Federated Learning Strategies for Enhanced Quality Inspection Processes in Manufacturing Industry

Vom Fachbereich Maschinenbau und Verfahrenstechnik

der Rheinland-Pfälzischen Technischen Universität Kaiserslautern-Landau

zur Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

genehmigte

Dissertation

von

Herrn

M.Sc. Vinit Hegiste

aus Mumbai, Indien

Tag der mündlichen Prüfung: 20.02.2026

Dekan:	Prof. Dr. rer. nat. Roland Ulber

Promotionskommision:

Vorsitzende:	Prof. Dr.-Ing. Kristin de Payrebrune
1. Berichterstatter:	Prof. Dr.-Ing. Martin Ruskowski
2. Berichterstatter:	Prof. Dr.-Ing. Patrick Rüdiger-Flore

D 386

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und ohne unerlaubte, fremde Hilfe angefertigt habe. Alle Ausführungen, die aus anderen Schriften wörtlich oder sinngemäß übernommen wurden, sind kenntlich gemacht. Alle verwendeten Quellen sind im Literaturverzeichnis zitiert.

Kaiserslautern, den 05. März 2026

_____

Vinit Vikas Hegiste

Acknowledgements

I never imagined that I would one day complete a PhD. However, with a lot of support, hard work, and perseverance, I have finally reached this point. This journey has been filled with many challenges and learning experiences, from sleepless nights spent working on research to attending conferences and meeting inspiring people who shared their knowledge and passion for their work. Along the way, I realized how important patience and persistence are in research. Most importantly, I was fortunate to have the support of many wonderful people who made this journey possible.

I would like to express my heartfelt gratitude to Professor Dr.-Ing. Martin Ruskowski, my doctoral supervisor (Doktorvater), for his valuable guidance and expertise throughout my doctoral journey. I am especially grateful for the trust he placed in me and for giving me the freedom to explore and pursue research in a new and challenging field. His encouragement and support were essential in shaping this work, and I feel truly privileged to have conducted my research under his supervision. I would also like to sincerely thank Prof. Dr.-Ing. Patrick Rüdiger-Flore, the second supervisor of my thesis, for his valuable feedback and support during the completion of this work. My sincere appreciation also goes to M.Sc. Simon Bergweiler, my PhD advisor, whose continuous guidance and readiness to discuss ideas greatly supported me throughout this research. His insights and encouragement were very important during the different stages of my doctoral work. Furthermore, I would like to thank Prof. Dr.-Ing. Kristin de Payrebrune for kindly agreeing to chair the doctoral committee and for supporting the successful completion of this doctoral examination process.

Furthermore, I would like to extend my appreciation to all my colleagues at WSKL / SmartFactory KL, who supported me in many ways during my PhD journey. Over the years, they became much more than colleagues and truly felt like a second family. From Feierabendbier to birthday celebrations and our Pizza and Flammkuchen nights, these moments created many wonderful memories that made this journey even more special. I am also grateful for the many conversations and for patiently helping me improve my German along the way. A heartfelt thank you to Tatjana, Julian, Jonathan, Khalil, Simon, Patrick, Mario, Leo, Thomas, Magnus, Nigora, Jonas, and Vassili for the supportive and friendly work environment and for making every day at the institute enjoyable. I would also like to sincerely thank Isabel and Julia from HR, who were always incredibly helpful and supportive whenever needed. I would also like to thank my colleagues at DFKI, especially Snehal, Jibiraj, Ali, and Hooman, for the great research collaborations and the

many insightful discussions during our weekly meetings. These exchanges were very valuable and contributed significantly to the research presented in this thesis. Beyond research, many shared experiences made this journey even more memorable, from Hannover Messe (especially the Beckhoff party) to quiz nights and the summer and winter institute parties. These moments added a lot of joy to the PhD journey and created memories that I will always cherish.

Regarding the topic of federated learning, I would like to express my special gratitude to Tatjana Legler for her constant support and collaboration throughout my PhD journey. From brainstorming research ideas to co-authoring many of the research papers in this thesis, her guidance and expertise played a major role in shaping this work. I am deeply thankful to have had such an inspiring mentor and team leader. Working together on research papers, traveling to conferences, and having countless discussions about ideas and challenges made this journey much easier and far more enjoyable. Having her as a discussion partner and mentor was truly invaluable. I am sincerely grateful for her continuous guidance, encouragement, and dedication throughout these years.

From the bottom of my heart, I would like to thank my family, my mother Pragati Hegiste, my father Vikas Hegiste, and my brother Atharva Hegiste, for their endless love, encouragement, and support throughout my life. Being far away from home while pursuing this journey was never easy, but their constant belief in me gave me the strength to keep going. I will always be grateful for the sacrifices my parents have made and for everything they have done to help me reach this point. Without them, this would never have been possible. I would also like to thank my best friend Kinjal Marwadi for always being there for me and supporting me from afar during all these years. A special thank you to my friends in Germany, who have been like a small family to me since my Bachelor of Engineering days in Mumbai. Having them here made life away from home much easier and filled with laughter and support. Thank you Adnan, Mayuri, Mayank, Karande, Uzma, and Drushti for being part of this journey and for all the wonderful memories we created together.

Lastly, I would like to express my deepest gratitude to my life partner and constant source of strength, my wife Mrs. Rhythm Arora-Hegiste. Without her support, this PhD would not have been possible. Thank you for all the love, encouragement, and motivation you have given me throughout this journey. I am especially grateful for the sacrifices you made in your own career so that I could pursue and complete this PhD. You were always there to lift me up during the difficult moments and to celebrate the good ones with me. Words cannot truly express how much this means to me, but I can simply say: Thank you, and I Love You.

To all those mentioned above, as well as everyone else who contributed to this work in any way, I offer my sincere thanks. Your support, encouragement, and collaboration played an important role in the successful completion of this thesis.

Table of Contents

Abstract

The increasing reliance on vision-based deep learning for automated quality inspection in manufacturing emphasizes the need for collaborative model development across distributed production sites. Also, legal, privacy, and proprietary constraints often prevent centralized aggregation of such industrial data. This thesis investigates the use of Federated Learning (FL) as a decentralized framework for training high-performance computer vision models without sharing raw data. A modular and scalable FL architecture is proposed, supporting both image classification and object detection under non-Identically and Independent Distributed (non-IID) client distributions, and leveraging lightweight, high-accuracy deep learning models suitable for industrial deployment.

The experimental validation covers diverse industrial scenarios, including USB port and cabin windshield classification, as well as YOLO-based object detection for localizing quality-relevant components. The framework is further extended to hybrid setups that incorporate synthetic and real clients to address data scarcity, and validated through live inference across heterogeneous domains. In addition, a novel Federated Ensemble (FedEnsemble) strategy is introduced, in which a centralized dataset is partitioned into clients for federated training, yielding models that outperform conventional centralized baselines under domain shift.

The main contributions of this thesis are:
(i) a validated FL framework tailored for privacy-preserving quality inspection in industrial environments,
(ii) empirical insights into training dynamics with a small number of heterogeneous clients,
(iii) evidence of robust cross-domain generalization through hybrid federated setups combining real and synthetic data, and
(iv) the introduction of FedEnsemble as a practical extension of FL that improves detection robustness in deployment.
Together, these findings establish FL as a viable foundation for building distributed, scalable, and secure AI systems in smart manufacturing, enabling the development of robust and generalizable quality inspection models.

Kurzfassung

Die zunehmende Abhängigkeit von vision-based Deep Learning Systemen für die automatisierte Qualitätsinspektion in der Fertigung betont die Notwendigkeit einer kollaborativen Modellentwicklung über verteilte Produktionsstandorte hinweg. Rechtliche, datenschutzrechtliche und proprietäre Einschränkungen verhindern jedoch häufig die zentrale Aggregation industrieller Daten. Diese Dissertation untersucht den Einsatz von Federated Learning (FL) als dezentralem Framework zum Training leistungsfähiger Computer-Vision-Modelle, ohne dass Rohdaten geteilt werden müssen. Hierfür wird eine modulare und skalierbare FL-Architektur vorgeschlagen, die sowohl Bildklassifikation als auch Objekterkennung unter non-IID-Verteilungen der Clients unterstützt und auf leichte, hochgenaue Deep-Learning-Modelle setzt, die für industrielle Anwendungen geeignet sind.

Die experimentelle Validierung umfasst verschiedene industrielle Szenarien, darunter die Klassifikation von USB-Anschlüssen und Kabinen-Windschutzscheiben sowie die YOLO-basierte Objekterkennung zur Lokalisierung qualitätsrelevanter Komponenten. Darüber hinaus wird das Framework auf hybride Setups erweitert, die synthetische und reale Clients kombinieren, um Datenknappheit zu überwinden, und durch Live-Inferenz über heterogene Domänen hinweg evaluiert. Zudem wird eine neuartige FedEnsemble-Strategie eingeführt, bei der ein zentrales Datenset in mehrere Clients partitioniert und föderiert trainiert wird. Diese Strategie führt zu Modellen, die konventionelle zentralisierte Baselines unter Domänenverschiebungen übertreffen.

Die Hauptbeiträge dieser Arbeit sind:
(i) ein validiertes FL-Framework, zugeschnitten auf datenschutzfreundliche Qualitätsinspektion in industriellen Umgebungen.
(ii) empirische Erkenntnisse über Trainingsdynamiken mit einer kleinen Anzahl heterogener Clients.
(iii) Nachweise für robuste Domänen-Generalisation durch hybride föderierte Setups mit realen und synthetischen Daten.
(iv) die Einführung von FedEnsemble als praxisnahe Erweiterung von FL, die die Robustheit der Objekterkennung im Einsatz verbessert.
Zusammengenommen etablieren diese Ergebnisse FL als tragfähige Grundlage für den Aufbau verteilter, skalierbarer und sicherer KI-Systeme in der intelligenten Fertigung und ermöglichen die Entwicklung robuster und generalisierbarer Qualitätsinspektionsmodelle.

List of Abbreviations

AGVs	Automated Guided Vehicles	5
AI	Artificial Intelligence	1
AP	Average Precision	78
CNN	Convolution Neural Network	11
CR	Communication Round	34
CSP	Cross Stage Partial	18
DL	Deep Learning	1
DP	Differential Privacy	3
FedAvg	Federated Averaging	11
FedBE	Federated Bayesian Ensembles	28
FedEnsemble	Federated Ensemble	VII
FedMA	Federated Matched Averaging	24
FedNova	Federated Normalized Averaging	25
FedOD	Federated Object Detection	8
FedProx	Federated Proximal	24
FL	Federated Learning	VII
FPS	Frames Per Second	19
GDPR	General Data Protection Regulation	1
HE	Homomorphic Encryption	34
IID	Independent and Identically Distributed	64
IoU	Intersection-over-Union	16
mAP	mean Average Precision	8
ML	Machine Learning	1
NMS	Non-Maximum Suppression	16
NN	Neural Network	12
non-IID	non-Identically and Independent Distributed	VII
PANet	Path Aggregation Network	18
R-CNN	Region-based Convolutional Neural Network	15
RPTU	Rheinland-Pfälzische Technische Universität	120
SMPC	Secure Multi-Party Computation	3
SOTA	State of the Art	8
SPPF	Spatial Pyramid Pooling Fast	18
SVM	Support Vector Machine	12
YOLO	You Only Look Once	15

x

1 Introduction

The manufacturing industry is undergoing a rapid digital transformation, with technologies like automation, Artificial Intelligence (AI), and Machine Learning (ML) playing a critical role in maintaining efficiency, competitiveness, and precision [Wa21]. Among the most significant areas benefiting from this technological shift is quality inspection, where AI-based systems ensure that products meet stringent standards before they reach consumers. According to a recent report, large industrial facilities lose more than a day's worth of production each month due to machine failures, amounting to an estimated $864 billion annually for Fortune Global 500 manufacturing and industrial firms [Se21]. This underscores the critical role of efficient quality inspection systems in preventing costly errors and maintaining product reliability.

