalanced datasets are considered to be ones where one class overpopulate the other one with ratio 100:1 or greater [50]. In this issue, none of the traditional models could not provide adequate solution appropriate for every-day use.

Data imbalance can derive from the nature of the data set, as in healthcare, and that is called intrinsic imbalance. Other types of imbalances regarding time, storage, etc. is called extrinsic imbalance. Beside these imbalances, which are called between-class imbalance, there is within-class imbalance meaning that the samples inside one class have multiple subconcepts which are not equally represented [50]. The issue regarding imbalanced data could be solved in multiple ways which can be roughly divided in three groups: modification of the dataset; modification of the learning process; and modification of the assessment techniques [51].

Modification of original dataset is called sampling and it involves increasing or decreasing minority or majority class, respectively and it is called oversampling and undersampling of the class. Random sampling consists of copying or removing random samples from the class which leads to more balanced data. Modified dataset is more convenient for learning, but it goes with unique problems such as losing important concepts with undersampling and replicating (not providing new information) with oversampling [48, 49].

Synthetic Sampling and Data Generation (SMOTE) is a more advanced technique for oversampling the minority class by overpopulating samples based on their neighborhood. As recognizable from the previous sentence, SMOTE is roughly based on K-Nearest Neighbors (KNN) [1]. Unlike random sampling, this method is more based on the features of the samples, meaning that for every sample algorithm generates the new one by interpolation with one of the random neighbors from the same minority class. This leads to oversampling and the generation of new information. SMOTE showed a big improvement in learning on imbalanced data so various modifications were developed by the researchers. Borderline-SMOTE and ADASYN are the most famous derivative of the classical SMOTE algorithm. Their contribution lies in selective oversampling. Borderline-SMOTE generates samples for every existing sample from the minority class which have more majority neighbors then neighbors of their own, except for ones with all majority neighbors which are considered noise. On the other hand, ADASYN is a combination of Adaboost and SMOTE algorithm. Firstly, the distribution function is estimated based on the difficulty of samples to be learned, and the new samples are generated only near hardest-to-learn samples. [3] Other similar algorithms use combination of SMOTE and K means clustering, firstly, into detecting clusters via standard clustering protocol and afterwards oversampling each cluster to the size of the biggest. In some realization, more clusters are defined then it's class in dataset which will both within-class imbalance and between-class imbalance improvement [48,49].

Cost-Sensitive approach introduced the idea of modifying the learning process by introducing a cost matrix associated with the ability to misclassify some samples. The cost matrix consists of

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

elements Ci,j which represents the number of samples from class j that are classified as class i. Naturally, on the diagonal of the cost matrix are shown correctly classified samples despite misclassified samples in other matrix cells. The main purpose of the matrix is to minimize the misclassification by minimizing the cost function, where P(i|x) is probability estimation of classifying sample x as class i [50, 51].

The classification is often assessed using accuracy, which is by the definition the ratio of correctly classified samples. This metric is not good enough for highly imbalanced data because it is possible to achieve high accuracy with just classifying all samples as majority class. This is the reason why researchers proposed more metrics such as precision, recall, F and G scores [53]. F score is an important metrics because it represents the combination of Precision and Recall and the impact of those two factors are manageable with parameter beta. Beta is usually 1 and then we have the most used metrics called F1 score. When beta is less then 1 the F score weights the precision more, while for beta greater than 1 it goes opposite. Even thou these metrics shows better assessment in imbalance data they cannot still compare different classifiers on different distribution of data [54].

ROC is an assessment technique special because of its possibility to present results graphicly. ROC shows the ratio of TP rate and FP rate. These metrics present the ratio of correctly/incorrectly classified positive sample to all positive/negative samples. Ideal classification is considered to be in top-left corner of the graph (0,1) while the values on the diagonal is considered to be as good as random classification. For hard classifiers, metrics are represented as dot, while for the soft classifiers the metrics are shown via lines (ROC curves). Comparison of two soft classifier is possible with the use of ROC curves where the better classifier is the one with greater area under the curve (AUC) [48, 49, 51].

1.4.2. Bias metrics

Bias metrics refer to descriptive statistics that we can compute both before training the model, and after training. It this division we look for whether the inequality is inherited with the dataset, or it is induced by the model. It his case we differ pre-training bias and post-training bias. For easier understanding we should define notation often used in literature. y and $\hat{y}$ stand for labels and predictions respectively; a and d are used as a notation for favorable and disfavored facet; while n_a and n_d stand for number of samples from actual and predicted class b respectively. The scenario where facet a has a higher proportion of positive predicted outcomes is called positive bias, while the scenario with the higher proportion of positive predicted outcomes for facet d is called negative bias [55].

- **Pre-training Bias**

1.4.2.1.1. Class imbalance (CI)

$$CI = (n_a - n_d)/(n_a + n_d)$$

Class imbalance is the first metric and one of the simplest. It represents the relative difference between the number of favored and disfavored classes. CI can take values between -1 and +1. Negative values indicate a greater disfavored class, positive values indicate a greater favored class, while values near zero indicate a balanced dataset [54, 55].

1.4.2.1.2. Difference in positive proportions of true labels (DPPL)

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

$$DPPL = q_a - q_d = \left(n_a^{(1)}\right)/n_a - \left(n_d^{(1)}\right)/n_d$$

This statistic shows us the difference between the ratios of positive labels in relation to the whole class for favored and disfavored class respectively. DPPL tries to describe, numerically, the ability of disfavored class to learn to predict positive outcome. The metric can take values between -1 and 1 for binary and multiclass classification and values between -∞ and ∞ for continuous labels. The negative values indicate negative bias, positive values indicate positive bias, while values near zero stand for demographic parity. [54, 56]

1.4.2.1.3. The Kullback-Leibler divergence (KL)

$$KL\left(P_a\middle|\middle|P_d\right) = \sum_x^{(} P_a(x)\log\left(\left(P_a(x)\right)/\left(P_d(x)\right)\right)\right)$$

The Kullback-Leibler divergence (KL) measures the statistical distance, and it is often called "relative entropy". It shows how distant the distribution of facet a, Pa(y) and the distribution of facet d, Pd(y). As the relative entropy of Pa(y) with respect to Pd(y) it quantifies the amount of information lost when moving from Pa(y) to Pd(y). KLd is not symmetric. KL metric can take values between 0 and +∞ for all types of classification. Values near zero mean the outcomes are similarly distributed for the different facets. The more positive value indicates larger divergence [54, 56,57].

1.4.2.1.4. Conditional demographic disparity in labels

$$DD_i = \left(n_d^{(0)}\right)/n^{(0)} - \left(n_d^{(1)}\right)/n^{(1)}$$

$$CDD_d = 1/n\sum_i n_i\, DD_i, \sum_i n_i = n$$

The demographic disparity metric (DD) determines whether a facet has a larger proportion of the rejected outcomes in the dataset than of the accepted outcomes.

The CDD metric is the weighted average of DDi, where every DDi is scaled according to the size of the subgroup. The value of the DDi can range between -1 and 1, where 1 indicates no rejections in a and no acceptance in d; -1 indicate no acceptance in a and no rejections in d.

If a particular set does not condition any class, the CDD is zero if and only if DPL is zero [54,56].

- **Post-training Bias**

1.4.2.2.1. Disparate Impact (DI)

$$DI = q'_a/q'_d,\ q'_a = \left(\hat{n}_a^{(1)}\right)/n_a,\ q'_d = \left(\hat{n}_d^{(1)}\right)/n_d$$

The disparate impact (DI) [60] metric is defined as the ratio of the proportion of positive predictions for facet d over the proportion of positive predictions for facet a.

DI can take only positive values for all types of classification.

Meaning of the DI value:

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

- For DI between 0 and 1 – positive bias
- For DI = 1 – demographic parity
- For DI greater than 1. [54, 59,60]

1.4.2.2.2. Difference in conditional outcome

This metric compares the observed labels to the labels predicted by the model. Two metrics are important: DCA (Difference in conditional acceptance) and DCR (Difference in conditional rejection). The idea behind these two metrics is the same except that the first one compares the positive labels with the positive predictions, while the second one compares negative ones.

$$c_a = \left(n_a^{(1)}\right) / \left(\hat{n}_a^{(1)}\right),\ c_d = \left(n_d^{(1)}\right) / \left(\hat{n}_d^{(1)}\right),\ CDA = c_a - c_d$$

$$r_a = \left(n_a^{(0)}\right) / \left(\hat{n}_a^{(0)}\right),\ r_d = \left(n_d^{(0)}\right) / \left(\hat{n}_d^{(0)}\right),\ CDR = r_a - r_d$$

Both metrics can take a value between -∞ and ∞. The values around zero indicate that both facets get accepted/rejected in a similar way. Positive values indicate for positive bias, and vice versa [1, 6].

1.4.2.2.3. Treatment equality (TE)

This metric compares the ratio of false positive (FP) and false negative (FN) values between disfavored and favored classes. Both of these parameters belong to the classical classification metrics.

$$TE = FN_d/FP_d - FN_a/FP_a$$

FN presents the number of misclassified negative samples, while FP presents the number of misclassified positive samples. The ratio shows the types of errors across favored and disfavored classes. It can take values between -∞ and ∞, and it is not defined for continuous labels. A positive value indicates a greater ratio of FN to FP in faced d compared to facet a, while negative value indicates smaller ration of FN to FP in faced d. Values near zero stand for balanced error ratios in facets a and d [54, 59].

1.4.2.2.4. Counterfactual flip test (FT)

The main idea behind this test is to check whether the samples from the disfavored set get the opposite prediction from the samples from favored set. For every sample in the disfavored set, the favored pair is selected using KNN algorithm.

$$FT = (F^{\pm}F^{-})/n_d$$

where F+ stands for the number of samples where the disfavored sample gets negative, while favored one gets positive prediction. F stands for number of where disfavored sample gets positive and favored one negative prediction.

FT can take the value between -1 and 1, where zero stands for parity, negative value for a negative bias, and positive value for positive bias [54, 59].

- Biased attributes

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

- Induced bias
  - Data poisoning
  - Data augmentation
- Non-induced bias
  - Situational
  - Data collection and sampling

Non-induced biases correspond to those biases that are present in the datasets but do not require intentionality on their creation. The presence of those biases might lead to unintended consequences such as:

- Data not being representative of the population or phenomenon to study. Between other effects, probably the most prevalent in the literature is the sample size disparity [61, 62].
- Data not including attributes that are good predictors of the phenomenon to study.
- Data including content generated by humans that shows biases against demographic groups. This has been referred before as “Data as social mirror” [60].

The analysis of data biases in social data, [65] identified two types of biases depending on the control data practitioners (mainly developers, researchers) might have on the biases reflected on the data.

Biases that occur without control of the data practitioner are named biases at source. Between them, the authors distinguished the following categories:

Functional biases. Result of platform-specific mechanisms or affordances, that is, the possible actions within each system or environment.

Normative biases. Are the result of written or unwritten norms and expectations of acceptable patterns of behavior on a given online platform or medium.

External sources. Biases resulting from factors obfuscating the distribution of the information.

Non-individual accounts. Interactions on platforms that are produced by organizations, automated agents or individual users owning multiple accounts in each platform.

On the other hand, there is the category of biases that are created at the time of building datasets. Those can be differentiated based on the stage of the data curation process where they appear:

Collection. Typically, those are introduced during the selection of the data sources, or by the way in which data is acquired from these sources and/or prepared.

Processing. Biases introduced during operations such as cleaning, enrichment and aggregation.

Analyzing. Biases introduced at the time of performing qualitative and/or quantitative analyses, obtaining descriptive statistics and performing observational studies.

Evaluating. Introduced at the time of selecting metrics, executing assessments and interpretation of results or not fostering reproducibility.

SPATIAL project is funded by the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Additionally, there is the problem of dataset generalization [53, 64] showing how biases affecting data might lead to unwarranted situations in which the knowledge extracted from one dataset might not generalize well to other datasets.

Similarly, distribution or domain shifts [67] lead to scenarios in which the collected data, often used for model training, is not representative from the reality anymore what might cause a drastic loose of performance in production.

1.5. BIAS DETECTION MECHANISMS

To analyze possible biases on the data, several mechanisms are available. In this section, we examine the most common and effective methods that have been adopted to overcome data quality issues, including fairness analysis, divergence testing and causality techniques.

Fairness analysis: Fairness implies fair and impartial treatment, meaning that all sides should be taken equally without favoritism towards one side or another [68]. With the appearance of modern machine learning algorithms and their implementation in everyday life, in addition to standard rules and data protection, there was a need for regulation and monitoring of decisions made by algorithms and ensuring that such decisions will not threaten basic human rights [69]. Features that could lead to any kind of discrimination, such as sex, race, age, income, etc. are called sensitive (protected) attributes and they are carefully monitored during predictions. One other factor that could lead to unfair predictions are proxy attributes. These attributes are not sensitive as our protected attributes, but they highly correlate with them which can be also source of unfair predictions [69]. Detection of why the model is unfair and detection of sensitive and proxy variables is called fairness analysis.

Fairness can be divided in individual fairness and group fairness. Individual fairness considers two individuals with similar characteristics that are treated similarly. Group fairness, also known as statistical parity and demographic parity, considers that the demographic of the group which receive particular positive outcome is identical to the demographic of the whole group. Many definitions arose from the group fairness and one of the important ones is causal fairness which states that the sensitive attributes should not have causal impact on the outcome of a task [68, 69].

Beside post-training metrics explained earlier, two additional metrics are often used.

Equal opportunity: True positive rate is defined as:

$$TPR = TP/(TP + FN)$$

Equal opportunity states that TPR should be equal across all groups, meaning that positive individual should have positive prediction in both favored and unfavored class. [67, 70, 71]

Equalized odds: Equalized odds is more in definition of fairness then rule of Equal opportunity. We need to introduce False positive rate as:

$$FPR = FP/(FP + TN)$$

Equalized odds states that both TPR and FPR needs to be equal across all groups, meaning that the probability of individual the probability of an individual from the unfavored class towards a true or false positive outcome should be equal to the probability of an individual from the favored class towards the same (true or false) outcome [54, 70, 71].

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Causality techniques: We can say that random variable X causes random variable Y only if there exists at least two different interventions on X that can cause two different probability distributions of Y [74]. Understanding the causality between our variables (features) can not only help us remove unwanted dependency between sensitive variables and outcomes but help us increase the transparency and explainability of the fair models [73]. For visual representation, researchers often use Causal diagram where every variable is represented with nodes and every causality between them with lines [75].

Insufficient knowledge of the data and the causality between them can lead to correlations that can harm the integrity of the model. One of a one of the paradoxes that often appear in literature is Simpson's paradox which says that a trend or result that is present when data is put into groups that reverses or disappears when the data is combined [76], meaning that in the case of omitting part of the variables and not observing the whole picture, there may be unintentional wrong conclusions, which when include sensitive attributes can be discriminatory.

One of suggested measures for causal fairness is counterfactual fairness. A predictor is said to satisfy *counterfactual fairness* if:

$$P(Y(a,U) = y|X = x, A = a) = P(Y(a',U) = y|X = x, A = a),$$

for all y, x, a, a' in domain of its variables. Variable A is defined as protected attribute and a and a' are considered its instances (such as male or female, race, etc.). This means that model is considered counterfactual fair if the prediction would not change with the change of sensitive attributes or attributes affected by them [71,75,59,76].

Given the rigidity of the previous definition, two new terms were introduced by scientific community: proxy discrimination and unresolved discrimination. The predictions do not suffer from proxy discrimination if predictions in graph are not connected to the proxy variable. Proxy variable is variable that is not sensitive but derived from sensitive attribute. The predictions do not suffer from unresolved discrimination if predictions in graph are not linked to resolved variables (variable influenced by sensitive attribute but is not considered to be discriminatory) [68,71,75,59,76].

Causal reasoning is very important for maintaining fair prediction but in real life it is very hard to implement correct causal model and removing correlated features found could seriously compromise accuracy of the predictions [73].

1.6. FROM CENTRALIZED TO DISTRIBUTED MACHINE LEARNING

While centralized machine learning can be used to build AI models over time, the process can be accelerated further by relying on distributed machine learning techniques. The key idea of distributed machine learning is to exploit distributed devices to train AI models with heterogeneous data. In this manner, AI models can be more robust and resilience when facing different situations. Figure 4 compares both paradigms. From this figure, a key difference is that contributors in distributed machine learning provide logic contributions (binary) to the overall model rather than just data.

Centralized machine learning: A centralized architecture was assumed in classical machine learning scenarios, where all the data used for training is gathered to a centralized site, and a single AI model is trained with the data. This architecture has limitations in scalability and privacy. Obviously, the centralized site has a limit in terms of the training data it can handle and

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

the consequent communication cost, and models that learn from sensitive personal data has the problem of copying it from users' devices and keeping in the centralized machines.

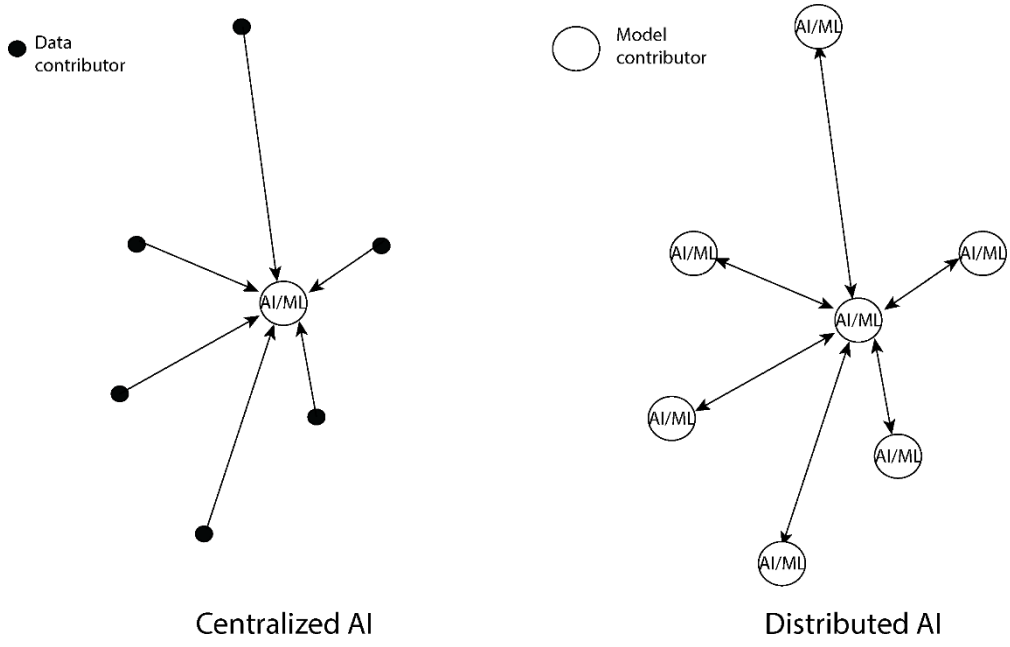


Figure 4: Centralized vs. Distributed Machine Learning Paradigms

Distributed machine learning: There are a number of variants in distributed machine learning architecture although they share common properties. These architectures are tailored for two particular tasks, training and inference of AI models. Model inference load can be distributed between multiple using collaborative processing and distributed computing methods [77]. Likewise, model training relies on distributed devices to build AI models and in this process, devices intercommunicate with each other to share and update parameters. Distributed machine learning addresses the scalability issue as the training load is divided across multiple machines There are two main approaches to distributed machine learning: first, data parallelism where the models are built with different subset of the data; second, model parallelism where the model is divided into multiple parts and distributed across the machines. For example, different layers may be trained on different nodes. Another method to train machine learning models is federated learning, which builds AI models in a privacy-preserving manner and can overcome the problem of data heterogeneity. Due to the large adoption and integration easiness of the approach, our experiments in this deliverable rely on federated learning.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

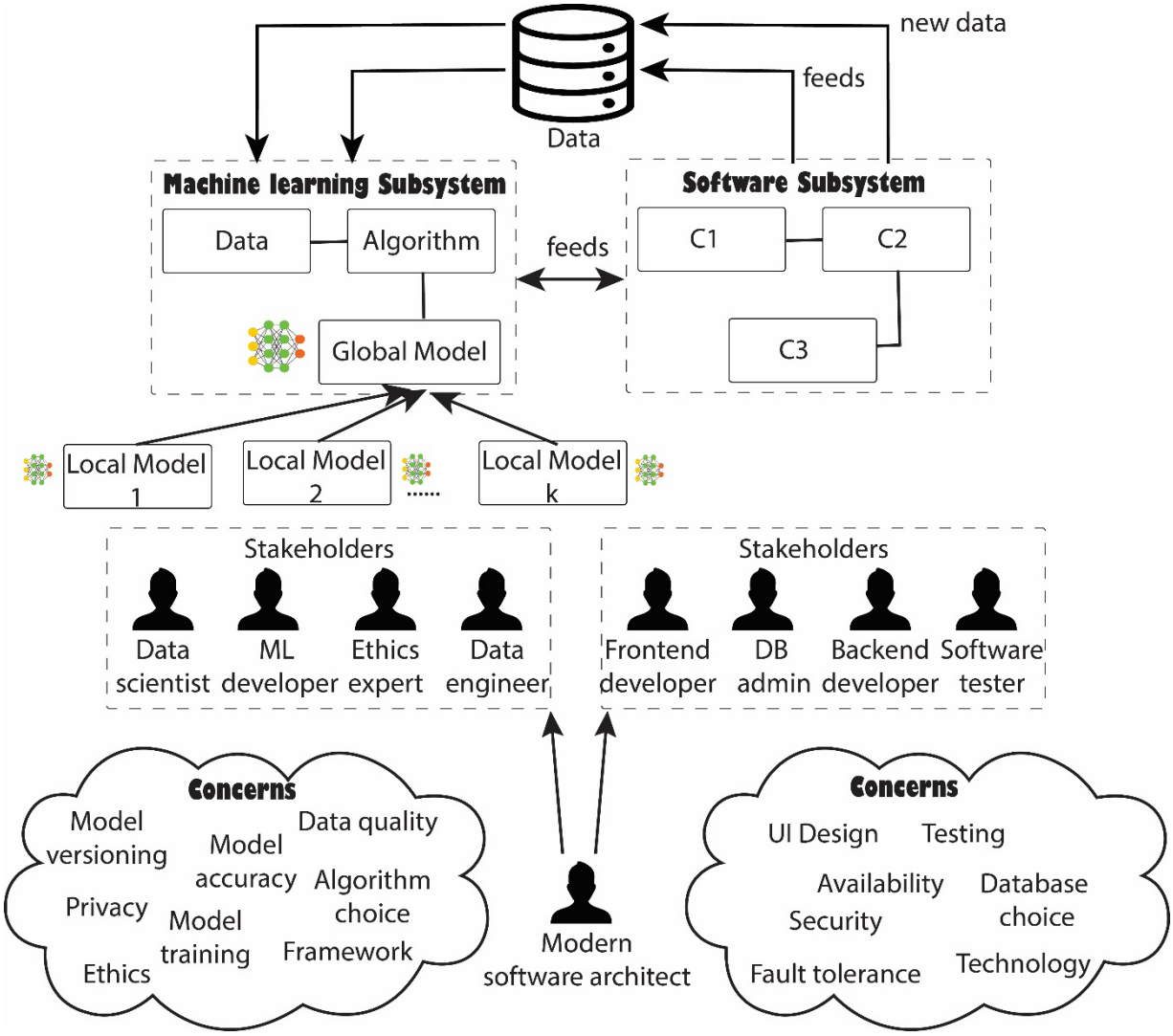


Figure 5: Distributed AI powered by federated learning in modern systems and applications

Federated Learning (FL) allows participating clients to jointly train a global ML model without revealing any available data. Figure 5 shows how FL extends further modern machine learning architectures (See section 1.0 - Background). In simple terms, the clients participating in the federated training procedure first train an ML model locally based on their available (possibly sensitive) local data [3]. Subsequently, each client only shares their learned model weights with a central server, which aggregates the individual model weights to a global model by using a mechanism called Federated Averaging [3]. By iteratively repeating this procedure, the global model can achieve performance comparable to a model trained centrally using all available data. However, the primary advantage of this approach is that the global model can be trained without requiring the clients to reveal any private or sensitive data, thus, protecting their privacy. Besides, it was shown that the FL training mechanism is capable of learning high-performing ML models even when the data is unbalanced and non-IID (identically independently distributed) distributed among the participating federated clients [3, 78].

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

1.7. DATA BIASES IN DISTRIBUTED MACHINE LEARNING

Deep learning has been widely used in everyday life for some time now. Such an application is conditioned by the large amount of data that is generated and its possibility of application in deep learning. Two challenges that have arisen are training large models on large amounts of data that cannot be stored in one place. For the sake of optimization, the training job is divided into several agents usually, GPUs, which work in parallel. This type of learning led to the creation distributed machine learning.

Two ways to distribute model are data parallelization or model parallelization, with the former being the most used approach. Data parallelization aims at distributing same model with chunks of data on multiple GPUs. The model is trained on smaller data and updates are sent to centralized server where aggregation across all agents is executed [81]. Model parallelism takes another approach, which allows you to distribute different parts of the model across different agents and to specify in your model definition which parts of the model should go on which device. Only important thing is that the labels and the output layer of the model must be stored on same agent [82].

In this basic type of machine learning data is firstly centralized and then distributed across multiple workers in data parallelism. In this type of learning, only “classical” bias can exist while inducing bias via distribution is not likely. Another kind of distributed learning, that is quite likely to diminish fairness in model is a situation where data is originally distributed across multiple sources who do not have sufficient trust to share the training data with each other (e.g., health records, location-based services, personalized recommendations, autonomous vehicle, etc.) and this use of this data can seriously jeopardize privacy [81,82]. This kind of distributed datasets are susceptible to traditional bias which is already known from centralized storage.

FL is the solution that is proposed for this kind of privacy sensitive learning, where only model is downloaded on every source, trained and the parameters of the model is returned to centralized server (weights and biases) for aggregation with other updates [83, 82]. In this approach, the data is not sent anywhere and sensitive data stays on the local device. The flaw of this approach is that the performance of FL largely depends on the data distribution on the local device and can decrease significantly if the data is not i.i.d. (Independent and identically distributed) [83, 82]. This is the main difference between “classical” and federated learning, because the random distribution of centralized data (in data parallelism) it is considered that every agent gets i.i.d. data.

Beside the different distribution of data, federated learning could be affected by other types of bias, most often selection bias. One of the ways to implement FL is the asynchronous approach. This approach is characterized by the fact that the centralized server does not wait for all agents to update the parameters but aggregates them as they arrive. This approach discriminates against weaker devices (i.e., slower devices, devices with poorer internet connection, etc.) that most often come from areas with poor socio-economic status, which means that the data set on which the central server is trained more often does not represent a global distribution of data. This type of bias is called bias via party selection, or drop outs.

Techniques for bias mitigation in FL are inherited from “centralized” learning and it could be divided based on the place where the algorithm is placed in pipeline on: pre-processing, in-processing and post-processing [86]. On one hand, pre-processing based bias mitigation techniques focus on processing the training set before the actual learning process. These techniques most often include populating minority classes using methods from the chapter on

SPATIAL project is funded by the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 101021808.

the imbalanced data and they are independent of the type of model used for learning in later process [84, 85]. On the other hand, in-processing based bias mitigation techniques are focused on modification of learning algorithm to improve fairness using discrimination-aware approaches or cost-sensitive methods which penalize the discrimination using cost matrix. Unlike the pre-processing techniques, this type of mitigation is highly dependent of the model and it's learning process. [86]. Likewise, post-processing based bias mitigation techniques cannot affect learning process and training data, so this approach helps algorithm to be fairer using metrics such as Equalized odd, Disparate Impact, etc. [84, 86].

2. DATA QUALITY TRADE-OFFS IN DISTRIBUTED TRAINING

As discussed so far, the robustness of distributed AI models is tied to the characteristics of the distributed data and devices that are involved in the training process. Thus, there are different trade-offs to consider when building AI models in a distributed environment. In this section, we explore 1) the effect of different spatial and temporal characteristics of data used for training; 2) the impact in model performance when removing sensitive and private information from datasets available for training and 3) the relation between model performance and training configuration of distributed devices.

2.1. SPATIAL AND TEMPORAL CHARACTERISTICS OF DATA FOR DISTRIBUTED MODEL TRAINING

We begin by demonstrating the impact of relying on distributed data for model training. Since distributed devices can contribute with data that has different spatial and temporal characteristics, it is important to determine the minimum configuration of devices and their contributors to achieve optimal model training. Our results are verified by comparing our benchmark to others that rely on a classical machine learning architecture. In the following, we describe the experimental setup in detail.

2.1.1 Experimental setup

Dataset and baseline: A state-of-the-art dataset is used in our experiments, such that it is possible to verify our results against other benchmarks reported in the literature. We rely on the TrashNet dataset to conduct our experiments. Initially, we replicated the centralized model training reported by TrashNet [89]. This centralized model is defined as our baseline. We used TensorFlow framework to construct a Convolutional Neural Network (CNN). We implemented a centralized network consisting of eleven layers, similar to AlexNet [90]. The CNN model was trained with a train/test split of 70/30, images of trash of size 256 x 256 px, 40 epochs, a batch size of 32, and learning rate of 0.0001.

System model and setup: To scale the functionality of our prototype, we rely on distributed training. Thus, TrashNet centralized training environment is transformed into a distributed learning one that uses federated learning. To enhance the performance of the model and achieve the baseline accuracy, we increase the amount of data by applying six types of image manipulation. Data was augmented using computer vision python package to expand the dataset size so that a better model could be trained. We applied well-known data augmentation methods, including Rotation, Vertical Flip, Horizontal Flip, Channel Shift. After applying data augmentation, the resulting dataset consists of 15,040 images belonging to 5 trash categories:

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

paper, glass, trash, metal, plastic. With this information, a CNN model for image classification is trained. The accuracy of the model reaches up to 76.8%, which is similar to the one reported extensively in the literature [89]. We attempted to maximize the data we had through augmentation. Along with that, more through hyperparameter search could be performed.