Traditionally, quality inspection relies on centralized machine learning models, which posed several challenges related to data handling, privacy, and communication overhead in distributed manufacturing environments. In recent years, Deep Learning (DL) has witnessed a surge in popularity due to the availability of fast and cost-effective GPUs, online platforms like Google Colab, AWS, and the proliferation of open-source datasets such as ImageNet [De09], VOC Pascal [Ev10], and CIFAR-10 [Kr09]. These advancements have enabled the development of complex DL architectures such as VGG [SZ15], ResNet [He16], DenseNet [Hu17], EfficientNet [TL19], etc., which have achieved high accuracy in a wide range of applications, including face recognition, multi object classification, action recognition, and more.

However, despite these advances, there is still a significant limitation in terms of data privacy, in terms of training dataset. For DL models to be trained effectively, large datasets from different sources need to be merged (centralized) in a large dataset. This raises serious privacy concerns, particularly in industrial contexts where data sharing among multiple entities, such as manufacturing plants or partner companies, can potentially lead to data breaches or theft of proprietary information [NSH18; Sh17]. The issue becomes even more pronounced when multiple companies collaborate, as one party may exploit the shared data, leading to a loss of intellectual property or competitive advantage [Ao17; Bo15].

In response to these concerns, legislative frameworks like the *General Data Protection Regulation (GDPR)* [VV17] have been enacted in European countries, making it legally challenging for organizations to share sensitive data. These regulations create additional barriers to data-sharing practices and hinder collaborative efforts that could otherwise lead to more powerful machine learning models.

This is where Federated Learning (FL) [Mc17] proves to be a revolutionary solution. FL is a decentralized machine learning approach in which the training data remains distributed across various devices or local servers (e.g., mobile devices, edge devices, or manufacturing plants), and only the locally computed model updates/weights are shared with a central server for aggregation. In this way, FL facilitates the collaborative development of a shared machine learning model without the need to transfer sensitive data, ensuring that the data stays on the local devices while contributing to the global model.

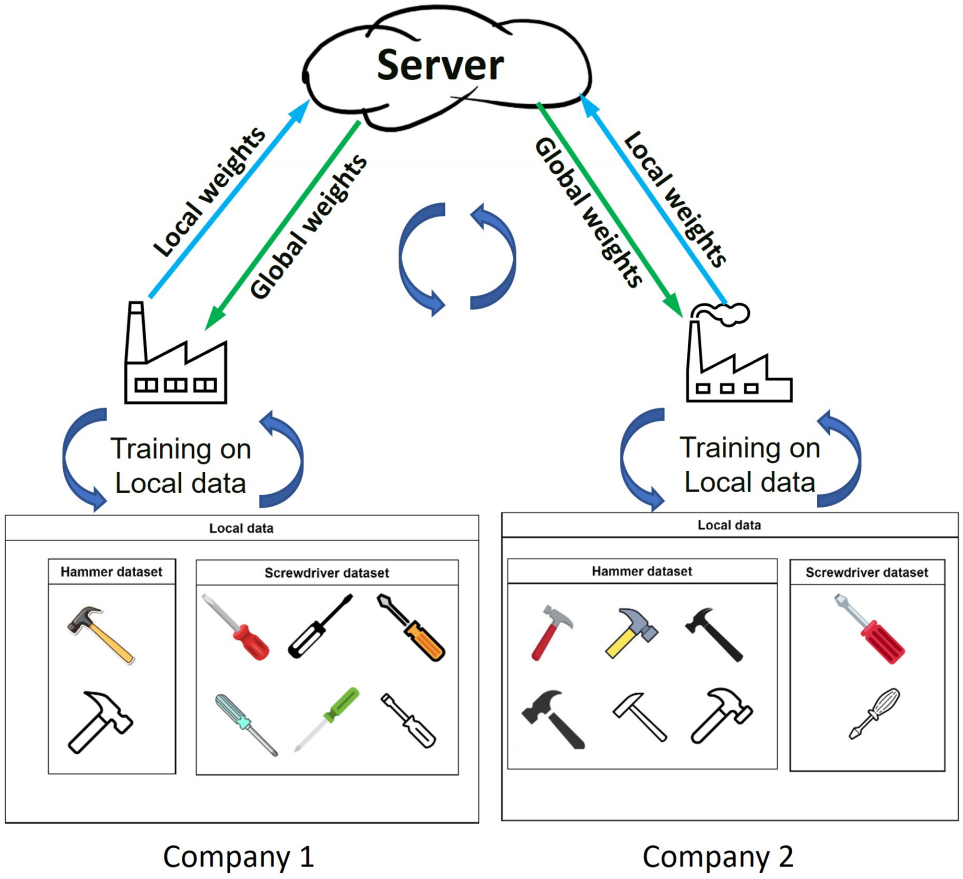

Figure 1.1: Illustration of horizontal FL between two companies producing the same product types in different quantities. Local datasets remain private while model updates are exchanged with a central server to train a global model.

FL is particularly relevant in manufacturing ecosystems where companies face similar tasks such as predictive maintenance, defect classification, and quality inspection, but are unable or unwilling to share raw data due to competitive, legal, or privacy constraints. Figure 1.1 illustrates such a scenario: Company 1 primarily produces screwdrivers with fewer hammer samples, while Company 2 manufactures more hammers but fewer screwdrivers. As a result, models trained locally by each company are biased toward their dominant class. If Company 1 were later required to classify hammers, its locally trained model would underperform due to insufficient representation of that class. Instead of investing additional resources in data collection and retraining, both companies can collaborate through FL to train a global model. This shared

model benefits from the diversity of both datasets, resulting in balanced performance across screwdrivers and hammers, without requiring either company to share its raw data. Furthermore, security advancements in FL techniques, such as Differential Privacy (DP) [Ab16; DR14] and Secure Multi-Party Computation (SMPC) [Bo17], enhances FL ability to protect sensitive data while maintaining high model performance, even in distributed and heterogeneous environments [Ka21a].

This thesis explores the application of FL strategies to enhance quality inspection processes in manufacturing. Specifically, it focuses on applying FL to image classification and object detection tasks, which are essential for identifying defects and anomalies in products. By developing and evaluating FL architectures for these tasks, this research aims to provide practical, scalable solutions for manufacturers that rely on distributed and heterogeneous data environments, where data privacy is of paramount concern. Parts of the research presented in this thesis have already been peer-reviewed and accepted for publication at international IEEE conferences [He23; He24; HLR22; HLR23; HLR24], demonstrating the scientific relevance and practical value of the developed methods.

1.1 Problem Definition

While DL has proven highly effective in automating quality inspection processes, the deployment of machine learning models in collaborative, multilocation manufacturing environments presents several challenges. Traditional centralized models require data from different sites to be collected and transferred to a central server for training, which is often impractical due to the following reasons:

- **Data privacy concerns**: Manufacturers work with proprietary and sensitive data related to their products, processes, or clients. Sharing this data with external entities or centralized servers poses significant risks to intellectual property and security. For example, a car manufacturer may hesitate to share defect detection data with suppliers due to fears of intellectual property theft or competitive disadvantage.
- **Communication overhead**: Transferring large datasets from multiple locations to a central server requires significant bandwidth and results in high communication costs, especially when data volumes are substantial. For instance, industrial-grade imaging systems used for quality inspection can generate terabytes of data daily, making centralized merging od data both time-consuming and cost-prohibitive.
- **Data heterogeneity**: Different manufacturing plants often use distinct equipment, produce various products, and implement unique inspection procedures. This leads to heterogeneous datasets that can negatively impact the performance of centralized models, which may fail to generalize across all sites. For example, factories may utilize different

types of sensors, resulting in data that varies in resolution, format, and distribution, making it challenging to create a unified model.

These challenges highlight the need for a decentralized machine learning approach that addresses the unique requirements of manufacturing quality inspection without compromising data privacy, communication efficiency, or model accuracy. FL offers a promising solution by allowing multiple manufacturing sites to train models locally while sharing only the model weights (and no raw training data) with a central server for aggregation.

However, research on the practical application of FL in real-world manufacturing environments, particularly in the context of quality inspection tasks, remains limited. Existing work as has largely focused on benchmark datasets such as MNIST or CIFAR-10, which do not accurately reflect the complexities of industrial settings [We23]. This gap highlights the need for research into FL architectures and strategies specifically tailored for industrial applications, paving the way for this thesis to explore these challenges and propose viable solutions.

1.1.1 Collaboration in Manufacturing

Collaboration among manufacturing companies to train machine learning models is inherently challenging due to several critical factors, primarily rooted in concerns over data privacy, market competition, and regulatory compliance. Companies are often reluctant to share data with third-party organizations or even other entities within the same industry for three main reasons:

1. **Competition in the Market**: Manufacturing is a highly competitive industry where organizations invest heavily in proprietary processes, product designs, and quality control systems. Sharing data can reveal critical operational insights, potentially eroding competitive advantages. For example, a semiconductor manufacturer may hesitate to share data about defect patterns with competitors, fearing the potential loss of proprietary knowledge and its edge in the market.
2. **Fear of Sharing Company Secrets**: Data collected in manufacturing environments often includes sensitive information about production methods, machinery performance, and defect detection processes. Companies are concerned that sharing such data, even for collaborative learning purposes, might expose trade secrets or proprietary knowledge. For instance, a supplier hesitant to share defect patterns with a manufacturing client risks exposing details about their proprietary quality control processes or production techniques.
3. **Lack of Trust and Understanding**: A major barrier to collaboration is the uncertainty regarding what happens to data once it is shared. Many companies lack sufficient understanding of how their data will be handled by third parties, leading to apprehension about unintended data leaks or misuse. For example, a lack of transparency in data handling

protocols can discourage small manufacturers from participating in data-sharing initiatives, even if it promises significant collaborative benefits.

Furthermore, European regulations such as the GDPR [VV17] impose stringent requirements on data privacy and usage, which discourages data sharing even further [HLR22; Mc17]. Similar frameworks, such as the *California Consumer Privacy Act (CCPA)* [Ca18], have also been introduced in the United States to regulate data sharing, emphasizing the global nature of these privacy concerns.

As mentioned above, FL presents a promising solution to address these challenges by enabling collaborative model training without requiring companies to share their raw data. In FL, participants collaboratively train a global model by transmitting only model updates (e.g., gradients or weights) to a central server, thereby preserving data privacy and ensuring compliance with data protection regulations [Ka20; Wa20b].

The decentralized nature of FL allows companies to maintain control over their sensitive information while benefiting from collaborative efforts. By participating in FL, multiple clients can contribute to the development of superior models that generalize better across diverse environments and applications. This enhanced collaboration can drive significant advancements in key areas such as:

- **Autonomous Driving**: Aggregating data from different environments improves object detection, path planning, and safety systems [SBD21].
- **Energy Management**: Collaborative learning across energy providers optimizes load balancing and renewable energy integration [Kh22].
- **Path Planning**: Federated models improve navigation algorithms for Automated Guided Vehicles (AGVs) in smart factories [LWL19].
- **Large Language Models**: Cross-enterprise FL enhances domain-specific language models while respecting privacy [Ya24].
- **Predictive Maintenance**: FL allows companies to pool insights on machinery failure patterns without exposing individual machine data [Ko23].
- **Quality Inspection**: Aggregating data from various production lines improves defect detection models across different manufacturing setups [HLR22].

In summary, FL fosters trust in collaborative AI development by offering privacy-preserving mechanisms and compliance with legal frameworks, enabling industries to reap the benefits of collective intelligence without compromising data confidentiality.

1.1.2 Federated Learning Research in Manufacturing

The focus of this research is placed on FL in manufacturing, where additional challenges related to its implementation in industrial use cases are explored. The discussion is initiated with the most fundamental aspect in ML, namely the ***dataset***. Most of the research is currently based on benchmark datasets such as MNIST or CIFAR-10, which are not realistic datasets as they comprise image sizes of 28×28 and 32×32 pixels, respectively [Li20; Mc17; Wa20a; Ya19]. While these datasets are useful for demonstrating the power of FL, they do not adequately reflect the complexities of real-world manufacturing scenarios. Applications such as ***quality inspection*** require higher-quality images (with higher resolutions) to classify or detect errors on a product with high accuracy and confidence. One significant challenge of implementing FL in manufacturing is dealing with fewer clients compared to general FL use cases. Unlike traditional FL applications that involve thousands or millions of clients (e.g., smartphones in mobile networks), manufacturing scenarios typically involve a limited number of participating entities, such as factories, production lines, or suppliers [Ya19]. This smaller number of clients makes it more challenging to generalize federated models effectively, while also increasing the risk of model bias when certain clients dominate the dataset. Another critical challenge is managing non-IID datasets in industrial setups. For instance, different factories may use distinct sensors, machinery, or inspection protocols, resulting in data distributions that vary significantly across clients. Current studies in FL show that it also works with heterogenous datasets, but lacks in details studies in manufacturing use cases.[Li20].

Specific datasets relevant to manufacturing, such as proprietary defect detection datasets or sensor generated time-series data, highlight these challenges. For example, in a manufacturing quality inspection use case, datasets may include high-resolution images of product defects captured under varying lighting conditions or from different camera angles [HLR22]. The heterogeneity of these datasets requires advanced FL strategies that account for client-specific variations while still achieving robust global model performance. When comparing FL research in manufacturing to other industries, such as healthcare or finance, unique challenges become evident. In healthcare, FL has been successfully applied to tasks like medical image analysis and predictive diagnostics, often benefiting from relatively standardized data formats (e.g., DICOM images) [Ri20]. In finance, FL has been used for fraud detection and credit scoring, where numerical data is often more uniform across institutions [Ka21a]. In contrast, manufacturing environments are far less standardized, with greater variability in data types, quality, and formats across clients. Furthermore, there is currently limited research that directly compares the performance of client-specific models with global federated models in manufacturing. Such comparisons are essential to demonstrate the practical value of FL in industrial settings. For example, manufacturers would benefit from understanding how FL models outperform or complement traditional, localized models when applied to tasks like defect detection or predictive maintenance [HLR22].

1.2 Objectives

The primary objective of this thesis is to assess the feasibility and potential of FL as a solution to address the unique challenges associated with quality inspection in manufacturing, as outlined in Section 1.1. However, FL itself presents challenges when adapting to manufacturing use cases, as discussed in Subsections 1.1.1 and 1.1.2. These include managing non-IID data, handling a limited number of clients, and ensuring compliance with data privacy regulations.

To address these, the following key research questions are formulated:

Question 1: Is FL a practically feasible and effective approach for real-world, custom quality inspection use cases in manufacturing, particularly where client data is independently collected and heterogeneous?

This question explores whether FL can effectively handle the distinct requirements of manufacturing environments, including the management of custom non-IID data distributions from diverse manufacturing setups, processing of larger input sizes critical for high-resolution quality inspection tasks, and deployment in scenarios with a limited number of participating clients.

Question 2: Building upon the insights gained from Question 1, can FL be extended to more complex computer vision tasks, such as object detection, where objects must not only be classified but also accurately localized within the frame?

Object detection introduces additional complexities, particularly in ensuring precise spatial localization of defects or components. This question investigates whether FL can scale and adapt to meet the demanding requirements of advanced applications in industrial contexts [He23; HLR23].

These two main questions give rise to more specific sub-questions, which focus on critical aspects of FL implementation in manufacturing:

1. **What architectural and system-level design considerations arise when deploying FL in manufacturing environments with few clients?** Unlike general FL applications involving thousands or millions of clients (e.g., mobile devices), manufacturing environments typically feature a limited number of participants, such as individual factories or production lines. This thesis examines the design and implications of FL architectures tailored for such scenarios, focusing on their impact on model convergence, performance, and scalability [Li20; Ya19].
2. **What practical benefits can FL provide to manufacturing companies?** For manufacturing companies to adopt FL, it is essential to demonstrate tangible benefits. This thesis conducts a comparative analysis between FL models and traditional client-specific models, evaluating improvements in accuracy, generalizability, and data privacy. The goal is to provide actionable insights that highlight the value of FL in enhancing operational effi-

ciency and decision-making processes [HLR22].

This thesis aims to bridge these research gaps by investigating FL architectures and methodologies specifically designed for manufacturing environments. By leveraging real-world datasets and addressing critical challenges such as limited client participation and non-IID data distributions, the study offers practical insights into the applicability and deployment of FL in industrial contexts.

1.3 Procedure

The research questions outlined in Section 1.2 form the foundation of this dissertation and guide the organization of the subsequent chapters. Each chapter is strategically designed to address a specific aspect of the problem statement and contribute to the overall objective of validating FL for quality inspection in industrial environments. Chapter 2 establishes the theoretical and technical foundations underpinning this research by introducing key concepts in DL and FL. It outlines essential definitions, architectural paradigms, and recent State of the Art (SOTA) advancements relevant to vision-based inspection systems. The chapter further presents a structured review of contemporary literature, highlighting persistent challenges such as data heterogeneity, communication efficiency, and privacy constraints. Through this analysis, critical research gaps are identified, thereby framing the motivation and direction for the contributions presented in this thesis.

Chapter 3 outlines the core methodological framework of the thesis. It details the practical implementation of FL techniques tailored to industrial computer vision use cases. This includes federated image classification, Federated Object Detection (FedOD), ensemble-based FL, and experiments incorporating synthetic datasets as synthetic client in FL process. The chapter discusses model architectures, dataset preparation, training strategies, and federated aggregation mechanisms. Each methodology is designed with scalability, privacy, and real-world feasibility in mind, ensuring its relevance for deployment in manufacturing systems. Chapter 4 provides an empirical evaluation of the methodologies described in the previous chapter. It analyzes the performance of each use case using quantitative metrics such as accuracy, mean Average Precision (mAP), and model convergence. Detailed comparisons are drawn between centralized, local, and federated approaches, with a specific focus on how FL impacts generalization under non-IID settings. This chapter serves to validate the effectiveness of the proposed FL frameworks in achieving the research objectives.

Finally, Chapter 5 summarizes the key findings and contributions of the thesis. It highlights how the research questions have been addressed through experimental and theoretical analysis. The chapter also discusses limitations encountered during implementation such as computational constraints, annotation challenges, or architectural limitations, and presents potential

directions for future work. These include exploring advanced privacy-preserving techniques, applying FL to more complex object detection tasks, and extending the framework to other industrial domains. The goal is to ensure the scalability and adaptability of federated machine learning in real-world production environments.

2 Fundamentals and State of the Art

This chapter provides a comprehensive foundation for the Deep Learning (DL) and Federated Learning (FL) techniques used in this thesis. It begins by reviewing the evolution of CNN-based architectures for image classification and object detection, including recent advances such as the YOLO family of models that are optimized for real-time performance. It then presents the architectural principles and algorithmic extensions of FL, including baseline methods like Federated Averaging (FedAvg) and more recent variants designed to address data heterogeneity, client drift, and communication constraints.

Finally, the chapter surveys the state of the art in applying FL to manufacturing use cases. It highlights pioneering studies in federated defect prediction, predictive maintenance, and industrial condition monitoring, while identifying key research gaps. These include the limited availability of real-world multi-client datasets, the underexplored application of FL in image classification and object detection, and the lack of robust benchmarking against local and centralized models. Addressing these gaps is a central motivation for the experimental work presented in later chapters and answering the research questions in this thesis.

2.1 Image Classification

Image classification is a foundational task in computer vision, aiming to categorize an input image into one of several predefined classes. This capability has enabled transformative applications across various domains, including autonomous systems, medical diagnostics, manufacturing, etc. In industrial settings, image classification is instrumental in automating quality inspection processes, detecting visual defects, and identifying anomalous patterns in products and components [Ke24].

The evolution of image classification methods has paralleled advances in ML, beginning with early rule-based and statistical approaches, followed by classical ML techniques, and culminating in the adoption of DL architectures. The advent of DL, particularly Convolution Neural Networks (CNNs), has revolutionized the field by allowing models to learn directly from raw pixel data, eliminating the need for manual feature engineering, and significantly improving performance on complex datasets [Ch25].

2.1.1 Neural Networks

In the early stages of ML, image classification tasks were typically addressed using traditional algorithms such as Support Vector Machines (SVMs) and decision trees. Although effective in certain domains, these methods often struggled with high-dimensional inputs like images due to their limited capacity to model complex patterns and spatial hierarchies [Bi06].

This limitation catalyzed the development of Neural Networks (NNs), which introduced a more flexible, layer-based learning paradigm. NNs are loosely inspired by the structure of the human brain and consist of interconnected artificial neurons arranged in sequential layers. Each neuron applies a non-linear activation function to its input, and the resulting output is propagated forward through the network. A basic NN typically comprises the following components:

- **Input Layer**: Receives the raw input data (e.g., pixel values of an image).
- **Hidden Layers**: Transform the input data through a series of learned weights and activation functions to capture intermediate features.
- **Output Layer**: Produces the final prediction, such as a class label, based on the transformed features.

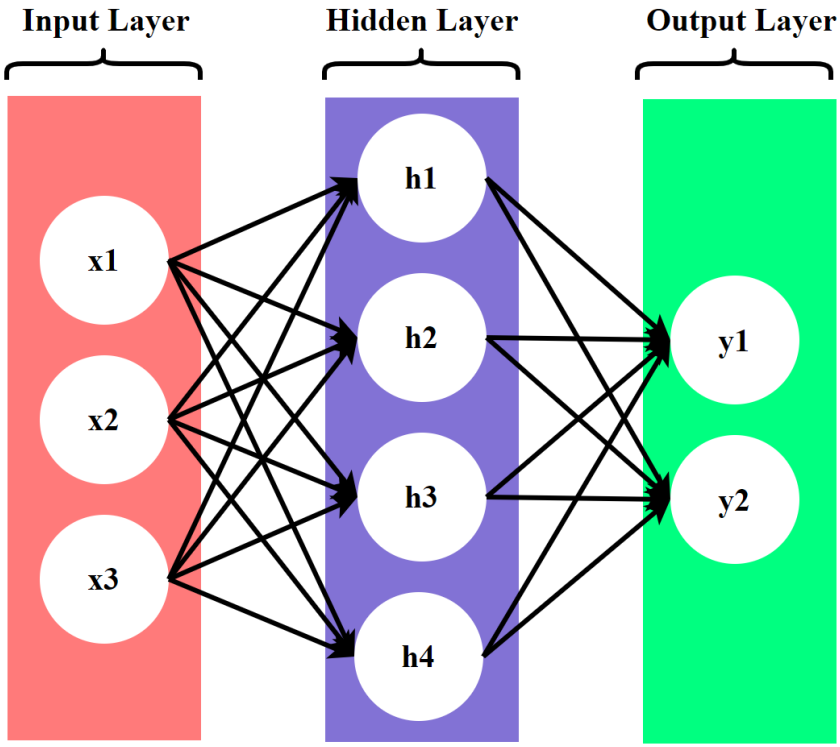

Figure 2.1: Simple Neural Network with one hidden layer for binary classification [GBC16].