Experiments: Since our goal is to demonstrate the effect of distributed data in model performance, we systematically divided the dataset to produce multiple configurations of devices for model training. Our experiments focus on analysing the amount of data samples that are distributed among multiple clients, and the effect of multiple device contributors toward model training.

2.1.2 Results

Federated learning performance: Figure 6(a) represents the effect on rounds required for the model to converge with different samples size per clients and the number of clients. From the results, it can be observed that the number of rounds required for model convergence is directly related to the number of clients participating in the training. The progression on the number of rounds when the number of clients grow twice is 32.76% (5 to 10: 35.38%, 10 to 20: 22.90% and 20 to 40: 40%) on average. It is due to the fact that as the number of clients increases the variability of data in the clients also increases which in turn induces the model to converge at a slower rate (resulting in a higher number of rounds). Simultaneously, in the same figure, it can be observed that the number of samples per clients number is inversely proportional to the rounds required for convergence. When the number of samples is increased by 100, the number of rounds decreases 15.8% on average (88 to 176: 25.1%, 176 to 276: 19% and 276 to 376: 3.3%). Here, we find another interesting observation if we discard the training using 5 clients.

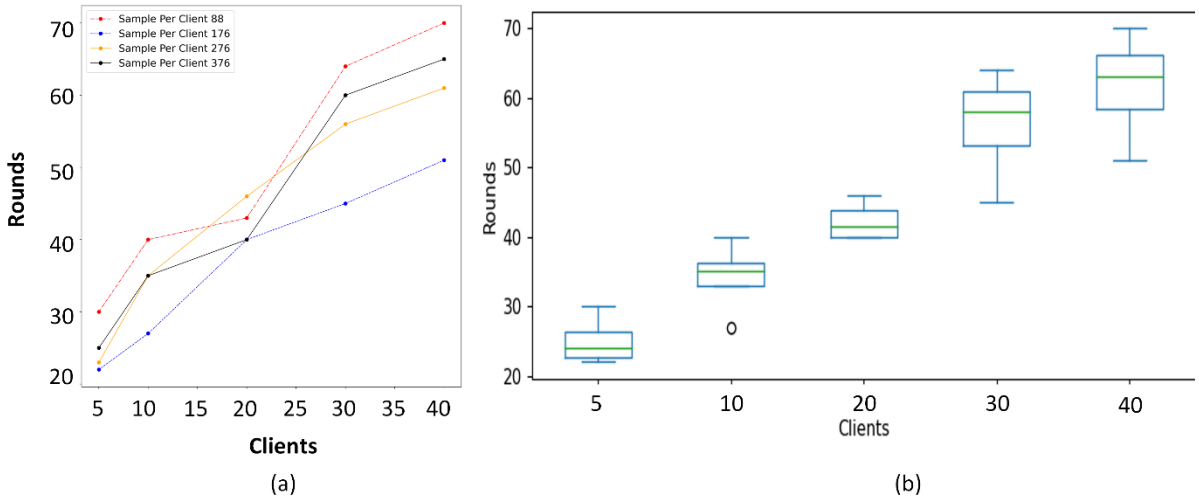


Figure 6: FL performance (a) Effects on rounds required for model convergence with different samples, (b) Variation in rounds for the increasing number of participating clients

Variations in training rounds: Figure 6(b) represents the variations in the number of rounds when considering different sample sizes. We observe that as the number of clients increases from 5 to 40, the standard deviation of the required number of rounds increases in the order of 3.55, 5.3, 5.2, 8.2, and 8.1. This indicates employing less number of clients during training can

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

make the rounds of training consistent. However, in order to have a trade-off between the number of rounds and the accuracy of the training for the number of clients, we explore the relation between the accuracy and clients in the next experiment.

Optimal training accuracy: From Figure 7(a), we observe the training accuracy when the model converges for different data samples and the number of clients. The optimal accuracy is more when there are more clients involved in the training process. We observe that the average decrease in accuracy for the same size of data sample to be 11.78% (88 samples: 5.2%, 176 samples: 6.3%, 276 samples: 8%, 376 samples: 27.63%) when the number of clients increases from 5 to 40. Furthermore, it can be observed that for the same amount of clients and different sample size the accuracy increases on an average of 30.95% (5 clients 43.39%, 10 clients 38.46%, 20 clients 29.16%, 30 clients 18.75% and 40 clients 25%). Here we find relation between the optimal accuracy model and the number of rounds the training needs to take place for that. It depends on the number of clients that can participate in the training process. If the intention is to have a robust model, it is important that the data distribution is IID and does not differ between participants (more samples provided). On contrary, when the aim is for the training to be faster less clients are recommended. In addition to these factors, coordination time is an important factor for stable collaboration FL. Hence next we explore the essence of time involved in FL.

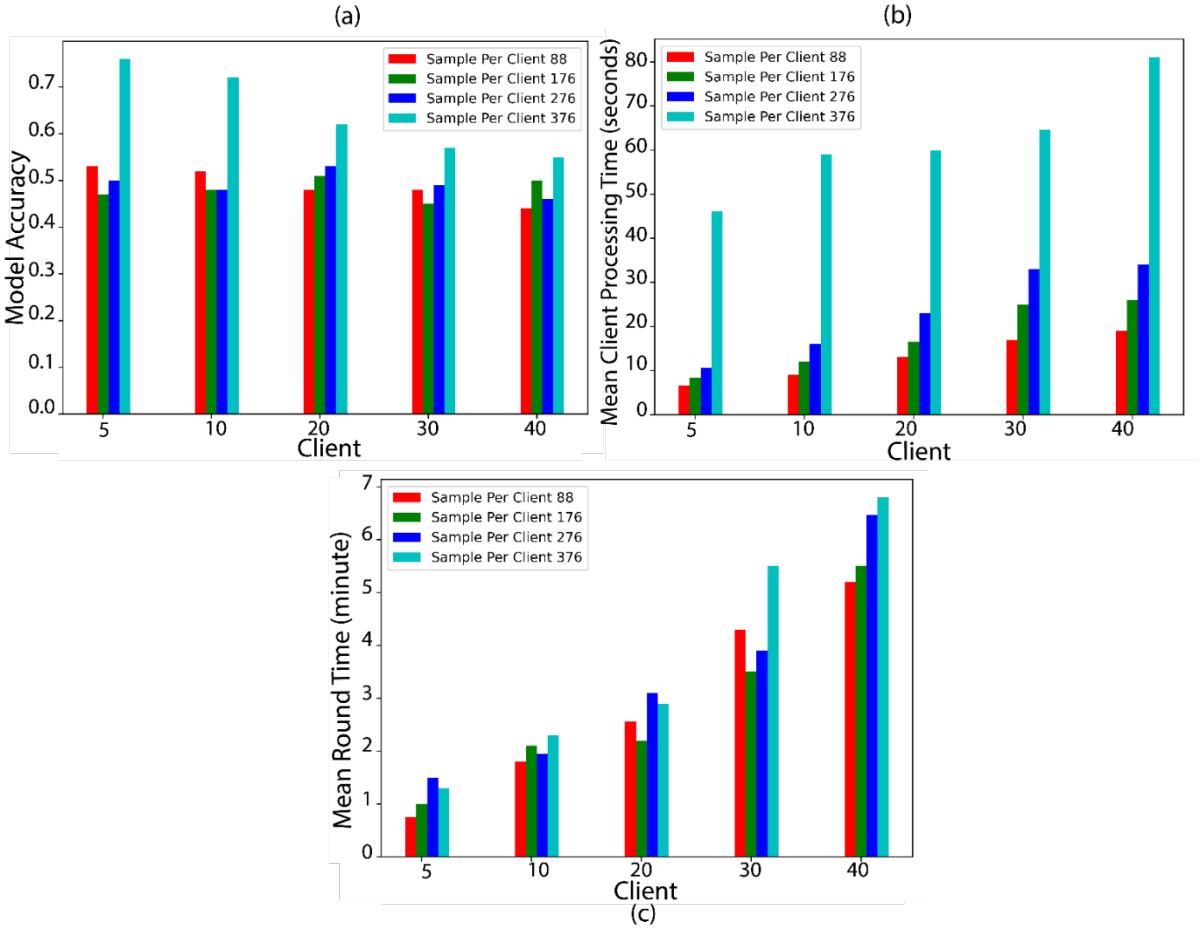


Figure 7: Model performance for different number of clients

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Processing time of the clients in FL: The mean client processing time is depicted in Figure 7 (b) when the clients are in training using various sample sizes. As the sample size increases, a steady increment of the client processing time increases with an average of 20.77 seconds (88 samples: 12.92 secs, 176 samples: 17.57 secs, 276 samples: 23.31 secs,376 samples: 75.25 secs). As the chunk of the data increases for training, it is evident that the processing time of the client increases. This phenomenon is more distinct when the client size is more than 10. Here, it can be also observed from the results that with respect to the same size of data samples, the time for the client processing also increases as the model upload and download is a part of the processing time of the client. The mean round time can be observed in the Figure 7 (c) when the clients are in training using various sample sizes. It can be seen here that the mean round time varies between different sample sizes when the number of collaborative clients is more than 20. The round time increases because it involves the waiting for updates, loading the model and aggregating the model. Waiting for the updates depends on the clients and hence more the round time. But this variation is significant when the number of clients is less (like 5 and 10). Furthermore, it can be observed that with the increment of sample size, the mean round time increases by 31.46% on average.

Collaboration time in FL: The total collaboration time from the agent's perspective is demonstrated in Figure 7 (c). This is done by considering the rounds required for optimal training and the mean time required for the agent to complete each round of training. It can be observed from the Figure 7 (c) that with involvement of more participant clients, the time of training increases a steady rate. We can observe that the number of samples size increases, the time of training increases proportionally. This result complements the insight from the Figure 6 (a) and provides us with a benchmark for collaboration of the agent for an opportunistic FL training in groups.

Summary: As we conduct experiments on FL training and benchmark them, we find that the time for collaboration and accuracy varies depending on the number of clients and size of data samples. This can be summarized in the Table 1. From this table, we can draw that when the sample size increases by 100, irrespective of the number of clients, the time required for the training increases by 10.75 times on average (marked in green). On the other hand, it can also be observed that the average accuracy increases by 0.15 percent marked with gray.

Table 1: Summary of results for different number of clients and samples

Client	376 Samples			276 Samples			176 Samples			88 Samples		
Available	Round	Time	Accuracy	Round	Time	Accuracy	Round	Time	Accuracy	Round	Time	Accuracy
5	25	80	0.76	23	92	0.5	22	88	0.47	30	120	0.53
10	35	98	0.72	35	140	0.48	25	108	0.48	40	160	0.52
20	40	110	0.62	46	184	0.53	40	160	0.51	43	176	0.48
30	60	198	0.57	56	224	0.49	45	180	0.45	64	256	0.48
40	65	200	0.55	61	244	0.5	51	204	0.5	70	280	0.44
Average	45	137.2	0.64	44.2	142.8	0.5	36.6	148	0.48	49.4	198.4	0.49

2.2 DATA FEATURE REMOVAL AND MODEL PERFORMANCE

Next, we analyze how data removal affects the performance of AI models. To evaluate this, we focus our analysis on a healthcare application designed to monitor fall detection using accelerometer data, e.g., wearable and smartphone devices. To successfully distinguish between falls and activities of daily living (ADLs), it is crucial to have access to as many data points representing as many different fall and ADL types as possible. However, to compile such a rich and representative dataset, it would be ideal to have access to the everyday motion and

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

acceleration data of end-users. One approach to achieve this would be simply asking end-users to record and label their motion and acceleration using their mobile end devices (e.g., mobile phones) in their everyday lives. However, this approach has two significant difficulties. On the one hand, users will perform very different activities in their daily lives and fall with varying frequency. Therefore, the gathered data will likely be unbalanced and non-IID distributed among the end-users. On the other hand, recording motion and acceleration data in everyday life raise serious privacy concerns, which is why users will probably not be willing to share their data with a central instance. Hence, federated learning could be a suitable and effective approach to overcome the two mentioned difficulties by 1) allowing the users to record motion and acceleration data securely and keeping this sensitive data private, and 2) training a global model in a privacy-preserving federated fashion based on possibly non-IID data.

From our healthcare application, we then investigate whether it is feasible to implement high-performing fall detection models using privacy-preserving federated learning mechanisms. In addition, we want to examine the effect of the varying distribution of ADL and fall (sub)classes among federated clients with respect to the binary classification capabilities of the global fall detection model. More precisely, we will perform and evaluate several experiments in which the participating clients contribute with a varying number of diverse ADL and fall types.

Since typical Edge-based federated learning settings are usually based on heterogeneous networks with heterogeneous clients (e.g., in terms of network latency, connectivity, bandwidth, or computational power) [78,89], the number of clients participating in the federated training rounds is likely to fluctuate. Therefore, we will also investigate the effect of reduced client availability in the federated training rounds on the classification performance of the global FL model.

2.2.1 Experimental setup

Dataset: In order to perform the described experiments, we leverage the UniMiB SHAR [92] dataset. Thanks to its unique properties, it is also particularly well suited for the envisioned FL experiments. The benchmark dataset published in 2018 contains acceleration samples recorded in controlled experiments using an Android smartphone placed in the right (50% of the time) and left front trouser pocket of 30 subjects (24 females, 6 men) aged from 18 to 60 years. As a result, the dataset contains 11771 data samples, of which 7579 samples represent human activities and 4192 samples represent falls. Hence, the UniMiB SHAR dataset is well suited to develop and benchmark applications for human activity recognition (HAR) and fall detection. For the above-described planned FL experiments, the UniMiB SHAR dataset is also highly predestined, as the individual data samples can be precisely assigned to each of the 30 subjects. Therefore, the dataset can be used to create a federated learning setting in which clients (i.e., the subjects) have proprietary and independent data. Moreover, the dataset features a granular distinction between different types of human activities and falls. Precisely, the dataset contains 9 (sub)classes describing activities of daily living (ADL) (e.g., falling backwards, falling rightward, falling forward) and 8 classes representing falls (e.g., falling backwards, falling rightward, falling forward). Therefore, by selecting subsets of the available data per client corresponding to only some of the subclasses, the dataset is also suited to simulate the varying distribution of ADL and fall types among the clients in the FL setting described above. For the completeness reasons, we want to mention that the Android smartphone used to record the acceleration data was equipped with a Bosch BMA220 triaxial low-g acceleration sensor. Therefore, the UniMiB SHAR dataset provides measurements of accelerations along each of the three Cartesian axes (x, y, and z direction) at a frequency of 50 Hz.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

ML model: As described above, the two types of relevant models exist in an FL setting. On the one hand, each client administers a local model that is first trained locally and then communicated to a central instance. Subsequently, the central server aggregates all client models to the second relevant model – the global model. By applying the Federated Averaging [70] algorithm, the global model represents the aggregated knowledge of all client models. In our performed FL experiments, we use a deep neural network (DNN) architecture to fulfil the binary classification task. The chosen architecture is similar to the DNNs centrally trained in the context of the robustness experiments (see Section 3.4). Precisely, we train DNNs consisting of five fully connected layers. The first layer represents the input layer equipped with 453 input neurons. The input layer is followed by three fully connected hidden layers consisting of 256, 128, and 64 neurons, respectively. The output layer represents the final layer and is equipped with a single neuron. We use the sigmoid function as the activation function for the output layer, whereas we use the rectified linear unit (ReLU) function to activate the neurons in the hidden layers. The binary cross-entropy is used as the loss function. Furthermore, stochastic gradient descent with a learning rate of 0.2 is applied to minimize the loss. In each of the 40 performed federated communications round, each local model initializes its weights according to the weights of the global model and then trains its local model for a total of 30 epochs before the global model is updated using Federated Averaging [70]. Subsequently, the global model is again propagated to all clients, and the procedure is repeated.

Evaluation criteria: To evaluate the effect of individual clients' varying subclass contributions on the resulting global fall detection model, we will measure the (binary) classification performance of the latter by using standard classification metrics. Precisely, we will investigate the accuracy, precision and recall, as well as metrics relevant to confusion matrices, such as true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). To measure the effect of different amounts of (sub)class contributions per client to the global model, we will distribute all possible combinations of the 9 types of ADL and 8 types of fall subclasses (within the dataset) to the individual clients. This will result in a total of 72 experiments. Besides, we will vary the number of clients contributing to each federated communication round to measure the effect of reduced client availability on the global model performance.

Setup and procedure: In the performed experiments, we used the open-source framework TensorFlow Federated (TFF) (version 0.44.0) [82] to simulate a federated learning setting. Furthermore, we leveraged the UniMiB SHAR dataset described above and separated the data of six randomly chosen subjects for testing purposes. The individual data of the remaining 24 subjects were used for the federated training procedure. Hence, we refer to the subjects as clients in the following. To simulate the varying subclass distribution, we began with modifying each client's data according to two parameters, the number of classes of ADLs and that of falls that each client contributes to the training process. Precisely, for each client, we selected a subset of its data corresponding to a fixed number n of ADLs ($1 \leq n \leq 9$) and a fixed number m of falls ($1 \leq m \leq 8$). The subclasses of ADLs and falls for each client were chosen so that all classes contribute equally to the global training process (i.e., client k, $1 \leq k \leq 24$, contributes ADL classes k to (k+n) modulo 9 and fall classes k to (k+m) modulo 8). With this selection of data, we then created a federated dataset for the simulation of FL. For the simulation, we performed 40 communication rounds (i.e., federated training iterations). In each round, a predefined number k $(1 \leq k \leq 24)$ of clients were randomly selected to participate in the federated training iteration, i.e., for which a local model was trained and propagated to the central server. In the baseline setting, we selected all clients $(k = 24)$ for each communication round. The

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

experiments were also replicated for the parameters $k = 1$ and $k = 8$ clients[1]. In our series of experiments, we simulated FL for each pair of parameters n and m for this fixed number of k clients, i.e., we performed in total 72 experiments. We repeated these experiments three times for statistical purposes. Afterwards, we evaluated the resulting global models according to the metrics described in *Evaluation Criteria* on the test set and averaged the values among the three runs. In order to avoid any impact of the choice of the randomly selected clients, we used the same random selection of clients for all 72 experiments of different values for parameters n and m (but fixed k). All experiments were performed on a MacBook Pro equipped with a 2,6 GHz 6-Core Intel Core i7 processor and 16 GB of DDR4 RAM, and running the macOS 13.0.1 operating system. The described experiments allow us to draw conclusions about the impact of the clients' varying (sub)class contribution to the global FL model. Furthermore, the setting enables us to compare the performance of FL models with different choices for the number of clients that are randomly selected in each communication round to contribute to the training of a local and thereby global model.

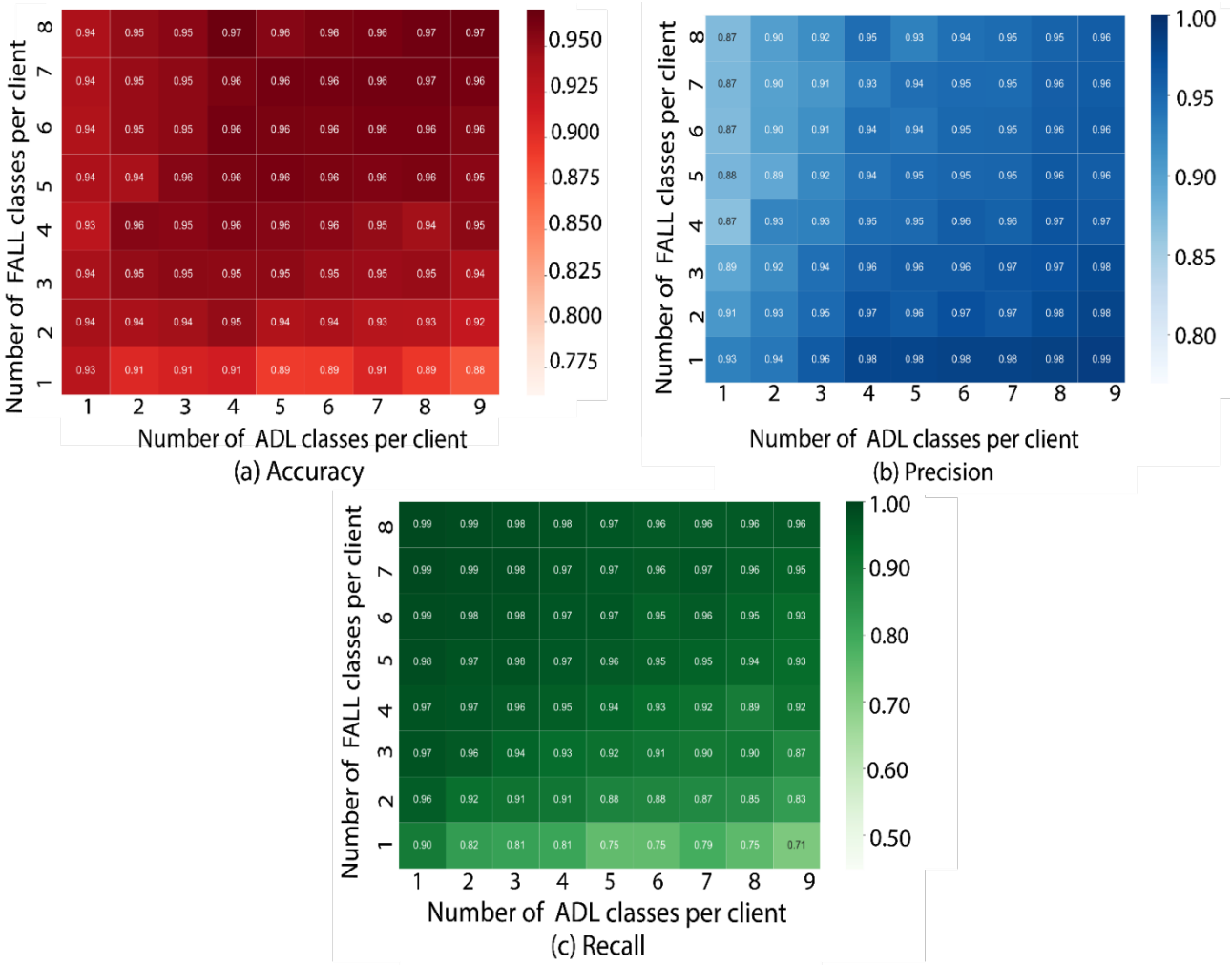


Figure 8: Heatmaps visualizing the effect of the varying (sub)class contributions of clients to the (a) accuracy, (b) precision, and (c) recall of the resulting global models originating from the 72 performed experiments

[1] Due to limited computational and time resources, further experiments with more granular partitioning of k were not feasible to execute in the scope of this deliverable.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

2.2.2 Results

Baseline: We next describe the baseline results of our performed FL experiments. Specifically, we refer to the baseline as the series of experiments in which a client availability of 100% per federated communication round was available. Thus, all 24 clients were involved in each federated training iteration of the global FL model. Figure 8 and Figure 9 show the obtained results in relation to the varying number of ADL and fall subclasses available to each client per individual experiment. As described, the dataset consists of eight types of falls and nine types of ADLs per client/subject, resulting in 72 individual experiments and 72 trained global ML models. For clarity, we refer to the different fall and ADL types synonymously as subclasses of the superclasses fall and ADL. In addition, to verify further our results, we conduct centralized training using the same dataset (See application scenario 3.4- Healthcare-based sensor data)

As we can observe from Figure 8 (a), the accuracy of the trained FL models ranges from 88% to 97%. In general, we can observe that the more subclasses of the two superclasses (fall and ADL) were available per client, the higher the binary classification accuracy to be measured. The accuracy achieves its maximum of 97% when training samples of all subclasses are represented in the clients' training data. However, we can also observe that even a few subclasses of both superclasses per client are sufficient to achieve an acceptable binary classification accuracy. The heatmap illustrated in Figure 8(a) demonstrates that three types of fall and ADL (sub)classes per client are sufficient to achieve an accuracy of more than 95% on average. We consider this result still sufficiently qualified for the fall detection task. In this context, we can also mention that the illustrated results suggest that a balanced contribution of the two superclasses results in higher accuracies. In contrast, we can observe the poorest accuracy results in the bottom row of the heatmap shown in Figure 8 (a). This row illustrates the experiments in which each client has access to only a single type of fall but many ADL subclasses. As we can see, the more the number of ADL subclasses deviates from the single fall subclass, the lower the measured accuracy. The minimum of 88% is reached when only one single fall subclass but all nine ADL subclasses are represented in the clients' training data. We assume that, in this case, the global model specializes in detecting and predicting ADLs due to the high imbalance. Thus, the model is no longer capable of reliably identifying falls. Figure 9 supports this assumption since it shows that the TN metric is maximized for this extreme case, whereas the TP and FN metrics are minimized.

On the contrary, the recall is higher the more fall subclasses each client contributes. This behaviour can be best explained by the metrics presented in Figure 9. As illustrated, the TP, TN, FP, and FN metrics exhibit similar behaviour. We can see that the values for the TP and FP range from 29% to 41% and 0% to 6%, respectively. The results for the TN and FN metrics vary from 53% to 59% and 0% to 12%, respectively. Interestingly, we can notice that as soon as the class distribution gets highly unbalanced (i.e., the number of subclasses of one superclass significantly exceeds that of the others), the corresponding metrics maximize or minimize in the respective direction. In those highly unbalanced cases, the resulting global ML model seems to be primarily able to identify the predominant class and struggles to classify the minority class correctly. Having a look at the respective extreme cases illustrates this behaviour. Let us begin with the extreme case where the number of fall subclasses per client significantly exceeds the available ADL subclasses (see the upper left corners in Figure 9). In this case, we can observe that the associated metrics TP (i.e., correctly classified falls) and FP (i.e., ADLs classified as falls) are maximized with 41% and 6%, respectively. In contrast, the number of TN (i.e., correctly classified ADLs) and FN (i.e., falls classified as ADLs) are minimized with 53% and 0%, respectively.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

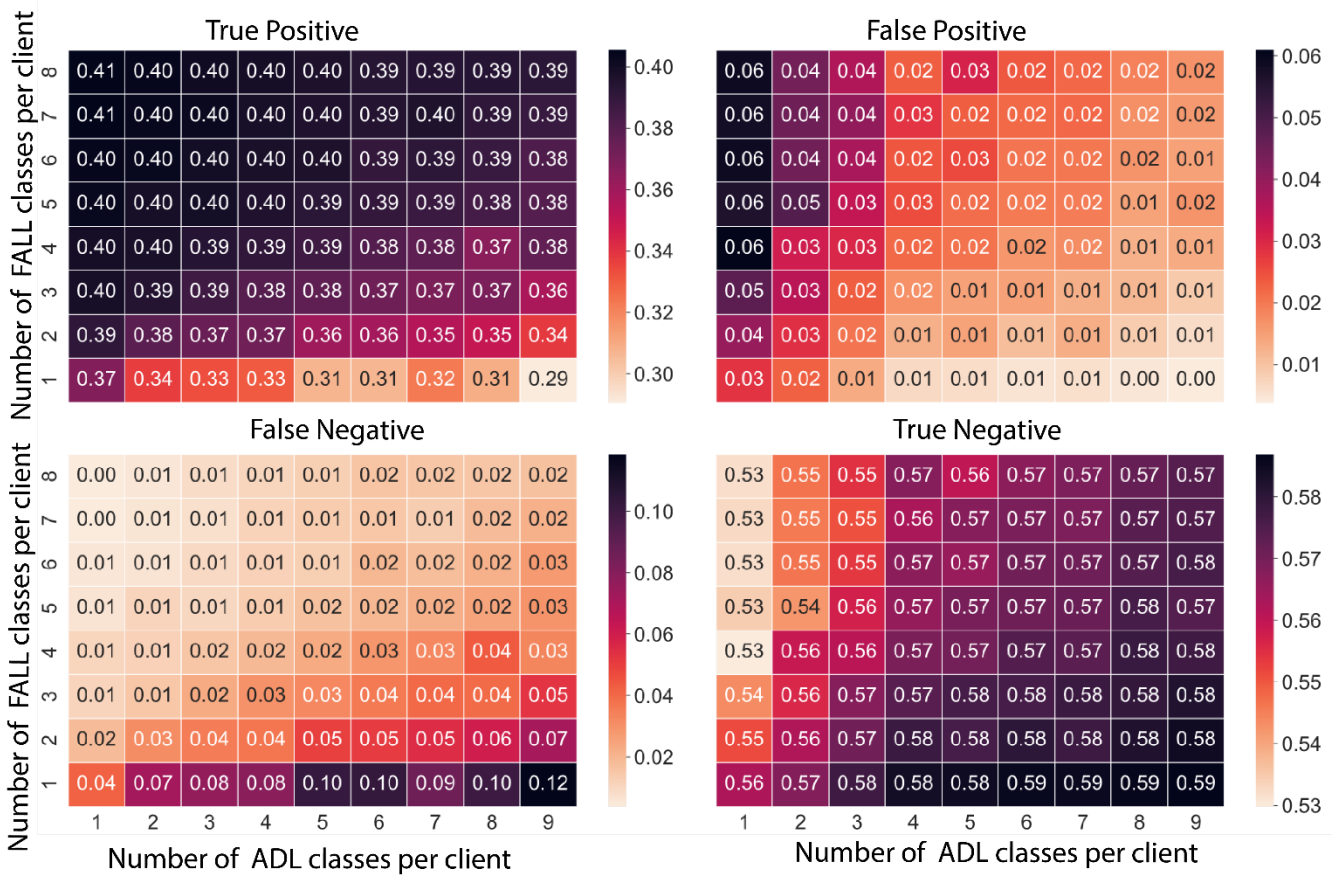


Figure 9: Heatmaps visualizing the effect on different number of clients on performance metrics

It seems that the global model prioritizes falls in this extreme case, which significantly increases ADL misclassification. As expected, the opposite extreme case shows reverse behaviour. When the number of ADL subclasses significantly exceeds the number of fall subclasses (see the lower right corners in Figure 9), TN and FN metrics are maximized with 59% and 12%, respectively. On the contrary, the TP and FP metrics are minimized with 29% and 0%, respectively. In this case, the model now primarily learns to detect ADL classes, which implies it no longer reliably detects falls. Since the precision[2] depends on the TP and FP and the recall[3] on the TP and FN, the above-described behaviour of these two metrics can be explained.