Figure 2.1 illustrates a simple feedforward NN with one hidden layer, and an output layer depicting a binary classification. The learning process in NNs is enabled by the backpropagation algorithm, which iteratively adjusts the weights based on the gradient of a loss function with respect to each parameter. A comprehensive overview of this technique is provided in the book

Deep Learning by Goodfellow et al. [GBC16]. Despite their flexibility, traditional NNs exhibit certain limitations when applied to image data. Specifically, they treat each pixel as an independent feature, disregarding the spatial correlations inherent in visual information. This shortcoming led to the development of CNNs, which are explicitly designed to exploit spatial hierarchies in images.

2.1.2 Convolutional Neural Networks

CNNs were introduced to address the shortcomings of traditional NNs in handling spatial data like images. Unlike NNs, which process images as a flat array of pixels, CNNs are designed to recognize spatial hierarchies in images. By applying convolutional filters to the input data, CNNs can automatically detect patterns such as edges, textures, and more complex structures [Le98]. This also led to reduced training and inference time for CNNs models as compared to NNs for images with larger dimensions. As shown in Fig. 2.2, a typical CNN include convolutional layers, which apply kernel filters that slide over the input image to detect local patterns; pooling layers, which perform downsampling operations to reduce the dimensionality of feature maps and enhance computational efficiency; and fully connected layers, which take the extracted high-level features and produce final class predictions.

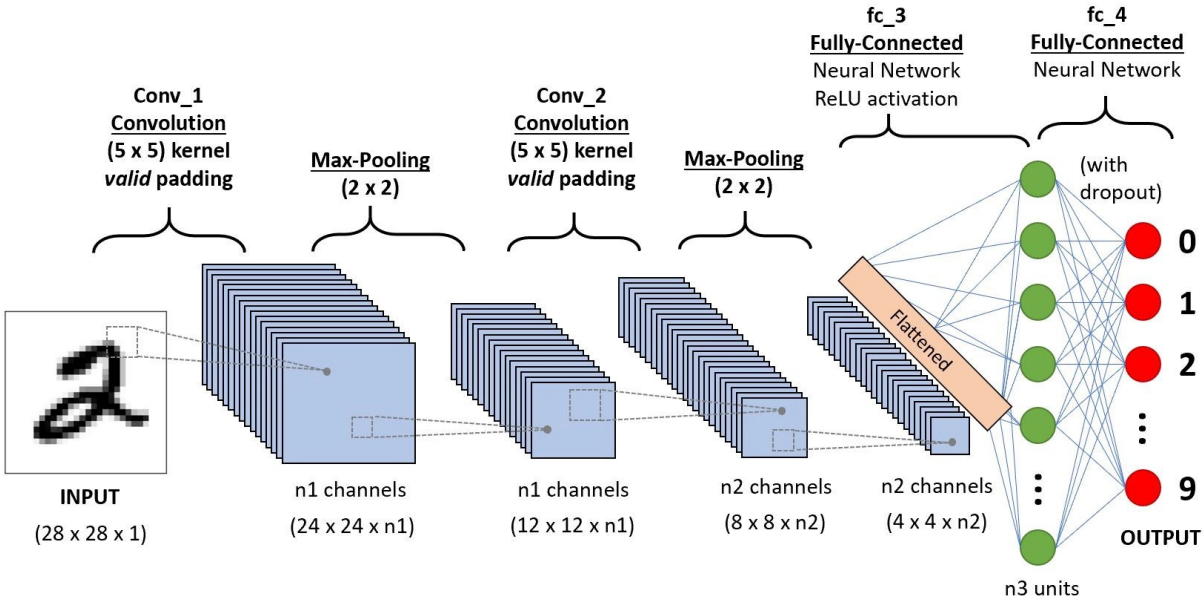

Figure 2.2: Illustration of a basic Convolutional Neural Network architecture[Ra20].

The innovation of CNNs significantly boosted the performance of image classification models. However, as the complexity of image classification tasks grew, so did the need for deeper and more powerful networks. This paved the way for the development of deep CNNs.

2.1.3 Deep Convolutional Neural Networks

While CNNs are highly effective at detecting low- and mid-level features such as edges or textures, more complex tasks such as recognizing objects across varying positions, scales, or orientations requires the network to capture higher-level abstractions. To address this, researchers began stacking more layers in CNNs, leading to the development of Deep CNNs. These networks typically consist of dozens to hundreds of layers, each learning increasingly complex representations. Several deep CNN architectures have been proposed, each introducing innovations to enhance performance and computational efficiency. Key architectures include:

- **AlexNet** [KSH12]: Pioneered the use of deep CNNs in large-scale image classification, introducing ReLU activations and dropout regularization to improve training efficiency and reduce overfitting.
- **VGGNet** [SZ15]: Emphasized depth by using multiple stacked 3x3 convolutional layers, demonstrating that deeper networks can capture more complex features, albeit with increased computational cost.
- **ResNet** [He16]: Introduced residual connections, allowing the training of much deeper networks by mitigating the vanishing gradient problem, thus enabling models like ResNet-50 and ResNet-101 to achieve superior performance.
- **DenseNet** [Hu17]: Connected each layer to every other layer in a feed-forward fashion, enhancing feature reuse and reducing the number of parameters, which leads to more efficient models.
- **Inception (GoogLeNet)** [Sz15]: Utilized inception modules that perform convolutions at multiple scales, capturing both fine and coarse features, and improving computational efficiency through dimensionality reduction.
- **EfficientNet** [TL19]: Employed a compound scaling method that uniformly scales network depth, width, and resolution, achieving state-of-the-art accuracy with fewer parameters and FLOPS.

Each architecture offers trade-offs between accuracy, computational efficiency, and model complexity. For instance, while ResNet's deep architecture excels in capturing intricate features, EfficientNet achieves comparable accuracy with fewer resources, making it suitable for deployment in resource-constrained environments. The choice of architecture should align with the specific requirements of the application, such as the need for real-time inference, available computational resources, and the complexity of the classification task. Moreover, studies have shown that the performance of DL architectures can vary depending on the dataset and task complexity. For example, in medical imaging tasks like chest radiograph classification, shallower networks like ResNet-34 have achieved performance comparable to deeper models, suggesting that model selection should consider the specific context of the application [Br20].

In summary, the evolution of CNN architectures reflects a continuous effort to balance model complexity, computational efficiency, and classification accuracy. Selecting the appropriate architecture necessitates a thorough understanding of the application's unique demands and constraints.

2.2 Object Detection

While image classification determines the presence of objects within an image, object detection extends this capability by identifying multiple object classes and their respective spatial locations through bounding boxes. This functionality is critical in scenarios where spatial information is essential, such as identifying defects in manufactured products or detecting objects in autonomous driving. For instance, as illustrated in Fig. 2.3, the left side presents an example of image classification, where the model identifies the presence of a single class (i.e., cat) without specifying its location. In contrast, the right side demonstrates object detection, wherein the model not only identifies the object class but also localizes it within the image. Moreover, object detection can identify multiple objects within a single frame, unlike image classification, which typically assigns a single label per image.

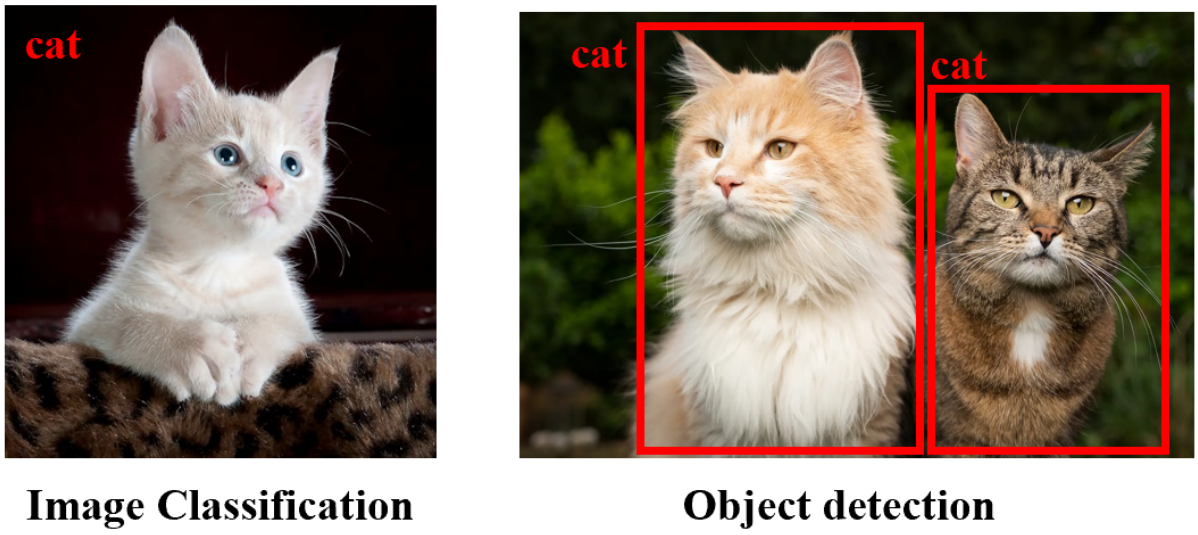

Figure 2.3: Comparison between image classification and object detection.

2.2.1 Why YOLO?

Traditional object detection frameworks such as Region-based Convolutional Neural Network (R-CNN) and Faster R-CNN adopt a two-stage pipeline. The first stage generates region proposals that may contain objects, and the second classifies these regions and refines their bounding boxes. Although these methods achieve high accuracy, their reliance on multiple sequential modules results in considerable computational overhead. This makes them unsuitable for real-time applications such as autonomous navigation or industrial quality inspection. You Only Look

Once (YOLO) redefines the object detection paradigm by formulating it as a single-stage regression problem [Re16]. Instead of separating region proposal and classification into discrete stages, YOLO processes the entire input image through a single CNN. This network simultaneously predicts bounding box coordinates, objectness scores, and class probabilities for each predefined grid location. The unified architecture eliminates the need for external region proposal modules and significantly reduces inference time, making it well-suited for high-throughput detection tasks.

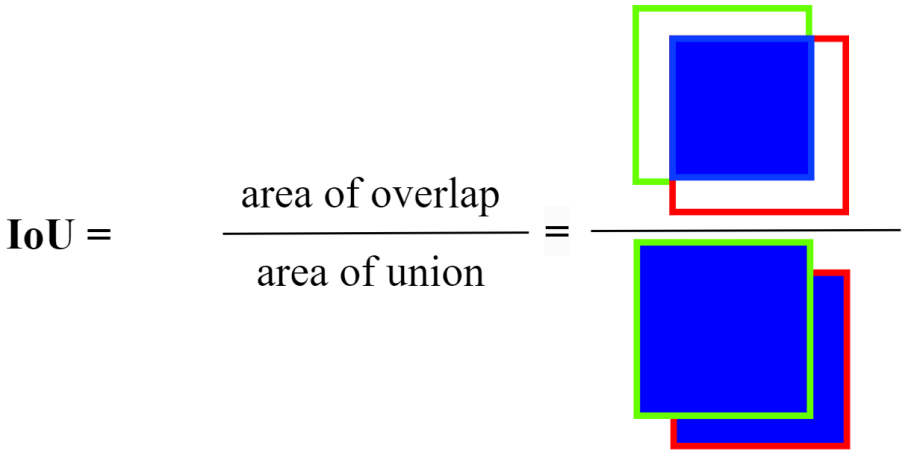

Figure 2.4: Schematic explanation of Intersection over Union (IoU), calculated as the ratio between the area of overlap and the area of union between predicted and ground truth bounding boxes.

To address variations in object scale and aspect ratio, YOLO incorporates the concept of ***anchor boxes***. Each grid cell is associated with multiple predefined bounding box shapes, which the network adjusts using offset regression. This allows the model to detect multiple objects per grid cell and improves its ability to represent objects with diverse geometries. Since each object may be predicted by several overlapping bounding boxes, YOLO applies Non-Maximum Suppression (NMS) as a post-processing step to suppress redundant detections. NMS retains only the highest-confidence prediction for each object and eliminates those with a high Intersection-over-Union (IoU) overlap. This combination of anchor boxes and NMS enables YOLO to efficiently detect multiple, variably-sized objects in a single image. Figure 2.4 presents a schematic explanation of the Intersection over Union (IoU), which quantifies the overlap between a predicted bounding box and the ground truth. Figure 2.5 shows examples of different IoU scores

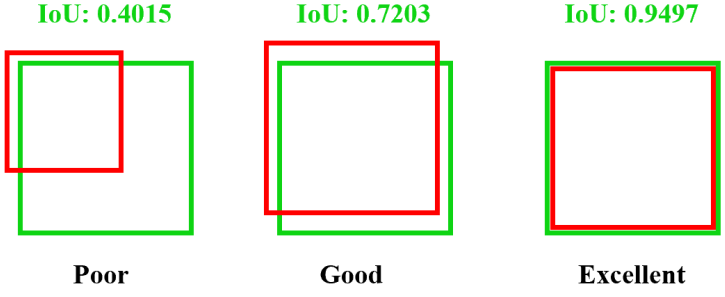

Figure 2.5: Examples of predicted bounding boxes with different IoU values. Higher IoU indicates better alignment with the ground truth.

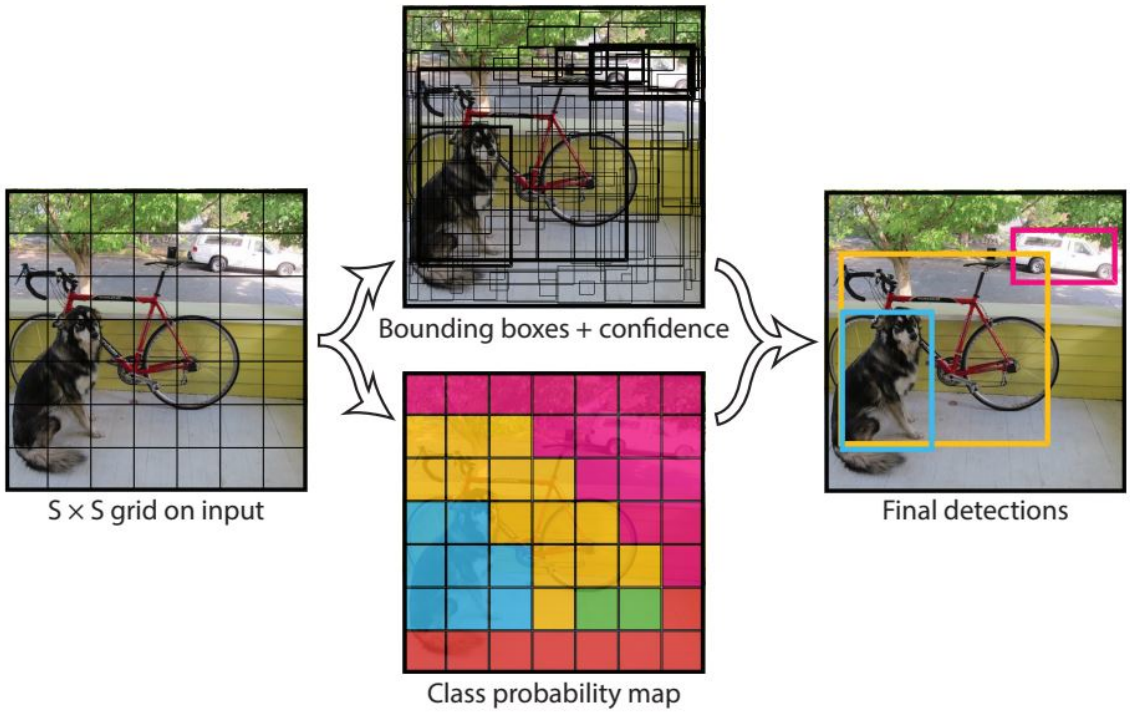

Figure 2.6: YOLOv3 architecture predicting bounding boxes in a single forward pass. [RF18]

and how they correspond to the quality of the detection.

Since its original formulation, the YOLO framework has undergone several architectural improvements. YOLOv3 introduced multi-scale feature prediction and a deeper backbone network, Darknet-53, which significantly improved detection performance, especially for small objects [RF18]. More recent versions, such as YOLOv5 and YOLOv8, have expanded on these capabilities through the introduction of auto-anchor optimization, enhanced feature pyramids, and anchor-free detection strategies. These enhancements have improved both accuracy and generalization, establishing YOLO as one of the most effective and widely adopted frameworks for object detection in real-world applications.

2.2.2 YOLOv5

YOLOv5, introduced by Ultralytics in 2020 [Jo20], marked a major step forward in the evolution of single-stage object detection architectures. While YOLOv3 had already established itself as a powerful and fast real-time detector, it exhibited limitations in its adaptability, training complexity, and modular flexibility. YOLOv5 was developed to address these shortcomings, offering improved performance, streamlined implementation, and greater architectural modularity through the PyTorch framework. Unlike its predecessors which were implemented in Darknet (a custom C-based framework), YOLOv5 was rewritten entirely in PyTorch, thus simplifying integration with modern DL pipelines and enabling rapid experimentation and deployment. From

an architectural standpoint, YOLOv5 continues the canonical split of object detection models into three major blocks: **the backbone, the neck, and the head**. Each of these components incorporates distinct innovations aimed at enhancing efficiency, scalability, and detection robustness.

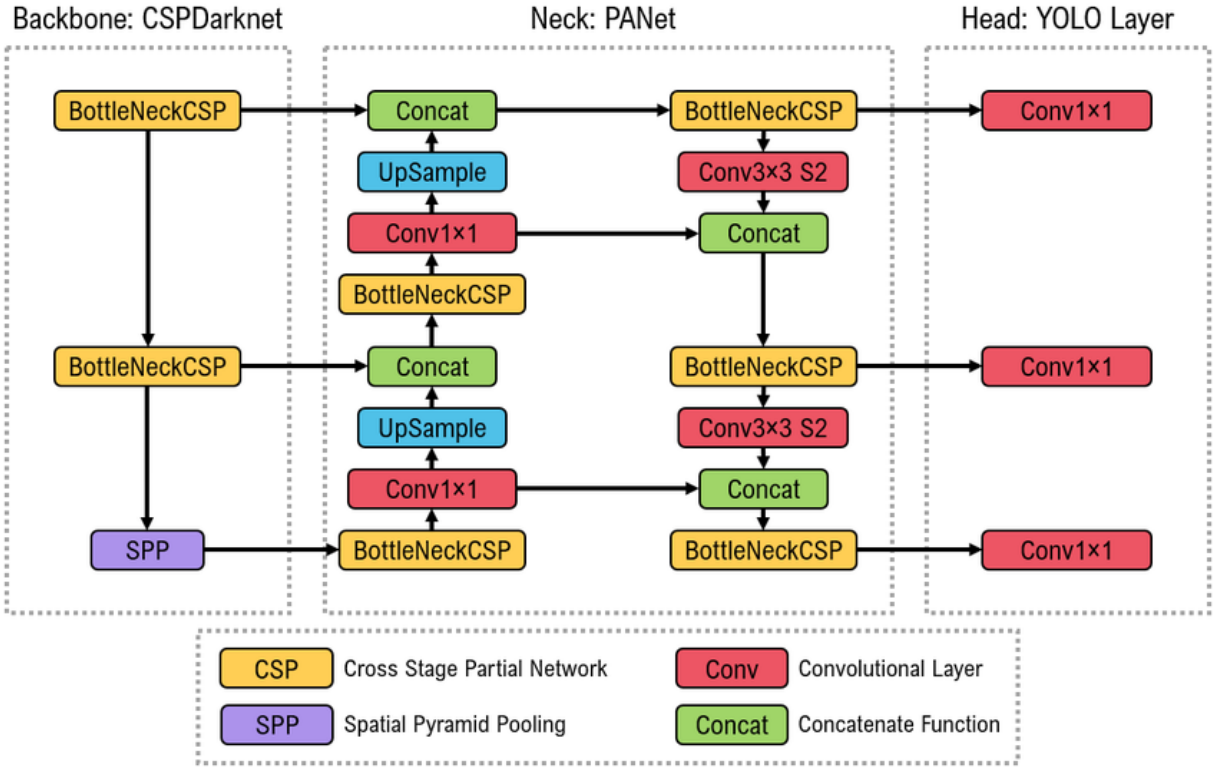

Figure 2.7: Overview of YOLOv5 architecture, showing the CSP-Darknet53 backbone, PANet neck, and the output head used for bounding box regression and classification [Ka22].

The backbone of YOLOv5 employs CSPDarknet53, a Cross Stage Partial (CSP) enhanced version of the Darknet53 network originally used in YOLOv3. CSPNet architecture was designed to mitigate gradient duplication and improve learning capability by splitting feature maps along the channel dimension and processing them in parallel. CSP-Darknet facilitates efficient feature extraction across multiple scales while significantly reducing computational cost and memory usage, particularly beneficial in lightweight or resource-constrained environments [WBL20].

The neck of YOLOv5 integrates two modules: a modified Path Aggregation Network (PANet) and a Spatial Pyramid Pooling Fast (SPPF) layer. PANet is responsible for enriching feature maps by aggregating information from different stages of the backbone. It enhances bottom-up and top-down paths to allow better propagation of localization and semantic information, which is critical for detecting small and large objects across scales. The SPPF module replaces the original Spatial Pyramid Pooling module introduced in YOLOv4. It applies a series of max pooling operations of varying kernel sizes in parallel, followed by a concatenation step. The “Fast” version optimizes this operation to improve inference speed without compromising the ability to capture contextual information from different receptive fields.

The head in YOLOv5 is responsible for dense predictions of bounding boxes, objectness scores,

and class probabilities. The decoupled detection head enables independent optimization of localization and classification tasks, improving convergence and generalization. The output is further refined using NMS during inference to eliminate redundant detections and preserve high-confidence bounding boxes.

YOLOv5 is released in five pre-configured model variants, namely YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large) which differ in depth and width multipliers. This flexibility allows users to select a model variant that aligns with their specific performance, latency, and memory constraints. Moreover, YOLOv5 supports mixed-precision training and native export to ONNX, TorchScript, and CoreML formats, making it suitable for deployment across a variety of hardware platforms, from GPUs to edge devices.