In contrast to the discussed extreme cases, we can also observe from Figure 8 (b), Figure 8 (c), and Figure 9 that the illustrated metrics improve with increasing class balance. The more subclasses of the fall and ADL superclasses are available to a client, the closer the global model approaches the observed optimal value. Although the individual metrics never achieve their observed maximums for the balanced class distributions, ALL metrics exhibit good and acceptable performance results. Therefore, we conclude that the global model performs the binary classification task best when all classes are balanced and as many types of ADL and falls as possible are contributed by each client. Nevertheless, we want to note that, again, already

[2] The precision is defined as *precision = TP / (TP + FP)*

[3] The recall is defined as *recall = TP / (TP + FN)*

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

three ADL and fall subclasses per client are sufficient to obtain excellent and satisfying results comparable to those of the maximum subclass distribution (i.e., 9 ADL and 8 fall subclasses per client).

In summary, we observed only a weak correlation between the investigated classification performance metrics and the respective class distribution in the performed FL experiments. Nevertheless, we conclude that the investigated federated learning models perform better when the amount of data from both classes per client is balanced. In addition, the more the data from each client covers the entire data distribution, the better results were observed during the performed experiments. In this context, a class distribution of at least 3 of 9 ADL subclasses and 3 of 8 fall subclasses among the clients was already sufficient to fulfil the binary classification task adequately. Therefore, we conclude that in the investigated federated learning setting, it is possible to recognize and correctly classify other (and possibly unknown) ADLs and falls from only a few training samples. Moreover, these observations are also promising for practical applications. The results prove that privacy-preserving FL training methods could be used to train good fall detection classifiers even when the underlying data is sparse and non-IID distributed among the clients.

Effect of reduced client availability: Since clients participating in typical decentralized Edge-based FL settings can exhibit heterogeneous system and network properties (e.g., high network latency, connectivity issues, heterogeneous bandwidth, or heterogeneous computational resources) [89, 91], (short-term) failures of individual clients must be expected. Consequently, these clients cannot participate in individual communication rounds in the federated training process or delay it significantly. To counteract this limitation and investigate possible effects on our use case, we also examined the FL models' performance when the number of participating clients per communication round is significantly reduced. In this case, not all clients participate in every communication round of the federated training process, either due to failures, to counteract the characteristics described above, or to save resources (e.g., for the usage of data or computational resources). Our aim was to investigate whether a classification performance comparable to the baseline results can be achieved at reduced client availability. For this purpose, we replicated all experiments discussed in the previous section and varied the number of clients participating in the decentralized training process. Specifically, we examined the cases for eight available but randomly altering clients per communication round as well as the minimum of only one available but randomly varying client per communication round. Thereby, 8 out of 24 clients correspond to a network availability of 33%, whereas 1 out of 24 clients reflects an availability of less than 5%. The difference between the obtained measurement results and the baseline results discussed above is illustrated in Figure 10, Figure 11, Figure 38, and Figure 39 (see Appendix C).

Figure 10 (b) visualizes the variation of the accuracy compared to the baseline results when only eight (varying) clients participate in a single federated communication round. As we can see, FL models trained with only eight participating clients per communication round shows an almost identical classification performance to the baseline results. The confusion matrix shown in Figure 11 (b) also supports this statement. No changes in classification performance are measurable for the illustrated cases with balanced class distribution and high numbers of ADL and fall types per client. Only for the strongly unbalanced class distributions among the clients, some performance declines are to be noted, which, however, are not to be regarded as significant and negligible.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

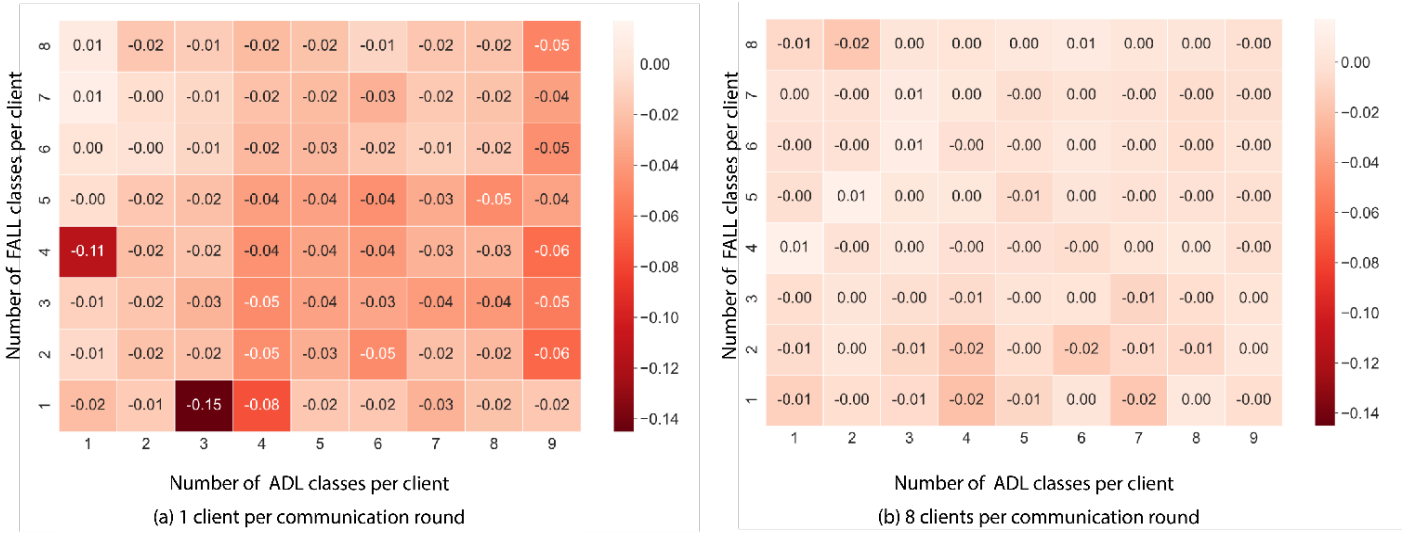


Figure 10: Variation in accuracy for different number of participants in FL

.

In contrast, Figure 10 (a) and Figure 11 shows that for the extreme case of only one participating but altering client in each of the performed communication rounds, the results of the global models sometimes significantly underperform those of the discussed baseline models. We can witness a reduction in accuracy of up to 15%. For the balanced class distribution with a high amount of ADL and fall types (see the upper right corner in Figure 10 (a)), we can still observe a deviation of 5%. The metrics presented in Figure 11 also diverge strongly from the baseline results. Again, we can observe a deviation of up to 15% for the TP and FN metrics. The balanced case's deviation is again 5% compared to the baseline. The deviations for the TN and FP are negligible for all but one of the 72 experiments. In summary, we note that for the client availability of one altering client per communication, the results are worse than those for eight altering clients per communication round. Based on the discussed results, we, therefore, do not recommend for our use case to reduce the number of participating clients per communication round to the minimum of one client (representing less than 5% client availability).

The global FL models trained with a fraction of 8 varying clients per communication round (33% client availability) show a binary classification performance equivalent to the baseline models. In contrast, a significant performance decline could be observed for a client availability of less than 5%. Therefore, we conclude for our use case that based on the obtained results, the number of participating clients per communication rounds can be significantly reduced by up to at least 66% without sacrificing classification performance. In a practical application, this can save costs (i.e., for network, resource, or data usage) as well as counteract the limitations mentioned above. Further studies would be necessary to investigate whether this lower limit can be further reduced. However, due to restricted computational and time resources, such experiments could not be performed within the scope of this deliverable.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

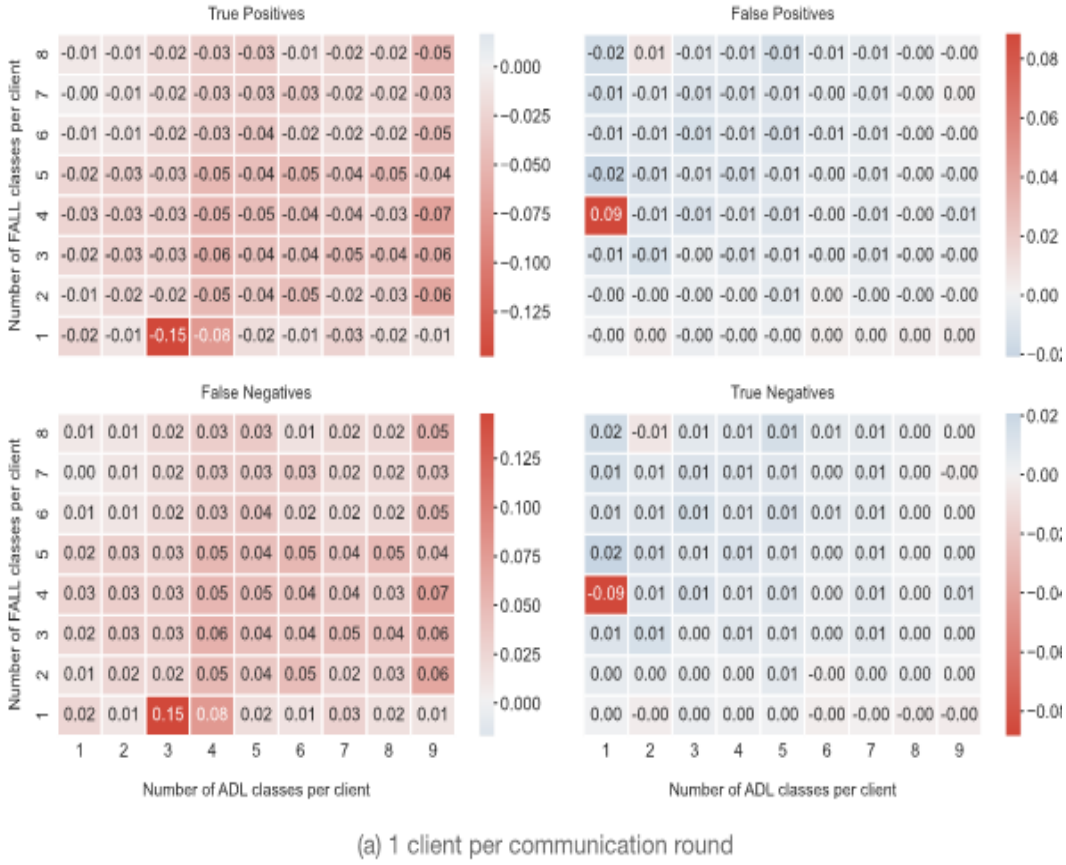


Figure 11: Modified confusion matrices visualizing the deviation of the TP, FP, TN, and FN metrics compared to the baseline results when only a single client participates in a single federated communication round. Negative values indicate a decline in the respective metric. See Appendix A for the confusion matrices visualizing the deviation of the TP, FP, TN, and FN metrics compared to the baseline results when just eight (8) client participates in a communication round.

Summary: We investigated the effect of the varying distribution of ADL and fall (sub)classes - among federated clients with respect to the binary classification capabilities of a fall detection model trained in a FL setting. In addition, we investigated the effect of reduced client availability in the federated training rounds on the classification performance of the global FL model. Our obtained results suggest that there is only a weak correlation between the examined classification performance metrics and the varying (sub)class distribution available to each participating client. Based on our results, we conclude that the investigated federated learning models perform better when the amount of data from both classes per client is balanced. Furthermore, we observed that the more the data from each client covers the entire data distribution, the better the results. Interestingly, the binary classification task can already be performed sufficiently well when each client has access to at least 3 out of 9 ADL types and at least 3 out of 8 fall types. This suggests that privacy-preserving FL training methods can be used to train good fall detection classifiers even if the underlying data is sparse and non-IID distributed among the clients. In addition to this, we report the training results of the model using a centralized architecture (See section 3.4). Both results are comparable in terms of robustness and performance. Hence, the decentralized trained FL models can perform similarly to the

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

centralized trained ML models in our examined scenario. Besides, we found that the fraction of clients participating in each communication round of the federated training procedure can be reduced by at least 66% (representing 8 of 24 clients) while maintaining high binary classification performance equivalent to the baseline setting. Thus, by reducing the number of participating clients, we can save costs (e.g., for network, resource, or data usage) and address potential network and resource limitations that can occur in typical Edge-based federated learning settings (e.g., latency, connectivity issues, or heterogeneous bandwidth) [91,89] without suffering any performance losses.

2.3 RELATION BETWEEN MODEL PERFORMANCE AND TRAINING CONFIGURATION OF DISTRIBUTED DEVICES.

We next analyze the relation between model performance and the configuration of distributed devices used for training. We rely on our FLaaS framework [FLAAS] [94] to conduct benchmarks that explore this effect of this trade-off as AI models are built. Based on that, we also conducted a user-study to assess FLaaS system overhead and ML performance while computing FL projects, in both client devices and server.

2.3.1 Experimental setup

Apparatus: we created three simple Android apps (named Red, Green and Blue or RGB). Each app includes a FLaaS Library v1.0.0 for supporting system functions and to demonstrate the simplicity and efficiency of a third-party app registering with a FLaaS.

Dataset: The dataset for the ML model training and testing is $CIFAR$-10 [95], a benchmark dataset commonly employed by FL researchers. We do not anticipate drastic changes in system overhead if we use other datasets. This dataset consists of $50k$ training samples and $10k$ test samples of images (32×32×3) representing one of 10 types of objects, or classes. We partition the training set into subsets of 150 samples, and each subset is assigned to an instance of a third-party app (Red, Green and Blue) to be installed on a user device. Two data distributions are constructed from $CIFAR$-10 data: i) Independent and Identically Distributed (IID), where an app has samples of all 10 classes; ii) Non-IID, where an app has samples only from two random classes.

ML model and experiments: As a base DNN model we use MobileNetV2 [96] , which is pre-trained with ImageNet [97] dataset with image size of 224x224. As a head of the DNN (used for model personalization on user local data), we use a single dense layer with 32 neurons and $RELU$ activation function, followed by $so\ f\ tmax$ activation function (SGD optimizer with 0.001 learning rate, 0.0001 kernel regularizer, and 0.0001 for bias regularizer). Furthermore, we applied typical parameter values used in other $F\ L$ works with $CIFAR$-10: 20 samples per batch and 20 epochs per batch. We experimented with 10 up to 30 FL rounds.

We use three different strategies to train models in client devices:

- **Single-app (SA)**. One model per app
- **Joint samples (JS)**. Data shared between apps.
- **Joint Models (JM)**. Model trained on data, then model shared between apps.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Metrics used to analyze performance: We measure the performance of the ML model trained in FL fashion with FLaaS on client devices, by computing Accuracy when testing the global FL model on 1000 samples of $CIFAR\text{-}10$ test set. This ML performance is measured on FL global models built in ideal scenarios (via simulation) when all user devices would participate in all FL modeling rounds, and compared to real scenarios with real user devices freely joining and completing, or dropping their FL rounds depending on their actual phone usage. The simulation environment uses Keras TensorFlow v2.6.0 under Python v3.8.12. It replicates the system implementation but depends on a JSON-based input for instructing the total rounds and user participation per round.

2.3.2 Results

Accuracy: Figure 12 shows the accuracy results achieved per FL round, for different experimental configurations: Joint Samples (left) vs. Joint Models (center) with IID vs. NonIID included. show that with a sufficient number of client devices joining an FL project (e.g., a few tens of devices), FLaaS can produce a useful model. In particular, even though the number of user devices joining and delivering ML models is much lower than the ideal, the model trained performs comparatively well. We also note that IID data lead to ML models with 36.43% better accuracy, on average, than NonIID data, at the end of the $10th$ round. However, this is to be expected, as it has also been found in other FL studies. NonIID data provide a more realistic view of how data would be distributed across users, since real users cannot.

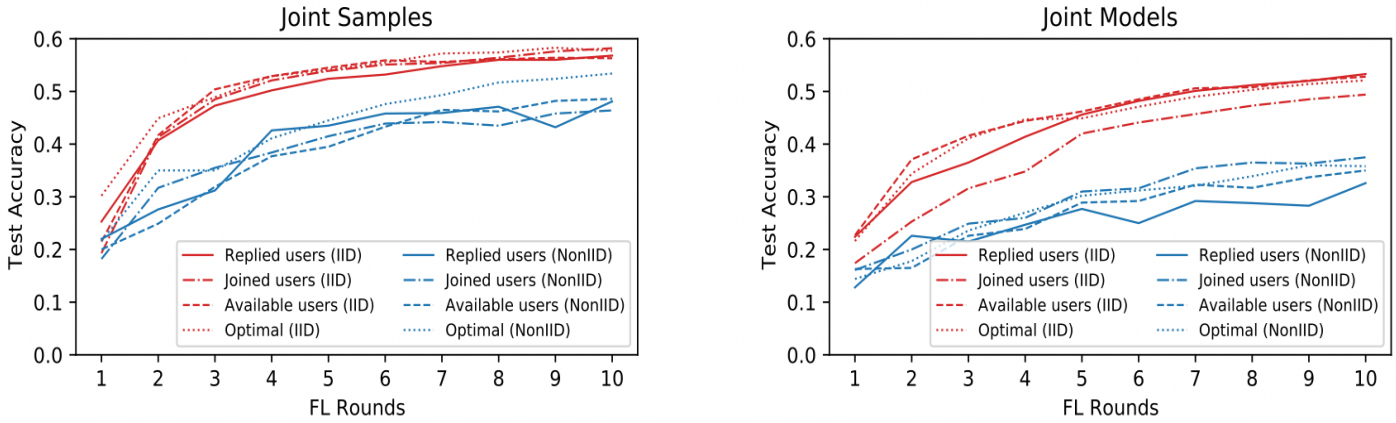


Figure 12: Test accuracy achieved per FL round, for different experimental configurations: joint samples (left) vs. Joint models (center) with iid vs. Non iid included.

Analyzing the right plot in the same figure, we note a higher ML performance of models trained by building them while sharing samples with FLaaS Local (i.e., JS), vs. training individual models, one per third-party app, and then averaging them at FLaaS Local (i.e., $J\ M$). In fact, JS mode leads to a model with 6.57% higher test accuracy than $J\ M$ mode for IID data, and 32.22% higher for NonIID data, at the end of the $10th$ round. This improved performance is expected since sharing data with FLaaS Local allows it to train the joint model on 3× more data and thus optimize it better, than averaging models from the RGB apps, trained on 1/3 of samples. Similar

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

improvements in test accuracy (10.53% for IID and 37.5% for NonIID) are also observed when comparing joint to single-app modeling (results are not shown due to space limitation).

Device cost: In Figure 13 we study the Cumulative Distribution Function (CDF) of total time taken to complete an FL round, for the joint modeling types (JS vs. JM). This duration incorporates potential pauses of the FL training process, if, for example the device OS put to

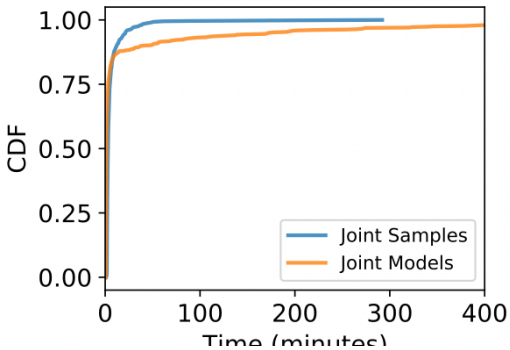


Figure 13: Total duration of FL round, across projects using JS or JM mode

sleep the ML training process. Interestingly, the great majority of FL rounds, in both JS and $J\ M$ modes have a short duration (Figure 13) the $80th$ percentile duration of an FL round is 6.15 minutes for JS and 4.65 minutes for $J\ M$. This means FLaaS or other FL-based ML systems can train full FL rounds in short periods of time. However, there are straggler devices needing hundreds of minutes to complete the same FL tasks, creating a long tail distribution. In fact, the $90th$($99th$) percentile for the two modes is 12.1(48.7) minutes for JS and 40.3 (2527.3) minutes for JM.

Figure 14 shows an average time taken to perform different functions related to participation in an FL project using JS (left) or JM (right) mode, we measure average time needed per FLaaS related device function. Note that the plots are independent of data distribution (IID vs. NonIID), but dependent on type of joint modeling (JS vs. $J\ M$). We find that the most expensive function is to load the data on device and prepare them for ML training, with median time 1.94 and 0.80 minutes, for JS and $J\ M$, respectively. The rest of functions are very fast: median time for both JS and $J\ M$ being 0.01 minutes when joining a round, and 0.02 minutes when downloading the global model parameters (weights). Finally, there is a difference when conducting ML training for JS and $J\ M$, with a median of 0.39 and 0.18 minutes, respectively.

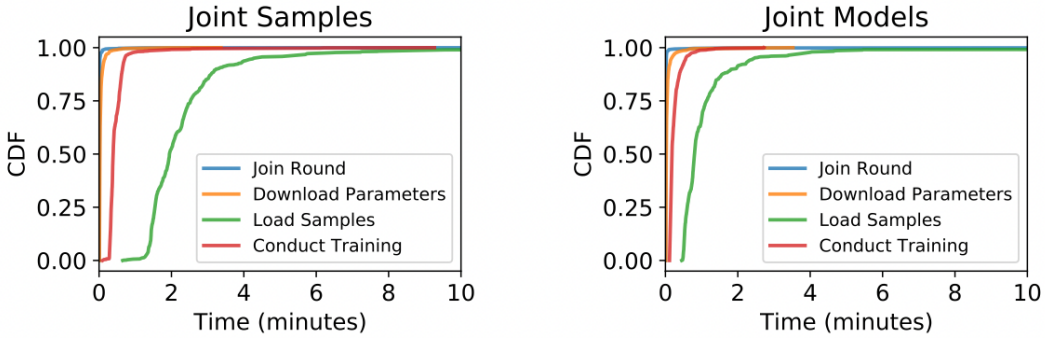


Figure 14: Average time taken to perform different functions related to participation in an FL project using JS (left) or JM (right) mode.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

In-lab test devices: Next, we dive deeper into the cost of FLaaS FL training function on a mobile device by performing several experiments with three popular Google Pixel devices (specs shown in Table 2) as test devices in a controlled setting: the devices are forced to perform the ML task on IID data, from beginning (received notification of new ML task) until the end (delivery of ML model to FLaaS Server) without any breaks. All devices were updated to the latest version of the OS (Android 12) and with deactivated OS-related automated process that would influence the measurements (i.e., automatic updates, adaptive battery, and brightness). During the evaluation, all devices are connected to a stable 5GHz WiFi network while CPU and Battery Discharge data are collected using the Android Device Bridge (ADB) tool. All RGB apps are whitelisted (i.e., set to Unrestricted Battery usage) aiming to measure the most optimal scenario where all apps are responsive to the ML training requests, without further scheduling delays introduced by Android's Work Manager.

Table 2. Device Specification (CPU and Memory)

Model	Device Specification
Pixel 3a (P3a)	Octa-core (2x2.0 & 6x1.7 GHz), 4GB RAM
Pixel 4 (P4)	Octa-core (1x2.84, 3x2.42 & 4x1.78 GHz), 4GB RAM
Pixel 5 (P5)	Octa-core (1x2.4, 1x2.2 & 6x1.8 GHz), 8GB RAM

Table 3. Cost for on-device ML training: Training duration, CPU usage, and Power discharge. Means and standard deviation (SD) computed on distribution of each metric on each device and modeling mode, over 10 rounds

	Duration (sec)		CPU usage (%)		Power Discharge (mAh)	
	JS Mean (SD)	JM Mean (SD)	JS Mean (SD)	JM Mean (SD)	JS Mean (SD)	JM Mean (SD)
P3a	167.3(13.1)	85.3 (1.5)	17.9 (2.1)	29.7 (0.6)	56.8 (8.6)	37.7 (0.5)
P4	117.1 (1.1)	62.1 (0.8)	15.40 (0.1)	27.8 (0.4)	41.3 (0.9)	29.7 (0.5)
P5	115.0 (0.4)	62.8 (0.9)	16.1 (0.1)	28.8 (0.4)	42.7 (1.0)	32.4 (0.7)

Table 3 shows the average time taken for a device to perform the ML training task, for the three test devices and two modes of training (JS vs. $J\ M$) with a total of 450 samples. On the one hand, we notice that the average time of the older device (Pixel 3a) is always higher than the other two, newer devices. This is true regardless of the modeling mode. Also, and as expected, the JS is slower than $J\ M$ since apps have to first transfer data to FLaaS Local, and then Local to train the model on them. On the other hand, $J\ M$ builds 3 individual models, even in parallel, if the OS's Work Manager supports it. Interestingly, this parallelized execution in $J\ M$ is reflected in the results for CPU usage, which is, on average, 74.7% higher on $J\ M$ than JS, across devices. Finally, the reduced execution time, even at higher CPU usage, leads to overall reduction in consumed power from $J\ M$: 41% lower than JS.

Summary: Overall, a regular user device is expected to spend small amount of time for completing the FL round: a median of 131.7s and 69.6s in optimal case (test devices) and 178.2s

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

and 85.2s in regular case (real users), for JS and $J\ M$, respectively. Notably, there are straggler devices that may require significantly more time to complete the same task. This means a practical FL system needs to account for such behavior and stop waiting for these devices very early on in the FL round. Also, the costliest function in the process is loading data for the ML training. Future work could further optimize this step. Further, JS, in comparison to $J\ M$, requires 90.00% more execution time, but uses 74.68% less CPU. However, it requires 40.96% more power to process the same number of samples but in 1 app (FLaaS Local). Finally, older devices (e.g., Pixel 3a) require more execution time (44.2% and 36.6% increase) and more power (35.2% and 21.7% increase) to perform the same ML task in JS and JM, respectively.

3. DETECTION OF DATA QUALITY ISSUES IN DISTRIBUTED MACHINE LEARNING APPLICATIONS

Since distributed devices can contribute towards the training of robust AI models, in this section, we next analyze how data quality issues can be detected in distributed environments. To do this, we describe different applications scenarios that rely on AI to support decision making. We consider four different application scenarios: 1) autonomous drones, 2) healthcare using images, 3) autonomous vehicles and 4) healthcare using sensor data. In the following section, we describe each of the scenarios in more details.

3.1. AUTONOMOUS DRONES

Application scenario: The integration of AI into drones is critical for enabling autonomous operations that require minimal or no human intervention. Indeed, AI is essential for navigation, trajectory estimation and localization [98] to mention some examples. As these techniques have matured, drone applications that automate our daily life activities have become a reality, e.g., delivery of food or medicine and applications that harness drones for environmental purposes, e.g., air quality monitoring [99], litter detection and separation [100], and water pollution monitoring [101].

City-scale deployments of autonomous drones, such as ground drones, cars or other vehicles, aerial drones, or even aquatic drones, are only possible if the operations of the drones are trustworthy, i.e., they can be guaranteed to operate safely without causing harm to the citizens, the urban infrastructure, or the environment. Modern autonomous drones largely rely on deep learning models which are black-box in nature, making it difficult to verify the decisions that are made [102]. In this scenario, we consider litter recognition on autonomous drones as a representative example of drone applications that rely on AI. We rely on performance metrics based on DCPI metric to analyze whether it is possible to observe changes in data quality while training.

Dataset and model: We perform the experiment using the TrashNet litter classification dataset which consists of 2527 litter images [103]. We rely on this dataset as it contains a large number of real-world images which makes it possible to analyze different environments and contexts for litter classification. We evaluate the training of a model in both centralized and decentralized manners. We transform the training of a CNN model from a centralized to a distributed manner. We transform TrashNet [102] model for litter recognition into a FL model. We chose the most optimal model for classification that is reported by TrashNet. The CNN model consists of 3 convolution layers with 3 x 3 kernel size and LeakyReLU activation function, each layer is

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

followed by a MAX-POOLING layer with the size of 2 times 2. A dense layer with ReLU and a softmax output layer are added at the end to build a CNN classifier. We use the Adam optimizer and set the learning rate to 0.005.

Adversary model and assumptions: To evaluate how the quality of data hampers model inference, we analyze how the performance of model is degraded as data is poisoned. We consider a data poisoning setup where the adversary has control over the input data that is used for training on one or more compromised client devices participating in the FL process. The attacker can poison the local training data of these compromised devices, but we assume that the adversary cannot modify the local models after they are trained (i.e., model poisoning) nor can they tamper with the performance monitoring of the compromised devices. This assumption can be enforced, e.g., by operating model training and performance monitoring inside a trusted execution environment [104], which would send local models and performance metrics over a secure channel directly to the FL server. Thus, the only way for the adversary to poison the local model is through poisoning the data. Similarly, we assume the FL server is not compromised. It performs a secure aggregation[105] that prevents it from analyzing the local model updates it receives, thereby protecting client's privacy.

Analysis pipeline: The performance metrics from when a device is training a model are analyzed using DPCI metrics. We developed a pipeline, called MODENA [90] that implements this metrics to identify possible attacks on data quality. Figure 15 depicts a high level overview of the pipeline used to analyze model performance with different types of data quality. As compromised data (data poisoning) enters the FL process, they eventually become integrated with the global model which is then propagated to all contributing devices. The individual devices keep track of the model's execution parameters before the model is updated and compare changes in the execution parameters after model updates have been carried out.

The model diagnostics of MODENA operate in three phases that are executed continuously. Below the phases are presented. In the first phase, performance metrics (CPU measurements) are collected while the device is training the model.

In the second phase, the point detection algorithms are applied to characterize the training process. The key insight is that from these characterizations is possible to establish fingerprints of training, such that later by looking at the changes in fingerprints, it is possible to detect whether training data was hampered or not.