2.2.3 YOLOv8

YOLOv8 [Ul23], introduced by Ultralytics in early 2023, represents a substantial advancement in the evolution of one-stage object detection models. As the successor to YOLOv5, it incorporates several architectural improvements that collectively enhance detection accuracy, speed, and generalization. A key innovation in YOLOv8 is the transition from anchor-based to anchor-free detection. Earlier YOLO versions relied on predefined anchor boxes to match predicted bounding boxes during training. This often introduced additional hyperparameters and reduced model adaptability. YOLOv8 eliminates this dependency by directly regressing object centers and dimensions from feature maps, simplifying the detection pipeline and improving robustness to variations in object scale and aspect ratio.

The architecture of YOLOv8 maintains a modular structure, comprising a backbone, a neck, and a detection head. The backbone is based on a refined Cross Stage Partial (CSP) network, which enhances gradient flow and reduces parameter redundancy by partially propagating feature maps through dense residual connections. This results in efficient feature extraction with improved representational capacity. The neck employs a top-down Feature Pyramid Network (FPN) design that aggregates multiscale features across different levels. This facilitates the detection of small, medium, and large objects within a single image, an essential requirement in applications such as manufacturing inspection and medical imaging. The detection head is **decoupled and anchor-free**, consisting of separate branches for classification and bounding box regression. This separation enables independent optimization of each task, resulting in improved convergence and detection precision. During inference, NMS is applied to remove redundant predictions and retain only the most confident detections.

In addition to architectural improvements, YOLOv8 offers substantial gains in runtime efficiency. According to Ultralytics' official benchmarks, the YOLOv8n (nano) variant achieves inference speeds exceeding 150 Frames Per Second (FPS) on high-end GPUs and delivers practical throughput on resource-constrained platforms such as ARM-based embedded devices. These perfor-

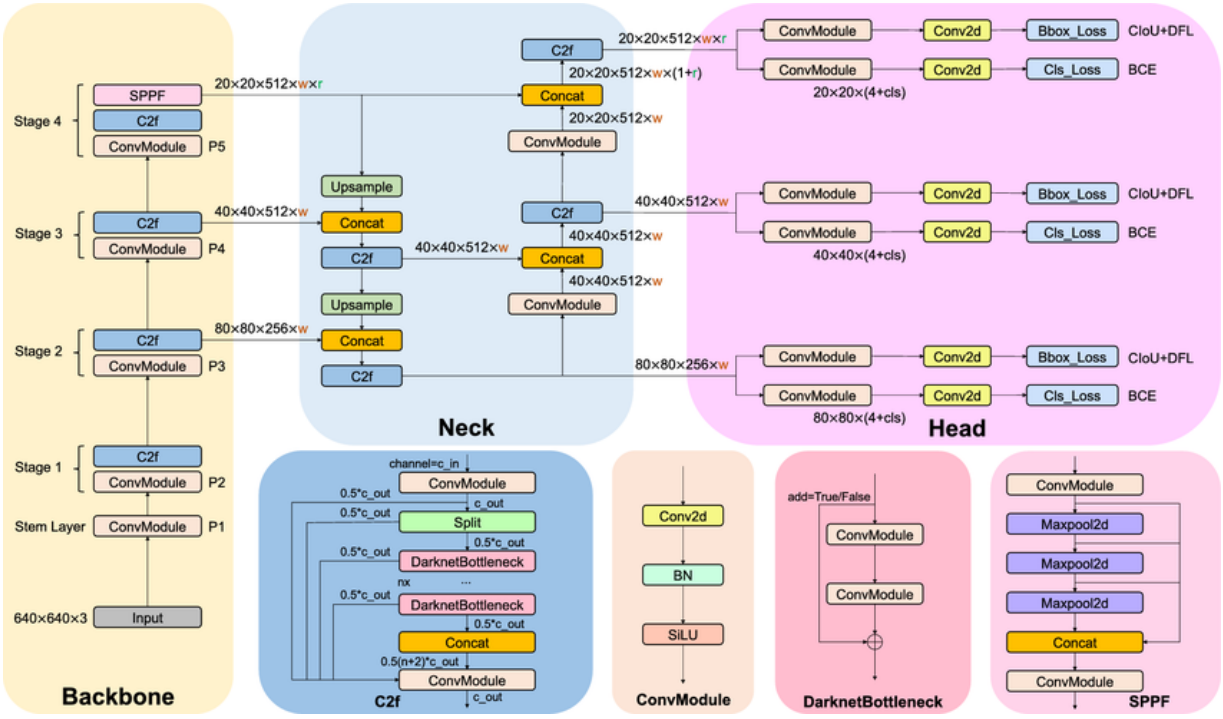

Figure 2.8: YOLOv8 architecture comprising a CSP-based backbone, an FPN-style neck, and a fully decoupled anchor-free detection head. Adapted from [JC23].

mance advantages are further supported by mixed-precision training and compatibility with multiple deployment formats, including ONNX, TorchScript, and TensorRT.

Similar to YOLOv5, YOLOv8 is released in five model variants: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x, each offering a different balance between computational efficiency and detection accuracy. This model scalability enables practitioners to select an appropriate configuration based on deployment constraints, making YOLOv8 a versatile and high-performing solution for object detection tasks across a wide range of application domains.

Note on YOLOv11 and YOLOv12

YOLOv11 and YOLOv12 are the most recent advancements in the YOLO object detection family. Both models build upon the modular architecture introduced in YOLOv8 and introduce incremental, yet meaningful, improvements in feature representation and detection accuracy. Importantly, they preserve the core design principles of real-time inference and end-to-end processing.

YOLOv11 enhances feature aggregation by introducing deeper residual pathways and lightweight convolutional modules optimized for computational efficiency. These architectural refinements improve the expressiveness of feature representations while reducing latency, particularly in environments that require GPU acceleration or real-time inference on edge devices [Ul24]. YOLOv12 further extends the model architecture by incorporating attention mechanisms, including the Area Attention Module and Residual Efficient Layer Aggregation Networks

(R-ELAN). These modules improve the model's ability to capture spatial dependencies and contextual information, which is particularly beneficial for detecting densely packed or visually ambiguous objects. This capability is especially useful in high-density manufacturing scenarios and medical imaging applications, where accurate boundary detection is critical [TZ25].

Although both YOLOv11 and YOLOv12 are compatible with FL workflow and conform to the PyTorch-based Ultralytics framework, they have not been employed within the scope of the experimental work presented in this thesis. Nevertheless, their architectural enhancements and reported runtime performance make them promising candidates for future investigation in FL environments. In particular, they warrant further benchmarking under non-IID data conditions and within decentralized hardware settings.

2.3 Federated Learning: Concepts and Architecture

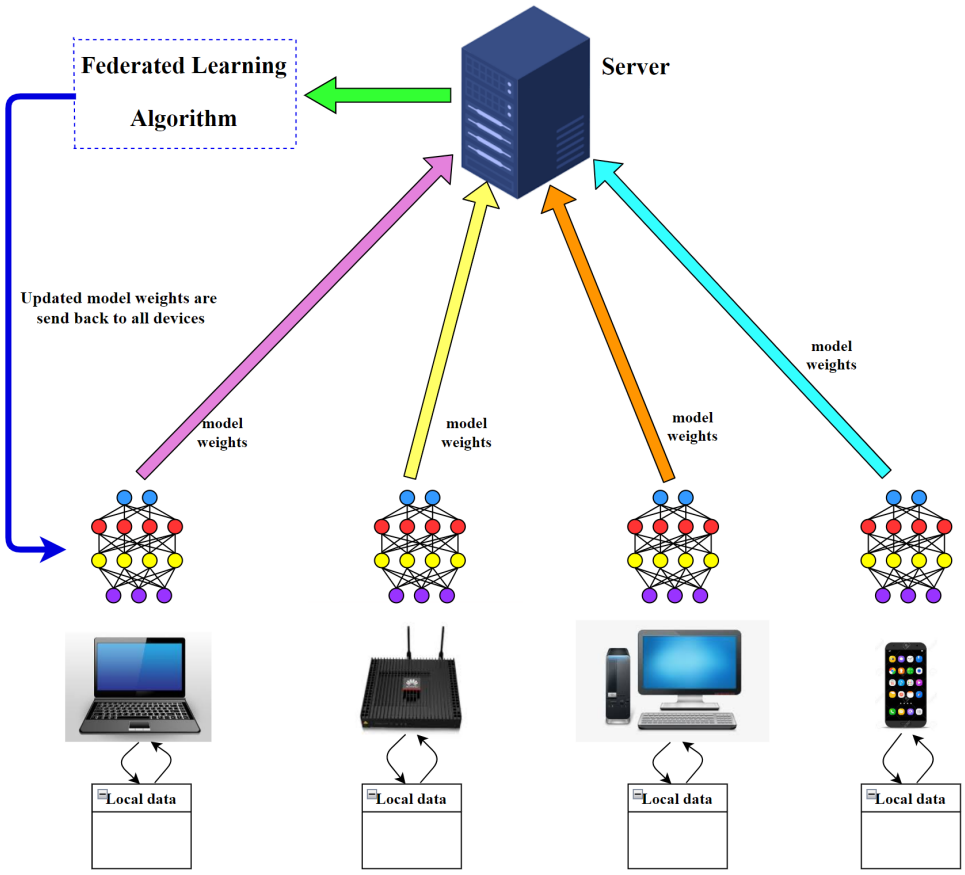

Figure 2.9: FL architecture: multiple local clients (devices or organizations) send model updates to a central server, which aggregates these updates to form the global model. Each client retains its private data.

FL is a decentralized machine learning paradigm introduced by Google researchers in 2016 [Mc17]. It represents a significant departure from conventional centralized training, where all data must be collected in a single repository. Instead, FL distributes the training process across multiple devices or organizations, referred to as clients, each of which retains its local

data and contributes to a shared global model through iterative communication. This paradigm was originally designed to address the growing concern over data privacy and communication efficiency in edge and mobile computing scenarios, where transmitting large volumes of raw data to the cloud is neither practical nor secure. As shown in Fig. 2.9, by enabling learning to occur directly at the data source, FL reduces dependence on centralized data storage and minimizes exposure of sensitive information.

The significance of FL lies in its ability to preserve data privacy while enabling large-scale collaboration across geographically distributed or institutionally isolated entities. It allows organizations such as hospitals, banks, and manufacturing companies to jointly train models without sharing sensitive or proprietary data, thereby addressing legal, ethical, and confidentiality constraints. By adhering to frameworks such as the General Data Protection Regulation (GDPR), FL ensures compliance with privacy-preserving principles while facilitating data-driven innovation. In addition to privacy benefits, FL enhances scalability and efficiency by performing computation locally, reducing communication overhead, and leveraging diverse datasets to improve model generalization in heterogeneous environments such as industrial production networks.

A typical FL system comprises several key components that enable decentralized collaborative learning while maintaining data locality:

- **Local Clients**: Entities such as smartphones, hospitals, or industrial units that store private data locally and independently train a copy of the global model.
- **Central Server**: The coordinating server responsible for initializing the global model and aggregating model updates received from clients during each communication round.
- **Global Model**: The shared model constructed through the aggregation of local client updates, iteratively distributed to synchronize learning progress across all clients.
- **Communication Rounds**: Each round involves transmitting the global model to clients, conducting local training on private data, and returning the updated parameters to the server for aggregation.

The overall training process in FL unfolds iteratively through the following stages:

1. **Initialization**: The central server initializes the global model and broadcasts it to participating clients.
2. **Local Training**: Each client performs local updates on its dataset, optimizing the model using its private data.
3. **Aggregation**: Clients send their updated model parameters to the server, which aggregates them using a weighted averaging scheme.
4. **Iteration**: The process is repeated over multiple communication rounds until the global model converges.