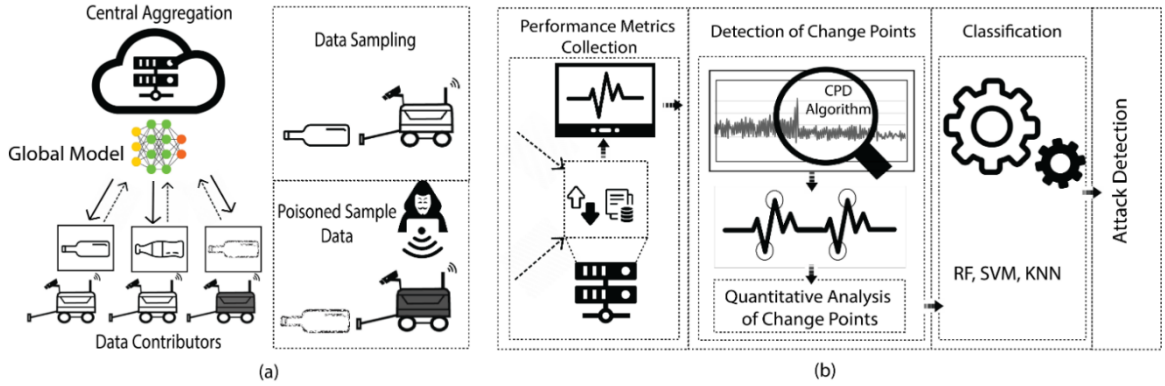


Figure 15: The pipeline showing: a) FL training (poisoned device - black AGV), b) Detection phases used in our pipeline

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Data quality trade-offs: To introduce changes in data quality, we consider four types of poisoning attacks, i.e., blurring [106], label flipping [107], occlusion [108], and steganography [107]. Blurring and occlusion reduce the informativeness of the data, making it low quality for proper class learning. Similarly, steganography potentially adds additional information to the binary descriptors of the data, which causes the model to deviate from the ground truth during prediction. Finally, label flipping is designed to confuse the model to perform wrong predictions from misleading labels. These attacks are selected as they can cause the AGVs to violate traffic signals (label flipping), misidentify targets in urban infrastructure (blurring and occlusion) and disrupt AGV's computing resources due to the overhead in processing the extra information (steganography). We perform these attacks incrementally to show their influence on model performance.

Procedure: We consider two testbeds which contain different types and amounts of devices, but which use the same generic procedure for experiments. In our experiments, we vary the number of devices that are compromised (from a single device up to all devices, i.e., ten or four, depending on testbed), the amount of data that is poisoned per device (5% to 30% in 5% increments), and the attack type (blurring, occlusion, steganography, and label flipping). To achieve this, we partition the available data into equal-sized partitions so that each device has access to one partition during each training round. For a given number of devices, we then poison the given fraction of data (5% to 30%) using the specific attack. This process is then repeated for all possible parameter combinations.

Testbed 1: FL aggregates contributions from multiple devices and thus we need to consider attacks where multiple devices are simultaneously compromised. We assess the impact of multiple devices using a virtual testbed that considers ten (virtual) devices that are implemented as threads on a single machine. Once each of ten devices has updated its local model, the parameters are aggregated by a separate aggregator thread and this constitutes one round of model training.

Testbed 2: is designed for practical deployments of FL and to assess its performance in a realistic environment, we develop a real-world testbed, consisting of four AGV nodes and a centralized server (Ubuntu 20.08, 4 GB RAM and 4 cores CPU with a storage capacity of 20 GB). We use Raspberry Pis (Raspberry Pis 4B model with 16 GB memory and 4 GB CPU) as they are the most common processing platforms used by commercial AGVs [109]. We use the performance of the FL model after 5 rounds as a reference point for comparison as it provided the same performance as the baseline. Performance metrics from the devices are collected using TIC stack. Telegraf is the data acquisition component, InfluxDB is the time series database and Chronograph is the visualization dashboard deployed in the form of a docker container in the Raspberry PI. The endpoint of Chronograph can be accessed to generate CSV files as time series data with intended matrices such as CPU usage and memory. Simultaneously we collect the percentage of used memory and CPU usage and log them in CSV form locally on the device. Our testbed is connected to a local router, emulating a base station that all devices would connect to. The latency to the server was stable as it was located in a private cloud (average RTT=102.5ms). Each training iteration took approximately 12.28 minutes on the devices, after that the server aggregates the global model and updates the models running on all devices. This process took on average, 56.4 minutes. We implement the FL procedure and the quantized CNN model using the Java-based deep learning library Deeplearning4j.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

3.2. HEALTHCARE-BASED IMAGING

Dataset and application scenario: We used a modification of the "NCT-CRC-HE-100K" and "CRC-VAL-HE-7K" data sets [110].This dataset contains a total of 107180 non-overlapping images of stained histological images of human colorectal cancer and normal tissues. All images are 224x224x3 shaped. Images of the tissues are divided into 9 classes: adipose, background, debris, lymphocytes, mucus, smooth muscle, normal colon mucosa, cancer-associated stroma, and colorectal adenocarcinoma epithelium. [110]. Modification of the dataset is part of the MedMNIST v2 collection of biomedical images [111]. This collection is inspired by the traditional MNIST dataset, which is widely used for assessment and universal comparison of the accuracy of the classifiers [112]. MedMNIST v2 contains 12 datasets for 2D and 6 sets for 3D classification. The key features of the proposed collection are:

- Diversity: Contains different sizes of data from 100 to 100000 and enables both binary and multiclass classification,
- Standardization: All sets are pre-processed in the same way which requires no additional background knowledge for users,
- Lightweight: All images are 28x28 for 2D and 28x28x28 for 3D, which is MNISTlike and enables shorter training with no great impact on accuracy. This resolution is also great for proof of concept and verification of the algorithm before going with the bigger resolution, and
- Education: The collection contains datasets from various research and with its CC License it enables education to the wider scientific circles. [111]

The dataset derived from the "NCT-CRC-HE-100K" and "CRC-VAL-HE-7K" data sets [110]- is called PathMNIST. This dataset does not contain NA values and the distribution of samples by classes ranges from 7886 and 12885 which hints about a slight imbalance of data that could impact our learning process. The "NCT-CRC-HE-100K" dataset is spit in training and validation set in ratio 9:1, while CRC-VAL-HE-7K is used as a test set.

Methodology and experiments: The goal of this section is to investigate the impact of data bias in distributed machine learning. Distributed machine learning is considered to be a very efficient way of training models in today's world with data distributed across multiple devices and servers. One branch of distributed learning, called Federated Learning, is quite susceptible to discrimination precisely because of the heterogeneous distribution of data among devices and the possible presence of different types of preferences.

We imitate the usual healthcare system with data distributed across multiple hospitals and clinical centers. In the case of the implementation of a machine learning algorithm for predicting the course of the disease in the central hospital, the model devices for making the central unique model. The process will be continued as long as we do not achieve the desired accuracy. Given the differences in population, demographics, customs, and prejudices, local data may contain inequities that can be transferred to the global model through learning and thus result in discrimination and violation of human rights.

All simulation performed in this article is done using Python 3.9, while the machine learning part is mostly done using TensorFlow 2.0 and sklearn libraries. [111,112,113].

Federated learning was simulated on 5 workers and a centralized model was used for the prediction and assessment of models on the test set, which was not available to local models. The federated learning is not performed in parallel which does not affect the results of the

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

experiment only the time of execution. The model used for image classification is ResNet neural network [116] without the last layer which was previously trained on the ImageNet dataset [111]. On the exit of ResNet50 were placed Flatten, two Dense, and two Dropout layers for 9-class classification. Because of the requirements, images were padded with zeros to reach 32x32x3 resolution, which is the minimum allowed resolution for ResNet. The code structure for FL simulation was inspired by [117] . The first training was done centralized on the training dataset for the assessment of the quality of the data, and because of the result classes 2 and 7 were excluded from the simulation because of their significant impact on the accuracy of the model. F1 scores in these two classes were under 0.58. To avoid imbalanced data, all classes were undersampled on the size of class 6 which is 7886 and then divided into 5 smaller datasets to simulate federated learning.

Learning was performed in 50 epochs using 64 batch size. The loss function was SparseCategoricalCrossentropy, Accuracy, and Adam optimizer with learning rate decay with decay steps 1000 and decay rate 0.96. Also, to avoid overfitting EarlyStopping was included using the validation dataset with a patience of 10 steps.

First Federated learning was performed without induction of bias and these results are used as a benchmark for discussion and comparison with other bias-related results.

Four scenarios were covered in this simulation:

1) undersampling of the best-performing class,
2) undersampling of the worst-performing class,
3) mislabeling one entire class,
4) inducing noise to compromise learning.

Data bias was, firstly, induced in one worker and after that increased up to 4 to examine the possibility of aggregation method to smooth these changes in global model.

3.3. AUTONOMOUS VEHICLES

Application scenario: Object classification and detection models trained over an autonomous driving dataset can inherit or generate biased models. Detecting such biases in a neural network is a challenging task, which requires a study with several datasets that combine different geographic, visual (traffic signs), and weather conditions. The scope of this experiment is to examine biases in the AI models trained/transfer-learning/validated for 3d object detection/classification and segmentation in the context of autonomous driving. As mentioned, previously these datasets are finite and are non-representative (typical or characteristic) as they do not comprehensively cover all possible detection settings (vehicle categories, viewpoint-angles, random distribution of the same class), and regions (rural/urban/variations with socio-economic regions in the city) in fair ratios.

In this work, we consider two different open-source datasets:

- **nuScenes**: nuScenes (Caesar H. et al, 2020) is a large-scale dataset for autonomous driving. It consists of 1000 driving video scenes collected from the cities of Boston and Singapore, two cities with dense traffic and challenging driving situations. It contains instances obtained from several sensors such as (camera + LiDar + Radar). As the dataset is extensive and consists of several classes, a split approach is used to train and test the model. Classes such as bicycle, bus, car, construction vehicle,

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

motorcycle, and pedestrian are used for measuring selectivity. Since the nuScenes dataset consists of 3D bounding box annotations, to have a comparative study, they are transformed into 2D bounding boxes before being utilized on the BIRANet architecture. The model performance is recorded for class: car and pedestrians [118].

- **Biased-Cars**: Biased-cars [119] is a synthetic dataset that consists of approximately 30000 images of five different car models with different colors in outdoor scenes[119]. It further includes different combinations for the viewpoint, which allows out of distribution generalization by studying the measure metrics. It provides labels for car model, color, viewpoint and scale. It also includes semantic label maps for background categories including road, sky, pavement, pedestrians, trees and buildings. The data generator ensures photo-realism by using physically based rendering.

Methodology: Object detection and classification tasks in vehicular datasets is performed using convolutional neural network, deep neural network and recurrent neural networks. In these neural networks some functions/filters/channels perform better than the others, and the reason behind this performance is unknown, as neural networks are generally described as black-box models. In general, the success or performance of these neural networks is measured on the basis of accuracy achieved and the loss function against a validation set/sample of the images or test data.

We aim to modify training data diversity to perform an empirical analysis it might cause during the inference phase. As the considered datasets are exclusive with different distributions and labels for classes, an independent approach is used for varying data diversity in the datasets. The biased car dataset can be separated into category and viewpoint combinations for each class. For a fair comparison, the number of images is kept constant during the training of all models.

Experiments are carried out by alternatively varying the category and viewpoint combinations for these classes. This process is carried out to check the selectivity of neurons and the model's accuracy, which is also very useful for understanding the model's generalization ability over an unseen dataset.

To vary data diversity in the nuScenes dataset using existing annotations, AI model BiraNet in [118] is trained in different conditions, which include day, night and rain situations [118].

To identify learning biases, we evaluate neuronal activity to provide information about the learning process of the neurons or layer over the object feature. Selectivity of neurons towards the *class-colour and class-label* have been investigated to understand the model learning ability against the input class/object [120].

Here selectivity can be defined as a measure of identifying the image and features that are not transformations of each other. If a neuron is activated for a particular class category, it can be assumed that it will have the maximum sum for this particular category. This property is expressed as the preferred category for the neuron for learning [121].

The selectivity score $S(m,n)$ thus computes the difference between the average activity $(\bar{X}_m^n)$ in the neuron for the preferred category with the average activity of the remaining categories $(\bar{X}_m^{-n})$.

In this document, the selectivity score has been used and integrated with the DNN models, being expressed as:

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

$$S(m,n) = \frac{\bar{X}_m^n - \bar{X}_m^{-n}}{\bar{X}_m^n + \bar{X}_m^{-n}}$$

The value of the *Selectivity score*: score $S(m,n)$ is generally measured between (0, 1). If the measure is 1, the neuron is *active* for a single class, and if the measured value is 0, the neuron is *identically active* for all category-classes.

Experiments: For each class, we will explore the relationship between the data and the learning performance using several metrics (selectivity score, specialization score and invariance). We take samples from MNIST rotation and biased car dataset (categorically distributed) for given variance and use architecture (Squeezenet, DenseNet, ReseNet) to understand the learning performance with variations.

All three models were first trained on the biased-cars dataset, which was followed by training on the nuScenes dataset for the car and pedestrian classes, as mentioned in dataset section. For a fair comparison of all models on the biased-car dataset, the models adapted to a common development environment using the PyTorch framework and are trained using the same loss function, ReLu activation function, learning rate, and optimizer. Adam optimizer with a learning rate of 0.001 is used for training these architectures, with a cross-entropy loss, which can be expressed as:

$$L_{ce} = -\sum_{i}^{j} \mathcal{P}(x_i)\log(q(x_i))$$

Here $\mathcal{P}$ is the probability distribution of the object label, q is the prediction, and j represents the number of samples present. The models are trained with 3000 train images, 390 validation, and 750 test images. All models are trained for 150 epochs.

As the nuScenes dataset is relatively large and consists of several test images, the approach proposed by [122] is adapted to train the models: the model is trained for 300 epochs with the learning rate and optimizer mentioned above.

In the case of the biased-cars dataset, data diversity with respect to distribution combinations has been considered to train the AI model, which allows capturing the selectivity and model performance measure during the model training.

3.4. HEALTHCARE-BASED SENSOR DATA

Application scenario: Considering ageing European society, the prompt detection of medical emergencies and immediate initiation of emergency medical assistance are becoming increasingly important. The faster an emergency is detected, and medical assistance is initiated, the more likely the patient is to survive. In this context, the advancing digitalization and the increasingly widespread availability of near-body IoT sensors can significantly contribute to the automated detection of emergencies. Therefore, in Scenario 4, we want to investigate the autonomous detection of emergencies based on IoT sensor data. Precisely, we aim to train and evaluate models for multivariate time series classification capable of detecting falls based on triangular acceleration data. However, in real-world scenarios, IoT sensor data used to train such models is often collected from untrusted sources (e.g., crowdsourcing or the Internet). Therefore, we also want to examine and compare the robustness of multivariate time series

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

classifiers with varying complexity against label flipping attacks, a specific variant of data poisoning attacks. Finally, we additionally want to investigate the effectiveness and performance of corresponding defence strategies against the applied attack. The deployment of such fall detection classifiers for early fall detection is a relevant application, especially for elderly people. Due to age or diseases, elderly people often suffer from motoric weaknesses such as gait, balance or coordination disorders [123]. Consequently, falls are a frequent cause of injury and even death in this part of the population [122,123]. In this context, the classifiers investigated in Scenario 4 could be employed to autonomously detect falls and then initiate medical assistance via digital emergency communication systems.

Dataset and model: For the planned robustness experiments, we will leverage the publicly available UniMiB SHAR dataset [92]. The UniMiB SHAR dataset is a benchmark dataset presented in 2018 that is used for human activity recognition (HAR) and fall detection. The dataset includes 11771 acceleration samples performed by 30 subjects (24 females, 6 men) aged from 18 to 60 years. The available data samples contain recorded examples for both human activities (7579 samples) and falls (4192 samples). Precisely, the dataset contains 9 classes describing activities of daily living (ADL) (e.g., standing up from laying, running, jumping, sitting down) and 8 classes representing falls (e.g., falling backwards, falling rightward, falling forward). The individual acceleration samples were recorded in controlled experiments using an Android smartphone placed in the right (50% of the time) and left front trouser pocket of the 30 subjects. The used Android smartphone was equipped with a Bosch BMA220 triaxial low-g acceleration sensor. As a result, the provided dataset contains measurements of accelerations along each of the three Cartesian axes (x, y, and z direction) at a frequency of 50 Hz.

We use the dataset to train and evaluate binary classifiers capable of using the provided triaxial acceleration data to distinguish between ADL and falls, thus, detecting falls. Precisely, we train, evaluate, and compare ML models with varying complexity, namely Logistic Regression (LR), Random Forest (RF), Multilayer Perceptron (MLP), Deep Neural Network (DNN), and Decision Tree (DT) models. To implement the DNN, we leveraged the TensorFlow library [114]. After performing the grid search approach to optimize the DNN architecture, we choose a DNN with an input layer of 453 neurons and an output layer with a single neuron. In addition, the DNN was equipped with three hidden layers consisting of 256, 128, and 64 neurons, respectively. The input layer and hidden layers applied the ReLu activation function, whereas the output layer applied the sigmoid activation function for the binary classification task. During the training process, we choose a batch size of 16, a learning rate of 0.001, and set the number of training epochs to 100. Furthermore, the DNNs were optimized with the ADAM optimizer [126] using the binary cross-entropy as the loss function. The LR, MLP, DT, and RF models were implemented using the scikit-learn library [115]. For the architecture and hyperparameter selection of these models, we used the default values of the scikit library. This is since an excessive and computationally intensive model and hyperparameter optimization of all examined models was not feasible due to time and resource constraints. Therefore, we refer the interested reader to the documentation of the scikit-learn library [127] for more information on the hyperparameters used.

Adversary model and assumptions: Paudice et al. described in [128] an optimal label flipping attack which finds per poisoning rate the optimal subset of poisoned training data that causes the most significant possible damage concerning the performance of the attacked model. This approach assumes a white-box attacker with access to all training data, the model to attack, and all related model parameters. The attackers' goal is to degrade the attacked model's performance e.g., accuracy, as much as possible. In the optimal label flipping attack, the attacker samples all possible poisoned permutations per poisoning rate, and re-trains and re-

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

evaluates the known model for each poisoned permutation. Finally, the attacker selects the permutation that caused the most significant damage and supplies the poisoned training data set to the victim. However, since this approach is a computationally intractable problem, [129] proposes a heuristic approach named randomized label flipping attack, which also takes into account a restricted attacker budget. In this approach, the attacker does not evaluate all permutations but just randomly samples a limited number of poisoned permutations, and subsequently re-trains and re-evaluates the model. The attacker continues with this approach until either a maximum number of iterations are performed (attacker budget exhausted) or a satisfying decrease in model performance is measured. Finally, the attacker again selects the permutation with the most significant damage in terms of model performance and supplies it to the victim.

In Scenario 4, we aim to implement this heuristic random label flipping attack. However, for all poisoning rates/percentages investigated in the robustness experiments, this would require poisoning, training, and evaluating all ML algorithms several times during the attack. Given the time and computational resources available, this was considered intractable. Therefore, we decided to poison only a single random permutation per poisoning rate/percentage and directly supply this poisoned subset to the victim without having to train and evaluate the models during the attack. Therefore, for the results described in Scenario 4, we can even relax the adversary model by assuming a black-box attacker model. The black-box attacker only has access to the training data and randomly poisons it only once per poisoning rate without evaluating the model. The attacker's goal remains to degrade the model performance as much as possible. Furthermore, we assume the victim has no information about whether the training dataset was poisoned or which specific label flipping attack was applied. In the scenario, we assume that the victim obtains data from untrusted sources and uses it to train an ML model. However, since the victim is aware that it is obtaining data from untrusted sources, we also investigate the effect of defence strategies against label flipping attacks. For this purpose, we apply the label sanitization method presented by [128] and described in Section 1.3 for which no knowledge about the attacked and defended model is required.

Evaluation criteria: In application scenario, we investigated the effect of the random label flipping attack with varying poisoning rates on the performance of the models described above. In this context, we evaluated model performance with respect to classification accuracy, precision, recall, true positives, false positives, true negatives, and false negatives. In addition, we evaluated the effect of the label sanitization defence strategy on the model performance. All experiments were performed for the following varying poisoning rates: 0% (baseline), 1%, 5%, 10%, 20%, 30%, 40%, and 50%. The evaluation results are described in Section 4.4.

Setup and procedure: The UniMiB SHAR dataset was used to evaluate the effect of random label flipping attacks with varying poisoning rates and respective label sanitization defence to DNN, DT, RF, MLP, and LR models. During the performed evaluation, we exploited a notable characteristic of the UniMiB Shar dataset: The dataset contains separable data from a total of 30 human subjects. Leveraging this characteristic, we performed the leave-one-subject-out cross-validation proposed by Micucci et al. [29] to evaluate the effect of the investigated attack and defence strategy on the model performance. This approach removes personalization effects from the model evaluation [29] and allows better conclusions about the generalizability of the developed models to unseen data from new subjects. Precisely, for each iteration of the leave-one-subject-out cross-validation, data from a single subject is used as the test data set, whereas the data of the remaining 29 subjects is used as the training data set. This results in a total number of 30 cross-validation splits per model training. Afterwards, we averaged all measured model performance evaluation metrics (e.g., accuracy, precision, recall). In summary, our

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

procedure for the investigated poisoning rates (0% (baseline), 1%, 5%, 10%, 20%, 30%, 40%, and 50%) and ML models can be described as follows: We applied the leave-one-subject-cross-validation for both the random label flipping attack and the label sanitization defence strategy for each combination of poisoning rate and ML model. First, a single subject is selected, and its data is filtered and put aside as a test data set. Subsequently, the data of the remaining subjects are used as the training data set. Based on the individual poisoning rate, a random subset of the training dataset is selected and poisoned according to the random label flipping attack. This means that for all randomly selected sample data points, the existing labels are replaced, i.e., "ADL" labels are flipped to "FALL" and vice versa (see Figure 29). Afterwards, the respective ML model (e.g., DNN, DT, RF, LR, or MLP) is trained on the poisoned training data set and then evaluated with the retained clean test data set based on the metrics described above. Next, we apply the label sanitization defence (see Section 1.3) on the poisoned training dataset. Thereby, we chose the parameters of nearest neighbours k=15 and the relabeling threshold $\mathrm{h} = 0.5$ see Section on bias detection mechanism). After the label sanitization was performed, the model is trained and re-evaluated on the cleaned dataset. According to the leave-one-subject-out cross-validation, this procedure is then repeated for the remaining 29 subjects. Finally, we averaged the measured model performance metrics. By performing 30 repetitions of the attacks and defences in the leave-one-subject-out cross-validation, we hope to have smoothed out statistical uncertainties and increased the significance and generality of the results. Again, we want to stress that the described procedure is repeated and executed for all combinations of poisoning rates and ML models.

4. RESULTS

Key insights from our experiments can be summarized as follows

- Changes in data caused by induced poisoning can be captured by CPU measurements as the device performs the distributed training (Results from Autonomous drones, see Section 4.1)
- Existing methods for bias detection can easily be applied to distributed machine learning (Results from Healthcare-based imaging, see Section 4.2)
- Metrics such as cosine similarity, selectivity score, and invariance can provide an approach to measure bias during the training process (Results from Autonomous vehicles, see Section 4.3)
- The performance of the label sanitization method is an excellent defence strategy against the applied random label flipping up to a poisoning rate of 30% in FL framework (Results from Healthcare-based sensor data, see Section 4.4)

4.1. IMPACT OF DATA POISONING

We also evaluated the FL implementation of the model considering three trials where AGVs serve as nodes that communicate with an Ubuntu server that aggregates the parameters. The result shows that the accuracy of the model is 76.07%, depending on the data that is used for training. Thus, the FL implementation achieves similar performance as the single model that uses the complete dataset for training, demonstrating the correctness of our FL implementation.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Effects of Poisoning: We first explore the model accuracy when poisoning a model with different methods using both, a classical centralized (one server) and a FL architecture (four devices + one server). As we increase the amount of poisoning from 5 % to 30 %, there was a steep decline in accuracy for both approaches. The poisoning attacks reduce the performance of the model at comparable rates for both architectures, Blurring (FL: 14.3 %; classic: 12.4 %) Steganography (FL: 10.09 %; classic: 16.2 %), Label flipping (FL: 13.2 %; classic: 15.5 %), and

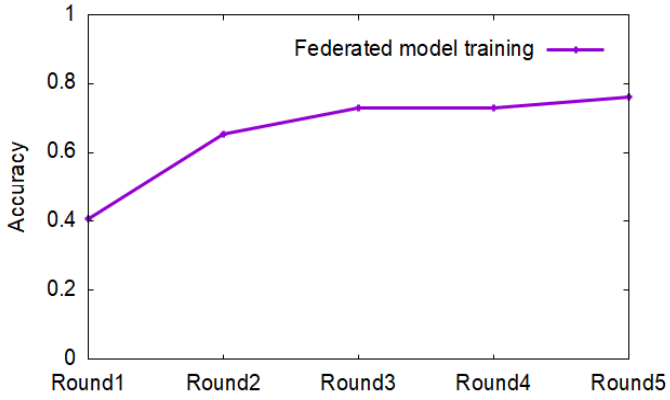


Figure 16: Model accuracy in FL as baseline

Occlusion (FL: 14.2 %; classic: 13.8 %). The result thus suggests that poisoning not only affects FL but can similarly decrease the performance of cloud-based training if equivalent part of the data is poisoned. Kendall correlation analysis indicates a negative correlation between data poisoning percentage and accuracy (τ = -1, $p<.05$), i.e., as the amount of poisoned data increases, the model's performance decreases further.

Distributed poisoning: The previous results show that the overall effect of data poisoning can be comparable in both the centralized and FL architectures. In practice, FL is subject to incremental poisoning by individual devices which accumulates over time, and thus can have a subtler and more difficult to discern effect. We next analyze the influence on model accuracy when different amounts of distributed devices contribute poisoned data to model training. We conduct these experiments using Testbed-1, i.e., virtual threads running on the same platform. Figure 17 shows the results when using up to 10 devices. We can observe the accuracy to be stable when data is clean (0 %, meaning clean data) and we can observe that the model's accuracy decreases gradually as the percentage of poisoned data increases. In parallel, we observe that from 3 devices, there are high declines in model accuracy (highlighted in red). Variance results from Table 4 on model accuracy also suggest that the severity of the performance drop depends on the poisoning method. Specifically, poisoning effects are significant after 3 devices with blurring, after just (1) one device with label flipping, and after 4 devices for occlusion and steganography. These results suggest that data poisoning is often subtle and possible to ignore when only a single source is compromised, but fast becomes severe as more devices are compromised. The results also show that the performance decline saturates eventually, and the performance starts to oscillate between improvements and decreases as the poisoned data takes control of the process. For instance, we can observe that

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

when ten devices poisoned the data, the model accuracy is about 70.6 %, which is just 3 % below its normal.

Table 4: Model performance degradation as incremental data poisoning is introduced by (virtual) individual devices gradually

Poisoning(%)	The number of devices									
	1/10	2/10	3/10	4/10	5/10	6/10	7/10	8/10	9/10	10/10
un-poisoned	74.5 ± 3.7	74.5 ± 3.7	74.56 ± 3.7	74.5 ± 3.7	74.5 ± 3.7	74.5 ± 3.76	74.5 ± 3.7	74.5 ± 3.7	74.5 ± 3.7	74.5 ± 3.7
Poisoning Type	Blurring									
5	70.9 ± 2.5	71.4 ± 0.6	69.8 ± 2.4	67.3 ± 2.3	66.7 ± 2.3	64.6 ± 2.2	65.9 ± 2.3	67.2 ± 2.3	67.5 ± 2.3	69.5 ± 2.4
10	72.9 ± 1.2	71.6 ± 2.3	71.8 ± 1.2	69.2 ± 1.2	68.6 ± 2.1	66.4 ± 2.2	67.8 ± 3.6	69.1 ± 1.1	69.4 ± 1.2	71.5 ± 1.2
15	69.3 ± 2.9	66.2 ± 3.5	68.3 ± 2.8	65.8 ± 2.7	65.2 ± 2.7	63.2 ± 2.6	64.5 ± 2.7	65.7 ± 2.7	66.0 ± 2.8	67.9 ± 2.8
20	72 ± 2.6	70.9 ± 1.5	70.9 ± 2.5	68.3 ± 2.2	67.8 ± 2.4	65.6 ± 2.1	67.0 ± 2.4	68.3 ± 2.5	68.6 ± 2.1	70.0 ± 2.5
25	68.9 ± 3.4	67 ± 2.3	67.8 ± 3.3	65.3 ± 3.2	64.8 ± 3.2	62.7 ± 2.6	64.1 ± 3.1	65.3 ± 1.2	65.6 ± 2.2	67.5 ± 3.3
30	72.0 ± 3.3	66.4 ± 3.1	70.9 ± 3.3	68.3 ± 3.2	67.8 ± 1.2	65.6 ± 3.0	67.0 ± 2.2	68.3 ± 3.1	68.2 ± 1.2	70.6 ± 3.3
Poisoning Type	Occlusion									
5	70.1 ± 1.0	69.6 ± 2.4	66.2 ± 1.2	66.3 ± 2.5	69.9 ± 1.5	64.2 ± 3.6	66.5 ± 2.2	67.3 ± 3.1	66.5 ± 2.6	70.1 ± 2.4
10	69.4 ± 1.5	59.1 ± 3.6	77.4 ± 2.2	66.1 ± 3.1	74.7 ± 2.6	64.6 ± 1.2	60.9 ± 2.7	73.5 ± 2.8	68.9 ± 2.3	67.1 ± 2.6
15	69.6 ± 1.6	63.3 ± 2.3	75.7 ± 1.7	68.0 ± 3.6	69.1 ± 2.3	59.6 ± 3.6	70.8 ± 2.4	70.2 ± 1.5	71.9 ± 3.0	72.1 ± 1.3
20	71.1 ± 1.2	60.2 ± 1.3	69.6 ± 1.3	63.6 ± 3.0	71.7 ± 1.0	64.4 ± 3.6	60.7 ± 2.3	71.7 ± 2.8	71.1 ± 2.2	71.0 ± 2.1
25	67.1 ± 2.3	64.2 ± 2.2	67.1 ± 1.8	69.6 ± 1.0	70.1 ± 3.6	66.5 ± 2.3	59.7 ± 1.2	66.5 ± 1.2	70.4 ± 2.7	72.6 ± 2.5
30	66.6 ± 1.0	73.1 ± 3.3	71.6 ± 3.1	69.6 ± 2.0	76.2 ± 2.6	63.8 ± 3.0	64.4 ± 2.2	74.2 ± 1.7	64.5 ± 3.4	68.1 ± 3.1
Poisoning Type	Steganography									
5	69.8 ± 3.3	68.5 ± 2.9	64.3 ± 2.4	67.9 ± 1.7	68.1 ± 3.5	69.3 ± 1.1	70.0 ± 2.0	71.5 ± 1.0	68.2 ± 3.9	71.6 ± 3.5
10	69.5 ± 3.7	68.4 ± 2.4	68.1 ± 2.0	70.0 ± 3.5	67.3 ± 2.8	69.7 ± 2.8	69.3 ± 2.4	69.4 ± 1.2	67.4 ± 3.7	69.4 ± 1.0
15	70.7 ± 3.3	67.1 ± 2.8	67.6 ± 3.4	72.6 ± 1.0	68.0 ± 1.3	69.7 ± 3.3	70.4 ± 2.5	69.7 ± 2.9	68.0 ± 3.6	69.8 ± 3.2
20	71.9 ± 1.4	67.5 ± 1.5	67.6 ± 1.9	69.9 ± 1.3	68.2 ± 3.4	67.3 ± 3.7	65.7 ± 2.4	67.8 ± 1.4	68.3 ± 1.0	67.9 ± 1.4
25	69.8 ± 3.2	66.0 ± 1.4	67.6 ± 1.5	69.7 ± 1.8	68.0 ± 2.6	72.2 ± 3.2	69.5 ± 1.2	70.0 ± 1.2	68.1 ± 2.7	70.0 ± 1.6
30	71.4 ± 1.4	66.0 ± 1.6	67.1 ± 3.8	66.2 ± 3.8	68.4 ± 3.2	69.0 ± 3.2	71.7 ± 3.3	67.5 ± 1.0	68.5 ± 1.2	67.5 ± 3.0
Poisoning Type	Label Flipping									
5	75.9 ± 3.6	74.8 ± 1.2	67.0 ± 3.0	68.0 ± 2.4	76.4 ± 1.4	75.1 ± 3.8	63.0 ± 3.9	59.8 ± 1.0	63.3 ± 3.9	60.5 ± 3.2
10	68.9 ± 2.4	64.7 ± 3.3	60.0 ± 2.8	64.0 ± 3.9	78.6 ± 1.0	75.1 ± 1.9	69.7 ± 1.2	65.1 ± 3.0	60.0 ± 3.6	68.9 ± 3.6
15	69.1 ± 1.5	69.1 ± 2.2	76.4 ± 2.0	77.8 ± 2.1	76.4 ± 1.9	77.8 ± 3.4	37.1 ± 2.1	64.7 ± 1.0	69.3 ± 1.5	68.5 ± 2.4
20	71.1 ± 1.8	70.3 ± 1.7	69.6 ± 2	72.8 ± 3.9	72.1 ± 3.4	72.2 ± 1.1	61.2 ± 1.4	65.1 ± 1.1	52.3 ± 2.3	63.5 ± 3.0
25	67.0 ± 2.2	61.5 ± 3.0	67.0 ± 3.8	66.7 ± 3.7	71.1 ± 2.7	73.3 ± 3.1	61.5 ± 2.8	64.9 ± 2.0	69.6 ± 1.3	61.5 ± 2.3
30	66.5 ± 3.7	66.0 ± 3.2	71.6 ± 1.8	72.6 ± 3.1	73.6 ± 3.2	74.0 ± 3.0	51.9 ± 2.7	67.7 ± 1.2	57.7 ± 3.2	52.9 ± 1.2

Identifying data poisoning: We next evaluate the first part of the SENTINEL pipeline by demonstrating that performance metrics can capture changes in model accuracy as data is poisoned. To do this, we rely on our real-world testbed (testbed 2) with 4 devices. From Figure 17, we can observe that the effects of data poisoning are highly similar between the virtual and the real-world deployment of testbed 2. Indeed, Figure 17 (a) shows marginal RMSE and accuracy differences between the two testbeds. Kolmogorov-Smirnov test on the accuracy difference verifies that the performance of the model between the virtual and the real devices. Figure 17 (b) indicates no significant differences for any of the attacks: KS=0.7563, p-value>0.05

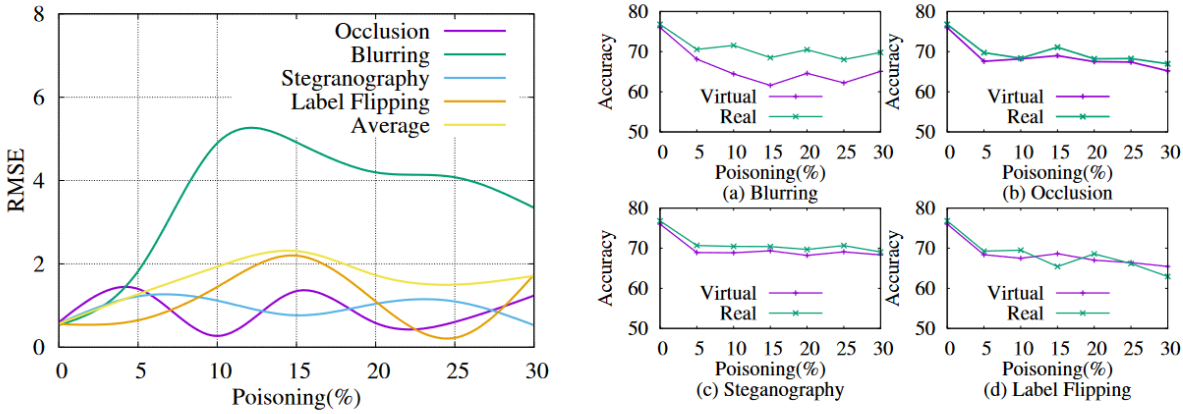


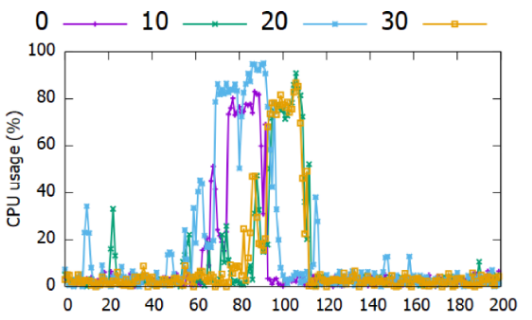
Figure 17: Testbed 2 virtual and real similarities

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

for blurring, KS=0.8351, p-value>0.05 for Occlusion, KS=0.8253, p-value>0.05 for steganography, and KS=0.8132, p-value> 0.05 for label flipping.

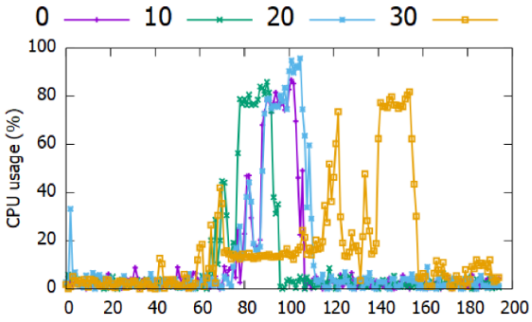
Performance metric quantification: We next capture and analyze the performance metrics of these devices when training models with poisoned data. We analyze primarily performance provided by CPU utilization, which constitutes a time series sampled in a period of time. We do so by identifying the change points and deriving the DCPIs from the performance metrics generated by the device in use. Figure 18 shows the results. As the amount of poisoning increases, the DCPI also increases. We can also observe increased variations in the number of CPI measurements. For instance, we can observe significant differences between data poisoning percentages just by looking at these metrics, 10% (CP=32, DCPI=0.069) and 30% (CP=40, DCPI=0.107) for blurring and 10% (CP=29, DCPI=0.067) and 30% (CP=37, DCPI=0.078) for steganography. To analyze the relationship between them, we further perform a Kendall correlation analysis between the percentage of poisoning and the DCPI: τ =0.7142, p<0.05 for blurring, τ =0.61905, p<0.05 for steganography, τ =0.90476, p<0.05 for occlusion, τ =0.80952, p<0.05 for label flipping, and τ =0.54997, p<0.05 for all poisonings together.

In all the cases, we observe a significant effect and correlation between the DCPI and the extent of poisoning. Figure 19 illustrates this further by showing the overall DCPIs calculated for incremental data poisoning levels for each attack type. We perform repeated-measures ANOVA test to assess the effect of DCPI on the extent of poisoning. The test shows significant effect on



Poisoning(%)	10	20	30
Change Points	32	38	40
DCPI	0.069	0.102	0.107

(a) Blurring



Poisoning(%)	10	20	30
Change Points	29	32	37
DCPI	0.067	0.069	0.078

(b) Steganography

Figure 18: Poisoned patterns captured by performance metrics

the amount of poisoning ($F(1.36,8.18)=10.318$, p-value<.05, $\eta^2=0.112$), demonstrating that DCPI can characterize the level of poisoning on the device.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

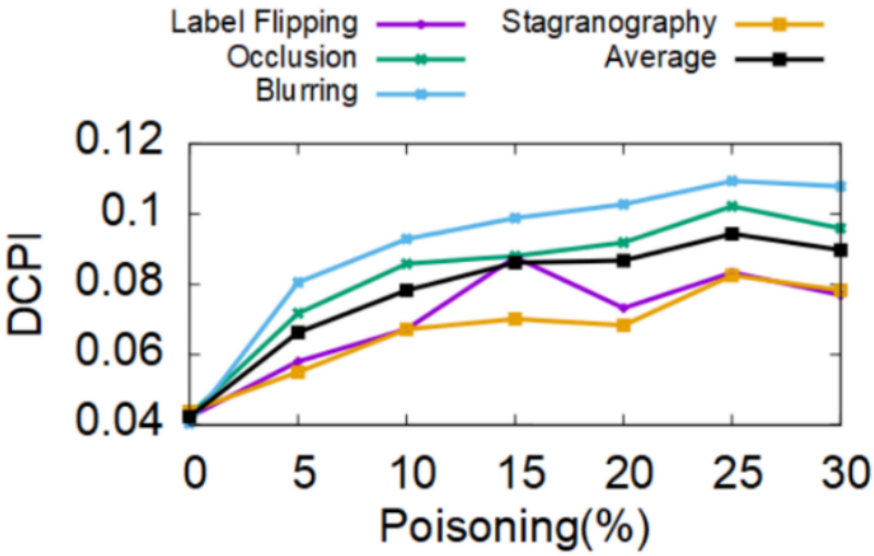


Figure 19: DCPI performance metric footprints with different levels of poisonings

4.2. QUANTIFICATION OF DATA BIASES

PathMNIST detailed distribution by classes is shown in Figure 20, demonstrating differences in class distribution. Figure 21 shows success in the classification of types of pathology by class.

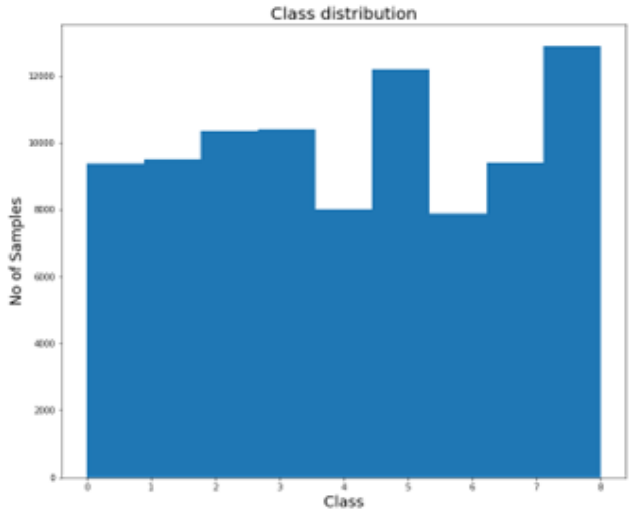


Figure 20: Distribution of samples by classes.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

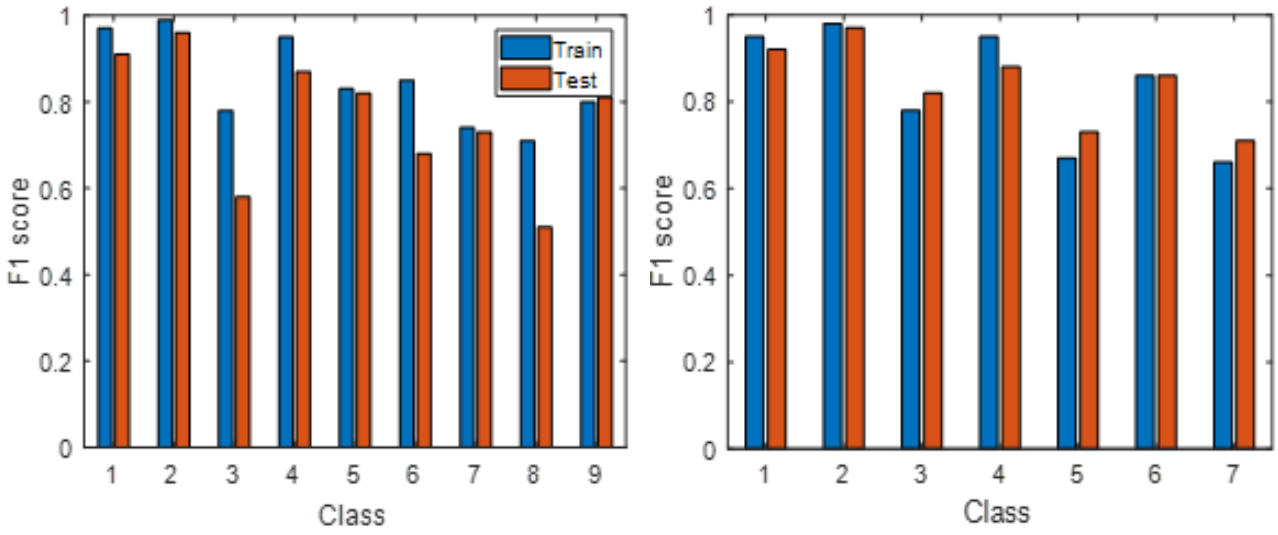


Figure 21: F1 score by classes before and after the exclusion of poor learning classes

This justifies our exclusion of Class 2 and Class 7 from the further training process because their F1 scores are 0.58 and 0.51 respectively. The accuracy of classifications is 0.85 and 0.80 on the train and test set respectively. The accuracy of classifications without Classes 2 and 7 are 0.92 and 0.87, respectively. Figure 22 represents the F1 score by class depending on the number of biased workers from 0 to 4. The undersampled class is chosen to be one that showed

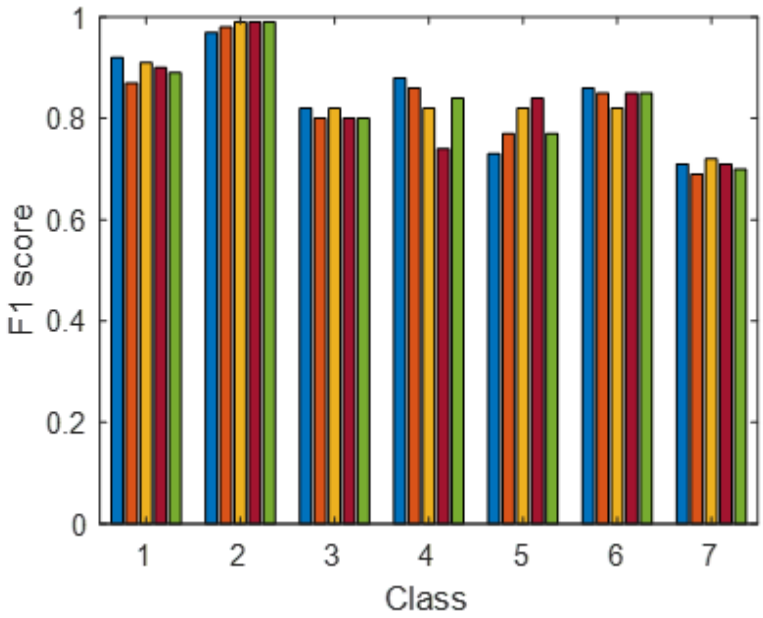


Figure 22: F1 score by classes represented in the order number of workers with undersampled best class on the test set (blue – 0, red – 1, yellow – 2, burgundy – 3, green –4)

the best results in the initial federated learning without bias, which is class 2. Accuracy of classifications on centralized training data are: 0.84, 0.83, 0.84, 0.83, 0.83 and accuracy of classification on centralized test data are: 0.84, 0.83, 0.85, 0.84, 0.83 for 0, 1, 2, 3, and 4 biased workers respectively.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Figure 23 shows a similar concept as Figure 22 only that in this case the undersampling class is chosen to be the class with the worst learning results in initial federated learning (F1 score) which is class 7 in this case. The result for both the best and worst classes could be seen in Figure 21 .Accuracy of classifications on centralized training data are: 0.84, 0.85, 0.82, 0.8, 0.79

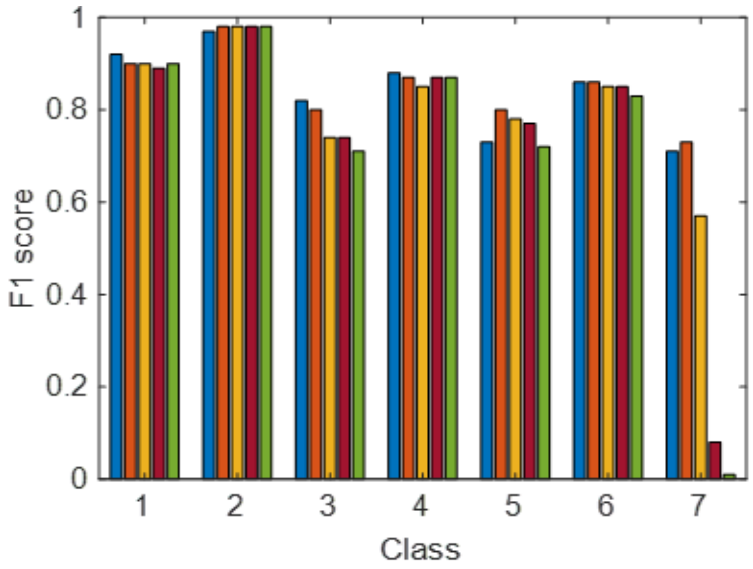


Figure 23: F1 score by classes represented in the order number of workers with undersampled worst class on the test set (blue – 0, red – 1, yellow – 2, burgundy – 3, green – 4)

and accuracy of classification on centralized test data are: 0.84, 0.85, 0.81, 0.79, 0.77 for 0, 1, 2, 3, and 4 biased workers respectively.

Figure 24 shows the F1 score by class in case of mislabeling the class. All samples from class 2 are artificially labeled as class 7 which prevents generalization. This case could be a result of

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

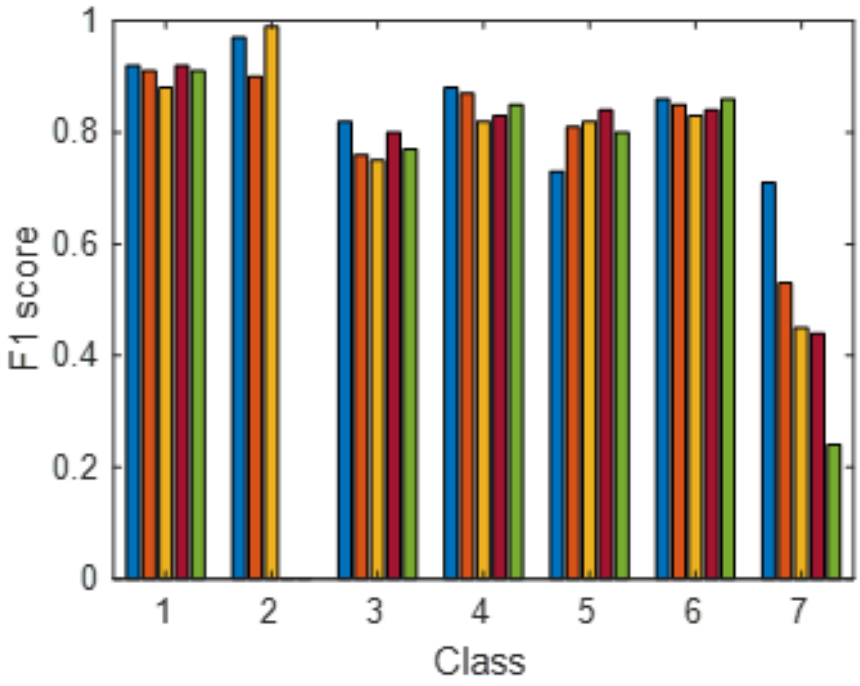


Figure 24: F1 score by classes represented in the order number of workers with mislabeled classes on the test set (blue – 0, red – 1, yellow – 2, burgundy – 3, green – 4)

human error, which can be interpreted as measurement bias, or a result of a malicious attack that should compromise the learning process. Results are represented in order of the number of biased workers from 0 to 4. The accuracy of classification on centralized training set are: 0.84, 0.84, 0.79, 0.71, 0.7 and the accuracy of classification on centralized test set are: 0.84, 0.83, 0.8, 0.72, 0.69 for 0, 1, 2, 3, and 4 biased workers respectively.

Figure 25 is described the F1 score by classes in case of present noise, in this case: salt and pepper (S&P) noise. The noise is present in every measurement of signal, even in image capturing and storage, and as such could seriously impact the learning process. In this case, we induce S&P noise in every image in the local dataset and watch the impact on the centralized model. As in previous cases, the results are shown in order of the number of biased workers from 0 to 4. The accuracy of classifications on centralized training set are: 0.84, 0.8, 0.78, 0.77, 0.65 and the accuracy of on centralized test set are: 0.84, 0.81, 0.78, 0.76, 0.69 for 0, 1, 2, 3, and 4 biased workers respectively.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

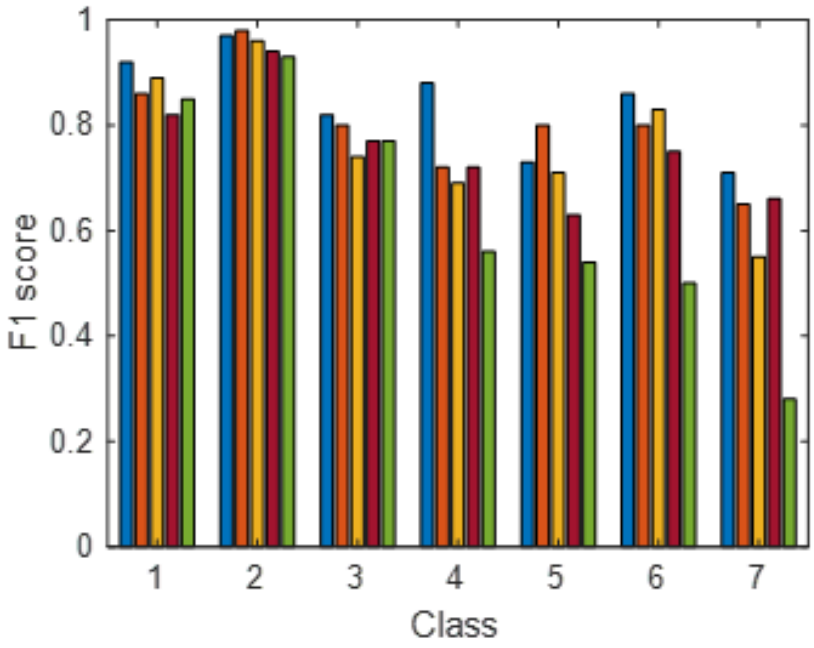


Figure 25: F1 score by classes represented in the order number of workers with induced S&P noise on the test set (blue – 0, red – 1, yellow – 2, burgundy – 3, green – 4)

4.3. COMPARISON OF DATA BIASES

Table 5 shows the cosine similarity analysis over the biased car dataset. For the equally distributed samples, the cosine similarity results for ResNet, SqueezeNet, and DenseNet were 0.38, 0.36, and 0.57 respectively. According to these results, we conclude that depending on the test sample and model's selectivity towards a category and class labels, SqueezeNet and ResNet could generalize better for the given input data.

Table 5: Performance of SOTA AI Models on the Biased-Car dataset

SOTA AI Model	Data Distribution	Average-Similarity	Accuracy (%)
ResNet18-cos	OOD	.38	74.1
SqueezeNet-cos	OOD	.36	74.8
DenseNet-cos	OOD	.57	71.5

Figure 26 shows the selectivity score of neurons on the biased-cars dataset with respect to the data diversity of classes. As the distribution combinations are increased for a fixed number of train-test-validation images, data diversity and class distribution make it difficult for the neural network to learn a challenging category or viewpoint. When the combination of train-test images is diverse, it leads to a drop in the neural network's performance (accuracy). The selectivity score shows a similar trend for SqueezeNet and ResNet once the data diversity is varied more than 50%.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

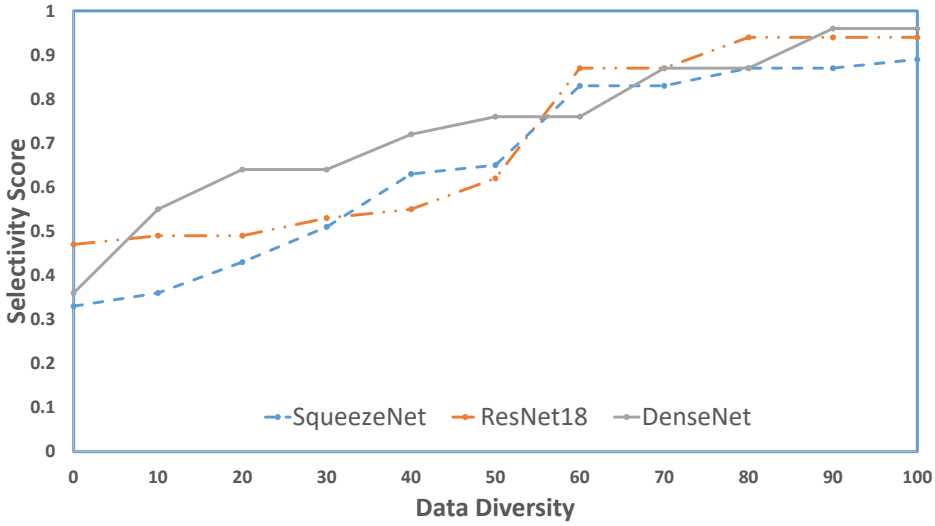


Figure 26: Selectivity Score on Biased-Car dataset with respect to data diversity

A small drop in accuracy (Figure 27) is observed for all neural networks between a data diversity of 40 – 60%. As shown in Table 6, similar to the biased car dataset, as the distribution combinations from different weather conditions are increased for a fixed number of train-test images, data diversity leads to a significant drop in the current model performance (precision). The selectivity score shows a similar trend for the two versions of the BIRANet architecture. BIRANet1 uses cross-entropy loss, and BIRANet2 uses joint loss. It is important to note that the selectivity score shows a similar trend for the two versions in data diversity, irrespective of separate loss functions. The models are trained using a train-validation split of 52-48%. The models show performance degradation in adverse conditions compared to normal conditions.

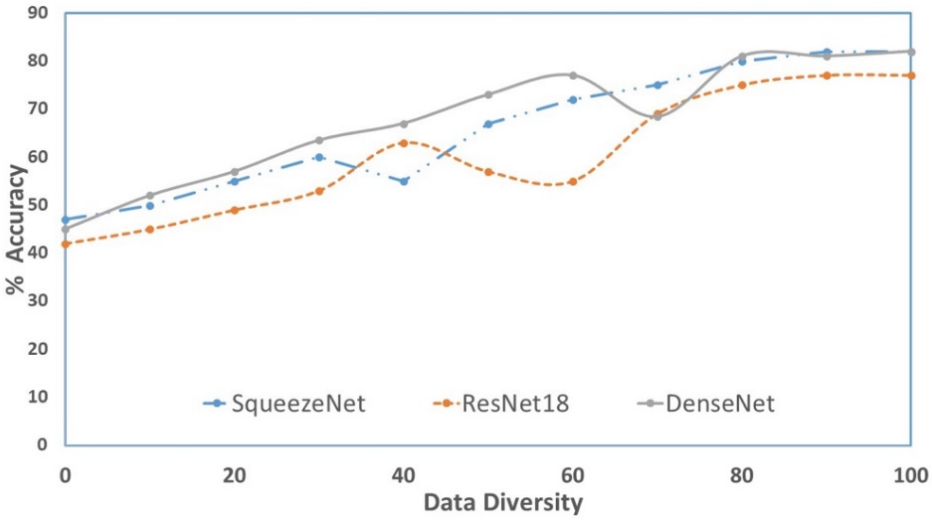


Figure 27: Accuracy on Biased-Car dataset with respect to data diversity

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Table 6: Models' performance on nuScenes dataset

Model	Dataset	Average Precision (car)	Average Precision (pedestrian)	mAP(0.5)
BIRANet1	nuScenes	64.1	46.4	55.2
BIRANet1	nuScenes (Night)	49.7	32.5	40.8
BIRANet1	nuScenes (Rain)	60.2	44.3	52.1
BIRANet2	nuScenes	64.8	46.1	56.3
BIRANet2	nuScenes (Night)	48.5	31.7	39.2
BIRANet2	nuScenes (Rain)	59.8	42.4	51.6

Figure 28 shows the selectivity score of neurons on the nuScenes dataset. The observed pattern is consistent across the considered models, with selectivity score getting increased for higher values of data diversity.