The most widely adopted aggregation method in FL is the Federated Averaging (FedAvg) algorithm, formulated as

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

where w_t^k denotes the local model of client k at round t, n_k represents the number of data samples available at client k, and $n = \sum_{k=1}^{K} n_k$ corresponds to the total number of samples across all K participating clients.

2.4 Federated Learning Algorithms

Having established the architectural components and training workflow of a federated system, this section details the algorithmic strategies used for model aggregation and optimization.

2.4.1 Federated Averaging Algorithm

FedAvg proposed by McMahan et al. [Mc17], is the foundational algorithm in FL and remains the most widely adopted due to its simplicity and communication efficiency. The algorithm proceeds in iterative communication rounds. In each round, a subset of clients receives the current global model and performs multiple local training epochs on their private datasets. These clients then return their updated model parameters to the central server, which aggregates them, typically via a weighted average based on local dataset sizes to produce the next global model. The strength of FedAvg lies in its straightforward design and minimal communication overhead, making it particularly well-suited for deployment in cross-silo settings such as healthcare, industrial automation, or manufacturing. In such scenarios, the number of clients is relatively small and the data, while potentially non-IID, tends to be structured and stable over time. Although FedAvg may encounter convergence challenges in massively distributed, highly heterogeneous environments, it remains a strong and reliable baseline in real-world applications.

Algorithm 1: Federated Averaging [Mc17]

Input: Initial global model w_0, number of rounds T, number of clients K

for *each round* $t = 0, \dots, T-1$ **do**

- Server selects a subset $\mathcal{S}_t \subseteq \{1, \dots, K\}$
- **for** *each client* $k \in \mathcal{S}_t$ ***in parallel*** **do**
  - Client receives w_t and performs E local epochs of SGD on local data
  - Client sends w_t^k to the server
- Server aggregates the updates: $w_{t+1} = \sum_{k \in \mathcal{S}_t} \frac{n_k}{n} w_t^k$

Output: Final model w_T

2.4.2 Extensions of FedAvg to Other FL Algorithms

While FedAvg remains the foundational algorithm in FL, and has demonstrated strong performance in many practical scenarios, including those involving non-IID data with a limited number of clients [He23; He24; HLR22], various algorithms have been proposed to build upon its core principles. These extensions aim to improve convergence behavior, enhance fairness across heterogeneous clients, and address scenarios involving large-scale deployments or severe system heterogeneity. The following subsections present several such algorithms that extend or refine the FedAvgframework to accommodate broader operational challenges in FL.

FedMA

Federated Matched Averaging (FedMA) [HAA20] addresses the challenge of aggregating heterogeneous model architectures across clients. Most traditional FL algorithms, such as FedAvgand Federated Proximal (FedProx), assume that all clients share an identical model structure. However, in real-world deployments, clients may operate with different neural architectures due to hardware constraints, deployment policies, or task-specific adaptations. This structural heterogeneity complicates weight aggregation and hinders global model convergence. FedMA mitigates this issue by adopting a layer-wise neuron matching strategy. Instead of averaging parameters blindly across structurally inconsistent models, FedMA uses similarity metrics to align neurons across clients based on their functional behavior. Once the matching is established, corresponding neurons are averaged to form each layer of the global model. This enables meaningful parameter aggregation even when clients use networks with differing internal activation patterns.

Algorithm 2: Federated Matched Averaging [HAA20]

Input: Local models $\{w_k\}$ with neuron representations

for *each layer* **do**

  Match neurons across client models using similarity metrics

  Average matched neurons to obtain global layer representation

Construct w_{t+1} from matched and averaged layers

Output: Updated global model w_{t+1}

FedMA has demonstrated strong performance in tasks such as federated image classification, particularly in scenarios where clients utilize different network architectures or experience internal neuron drift. However, its reliance on pairwise neuron matching increases computational complexity and may limit scalability to very deep or large models. Nevertheless, it presents a promising direction for extending FL, to structurally diverse and dynamically evolving edge environments.

FedProx

FedProx [Li20] addresses a core challenge in FL: client heterogeneity. When local datasets are non-IID, the client updates may deviate significantly from the global objective, leading to divergence or slow convergence. To mitigate this, FedProx modifies the client-side optimization objective by introducing a proximal term that penalizes large deviations from the current global model. This proximal regularization term encourages the local models to remain close to the global reference during training, thus stabilizing the optimization process and improving convergence under heterogeneous data and system conditions. This makes FedProx particularly effective in scenarios where client data distributions vary widely or where clients are intermittently available with varying computational resources.

Algorithm 3: Federated Proximal [Li20]

Input: Global model w_t, proximal coefficient μ

for *each client* $k \in \mathcal{S}_t$ **do**

  Minimize: $f_k(w) + \frac{\mu}{2}\|w - w_t\|^2$

  Send updated w^k to server

Aggregate as in FedAvg: $w_{t+1} = \sum_k \frac{n_k}{n} w^k$

Output: Updated model w_{t+1}

While FedProx improves stability in heterogeneous settings compared to FedAvg, it does not completely eliminate the challenges posed by non-IID data. For instance, it does not incorporate any correction mechanism for client-side drift or variance reduction, which motivates the development of further extensions such as SCAFFOLD and Federated Normalized Averaging (FedNova). Nevertheless, FedProx remains one of the most widely cited and practically relevant algorithms for FL in heterogeneous environments.

FedNova

FedNova [Wa20b] was proposed to address a critical limitation of FedAvg: imbalance in client contributions due to differences in local computation, such as the number of training iterations or batch sizes. In standard FedAvg, clients that perform more local updates can unintentionally dominate the global model, introducing bias and instability in the aggregation process. This is particularly problematic when clients have non-uniform compute capabilities or data volumes. FedNova introduces a normalization mechanism that scales each client's local update according to the number of local gradient steps performed. This ensures that each client contributes fairly to the global model, regardless of its local training length. The normalization is performed directly on the update vectors, resulting in an unbiased aggregation scheme that improves convergence and stability, especially in heterogeneous environments.

Algorithm 4: Federated Normalized Averaging [Wa20b]

Input: Each client performs T_k local steps

for *each client* k **do**

Compute local update: $\Delta w_k = \sum_{i=1}^{T_k} \nabla f(w_i^k)$

Normalize: $\Delta \tilde{w}_k = \frac{T_k}{\sum_k T_k} \Delta w_k$

Aggregate: $w_{t+1} = w_t - \eta \sum_k \Delta \tilde{w}_k$

Output: Updated model w_{t+1}

FedNova offers a simple yet effective approach to correct for client contribution imbalance and has shown improved convergence in both convex and non-convex objectives. However, the method does not address the variance in model directions that can arise under non-IID conditions. It also assumes accurate tracking of local update steps, which may introduce implementation complexity in certain asynchronous or unreliable communication settings. Nonetheless, FedNova remains a robust improvement over FedAvg, particularly in scenarios with unequal local training durations or resource-constrained federated deployments.

Adaptive Federated Optimization: FedAdam, FedYogi, FedAdagrad

Adaptive Federated Optimization [Re21], is a general framework that enhances server-side optimization in FL. It addresses key limitations of FedAvg, particularly its inefficiency under high gradient variance and non-IID data distributions. Unlike FedAvg, which performs naive weighted averaging of client model updates, this framework incorporates adaptive optimization techniques such as Adam, Yogi, and Adagrad directly at the server during aggregation. These techniques are known for their ability to adjust learning rates dynamically based on first and second-order moment estimates of the gradients.

FedAdam applies exponential moving averages to both the gradients and their squared values, allowing for smoother and more stable updates over time. This is particularly useful when client updates are noisy or diverse, as it dampens sharp oscillations. **FedYogi** offers a more conservative variant by preventing aggressive growth in second-moment estimates. Instead of exponential accumulation, it adjusts the second moment based on the sign of the difference between the previous estimate and the current gradient squared. This design helps avoid over-adaptation and enhances robustness. **FedAdagrad** accumulates squared gradients without decay, thereby reducing the learning rate over time for frequently updated parameters and improving convergence on sparse or unstable updates.

Adaptive optimization in the Federated framework improves convergence in heterogeneous federated settings and has been shown to outperform FedAvgand FedProx on benchmark datasets. However, the increased complexity of maintaining and tuning multiple moment vectors at the server introduces additional computational overhead and requires careful hy-

perparameter calibration. Despite these trade-offs, FedAdam, FedYogi, and FedAdagrad have emerged as strong baselines in modern FL research, particularly when fast and stable convergence is critical.

Algorithm 5: Adaptive Federated Optimization [Re21]

Input: Initial model w_0, learning rate η, decay rates β_1, β_2, optimizer variant `opt` in {`FedAdam`, `FedYogi`, `FedAdagrad`}

Initialize moment vectors: $m_0 = 0, v_0 = 0$

for *each round* $t = 1, \dots, T$ **do**

  Server receives client updates Δw_t^k

  Compute aggregated gradient: $g_t = \sum_k \frac{n_k}{n} \Delta w_t^k$

  Update first moment: $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$

  if *`opt == FedAdam`* **then**

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

  else

    if *`opt == FedYogi`* **then**

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

    else

      if *`opt == FedAdagrad`* **then**

        $v_t = v_{t-1} + g_t^2$

  Compute bias-corrected estimates:

  $\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$

  Global model update:

  $w_{t+1} = w_t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$

Output: Final model w_T

2.4.3 Other Notable FL Algorithms

In addition to the foundational and extended algorithms discussed previously, several recent methods have been introduced to improve the robustness, convergence stability, and adaptability of FL, particularly in environments with non-IID data and heterogeneous client capabilities. Although these methods offer valuable insights and novel strategies, they are not as widely deployed in practice. In most cases, evaluations are limited to small-scale experimental setups.

SCAFFOLD (Stochastic Controlled Averaging for FL) [Ka20] addresses the issue of client drift, which arises when local updates diverge significantly due to non-IID data. The algorithm introduces control variates, which are auxiliary variables maintained by both the server and clients. These vectors are used to correct the direction of local updates, thereby aligning client optimization trajectories with the global objective and improving overall convergence stability.

MOON (Model-Contrastive FL) [LHS21] integrates contrastive learning principles into federated optimization. In addition to the standard task-specific loss, MOON introduces a contrastive loss that encourages local representations to remain close to the global model while diverging from outdated versions. This alignment improves generalization and mitigates the impact of local overfitting in non-IID settings.

Federated Bayesian Ensembles (FedBE) [CC21] proposes a Bayesian approach to model aggregation. Rather than relying on deterministic averaging, FedBE draws samples from a posterior distribution over model parameters, forming a probabilistic ensemble. This methodology enhances the global model's robustness to client noise and facilitates uncertainty quantification, which is valuable for risk-sensitive applications.

Despite the growing algorithmic diversity in the literature, as previously discussed in Section 1.1.2, it is noteworthy that many of these approaches have been predominantly evaluated on benchmark datasets such as CIFAR-10 and MNIST. Consequently, **FedAvg remains the most widely adopted algorithm** in practical deployments, particularly in scenarios involving a limited number of clients. Its simplicity and ease of integration with diverse model architectures make it a robust baseline for real-world FL applications.