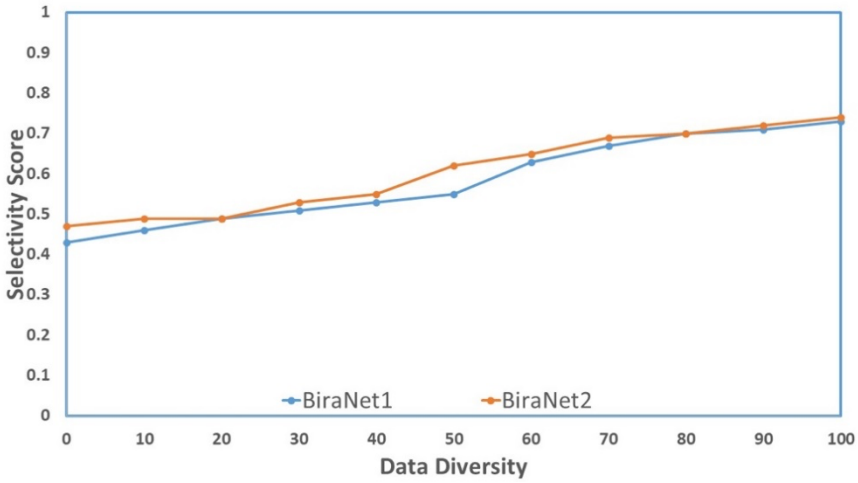


Figure 28: Selectivity score on nuScenes dataset with respect to Data Diversity

The topics and methods covered in this work [131] show approaches to identify and detect bias in an AI model used for real-world applications, such as autonomous driving. The focus given to the biased-car dataset shows that the requirement to have wide distributions of the objects in the training/testing set. It provides an unbiased dataset and further helps in the generalization of the AI model, thus preventing algorithmic bias.

We investigated bias detection using metrics such as selectivity score and cosine similarity during the learning process by varying the data diversity of the test set. The learned model is further used on the nuScenes dataset to detect pedestrians and cars. A further investigation can be carried out using distributed machine learning at the vehicle-edge-cloud for the vulnerable pair of classes such as pedestrians and cyclists. With respect to the other class (such as vehicles or traffic signs), the vulnerable classes generally have less representation within the dataset.

As the self-driving domain advances, studying metrics such as cosine similarity, selectivity score, and invariance can provide an approach to measure bias during the training process and further develop an unbiased AI model for inference. Exploring such metrics on a layer and block

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

level for a neural network by having a direct comparison with a similar object (e.g., pedestrian, cyclist, motorcyclist) can help understand the interpretability of the neural network, which further helps in generalization and preventing biases.

4.4. DEFENCE PERFORMANCE AGAINST DATA POISONING

Figure 29 illustrates the effect of the performed random label flipping attack and respective label sanitization on the label distribution of the used training data sets. As shown in the upper graph in Figure 29 the random label flipping attack successfully shifts the distribution of labels in the training dataset away from the original distribution. On the other hand, the lower graph in Figure 29 shows that the label sanitization defence successfully manages to reverse the effect of the random label flipping attack and recover original sample labels up to a poisoning rate of 30%. In Section 4.4, we will discuss the effect of the respective shifted and recovered label distributions on the performance of the investigated ML models.

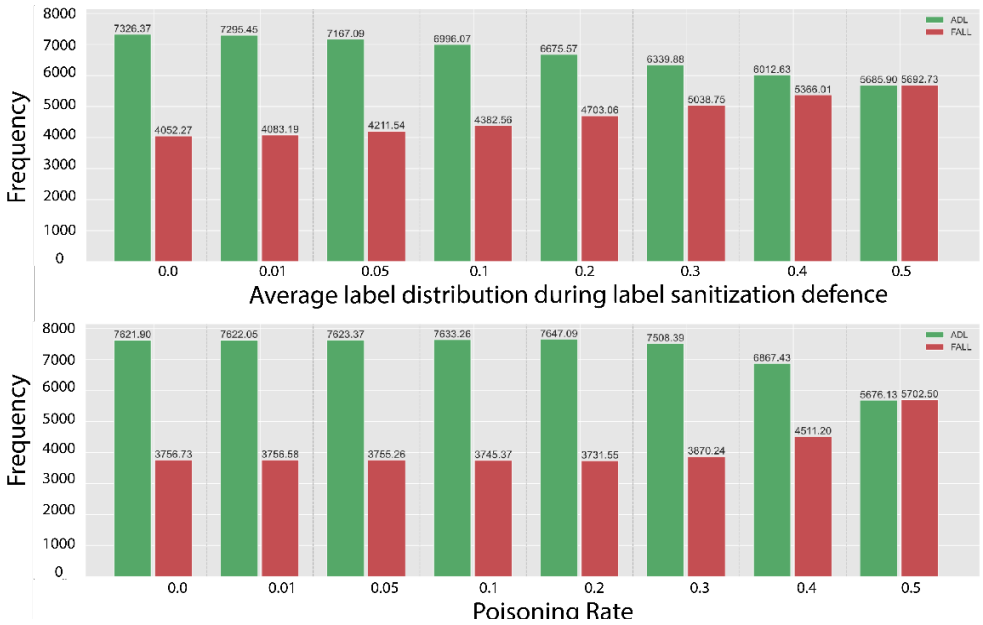


Figure 29: Average Label distribution of the training dataset after the random label flipping attack and respective label sanitization defence. The average values are calculated based on the results from the performed leave-one-subject-out cross-validation.

Baseline results: Table 9, which can be found in Appendix B, presents the obtained measurement results of the performed and above-described robustness experiments for the LR, DNN, RF, DT, and MLP models. In addition, Figure 30 and Figure 31 illustrate some of the acquired measurement results. For all results presented, the poisoning rate of 0% represents the baseline results unless otherwise stated. As shown in the left column of Figure 30 , the DNN, MLP and RF models achieve the best baseline results in the investigated fall detection use case. Table 9 indicates that the DNN, MLP, and RF achieve an accuracy and precision of 97% in the performed binary classification task. In contrast, the recall metric differs for the mentioned models. Here, the DNN achieves a value of 96%, whereas the RF and MLP manifest a recall of 95% and 94%, respectively. The obtained measurements for the false negatives and false

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

positives also demonstrate the excellent fall detection capabilities of the discussed models. As shown in Table 9 and Figure 31, the DNN, RF, and MLP exhibit a false alarm rate (false positives) of only 1%. In contrast, the values for the false negatives suggest that the DNN fails to detect falls in 1% of the cases, whereas the MLP and RF fail to detect falls in 2% of the cases. For our use case, these values are reasonably low. Therefore, the described DNN, MLP, and RF models can be considered very good classifiers for the performed fall detection and are comparable to the results of [92].

In contrast to the above-analysed high-performing fall detection models, the less complex but more interpretable DT and LR models perform more poorly. The DT model still achieves an accuracy of 90%, a precision of 88%, and a recall of 84% in the binary classification task. Besides, we can infer from Table 9 that the DT model fails to detect 6% of the falls (false negatives) and generates 4% of false positives. The LR model performs even worse. In the analysed fall detection scenario, it manifests an accuracy of only 73%. The precision and recall values of 63% and 46% are also significantly worse than those of the other presented models. The false negatives and false positive metrics also directly reflect this. Precisely, the LR model generates false alarms in 9% of the cases and even fails to detect 18% of the falls, which is

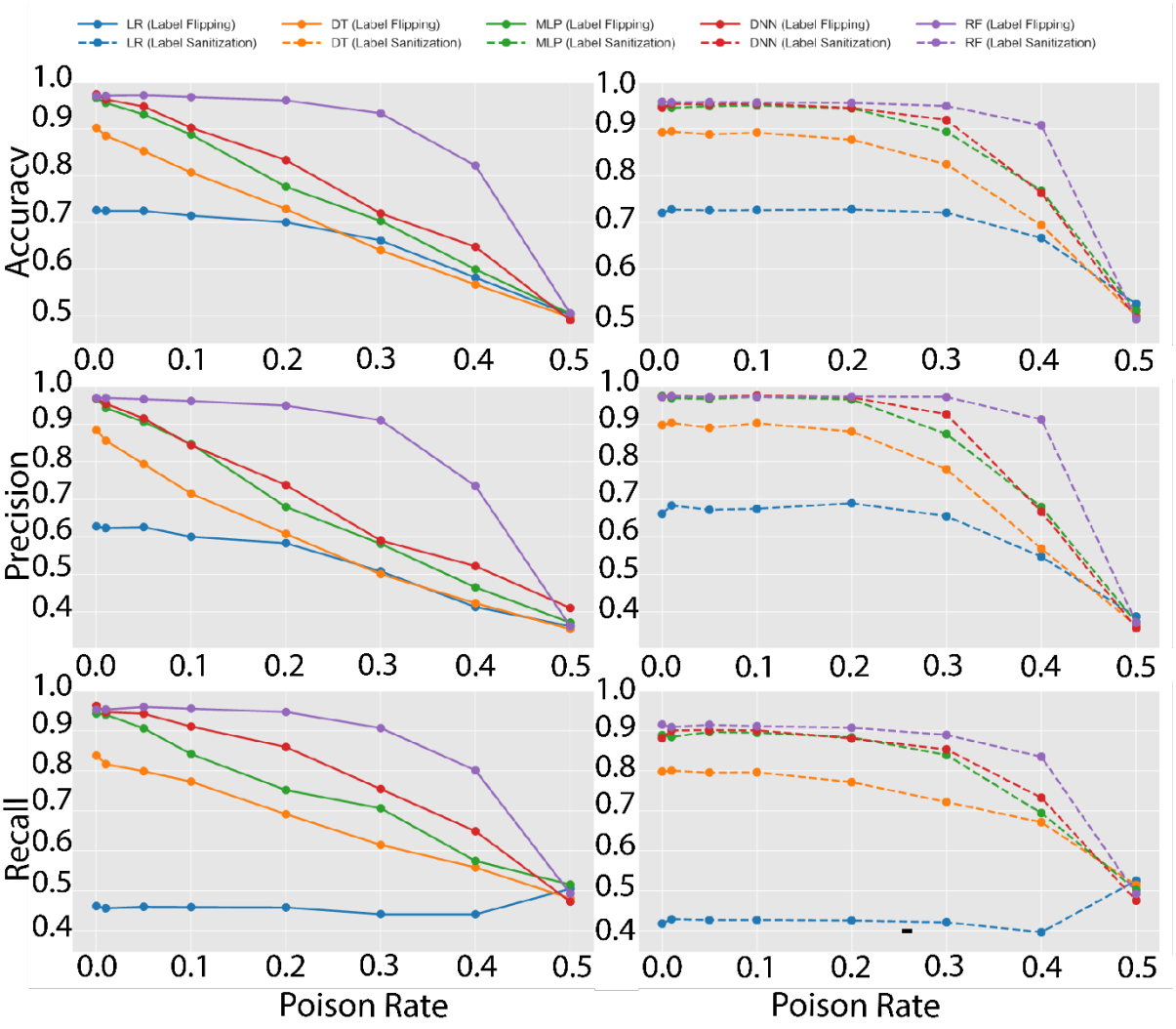


Figure 30: Effects of poisoning rate on model performance metrics

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

considered critical. In conclusion, the analysed LR and DT models with the selected hyperparameters are not suitable for practical application as fall detection models.

Instead, we recommend deploying the DNN, MLP, or RF models presented above for the fall detection described in Scenario 4.

Effects of data poisoning: In the robustness experiments performed in Scenario 4, we investigated the robustness of DNN, MLP, DT, RF, and LR machine learning models against random label flipping attacks. Thereby, we investigated the data poisoning attacks with varying poisoning rates in the experiments conducted. Specifically, the effect of the attacks was examined for poisoning rates of 1%, 5%, 10%, 20%, 30%, 40%, and 50%. Again, we refer to Table 9 for a numerical representation of the results obtained. In addition, the left column in Figure 30 and Figure 31 illustrates the effect of the poisoning rate on some of the investigated metrics[4].

As it can be derived from the left column of Figure 30, the investigated random label flipping attack has a significant impact on all performance metrics of all examined ML models. We can observe that all metrics decrease monotonically with an increasing poisoning rate. In some cases, even a significant performance decline can be observed. For example, the accuracy of the DNN model drops from 97% to only 90% at a poisoning rate of 10%. For a 20% and 30% poisoning rate, the model even manifests an accuracy of only 83% and 72%, respectively. Similar results can be observed for the precision and recall depicted in Figure 30. Since we consider poisoning rates of up to 30% as realistic and practical poisoning rates for label flipping attacks in real-world scenarios, this demonstrates the vulnerability of some ML models against this kind of adversarial ML attacks. Therefore, in our example of the DNN model, it can be concluded that the model is significantly limited in its fall detection capabilities at such poisoning rates. As a result, it becomes unusable for practical applications. Figure 31(a) illustrates the confusion matrix of the DNN as a function of the poisoning rate, which supports this assumption. As we can derive from the continuous line shown in red, the number of false alarms (false positives) for the DNN model increases drastically with an increasing poisoning rate. For example, at a poisoning rate of 20%, the DNN model detects ADLs as falls in 11% of cases. At a poisoning rate of 30% and 40%, the model even generates false alarms in 19% and 22% of the cases, which renders the model useless in practice. Similarly, the number of undetected falls (false negatives) also increases with an increasing poisoning rate. This raises critical safety concerns about the model's applicability, as undetected falls could result in serious harm. While the model fails to detect only 3% of falls at a poisoning rate of 10%, this value even increases to 9% at a poisoning rate of 30%. Thus, the model is no longer a reliable fall detector. Despite these negative aspects, we can also report some positive results from our experiments. Precisely, we can observe that for low poisoning rates of 1% to 5%, the performance losses of the high-performing models DNN, MLP and RF are reasonably small. Therefore, at such poisoning rates, they can preserve their fall detection capabilities, which makes such poisoning rates appear acceptable in practice. In contrast, a higher poisoning rate of 10% or more appears to be safety critical in the use case discussed in Scenario 4.

Apart from the DNN model, similar behaviour can also be observed for the performance metrics of the MLP, LR, and DT models. Therefore, we do not discuss these in detail and refer the interested reader to Table 9, Figure 30, and Figure 31. In contrast, we must mention the behaviour of the RF as exceptional among the studied ML models. The RF model behaves differently than all other investigated ML algorithms and seems much more robust against the

[4] Further visualizations are available in Figures 35-37 in Appendix C.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

applied random label flipping attack. As shown in Table 9 and Figure 30, the RF model still exhibits an accuracy of 97% at a poisoning rate of 10%, which is equal to its baseline results. Even at a poisoning rate of 30%, the RF model still manifests an accuracy of 93%, which is still very close to the baseline results considering the high poisoning rate. Only at a poisoning rate of 40%, a significant performance decrease in all investigated metrics can be observed for the RF model. However, this is to be expected since the original distribution of the training data has already been fundamentally shifted at such a high poisoning rate. Thus, ML models, in principle, can no longer learn the correct behaviour. As shown in Figure 30 and can derived from Table 9, the recall and precision metrics for the RF are also relatively stable at poisoning rates of up to 30%. The observed decreases are neglectable for the applicability of the model. Again, only beginning at a poisoning rate of 40%, a significant performance decrease can be observed, which would render the model unusable in practice. The confusion matrix shown in Figure 31 (b) also indicates that the RF remains an excellent fall detector up to a poisoning rate of 30%. Only at a poisoning rate of 40%, the number of false alarms (11% false positives) and undetected cases (7% false negatives) increase seriously, which results in an unusable model. Therefore, for fall detection based on multivariate time series data, the RF model is robust to the investigated random label flipping attack up to a poisoning rate of 30%. For such poisoning rates, the RF would remain a reliable fall detector and could be used in practice.

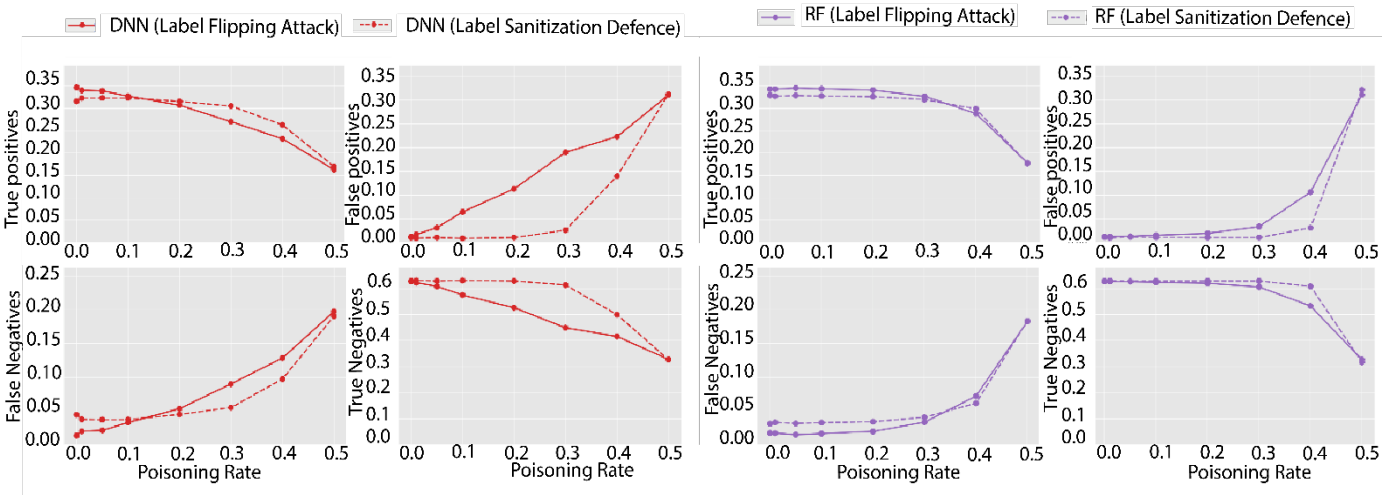


Figure 31: Effects of poisoning and defence mechanisms on model performance metrics

As another interesting fact, we can observe that all models converge in all metrics of interest at a very high poisoning rate of 50%, regardless of their underlying complexity. For this poisoning rate, all models examined show an average accuracy of 50%. Similar behaviour can be observed for the precision (40% on average) and recall (50% on average) metric. Therefore, we conclude that at such a high poisoning rate, the original data distribution has been manipulated so drastically that the binary classification task can no longer be performed, regardless of the ML algorithm used. Therefore, the ML model selection is no longer relevant here. The confusion matrix shown in Figure 31 supports these assumptions. As can be seen, the number of false positives (32% on average) clearly exceeds the number of true positives (18% on average) for all models at a poisoning rate of 50%. In addition, false negatives increase to 20% on average, whereas true negatives decrease to an average of 31%.

In summary, all investigated ML models are sensitive to random label flipping attacks in the context of fall detection based on multivariate time series data. During the performed

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

experiments, even significant performance losses at realistic poisoning rates were observed for the DNN, MLP, DT, and LR models. In contrast, the RF model is more robust against the applied data poisoning attack. While we observed that the other models could no longer be used as reliable fall detectors beginning at a poisoning rate of 10%, the RF could still reliably detect falls at a poisoning rate of up to 30%. The SPATIAL deliverables D2.1 and D1.2 support these observations and argue that ensemble learning methods such as the RF are more robust against adversarial ML attacks in many use cases. In conclusion, we state that the investigated random label flipping attack is a severe attack vector for the studied ML models in the context of multivariate time series classification. This demonstrates that data from untrusted sources should only be used with reservations and exhaustively analysed before using it for security-critical or safety-critical applications. Furthermore, the results also highlight the need for appropriate defence strategies. Therefore, we will investigate the effect of label sanitisation defence against random label flipping attacks in the next section.

Effects of data sanitization: Besides the robustness against label flipping attacks, we also investigated the effect and defence capabilities of the label sanitization method against this type of adversarial ML attack. This method was presented by [128] and was already described above. In simple terms, the label sanitization method attempts to identify and subsequently relabel suspicious data points within a local neighbourhood. In the experiments performed, we investigated whether this defence method can recover the model performance degraded by the applied random label flipping attack. Therefore, similar to the experiments for the label flipping attacks, we investigated poisoning rates of 0%, 1%, 5%, 10%, 20%, 30%, 40% and 50%. In this context, it can also be seen that we performed label sanitization experiments on the clean training data. These experiments are motivated by the fact that, in practice, a developer does not certainly know whether a supplied data set is poisoned or not and whether a defence method should be applied. Therefore, we want to investigate whether label sanitization is recommendable as a general and always applicable defence strategy. Again, the results of the experiments are listed in Table 9 and illustrated in Figure 31 as well as the right column of Figure 30.

As depicted in Figure 30, the applied label sanitization defence method can maintain relatively stable performance metrics for all investigated models for moderate poisoning rates. We can observe that up to a poisoning rate of 30%, all metrics decrease only moderately and remain comparable to the baseline results. For example, at a poisoning rate of 30%, the DNN model still exhibits an accuracy of 92% when applying the label sanitization defence, which is surprisingly close to the baseline accuracy of 97%, considering the high poisoning rate. In this case, the RF even manifests an accuracy of 95% compared to 97% in the baseline configuration. Similar moderate performance declines can be observed up to this poisoning rate (30%) for the other models and metrics examined. Such changes would be insignificant for the practical application. The confusion matrices shown in Figure 31 also support this statement. As we can observe from the dashed lines, the DNN and RF models produce neglectable false positives at up to 30% poisoning rates. Furthermore, we can conclude from the low false negative values that the models still reliably detect falls. However, the values here are slightly increased compared to the baseline configuration. Nevertheless, we can still consider the DNN, MLP and RF models as good fall detectors up to a poisoning rate of 30%.[5] Thus, up to a poisoning rate of 30%, the label sanitization method can identify a large proportion of the suspicious and previously poisoned data samples and recover their original label accordingly. Only from a poisoning rate of 40%, this ability seems to diminish significantly, as we can observe a significant

[5] As described in the baseline results, the LR and DT models are not recommended fall detectors even in the baseline configuration so that we will neglect them here.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

performance decline for all models. However, this behaviour is to be expected since the KNN algorithm used in label sanitization depends on the majority of labels within a local neighbourhood of a data sample. It is clear that at such high poisoning rates, the likelihood increases that a sample is located local neighbourhood with a majority of poisoned data samples. Hence, poisoned samples can no longer reliably be recognized as suspicious and relabeled. In addition, a poisoning rate of 40% can even lead to effects in which previously clean samples are mapped into a neighbourhood with a majority of poisoned data samples. Thus, the label sanitization method could even incorrectly relabel the clean label, which is equivalent to poisoning. At a poisoning rate of 50%, we can even observe that these effects are taken to their extreme. As we can see in Figure 30 and Figure 31, at a poisoning rate of 50%, the label sanitization loses all effectiveness, and the measurement results are comparable to those of the label flipping attack reported above. As seen in Figures and Table 9, all models also converge to the values described for the label flipping attack with a 50% poisoning rate. Thus, we conclude that the label sanitization defence can only be used effectively for moderate poisoning rates of up to 30%.

In addition, we have investigated the effect of the label sanitization method on clean training data. The goal of this approach is to determine whether this method can be recommended as a fundamental and generally applicable defence strategy. As we can observe in Table 9, Figure 30, and Figure 31, there is a very slight performance decline compared to the baseline results when applying label sanitization to clean data. For example, the DNN and MLP still show an accuracy of 95% compared to 97% in the baseline configuration. The RF model even manifests 96% accuracy compared to 97% in the original setting. A similar light performance decline can be observed for the precision and the number of false positives. Only the values for the recall decrease more significantly. For example, the DNN shows a recall of only 88% compared to 96% in the baseline. For the MLP and RF, the values decreased from 94% to 89% and 95% to 92%, respectively. This is also reflected in the number of false negatives. For example, while the DNN model fails to detect falls in 1% of cases in the baseline configuration, the number of false negatives increases to already 4% in this scenario (see Figure 31). In most practical use cases, these performance declines should be insignificant. However, it depends on the specific use case and the optimization goals. Therefore, a use-case-specific trade-off has to be defined, determining whether the observable moderate performance declines and the slight increase of false negatives are acceptable. In use cases where it is particularly important to keep the number of false negatives as low as possible, a general application of label sanitization to data from untrusted sources is not recommendable. If, however, a slight decline in performance is acceptable, the fundamental use of label sanitization is recommended due to the excellent defence capabilities up to a poisoning rate of 30%. However, we want to emphasize that all recommendations only refer to the investigated use case of fall detection in the context of multivariate time series acceleration data.

In summary, we conclude that the label sanitization method is an excellent defence strategy against the applied random label flipping up to a poisoning rate of 30%. Only from higher, but no longer realistic, poisoning rates of more than 30%, the label sanitization defence becomes ineffective against random label flipping attacks. Besides, we could observe a slight performance decrease when applying label sanitization on clean data. Therefore, it depends on the particular use case and the defined trade-offs whether it's recommended to apply this method in general for the benefit of increased security and robustness or whether it should not be applied due to high performance requirements. Furthermore, the label sanitization defence could be used to detect whether datasets from untrusted sources was poisoned via random label flipping attacks. However, such an analysis is out of the scope of this deliverable.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Limitations: For the experiments and results discussed in Scenario 4, we have to mention a couple of limitations and suggestions for improvement in the context of future work. The first limitation is given by the fact that we only poisoned a single randomly and model-independently chosen subset of the training dataset. As declared above, this was due to the limited time and computational resources. However, as shown in the work of [128] and [129] discussed above, other label flipping attacks exist that select even more effective subsets from an attacker's perspective. Poisoning these could impact our discussed results and lead to even more significant performance decreases. However, the simplicity of our attack and the nevertheless observed significant performance decline illustrate the severe threat that random label flipping attacks pose to ML models, even for black-box attackers with a limited budget and no model knowledge.

A further limitation was already described above. Again, due to limited time and computational resources, we have not performed hyperparameter optimizations for the investigated LR, DT, MLP, and RF models. Instead, we used default parameters provided by the used libraries. Furthermore, we used the same model parameters for all investigated poisoning rates to increase the comparability of the results. However, in real-world scenarios, the attacked developer to which the poisoned data was supplied would probably try to optimize the models for the received (but unknown) poisoning rate. Therefore, individually optimizing all studied models and hyperparameters for all investigated poisoning rates would be an approach closer to the practical use case. However, as model selection and hyperparameter optimization is a time and resource-intensive method, such an approach was not feasible due to time constraints and available computational resources within the scope of the project.

5. XAI FOR MODEL ANALYSIS

Explainable AI (XAI) methods provide also a natural point to dissect the logic of AI models. In the following, besides quantifying the important features of data that AI models used to boost their inference process, we also analyze whether induce and non-induced changes in data can be detected using XAI.

5.1. XAI PERFORMANCE

We analyze the complexity of using XAI methods to monitor drone behavior. We rely on the TrashNet litter classification dataset as in our main experiment. Here, as AI model we consider a convolutional deep learning model (CNN). Images are resampled to 300 x 300 to have consistent input dimensionality. We augment the training data using horizontal and vertical flipping and rescaling. We train the dataset using 2276 images with a batch size of 32 for each epoch iteration. The remaining images are used for testing, and we separately consider a collection of 10 poisoned and non-poisoned images for illustrating the performance of XAI methods. Our experiment was conducted on the Google Colab platform using the latest version of the Keras library (2.8.0) with TensorFlow (v2.8.2).

Threat model: We consider a generic threat model where the attacker attempts to cause the AI model used in the autonomous drone to fail. This can be a targeted attack that results in a specific misbehaviour, e.g., causing accidents by making the navigation support to fail to recognize pedestrians or cars, or an attack that simply causes the AI to malfunction, e.g., a sponge attack that drains the drone's resources or a ransomware attack that prevents normal

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

operations. The motivation for the attack can be causing damage or harm to the city or the citizens, financial incentive, or desire for fame.

XAI methods and analysis pipeline: We consider three model-agnostic XAI methods that can be applied for any type of AI model: LIME (Local the Interpretable Model-agnostic Explanation), SHAP (Shapley Addictive Explanations) and Occlusion sensitivity. As these methods are model-agnostic, they do not require any information about the CNN gradients to analyze model behaviour. These methods are also perturbation-based, which means that they manipulate the input (i.e., image pixels) to extract details that can be linked with the predictions. In our analysis, the XAI methods are applied separately for images where the background is removed (i.e., only the litter object) and for the original input image. The process for extracting the object is shown in Figure 32 and works by applying a dynamic patch (determined using object detection) on the image to isolate it. From the final output of the XAI methods, we calculate a pixel percentage metric that captures the importance of a pixel.

Data samples and quality trade-offs: We considered six litter categories: glass, paper, cardboard, trash, metal and plastic. To change the level of data quality, we rely on poisoning attacks. For the poisoning we consider two attacks, blurring and steganography, as described in the previous section. Blurring can make autonomous drones to misidentify targets in urban areas, e.g., crossing signals and pedestrian sides. Steganography introduces extra information in the binary information of the images, which can become resources intensive for the autonomous drone as more processing power is required to extract relevant information (similar to a sponge attack). We systematically assess how the level of poisoning affects the results by poisoning the data in 10% increments from 10% to 40%.

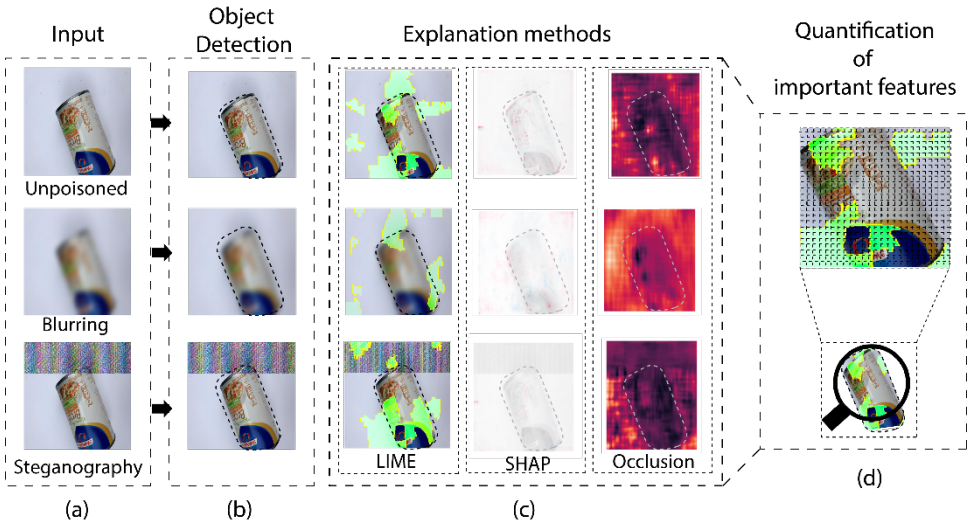


Figure 32: Pipeline to analyze objects with XAI methods by removing the background from images.

Model performance under poisoning: The performance to classify litter of our CNN is 0.7 when no data is poisoned, but this performance is reduced as data is gradually poisoned. After blurring attack, the model accuracy is reduced to 0.61 (10% poisoned); 0.53 (20% poisoned); 0.53 (30% poisoned) and 0.60 (40% poisoned). Similarly, after steganography, the model accuracy is reduced to 0.52 (10% poisoned); 0.52 (20% poisoned); 0.62 (30% poisoned) and 0.67 (40% poisoned). Table 7 summarizes the results. In both cases, we observe a clear drop in accuracy. Unlike our earlier experiment, the performance drop is higher for data poisoned with steganography than with blurring. This difference in results is simply due to differences in

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

the sensors (RGB vs thermal camera) and the processing pipeline and highlights how the effectiveness of the attack is influenced by the task and the specifics of the AI that is being used.

Table 7: Model performance as the data quality of the model is influenced by data poisoning

Percentage of data attack	Model performance (Blurring)	Model performance (Steganography)
Not poisoned	0.70	0.70
10%	0.61	0.52
20%	0.53	0.52
30%	0.53	0.62
40%	0.60	0.67

The performance drop resulting from poisoning depends on how much the attack affects the patterns in the data. In general, once larger amounts of the data become poisoned, the inference process starts to be dominated by the poisoned patterns whereas in smaller amounts they result in distortions that can confuse the model. This pattern is observed with both attacks with the sole exception being blurring at 10% rate. The reason for this exception stems from a small amount of blurring failing to distort the patterns of the litter object.

Analysis of XAI methods: We analyze the effectiveness of XAI methods by considering 10 randomly chosen poisoned samples from each litter category and report the accuracy of estimating the correct class for each sample. Table 8 summarizes the results for the different XAI methods. The effect of poisoning depends on the litter category and the extent of the poisoning. Paper and cardboard objects with regular shapes are easiest for the XAI methods, whereas classes containing irregular shapes (metal, plastic, trash) show the highest variation in results. As with the results for the CNN model, in some cases, a higher level of poisoning can result in a smaller drop or in some cases even an increase in performance. This pattern is more common for steganography as the poisoned data starts to dominate the inference process once a higher fraction of the data is poisoned. The robustness of the model can also be improved by incorporating transformations that resemble the poisoned data as part of the training data. For example, blurring is a commonly used data augmentation technique for improving the training of AI models. From our experiments, we also visually observed that the attacks tend to affect more the background and thus processing techniques that separate the foreground object from background are likely to improve performance.

Table 8: Individual performance of XIA methods on selected samples

	LIME					SHAP					Occlusion Sensitivity				
Poisoning Level	**0%**	**10%**	**20%**	**30%**	**40%**	**0%**	**10%**	**20%**	**30%**	**40%**	**0%**	**10%**	**20%**	**30%**	**40%**
Poisoning type	**Blurring**														
Cardboard	1	0.9	0.9	0.8	0.9	1	0.8	0.8	0.8	0.9	1	0.8	0.9	0.8	0.9
Glass	1	0.7	0.7	0.8	0.6	0.9	0.8	0.8	0.8	0.6	1	0.8	0.8	0.8	0.6
Metal	0.7	0.8	0.7	0.8	0.4	0.6	0.8	0.7	0.8	0.4	0.7	0.7	0.7	0.8	0.4
Paper	0.9	0.9	0.7	0.7	1	0.9	0.9	0.9	0.7	0.9	0.9	0.9	0.9	0.7	1
Plastic	0.8	0.7	0.7	0.7	0.7	0.8	0.7	0.7	0.7	0.7	0.8	0.7	0.7	0.7	0.7
Trash	0.9	0.6	0.6	0.8	0.7	0.9	0.6	0.6	0.8	0.6	0.9	0.6	0.6	0.8	0.7
Average	**0.9**	**0.8**	**0.7**	**0.8**	**0.7**	**0.9**	**0.8**	**0.7**	**0.8**	**0.7**	**0.9**	**0.8**	**0.8**	**0.8**	**0.7**
Poisoning type	**Steganography**														
Cardboard	1	1	1	0.8	0.8	1	1	1	0.8	0.8	1	1	1	0.8	0.8
Glass	1	0.7	0.6	1	1	1	0.6	0.6	1	0.9	1	0.7	0.6	1	0.9
Metal	0.7	0.6	0.6	0.7	0.7	0.7	0.6	0.6	0.7	0.8	0.7	0.6	0.6	0.7	0.8
Paper	0.9	1	1	1	1	0.9	1	1	0.9	0.9	0.9	1	1	0.9	0.9
Plastic	0.8	0.7	0.7	0.7	0.8	0.8	0.7	0.7	0.7	0.8	0.8	0.7	0.7	0.7	0.8
Trash	0.9	0.8	0.6	0.9	1	0.9	0.6	0.6	0.9	0.9	0.9	0.7	0.6	0.9	0.9
Average	**0.9**	**0.8**	**0.8**	**0.9**	**0.9**	**0.9**	**0.8**	**0.8**	**0.8**	**0.9**	**0.9**	**0.8**	**0.8**	**0.8**	**0.9**

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Diagnosing objects with XAI: Lastly, we examine the effect of data poisoning over the important features of the object when it is isolated from the background. As the metric we consider the coefficient of variation of the poisoned pixels, which depicts the ratio of the standard deviation to the mean. The higher the value of the coefficient, the higher the dispersion and thus the better the method is at identifying poisoned data. Figure 33 shows the results for the 10 test samples of each class. For the blurring attack, the average values of the XAI methods are 0.35 (LIME), 0.17 (SHAP) and 0.3 (Occlusion). For data poisoned with steganography, the corresponding values are 0.22 (LIME), 0.10 (SHAP) and 0.26 (Occlusion). One-way ANOVA between the three XAI methods indicates statistical significance, ($F(2,1794)=118.4$, p-value<0.001), indicating that there are differences in the applicability of the different XAI methods. The higher average values of LIME and Occlusion indicate that they generally are better at identifying poisoned data. SHAP performs well for metal objects which are the most irregular but struggles with other categories. We also used one-way ANOVA test to verify that the difference in variation across classes is significant across all XAI methods, poisoning attacks, and levels of poisoning ($F(5,1791)=14.76$, p-value<0.001). Across all XAI methods, the coefficients of variation are larger for steganography than for blurring indicating that XAI methods can also provide clues about the nature of the error. We also investigated the effect between attack type and data poison level. Two-way ANOVA test between attack type and data poisoning level indicates a significant effect ($F(1,4)=3.396$, p-value<0.01), i.e., the coefficients of variation depend not only on the attack type but also the extent of poisoning. Taken together, these results show that XAI methods help to identify the important features of the image, even after data is poisoned but their effectiveness is affected by the object, the type of attack, and the extent of poisoning generated by the attack. In any case, even when the objects can be separated and analyzed, this requires more processing and more elaborate processing pipelines which drains the resources of the drone faster and limits their operations.

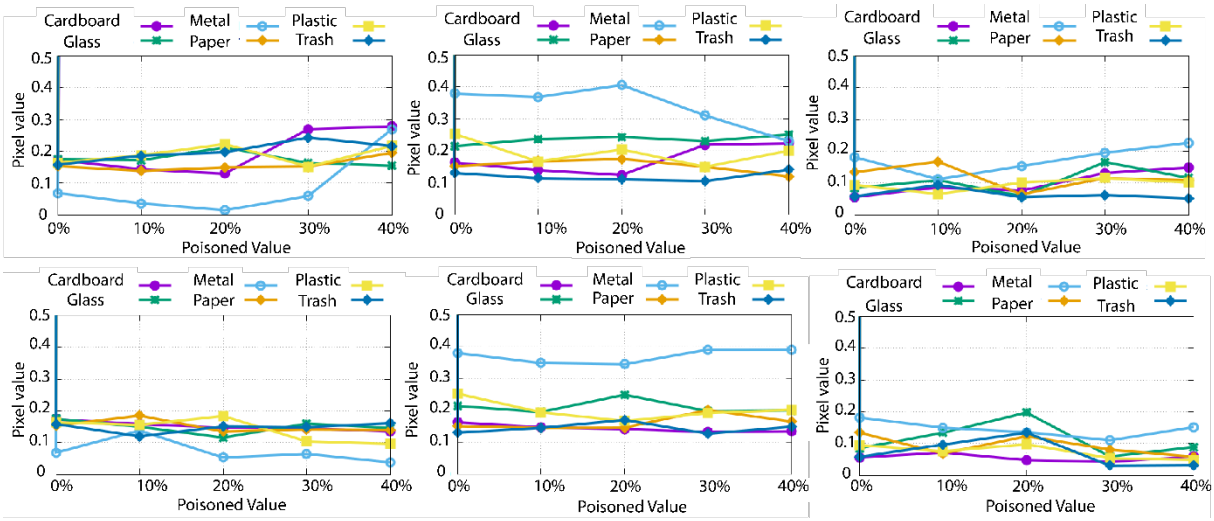


Figure 33: Object analysis with each XAI method as data is poisoned with (a-c) Blurring and (d-f) Steganography object analysis with each XAI method as data is poisoned with (a-c) Blurring and (d-f) Steganography

Lessons learned and experiences with XAI: To analyze AI models running in autonomous drones deployed in the wild. First, it is necessary to produce a reference model from which normal behavior can be characterized. After that, every update in a model that changes its inference process can be just tracked by re-training the model and re-playing the updates.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

Models can be analyzed by XAI methods after each training iteration. This suggests that performing this type of troubleshooting requires access to the data to train the model and the structure of the model to be trained. Moreover, the complexity of the analysis can be resource intensive and require a significant amount of time. In parallel to this, since different autonomous drones can have different model versions over time, troubleshooting becomes specific for each individual drone. Advanced secure communication mechanisms and trusted execution environments could be adopted to upload models to remote locations. The limitations of these mechanisms, however, can still pose problems for achieving large scale deployments in the wild. Indeed, substantial amounts of data to be exchanged and sensitive model information are key issues to consider when analyzing AI model behavior.

5.2. TOWARD GUIDELINES FOR XAI IN DISTRIBUTED MACHINE LEARNING

Based on the insights gained in our experiment**,** we next discuss the implications of using XAI for distributed machine learning and federated environments. As demonstrated in our experiment, there can be various configurations and consequent complexity that can have an impact on users' devices and affect the experience. In our experiment, we explored the impact of important parameters, the degree of data sharing among the services within a user device, and the distribution of the data used for training.

Our experiment shows the different impact of the configuration on the performance of the ML model used in the services, especially in the early phase of the training rounds. There are also additional environmental factors, such as the variations in the pool of devices participating in the federated environment and the consequent diversity of the samples for training. From the users' perspective, it becomes necessary to understand the dynamics and the impact on them.

Considering the complexity and its relation with the experience or quality of the service, providing explanations to the users will be useful for transparency purposes. The expected explanations, however, are different from typical explanations that are made for the outputs of ML models in a number of ways. First, the impact on users rather comes from the federated environment of the ML model than the ML model itself. Therefore, the explanations should be designed to convey the environmental situation. Second, the explanations are provided to end-users, not ML model developers. Thus, the explanations should translate the impact on user devices to user-friendly and intuitive descriptions.

Based on our experiment, we suggest three dimensions for designing the explanations of the federated environment: overall device participation status, consequent data distribution, and local configuration parameters affecting the ML model. The quality of the service (e.g., ML model performance) or the device load status (e.g., resource consumption such as CPU and memory) can be associated with the dimensions to provide the explanations.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

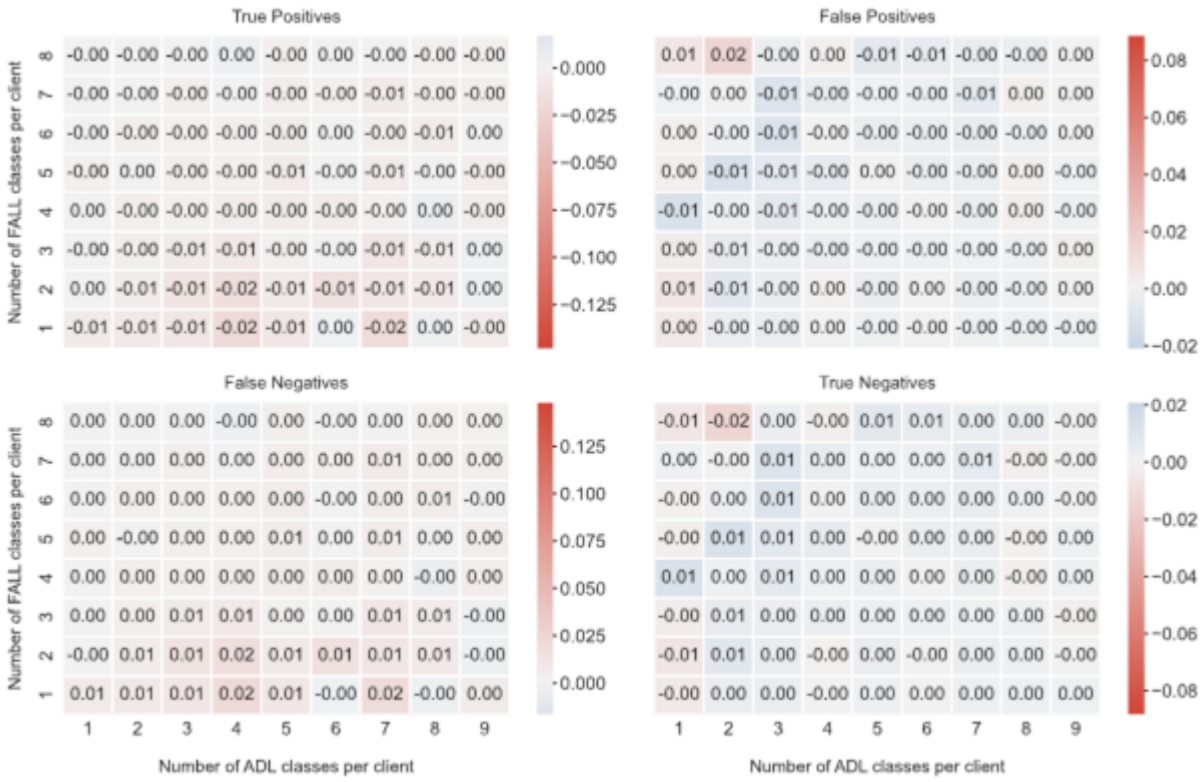
6. CONCLUSIONS

Data quality is fundamental for the training of robust AI models. Besides this, data quality plays an important role in making the AI models trustworthy. Indeed, induced change, like data poisoning and non-induced change, like collection biases, can be introduced to the model over time as the model learns continuously. In this deliverable, we analyzed different data trade-offs that arise when changing different characteristics of data in distributed machine learning. Through rigorous experiments and evaluations that consider different application scenarios, we analyze how changes in data quality impact the model performance. In addition, we analyze how the training process impacts the performance of the devices involved in the training. In parallel to this, we also studied how different methods can be used to detect biases and induced data changes over distributed environments. Since the XAI is a natural point to analyze model performance and evaluate quality of the data, we also presented initial insights that demonstrate to what extent XAI can characterize and monitor changes in data. Lastly, we discussed the implications of this work and provided new insights on how to use XAI to analyze data quality in distrusted AI. Our work paves the way towards understanding how data quality can contribute in achieving trustworthy AI models.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

7. APPENDICES

7.1. APPENDIX A



(b) 8 clients per communication round

Figure 34: Modified confusion matrices visualizing the deviation of the TP, FP, TN, and FN metrics compared to the baseline results when eight (8) participates in a single federated communication round. Negative values indicate a decline in the respective metric.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

7.2. APPENDIX B

Table 9: Summary of effects of poisoning attack on performance metrics

	Attack/ Defence	Poisoning Rate	Accuracy	Precision	Recall	F1 Score	TP	FP	FN	TN
MLP	Random Label Flipping	0.00	0.97	0.97	0.94	0.95	0.34	0.01	0.02	0.63
		0.01	0.96	0.94	0.94	0.94	0.34	0.02	0.02	0.62
		0.05	0.93	0.91	0.91	0.90	0.33	0.03	0.03	0.60
		0.10	0.89	0.85	0.84	0.84	0.30	0.06	0.06	0.58
		0.20	0.78	0.68	0.75	0.70	0.27	0.13	0.09	0.51
		0.30	0.70	0.58	0.71	0.63	0.25	0.19	0.11	0.45
		0.40	0.60	0.47	0.57	0.51	0.21	0.24	0.16	0.39
		0.50	0.50	0.37	0.52	0.43	0.19	0.32	0.17	0.32
	Label Sanitization	0.00	0.95	0.98	0.89	0.93	0.32	0.01	0.04	0.63
		0.01	0.95	0.97	0.88	0.92	0.32	0.01	0.04	0.63
		0.05	0.95	0.97	0.90	0.93	0.32	0.01	0.04	0.63
		0.10	0.95	0.97	0.90	0.93	0.32	0.01	0.04	0.63
		0.20	0.94	0.97	0.88	0.92	0.32	0.01	0.04	0.63
		0.30	0.89	0.87	0.84	0.85	0.30	0.05	0.06	0.59
		0.40	0.77	0.68	0.69	0.68	0.25	0.12	0.11	0.52
		0.50	0.51	0.37	0.50	0.42	0.18	0.31	0.18	0.33
DT	Random Label Flipping	0.00	0.90	0.88	0.84	0.86	0.30	0.04	0.06	0.60
		0.01	0.89	0.86	0.82	0.83	0.30	0.05	0.06	0.59
		0.05	0.85	0.79	0.80	0.79	0.29	0.08	0.07	0.56

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

		0.10	0.81	0.72	0.77	0.74	0.28	0.11	0.08	0.53
		0.20	0.73	0.61	0.69	0.64	0.25	0.16	0.11	0.48
		0.30	0.64	0.50	0.61	0.55	0.22	0.22	0.14	0.42
		0.40	0.57	0.42	0.56	0.48	0.20	0.27	0.16	0.37
		0.50	0.50	0.35	0.48	0.40	0.17	0.31	0.19	0.33
	Label Sanitization	0.00	0.89	0.90	0.80	0.84	0.29	0.04	0.07	0.60
		0.01	0.89	0.90	0.80	0.85	0.29	0.03	0.07	0.61
		0.05	0.89	0.89	0.80	0.84	0.29	0.04	0.07	0.60
		0.10	0.89	0.90	0.80	0.84	0.29	0.03	0.07	0.61
		0.20	0.88	0.88	0.77	0.82	0.28	0.04	0.08	0.60
		0.30	0.82	0.78	0.72	0.75	0.26	0.08	0.10	0.56
		0.40	0.69	0.57	0.67	0.61	0.24	0.19	0.12	0.45
		0.50	0.50	0.36	0.51	0.42	0.19	0.32	0.17	0.31
RF	Random Label Flipping	0.00	0.97	0.97	0.95	0.96	0.34	0.01	0.02	0.63
		0.01	0.97	0.97	0.95	0.96	0.34	0.01	0.02	0.63
		0.05	0.97	0.97	0.96	0.96	0.35	0.01	0.02	0.63
		0.10	0.97	0.96	0.96	0.96	0.34	0.02	0.02	0.62
		0.20	0.96	0.95	0.95	0.95	0.34	0.02	0.02	0.62
		0.30	0.93	0.91	0.91	0.91	0.33	0.03	0.03	0.61
		0.40	0.82	0.74	0.80	0.76	0.29	0.11	0.07	0.53
		0.50	0.51	0.36	0.49	0.41	0.18	0.31	0.18	0.33
	Label	0.00	0.96	0.97	0.92	0.94	0.33	0.01	0.03	0.63

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

	Sanitization	0.01	0.96	0.97	0.91	0.94	0.33	0.01	0.03	0.63
		0.05	0.96	0.97	0.92	0.94	0.33	0.01	0.03	0.63
		0.10	0.96	0.97	0.91	0.94	0.33	0.01	0.03	0.63
		0.20	0.96	0.97	0.91	0.94	0.33	0.01	0.03	0.63
		0.30	0.95	0.97	0.89	0.93	0.32	0.01	0.04	0.63
		0.40	0.91	0.91	0.84	0.87	0.30	0.03	0.06	0.61
		0.50	0.49	0.37	0.49	0.41	0.18	0.32	0.18	0.32
LR	Random Label Flipping	0.00	0.73	0.63	0.46	0.51	0.18	0.09	0.18	0.55
		0.01	0.72	0.62	0.46	0.51	0.17	0.09	0.19	0.55
		0.05	0.72	0.63	0.46	0.51	0.17	0.09	0.19	0.55
		0.10	0.71	0.60	0.46	0.51	0.17	0.10	0.19	0.54
		0.20	0.70	0.58	0.46	0.50	0.17	0.11	0.19	0.53
		0.30	0.66	0.51	0.44	0.47	0.17	0.14	0.19	0.50
		0.40	0.58	0.41	0.44	0.42	0.16	0.22	0.20	0.42
		0.50	0.50	0.36	0.51	0.42	0.18	0.32	0.18	0.32
	Label Sanitization	0.00	0.72	0.66	0.42	0.49	0.16	0.08	0.20	0.56
		0.01	0.73	0.68	0.43	0.51	0.16	0.07	0.20	0.57
		0.05	0.73	0.67	0.43	0.51	0.16	0.08	0.20	0.56
		0.10	0.73	0.67	0.43	0.50	0.16	0.07	0.20	0.56
		0.20	0.73	0.69	0.43	0.51	0.16	0.07	0.20	0.57
		0.30	0.72	0.65	0.42	0.50	0.16	0.08	0.20	0.56
		0.40	0.67	0.55	0.40	0.45	0.15	0.12	0.21	0.52
		0.50	0.53	0.39	0.53	0.44	0.19	0.30	0.17	0.33

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

DNN	Random Label Flipping	0.00	0.97	0.97	0.96	0.96	0.35	0.01	0.01	0.63
		0.01	0.96	0.95	0.95	0.95	0.34	0.02	0.02	0.62
		0.05	0.95	0.92	0.94	0.93	0.34	0.03	0.02	0.61
		0.10	0.90	0.84	0.91	0.87	0.33	0.06	0.03	0.58
		0.20	0.83	0.74	0.86	0.79	0.31	0.11	0.05	0.53
		0.30	0.72	0.59	0.75	0.66	0.27	0.19	0.09	0.45
		0.40	0.65	0.52	0.65	0.57	0.23	0.22	0.13	0.42
		0.50	0.49	0.41	0.47	0.39	0.16	0.31	0.20	0.33
	Label Sanitization	0.00	0.95	0.97	0.88	0.92	0.32	0.01	0.04	0.63
		0.01	0.95	0.98	0.90	0.93	0.32	0.01	0.04	0.63
		0.05	0.95	0.97	0.90	0.93	0.32	0.01	0.04	0.63
		0.10	0.95	0.98	0.90	0.94	0.32	0.01	0.04	0.63
		0.20	0.94	0.97	0.88	0.92	0.32	0.01	0.04	0.63
		0.30	0.92	0.93	0.85	0.88	0.31	0.03	0.06	0.61
		0.40	0.76	0.67	0.73	0.69	0.26	0.14	0.10	0.50
		0.50	0.50	0.36	0.48	0.40	0.17	0.31	0.19	0.33

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

7.3. APPENDIX C

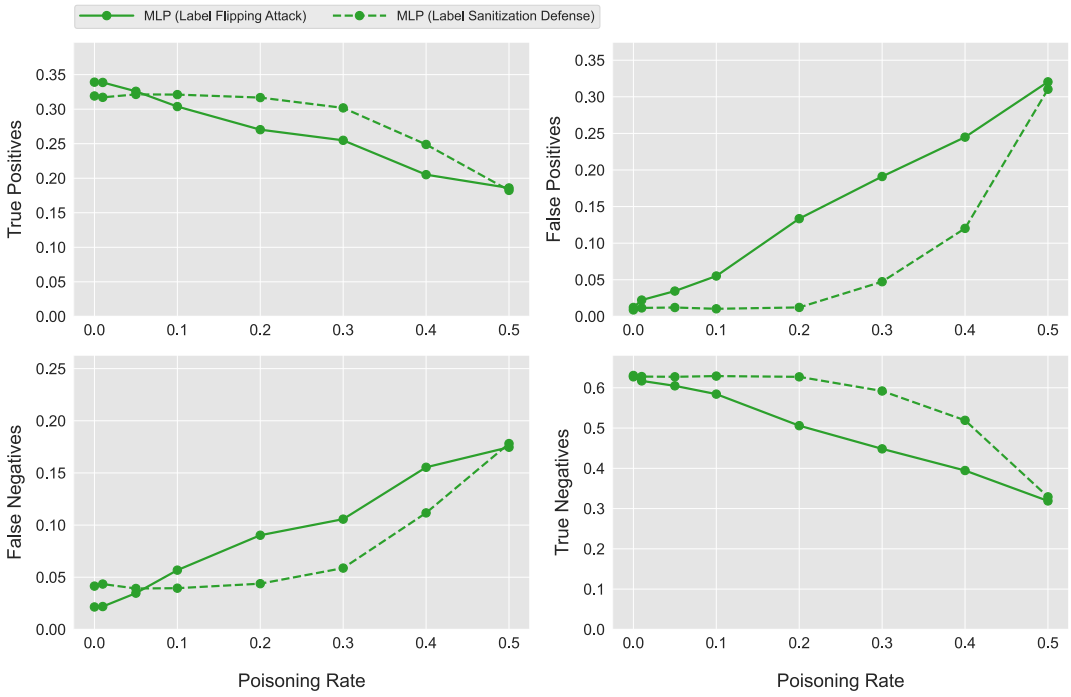


Figure 35: Confusion Matrix in dependence on the poisoning rate of the performed random label flipping attack (straight line) and corresponding label sanitization defence (dashed line) for the MLP model

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

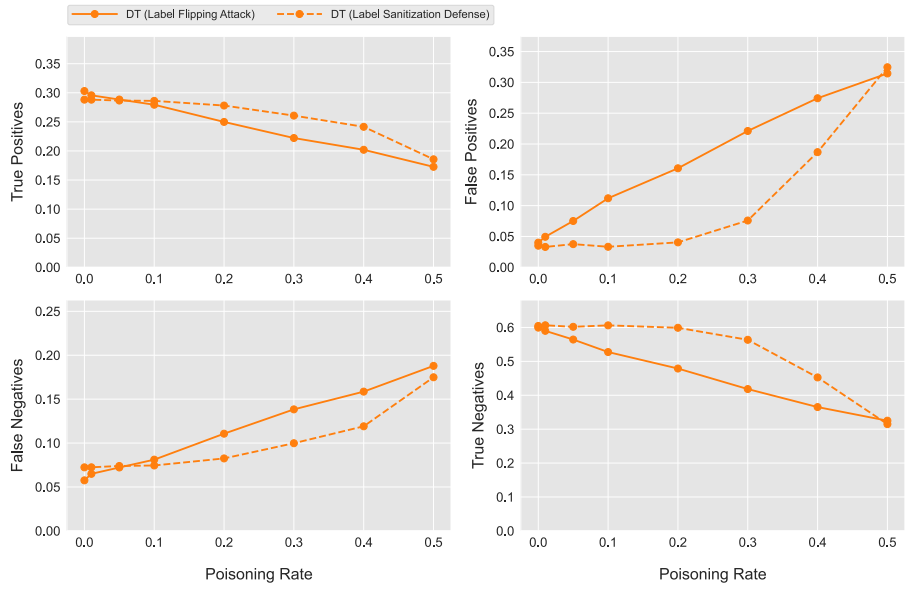


Figure 36: Confusion Matrix in dependence on the poisoning rate of the performed random label flipping attack (straight line) and corresponding label sanitization defence (dashed line) for the DT model.

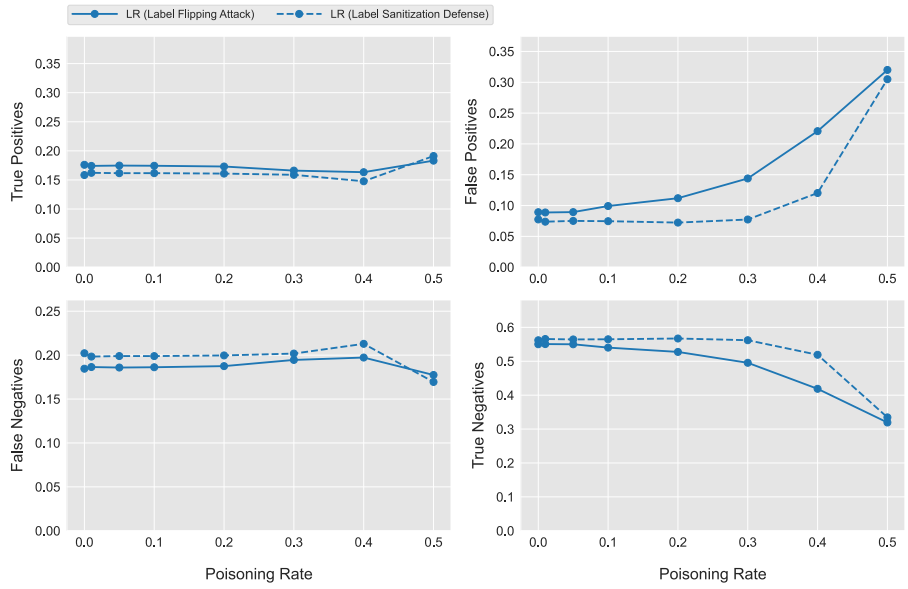
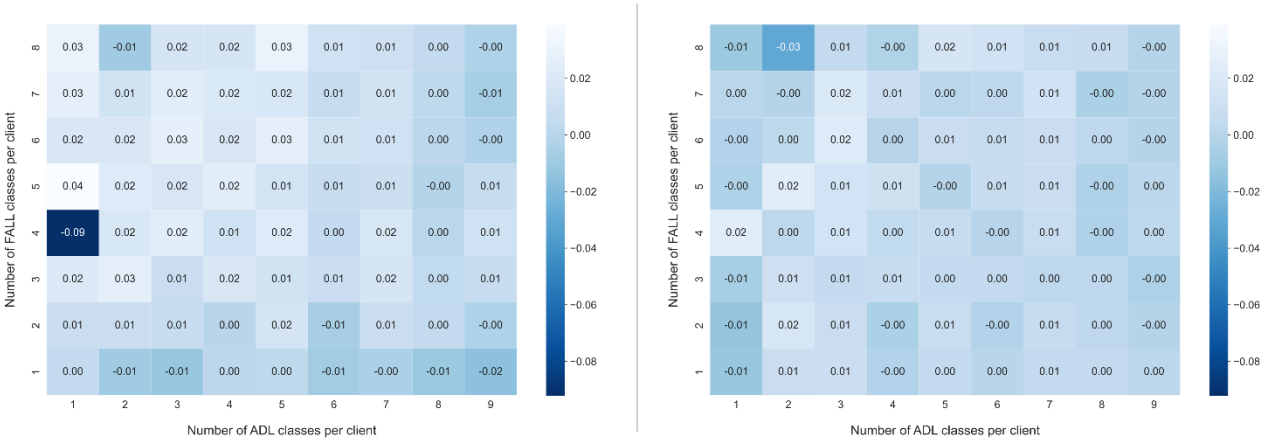


Figure 37: Confusion Matrix in dependence on the poisoning rate of the performed random label flipping attack (straight line) and corresponding label sanitization defence (dashed line) for the LR model.

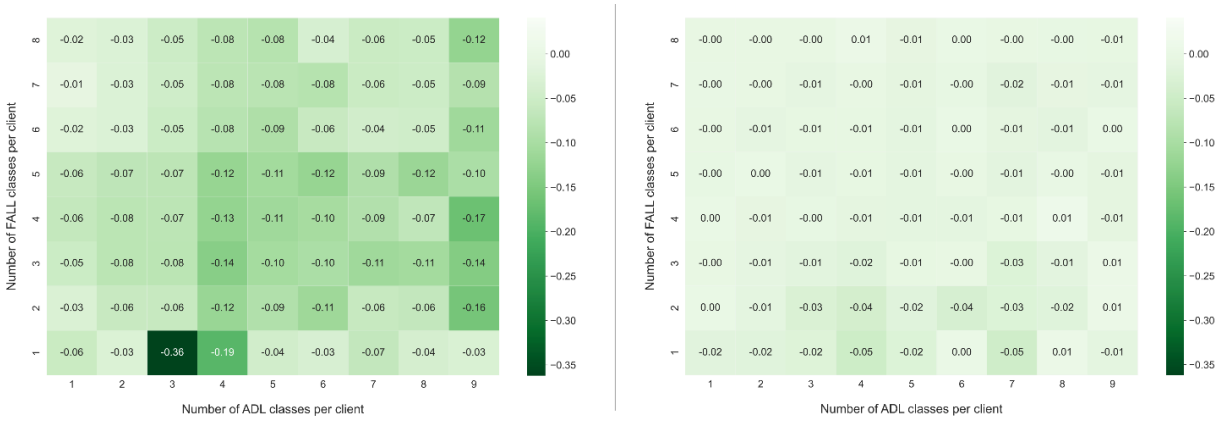
SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.



(a) 1 client per communication round

(b) 8 clients per communication round

Figure 38: Heatmaps visualizing the deviation of the precision compared to the baseline results when (a) only a single and (b) eight random clients participate in a single federated communication round. Negative values indicate a decrease in the precision metric.



(a) 1 client per communication round

(b) 8 clients per communication round

Figure 39: Heatmaps visualizing the deviation of the recall compared to the baseline results when (a) only a single and (b) eight random clients participate in a single federated communication round. Negative values indicate a decrease in the recall metric.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

REFERENCES

[1] M. Yalaoui und S. Boukhedouma, „A survey on data quality: principles, taxonomies and comparison of approaches“, in *2021 International Conference on Information Systems and Advanced Technologies (ICISAT)*, Dez. 2021, S. 1–9. doi: 10.1109/ICISAT54145.2021.9678209.

[2] F. Sidi, P. H. S. Panahy, L. S. Affendey, M. A. Jabar, H. Ibrahim, und A. Mustapha, „Data quality: A survey of data quality dimensions“, in *2012 International Conference on Information Retrieval & Knowledge Management*, 2012, S. 300–304.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, und B. A. y Arcas, „Communication-efficient learning of deep networks from decentralized data“, in *Artificial intelligence and statistics*, 2017, S. 1273–1282.

[4] M. Usman, M. A. Jan, X. He, und J. Chen, „A survey on big multimedia data processing and management in smart cities“, *ACM Comput. Surv. CSUR*, Bd. 52, Nr. 3, S. 1–29, 2019.

[5] M. Ma, P. Wang, und C.-H. Chu, „Data Management for Internet of Things: Challenges, Approaches and Opportunities“, in *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, Aug. 2013, S. 1144–1151. doi: 10.1109/GreenCom-iThings-CPSCom.2013.199.

[6] D. M. Strong, Y. W. Lee, und R. Y. Wang, „Data quality in context“, *Commun. ACM*, Bd. 40, Nr. 5, S. 103–110, 1997.

[7] K. Orr, „Data quality and systems theory“, *Commun. ACM*, Bd. 41, Nr. 2, S. 66–71, 1998.

[8] K. Sha und S. Zeadally, „Data quality challenges in cyber-physical systems“, *J. Data Inf. Qual. JDIQ*, Bd. 6, Nr. 2–3, S. 1–4, 2015.

[9] C. Batini, C. Cappiello, C. Francalanci, und A. Maurino, „Methodologies for data quality assessment and improvement“, *ACM Comput. Surv.*, Bd. 41, Nr. 3, S. 16:1-16:52, Juli 2009, doi: 10.1145/1541880.1541883.

[10] Y. U. Huh, F. R. Keller, T. C. Redman, und A. R. Watkins, „Data quality“, *Inf. Softw. Technol.*, Bd. 32, Nr. 8, S. 559–565, 1990.

[11] S. Liu, Z. Zheng, F. Wu, S. Tang, und G. Chen, „Context-aware data quality estimation in mobile crowdsensing“, in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, Mai 2017, S. 1–9. doi: 10.1109/INFOCOM.2017.8057033.

[12] A. R. Munappy, J. Bosch, H. H. Olsson, A. Arpteg, und B. Brinne, „Data management for production quality deep learning models: Challenges and solutions“, *J. Syst. Softw.*, Bd. 191, S. 111359, Sep. 2022, doi: 10.1016/j.jss.2022.111359.

[13] W. Fan, „Data quality: From theory to practice“, *Acm Sigmod Rec.*, Bd. 44, Nr. 3, S. 7–18, 2015.

[14] A. Y. Sun und B. R. Scanlon, „How can Big Data and machine learning benefit environment and water management: a survey of methods, applications, and future directions“, *Environ. Res. Lett.*, Bd. 14, Nr. 7, S. 073001, 2019.

[15] A. L'heureux, K. Grolinger, H. F. Elyamany, und M. A. Capretz, „Machine learning with big data: Challenges and approaches“, *Ieee Access*, Bd. 5, S. 7776–7797, 2017.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[16] E. Curry, „The big data value chain: definitions, concepts, and theoretical approaches“, *New Horiz. Data-Driven Econ. Roadmap Usage Exploit. Big Data Eur.*, S. 29–37, 2016.

[17] R. Kitchin und G. McArdle, „What makes Big Data, Big Data? Exploring the ontological characteristics of 26 datasets“, *Big Data Soc.*, Bd. 3, Nr. 1, S. 2053951716631130, 2016.

[18] C. Chai, J. Wang, Y. Luo, Z. Niu, und G. Li, „Data management for machine learning: A survey“, *IEEE Trans. Knowl. Data Eng.*, 2022.

[19] P. D. Allison, *Missing Data*. SAGE Publications, 2001.

[20] V. Gudivada, A. Apon, und J. Ding, „Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations“, *Int. J. Adv. Softw.*, Bd. 10, Nr. 1, S. 1–20, 2017.

[21] M. Soley-Bori, „Dealing with missing data: Key assumptions and methods for applied analysis“, *Boston Univ.*, Bd. 4, Nr. 1, S. 19, 2013.

[22] A. Halevy *u. a.*, „Goods: Organizing google's datasets“, in *Proceedings of the 2016 International Conference on Management of Data*, 2016, S. 795–806.

[23] A. R. T. Donders, G. J. Van Der Heijden, T. Stijnen, und K. G. Moons, „A gentle introduction to imputation of missing values“, *J. Clin. Epidemiol.*, Bd. 59, Nr. 10, S. 1087–1091, 2006.

[24] P. J. García-Laencina, J.-L. Sancho-Gómez, A. R. Figueiras-Vidal, und M. Verleysen, „K nearest neighbours with mutual information for simultaneous classification and missing data imputation“, *Neurocomputing*, Bd. 72, Nr. 7–9, S. 1483–1493, 2009.

[25] J. M. Jerez *u. a.*, „Missing data imputation using statistical and machine learning methods in a real breast cancer problem“, *Artif. Intell. Med.*, Bd. 50, Nr. 2, S. 105–115, 2010.

[26] S. I. Khan und A. S. M. L. Hoque, „SICE: an improved missing data imputation“, 2020.

[27] I. Myrtveit, E. Stensrud, und U. H. Olsson, „Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods“, *IEEE Trans. Softw. Eng.*, Bd. 27, Nr. 11, S. 999–1013, 2001.

[28] X. L. Dong und F. Naumann, „Data fusion: resolving data conflicts for integration“, *Proc. VLDB Endow.*, Bd. 2, Nr. 2, S. 1654–1655, 2009.

[29] M. D. Mambe, S. Oumtanaga, und G. N. Anoh, „A belief entropy-based approach for conflict resolution in IoT applications“, in *2018 1st International Conference on Smart Cities and Communities (SCCIC)*, 2018, S. 1–5.

[30] W. Fan, F. Geerts, S. Ma, N. Tang, und W. Yu, „Data quality problems beyond consistency and deduplication“, in *In Search of Elegance in the Theory and Practice of Computation*, Springer, 2013, S. 237–249.

[31] X. Ding, H. Wang, Y. Gao, J. Li, und H. Gao, „Determining the currency of dynamic data“, in *Proceedings of the ACM Turing 50th Celebration Conference-China*, 2017, S. 1–6.

[32] L. Brkić und I. Mekterović, „A time-constrained algorithm for integration testing in a data warehouse environment“, *Inf. Technol. Control*, Bd. 47, Nr. 1, S. 5–25, 2018.

[33] X. Kou, X. Duan, X. Zhao, und Y. Han, „Data Repair Method based on Timeliness and Conditional Function Dependency Rules“, in *Proceedings of the 2019 8th International Conference on Computing and Pattern Recognition*, 2019, S. 57–64.

[34] B. Saha und D. Srivastava, „Data quality: The other face of big data“, in *2014 IEEE 30th international conference on data engineering*, 2014, S. 1294–1297.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[35] J. Park, A. Patel, D. Curtis, S. Teller, und J. Ledlie, „Online pose classification and walking speed estimation using handheld devices“, in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, New York, NY, USA, Sep. 2012, S. 113–122. doi: 10.1145/2370216.2370235.

[36] Y. Xu, X. Zhu, J. Shi, G. Zhang, H. Bao, und H. Li, „Depth Completion From Sparse LiDAR Data With Depth-Normal Constraints“, gehalten auf der Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, S. 2811–2820. Zugegriffen: 10. Januar 2023. [Online]. Verfügbar unter: https://openaccess.thecvf.com/content_ICCV_2019/html/Xu_Depth_Completion_From_Sparse_LiDAR_Data_With_Depth-Normal_Constraints_ICCV_2019_paper.html

[37] Y. Roh, G. Heo, und S. E. Whang, „A survey on data collection for machine learning: a big data-ai integration perspective“, *IEEE Trans. Knowl. Data Eng.*, Bd. 33, Nr. 4, S. 1328–1347, 2019.

[38] „Premise - Data for Every Decision™“. https://www.premise.com/ (zugegriffen 8. November 2022).

[39] M. J. Cafarella, A. Halevy, und N. Khoussainova, „Data integration for the relational web“, *Proc. VLDB Endow.*, Bd. 2, Nr. 1, S. 1090–1101, 2009.

[40] H. Park und J. Widom, „Crowdfill: collecting structured data from the crowd“, in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, S. 577–588.

[41] C. Olston, F. Korn, N. Noy, N. Polyzotis, S. Whang, und S. Roy, „Managing Google’s data lake: an overview of the Goods system“, 2016.

[42] M. Chew und J. D. Tygar, „Image recognition captchas“, in *International Conference on Information Security*, 2004, S. 268–279.

[43] M. Hirth, T. Hoßfeld, und P. Tran-Gia, „Anatomy of a crowdsourcing platform-using the example of microworkers. com“, in *2011 Fifth international conference on innovative mobile and internet services in ubiquitous computing*, 2011, S. 322–329.

[44] K. Woodward, E. Kanjo, A. Oikonomou, und A. Chamberlain, „LabelSens: enabling real-time sensor data labelling at the point of collection using an artificial intelligence-based approach“, *Pers. Ubiquitous Comput.*, Bd. 24, Nr. 5, S. 709–722, Okt. 2020, doi: 10.1007/s00779-020-01427-x.

[45] Z. Jia, S. Lin, C. R. Qi, und A. Aiken, „Exploring Hidden Dimensions in Parallelizing Convolutional Neural Networks.“, in *ICML*, 2018, S. 2279–2288.

[46] A. I. Hleg, „Ethics guidelines for trustworthy AI“, *B-1049 Bruss.*, 2019.

[47] E. Parliament, „Data Act | Legislative Train Schedule“, *European Parliament*. https://www.europarl.europa.eu/legislative-train/theme-a-europe-fit-for-the-digital-age/file-data-act (zugegriffen 14. Februar 2023).

[48] I. Shumailov, Y. Zhao, D. Bates, N. Papernot, R. Mullins, und R. Anderson, „Sponge examples: Energy-latency attacks on neural networks“, in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2021, S. 212–231.

[49] A.-R. Ottun *u. a.*, „Toward Trustworthy and Responsible Autonomous Drones in Future Smart Cities“, 2022.

[50] H. He und E. A. Garcia, „Learning from imbalanced data“, *IEEE Trans. Knowl. Data Eng.*, Bd. 21, Nr. 9, S. 1263–1284, 2009.

SPATIAL project is funded by the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[51] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, und F. Herrera, *Learning from imbalanced data sets*, Bd. 10. Springer, 2018.

[52] C. Elkan, „The foundations of cost-sensitive learning“, in *International joint conference on artificial intelligence*, 2001, Bd. 17, Nr. 1, S. 973–978.

[53] C. X. Ling und V. S. Sheng, „Cost-sensitive learning and the class imbalance problem“, *Encycl. Mach. Learn.*, Bd. 2011, S. 231–235, 2008.

[54] „sklearn.metrics.fbeta_score“, *scikit-learn*. https://scikit-learn/stable/modules/generated/sklearn.metrics.fbeta_score.html (zugegriffen 17. November 2022).

[55] T. Tommasi, N. Patricia, B. Caputo, und T. Tuytelaars, „A deeper look at dataset bias“, in *Domain adaptation in computer vision applications*, Springer, 2017, S. 37–55.

[56] E. Webber, „Dive into Bias Metrics and Model Explainability with Amazon SageMaker Clarify“, *Medium*, 11. Dezember 2020. https://towardsdatascience.com/dive-into-bias-metrics-and-model-explainability-with-amazon-sagemaker-clarify-473c2bca1f72 (zugegriffen 17. November 2022).

[57] N. Japkowicz und S. Stephen, „The class imbalance problem: A systematic study“, *Intell. Data Anal.*, Bd. 6, Nr. 5, S. 429–449, 2002.

[58] „Measure Pretraining Bias - Amazon SageMaker“. https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-measure-data-bias.html (zugegriffen 17. November 2022).

[59] Y. Huang, „Kullback-Leibler (KL) Divergence and Jensen-Shannon Divergence“, 8. Juli 2020. https://yongchaohuang.github.io/2020-07-08-kl-divergence/ (zugegriffen 17. November 2022).

[60] S. Barocas und A. D. Selbst, „Big data’s disparate impact“, *Calif Rev*, Bd. 104, S. 671, 2016.

[61] „Disparate Impact (DI) - Amazon SageMaker“. https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-di.html (zugegriffen 17. November 2022).

[62] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, und S. Venkatasubramanian, „Certifying and removing disparate impact“, in *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, S. 259–268.

[63] J. Buolamwini und T. Gebru, „Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification“, S. 77–91, Jan. 2018.

[64] M. Hardt, „Understanding unintended sources of unfairness in data driven decision making“.

[65] A. Olteanu, C. Castillo, F. Diaz, und E. Kıcıman, „Social data: Biases, methodological pitfalls, and ethical boundaries“, *Front. Big Data*, Bd. 2, S. 13, 2019.

[66] A. Torralba und A. A. Efros, „et almbox. 2011. Unbiased look at dataset bias“, in *IEEE conference on computer vision and pattern recognition (CVPR). Google Scholar Google Scholar Digital Library Digital Library*.

[67] H. Singh, R. Singh, V. Mhasawade, und R. Chunara, „Fairness violations and mitigation under covariate shift“, in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 2021, S. 3–13.

[68] „Definition of FAIRNESS“. https://www.merriam-webster.com/dictionary/fairness (zugegriffen 17. November 2022).

SPATIAL project is funded by the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[69] C. O'Sullivan, „Analysing Fairness in Machine Learning (with Python)", *Medium*, 2. September 2022. https://towardsdatascience.com/analysing-fairness-in-machine-learning-with-python-96a9ab0d0705 (zugegriffen 17. November 2022).

[70] B. Catania, G. Guerrini, und C. Accinelli, „Fairness & friends in the data science era", *AI Soc.*, S. 1–11, 2022.

[71] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, und R. Zemel, „Fairness through awareness", in *Proceedings of the 3rd innovations in theoretical computer science conference*, 2012, S. 214–226.

[72] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, und A. Galstyan, „A survey on bias and fairness in machine learning", *ACM Comput. Surv. CSUR*, Bd. 54, Nr. 6, S. 1–35, 2021.

[73] D. Pessach und E. Shmueli, „Algorithmic fairness", *ArXiv Prepr. ArXiv200109784*, 2020.

[74] J. R. Loftus, C. Russell, M. J. Kusner, und R. Silva, „Causal reasoning for algorithmic fairness", *ArXiv Prepr. ArXiv180505859*, 2018.

[75] J. Pearl, „Causal diagrams for empirical research", *Biometrika*, Bd. 82, Nr. 4, S. 669–688, 1995.

[76] G. Vivaldi, „Simpson's Paradox", *Medium*, 18. Mai 2019. https://towardsdatascience.com/simpsons-paradox-decae6a4de88 (zugegriffen 17. November 2022).

[77] M. Silva, „The 10 Bias and Causality Techniques of that Everyone Needs to Master", *Medium*, 19. November 2019. https://towardsdatascience.com/the-10bias-and-causality-techniques-of-that-everyone-needs-to-master-6d64dc3a8d68 (zugegriffen 17. November 2022).

[78] A. Rothman, „Causal Inference in Data Science: Structure of M-Bias with Confounding Adjustment", *Medium*, 29. Dezember 2020. https://towardsdatascience.com/causal-inference-in-data-science-structure-of-m-bias-with-confounding-adjustment-70e4a263ad08 (zugegriffen 17. November 2022).

[79] H. Flores *u. a.*, „COSINE: Collaborator selector for cooperative multi-device sensing and computing", in *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2020, S. 1–10.

[80] S. AbdulRahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, und M. Guizani, „A survey on federated learning: The journey from centralized to distributed on-site learning and beyond", *IEEE Internet Things J.*, Bd. 8, Nr. 7, S. 5476–5497, 2020.

[81] D. Poulopoulos, „Distributed Deep Learning 101: Introduction", *Medium*, 26. Januar 2021. https://towardsdatascience.com/distributed-deep-learning-101-introduction-ebfc1bcd59d9 (zugegriffen 17. November 2022).

[82] R. D. PhD MD, „Multi-GPU Training in Pytorch", *Medium*, 5. März 2020. https://towardsdatascience.com/multi-gpu-training-in-pytorch-dbdb3389fd4a (zugegriffen 17. November 2022).

[83] D. Y. Zhang, Z. Kou, und D. Wang, „Fairfl: A fair federated learning approach to reducing demographic bias in privacy-sensitive classification models", in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, S. 1051–1060.

[84] A. Abay, Y. Zhou, N. Baracaldo, S. Rajamoni, E. Chuba, und H. Ludwig, „Mitigating bias in federated learning", *ArXiv Prepr. ArXiv201202447*, 2020.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[85] M. S. Ozdayi und M. Kantarcioglu, „The Impact of Data Distribution on Fairness and Robustness in Federated Learning“, in *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, 2021, S. 191–196.

[86] L. Ferraguig, Y. Djebrouni, S. Bouchenak, und V. Marangozova, „Survey of Bias Mitigation in Federated Learning“, in *Conférence francophone d'informatique en Parallélisme, Architecture et Système*, 2021.

[87] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, und A. Galstyan, „A survey on bias and fairness in machine learning“, *ACM Comput. Surv. CSUR*, Bd. 54, Nr. 6, S. 1–35, 2021.

[88] P. K. Lohia, K. N. Ramamurthy, M. Bhide, D. Saha, K. R. Varshney, und R. Puri, „Bias mitigation post-processing for individual and group fairness“, in *Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*, 2019, S. 2847–2851.

[89] G. Thung und M. Yang, „Classification of Trash for Recyclability Status; CS229 Project Report“. Stanford University: Palo Alto, CA, USA, 2016.

[90] Abdul-Rasheed Ottun *u. a.*, „MODENA: Model Diagnostics for Identifying Poisoning Attacks in Multi-Drone Systems“.

[91] T. Li, A. K. Sahu, A. Talwalkar, und V. Smith, „Federated Learning: Challenges, Methods, and Future Directions“, *IEEE Signal Process. Mag.*, Bd. 37, Nr. 3, S. 50–60, Mai 2020, doi: 10.1109/MSP.2020.2975749.

[92] D. Micucci, M. Mobilio, und P. Napoletano, „UniMiB SHAR: A Dataset for Human Activity Recognition Using Acceleration Data from Smartphones“, *Appl. Sci.*, Bd. 7, Nr. 10, Art. Nr. 10, Okt. 2017, doi: 10.3390/app7101101.

[93] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, und M. Guizani, „A Survey on Federated Learning: The Journey From Centralized to Distributed On-Site Learning and Beyond“, *IEEE Internet Things J.*, Bd. 8, Nr. 7, S. 5476–5497, Apr. 2021, doi: 10.1109/JIOT.2020.3030072.

[94] N. Kourtellis, K. Katevas, und D. Perino, „Flaas: Federated learning as a service“, in *Proceedings of the 1st workshop on distributed machine learning*, 2020, S. 7–13.

[95] A. Krizhevsky und G. Hinton, „Learning multiple layers of features from tiny images“, 2009.

[96] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, und L.-C. Chen, „Mobilenetv2: Inverted residuals and linear bottlenecks“, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, S. 4510–4520.

[97] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, und L. Fei-Fei, „Imagenet: A large-scale hierarchical image database“, in *2009 IEEE conference on computer vision and pattern recognition*, 2009, S. 248–255.

[98] Y. Fu, C. Li, F. R. Yu, T. H. Luan, und Y. Zhang, „A survey of driving safety with sensing, vehicular communications, and artificial intelligence-based collision avoidance“, *IEEE Trans. Intell. Transp. Syst.*, 2021.

[99] N. H. Motlagh *u. a.*, „Toward Blue Skies: City-Scale Air Pollution Monitoring using UAVs“, *IEEE Consum. Electron. Mag.*, 2022.

[100] Z. Yin *u. a.*, „Toward City-Scale Litter Monitoring Using Autonomous Ground Vehicles“, *IEEE Pervasive Comput.*, 2022.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[101] H. Flores *u. a.*, „Toward large-scale autonomous marine pollution monitoring“, *IEEE Internet Things Mag.*, Bd. 4, Nr. 1, S. 40–45, 2021.

[102] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, und F. Turini, „Meaningful explanations of black box AI decision systems“, in *Proceedings of the AAAI conference on artificial intelligence*, 2019, Bd. 33, Nr. 01, S. 9780–9784.

[103] R. A. Aral, Ş. R. Keskin, M. Kaya, und M. Hacıömeroğlu, „Classification of trashnet dataset based on deep learning models“, in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, S. 2058–2062.

[104] J.-E. Ekberg, K. Kostiainen, und N. Asokan, „The untapped potential of trusted execution environments on mobile devices“, *IEEE Secur. Priv.*, Bd. 12, Nr. 4, S. 29–37, 2014.

[105] H. Fereidooni *u. a.*, „SAFELearn: secure aggregation for private federated learning“, in *2021 IEEE Security and Privacy Workshops (SPW)*, 2021, S. 56–62.

[106] A. Taïk, H. Moudoud, und S. Cherkaoui, „Data-quality based scheduling for federated edge learning“, in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, 2021, S. 17–23.

[107] Q. Liu *u. a.*, „When deep learning meets steganography: Protecting inference privacy in the dark“, in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, 2022, S. 590–599.

[108] R. Zhu, K. Yin, H. Xiong, H. Tang, und G. Yin, „Masked face detection algorithm in the dense crowd based on federated learning“, *Wirel. Commun. Mob. Comput.*, Bd. 2021, 2021.

[109] M. Sajjad *u. a.*, „An efficient and scalable simulation model for autonomous vehicles with economical hardware“, *IEEE Trans. Intell. Transp. Syst.*, Bd. 22, Nr. 3, S. 1718–1732, 2020.

[110] J. N. Kather, N. Halama, und A. Marx, „100,000 histological images of human colorectal cancer and healthy tissue (2018)“, *DOI Httpsdoi Org105281zenodo*, Bd. 1214456, 2018.

[111] J. Yang, R. Shi, und B. Ni, „Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis“, in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, 2021, S. 191–195.

[112] L. Deng, „The mnist database of handwritten digit images for machine learning research [best of the web]“, *IEEE Signal Process. Mag.*, Bd. 29, Nr. 6, S. 141–142, 2012.

[113] G. Van Rossum und F. L. Drake Jr, „Python reference manual (Centrum voor Wiskunde en Informatica)“. Amsterdam Netherlands, 1995.

[114] T. Developers, „TensorFlow“, *Zenodo Doi*, Bd. 10, 2021.

[115] F. Pedregosa *u. a.*, „Scikit-learn: Machine learning in Python“, *J. Mach. Learn. Res.*, Bd. 12, S. 2825–2830, 2011.

[116] K. He, X. Zhang, S. Ren, und J. Sun, „Deep residual learning for image recognition“, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, S. 770–778.

[117] eceisik, „fl_public“. 27. Oktober 2022. Zugegriffen: 17. November 2022. [Online]. Verfügbar unter: https://github.com/eceisik/ece_fl_public

[118] R. Yadav, A. Vierling, und K. Berns, „Radar+ rgb attentive fusion for robust object detection in autonomous vehicles“, *ArXiv Prepr. ArXiv200813642*, 2020.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.

[119] S. Madan *u. a.*, „When and how do CNNs generalize to out-of-distribution category-viewpoint combinations?“, *ArXiv Prepr. ArXiv200708032*, 2020.

[120] I. Rafegas, M. Vanrell, L. A. Alexandre, und G. Arias, „Understanding trained CNNs by indexing neuron selectivity“, *Pattern Recognit. Lett.*, Bd. 136, S. 318–325, 2020.

[121] B. Zhou, Y. Sun, D. Bau, und A. Torralba, „Revisiting the importance of individual units in cnns via ablation“, *ArXiv Prepr. ArXiv180602891*, 2018.

[122] H. Caesar *u. a.*, „nuscenes: A multimodal dataset for autonomous driving“, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, S. 11621–11631.

[123] S. Patil, S. Suryanarayana, R. Dinesh, N. Shivraj, und N. Murthy, „Risk factors for falls among elderly: A community-based study“, *Int. J. Health Allied Sci.*, Bd. 4, Nr. 3, S. 135–135, 2015.

[124] S. P. Baker und A. H. Harvey, „Fall injuries in the elderly“, *Clin. Geriatr. Med.*, Bd. 1, Nr. 3, S. 501–512, 1985.

[125] E. Burns und R. Kakara, „Deaths from falls among persons aged≥ 65 years—United States, 2007–2016“, *Morb. Mortal. Wkly. Rep.*, Bd. 67, Nr. 18, S. 509, 2018.

[126] D. P. Kingma und J. Ba, „Adam: A Method for Stochastic Optimization“. arXiv, 29. Januar 2017. doi: 10.48550/arXiv.1412.6980.

[127] „scikit-learn: machine learning in Python — scikit-learn 1.2.0 documentation“. https://scikit-learn.org/stable/index.html (zugegriffen 9. Januar 2023).

[128] A. Paudice, L. Muñoz-González, und E. C. Lupu, „Label sanitization against label flipping poisoning attacks“, in *Joint European conference on machine learning and knowledge discovery in databases*, 2018, S. 5–15.

[129] A. R. Shahid, A. Imteaj, P. Y. Wu, D. A. Igoche, und T. Alam, „Label Flipping Data Poisoning Attack Against Wearable Human Activity Recognition System“, *ArXiv Prepr. ArXiv220808433*, 2022.

[130] Muccini, H., & Vaidhyanathan, K. (2021, May). Software architecture for ML-based systems: What exists and what lies ahead. In 2021 IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN) (pp. 121-128). IEEE.

[131] D. Katare, N. Kourtellis, S. Park, D. Perino, M. Janssen, A.Y. Ding, "Bias Detection and Generalization in AI Algorithms on Edge for Autonomous Driving", in Proceedings of the Seventh ACM/IEEE Symposium on Edge Computing (SEC), 2022.

SPATIAL project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101021808.