2.5 Related Work and Research Gaps

At the commencement of this research in May 2021, the application of FL within the manufacturing sector was relatively unexplored. The foundational concept of FL was introduced by McMahan et al. in 2017 [Mc17], primarily focusing on mobile device scenarios such as Google's Gboard [Ha18]. However, its adoption in industrial contexts, particularly for quality inspection tasks, was very limited. One of the pioneering efforts to apply FL in manufacturing was presented by Dib et al. [DRP21], where FL was utilized for defect prediction in sheet metal forming processes. This study demonstrated the feasibility of training models across decentralized data sources without compromising data privacy. Further, Ge et al. [Ge21] conducted an empirical study on failure prediction in production lines using FL, comparing federated and centralized learning approaches. Their findings indicated that FL can achieve comparable performance to centralized methods, even in heterogeneous data environments. Kanagavelu et al. [Ka21b] proposed a decentralized FL solution integrated with an Industrial Internet of Things smart manufacturing platform, demonstrating its effectiveness with real-world datasets and various machine learning algorithms.

These early studies laid the groundwork for applying FL in manufacturing, addressing challenges such as data privacy and collaboration across different production sites. However, they primarily focused on centralized datasets or simulations, with limited exploration of real-world, heterogeneous data scenarios. Additionally, comprehensive surveys like those by Zhou et al. [Zh21] and

Parimala et al. [Pa21] highlighted the potential of FL in accelerating Industry 4.0 initiatives, emphasizing the need for further research into handling non-IID data, communication efficiency, and robust aggregation methods tailored to industrial applications.

Since early 2022, there has been a significant increase in research exploring FL applications in manufacturing. For instance, Pruckovskaja et al. [Pr23] evaluated different FL aggregation methods for predictive maintenance and quality inspection, comparing them to centralized and local training approaches. Their study, based on four datasets with varying data distributions, indicated that FL performance is highly dependent on data characteristics and client distributions. In another study, Becker et al. [Be22] proposed an autoencoder-based FL method utilizing vibration sensor data from rotating machines for condition monitoring in the Industrial Internet of Things. Their approach enabled distributed training on edge devices, preserving data privacy while achieving competitive performance compared to centralized methods. Additionally, Siemens Digital Industries, in collaboration with Katulu, demonstrated the practical implementation of FL in industrial settings, highlighting its effectiveness in handling sensitive manufacturing data across multiple sites [KS24]. These studies underscore the growing interest and practical applications of FL in manufacturing, addressing challenges such as data privacy, heterogeneity, and the need for collaborative learning across decentralized data sources.

In the realm of object detection, Chen et al. [Ch19] implemented FL using YOLOv3 and Faster R-CNN algorithms on a real-world image dataset, providing extensive benchmarks on model performance, efficiency, and communication in a federated setting. This work demonstrated the applicability of FL in complex computer vision tasks such as object detection on real life objects. Furthermore, the integration of FL in predictive maintenance has been explored by various studies. For example, Ahn et al. [Ah23] proposed a 1D-CNN-BiLSTM model combined with FL for time series anomaly detection and predictive maintenance in manufacturing processes. Their approach addressed the challenges of data heterogeneity and distributional shifts, achieving a test accuracy of 97.2%. Despite the progress in applying FL within manufacturing, several research gaps persist. Many studies focus on simulations or controlled environments, with a paucity of research involving vision based real-world heterogeneous manufacturing data. There is a lack of comprehensive comparisons between FL models and locally trained models in industrial settings, hindering the understanding of FL's practical benefits. The application of FL to object detection tasks in manufacturing remains underexplored, with few studies addressing the unique challenges posed by such tasks. Addressing non-IID data across clients continues to be a significant challenge in FL, necessitating further research into robust aggregation methods and model personalization techniques.

While the reviewed studies provide valuable insights into the applicability of FL in manufacturing, several critical limitations persist that hinder the transition from proof-of-concept implementations to robust industrial deployments. A recurring shortcoming across many works is the reliance on synthetically partitioned datasets, where a centralized dataset is evenly split

Table 2.1: Feature Coverage in Selected FL Studies

Feature	Dib et al.	Ge et al.	Kanagavelu et al.	Pruckovskaja et al.	Becker et al.	Siemens & Katulu	Chen et al.	Ahn et al.
Vision Dataset	✗	✗	✗	✓	✗	✓	✓	✗
Real Clients	✓	✓	✓	✓	✓	✓	✗	✓
Mixed Clients (Real + Synthetic)	✗	✗	✗	✗	✗	✗	✗	✗
Open Implementation	✗	✗	✗	✓	✓	✗	✓	✓
Industrial Use Case	✓	✓	✓	✓	✓	✓	✗	✓
FL Image Classification	✓	✗	✗	✓	✗	✓	✗	✗
FL Object Detection	✗	✗	✗	✗	✗	✗	✓	✗
External Test Data	✗	✓	✗	✓	✓	✗	✗	✗
Comparison with Local Models	✗	✓	✗	✓	✓	✗	✗	✗

across clients to simulate federated settings. This fails to represent the heterogeneity and imbalance that naturally arise when data is collected from distinct manufacturing environments. As summarized in Table 2.1, only a few studies involve truly distinct clients, and even fewer combine real and synthetic data sources to emulate industrial variance across sites. Moreover, real-world manufacturing scenarios often involve site-specific imaging pipelines, domain-specific object categories, or custom sensor configurations, factors seldom represented in current FL experiments. Another notable gap is the lack of rigorous, quantitative comparisons between FL models and locally trained models at the client level. Although some studies reference local baselines, they frequently omit consistent evaluation metrics or overlook tasks where spatial precision is critical, such as object detection using mAP as a metric. For instance, FedOD remains relatively underexplored; Chen et al.'s implementation with YOLOv3 provided a useful starting point but lacked robust benchmarking against local and centralized models, and did not address crucial issues such as anchor alignment or bounding box variability across clients. Furthermore, most existing work assumes homogeneous client configurations, ignoring challenges like model drift or aggregation instability in the presence of highly skewed, non-IID data. Very few studies account for manufacturing-specific complexities such as small object detection, production-line class imbalance, or real-time inference limitations at the edge. These shortcomings, many of which are clearly reflected across the feature set summarized in Table 2.1, highlights the research gap for custom-designed, vision-based FL architectures. Such frameworks must support realistic, heterogeneous deployment conditions, offer direct benchmarking against local and centralized counterparts, and ensure model robustness and generalization across diverse manufacturing sites.

2.6 Summary

This chapter provided the theoretical and algorithmic foundation for the research presented in this thesis. It began by tracing the evolution of DL methods for visual quality inspection, with a focus on CNNs and their application to image classification and object detection. Architectures such as VGG, ResNet, DenseNet, and EfficientNet were discussed in terms of their trade-offs between accuracy and computational efficiency. Particular emphasis was placed on the YOLO family of models (specifically YOLOv5 and YOLOv8), due to their real-time inference capabilities and suitability for deployment in resource-constrained manufacturing environments.

The chapter then introduced FL as a decentralized training paradigm that addresses data privacy and ownership concerns by enabling collaborative model training across clients without raw data exchange. Core architectural components and training workflows were explained, followed by a comprehensive review of key algorithms such as FedAvg, FedProx, FedNova, FedMA, and adaptive optimizers like FedAdam and FedYogi. These methods were presented in the context of their ability to handle data heterogeneity, client drift, and communication bottlenecks.

Finally, the chapter reviewed the current state of research on FL applications in manufacturing. While early studies have demonstrated the feasibility of FL in tasks such as defect prediction and condition monitoring, substantial gaps remain, particularly in the application of FL to object detection tasks, in the use of real-world heterogeneous client data, and in the availability of open, reproducible implementations. These research gaps motivate the methodological contributions and experimental evaluations introduced in the next chapter, where a modular FL framework for vision-based quality inspection is proposed and evaluated under realistic, non-IID deployment scenarios.

3 Methodology and Implementation

This chapter presents a comprehensive discussion on the implementation of FL, highlighting best practices, algorithmic frameworks, and optimization strategies tailored for industrial applications. It begins by introducing the general FL architecture specifically developed in this thesis, designed to enable detailed experimental analysis and support deployment in real-world manufacturing environments. The chapter further elaborates on model aggregation techniques, local weight selection strategies, and integrated security mechanisms that collectively ensure privacy preservation, robustness, and scalability in decentralized FL systems.

Additionally, the integration of FL with visual quality inspection is discussed in greater detail, emphasizing the architectural adaptations and task-specific considerations required for effective deployment. This includes an analysis of the challenges associated with federated image classification and object detection, addressing key aspects such as dataset selection, model architecture design, and training methodologies. The chapter also introduces the FedEnsemble approach, which leverages multiple client models to improve generalization and enhance defect detection performance in industrial scenarios. Finally, a use case is presented where a synthetic dataset is incorporated as an additional client in the FL process to compensate for the limited availability of real-world training data.

This implemented framework directly supports the research objectives outlined in Section 1.2. Specifically, it addresses the feasibility of applying FL in manufacturing environments characterized by a limited number of clients, non-IID data distributions, and custom high-resolution visual inspection tasks (Question 1). Furthermore, by extending the FL pipeline to object detection, the chapter investigates the scalability of federated models for spatially aware tasks involving bounding box localization and anchor box variability (Question 2). The structured progression from image classification to object detection, followed by ensemble learning and the integration of synthetic data, enables a comprehensive evaluation of FL performance across increasing levels of task complexity in industrial quality inspection scenarios.

3.1 General Setup for Federated Learning Process / Architecture

As outlined in [Mc17], the primary objective of FL is to train a ML model while ensuring data privacy by decentralizing model training. In this research, it is assumed that all clients participating in the FL process are trustworthy or that FL operates within a secure framework, such as Gaia-X [Ga22], where security and client authenticity are inherently managed. Although FL introduces challenges such as privacy leakage and adversarial model updates, security measures, including DP, Homomorphic Encryption (HE), and SMPC, have been proposed to mitigate these risks [Ab16; Ao17; DR14; Sh17]. However, these security enhancements are not within the scope of this research.

In the proposed implementation, each client functions independently, training its local model without sharing raw data. Instead, only the model weights are transmitted to the central server for aggregation, thereby maintaining privacy throughout the learning process. The overall FL architecture is depicted in Figure 3.1. It is shown that clients receive the initial global model weights from the server. These weights are utilized as the starting point for local training. Conventionally, the weights for the 0th Communication Round (CR) are initialized using pre-trained ImageNet weights specific to the training model. The rationale behind distributing the same initial weights to all clients is grounded in the findings of [Mc17], which demonstrate that federated models achieve superior convergence when all clients initialize with the same parameters rather than random weights. Additionally, utilizing pre-trained weights significantly reduces computational costs, as demonstrated in the research [HLR25b], where training with pre-trained weights required only one-third of the GPU time compared to training from scratch which makes it sustainable and efficient for industrial use case.

Each client evaluates the received global model on its local test dataset prior to local training (as shown in Fig. 3.1, step 2). This step is crucial for assessing how well the initial global model generalizes across different clients, offering valuable insights into model performance before local updates. It is particularly beneficial for clients joining later in the FL process, as it provides a baseline evaluation of the global model's performance prior to personalization. The evaluation results are saved locally as *test_metrics.txt*. After training for the specified number of local epochs, the best-performing weights (explained in Subsection 3.1) are sent to the server along with the corresponding *test_metrics.txt*.

On the server side, both model weights and the test metrics are received from all the clients. Once received the server first averages out the test metrics (e.g., accuracy for multi-class classification) and checks if the previous global model achieves the threshold accuracy. If not, then the weights from all clients are aggregated using one of the federated algorithms (mentioned in Section 2.4). The weight aggregation process uses FedAvg algorithm, wherein each neuron in a given layer is updated by computing the mean of the corresponding neuron values across all

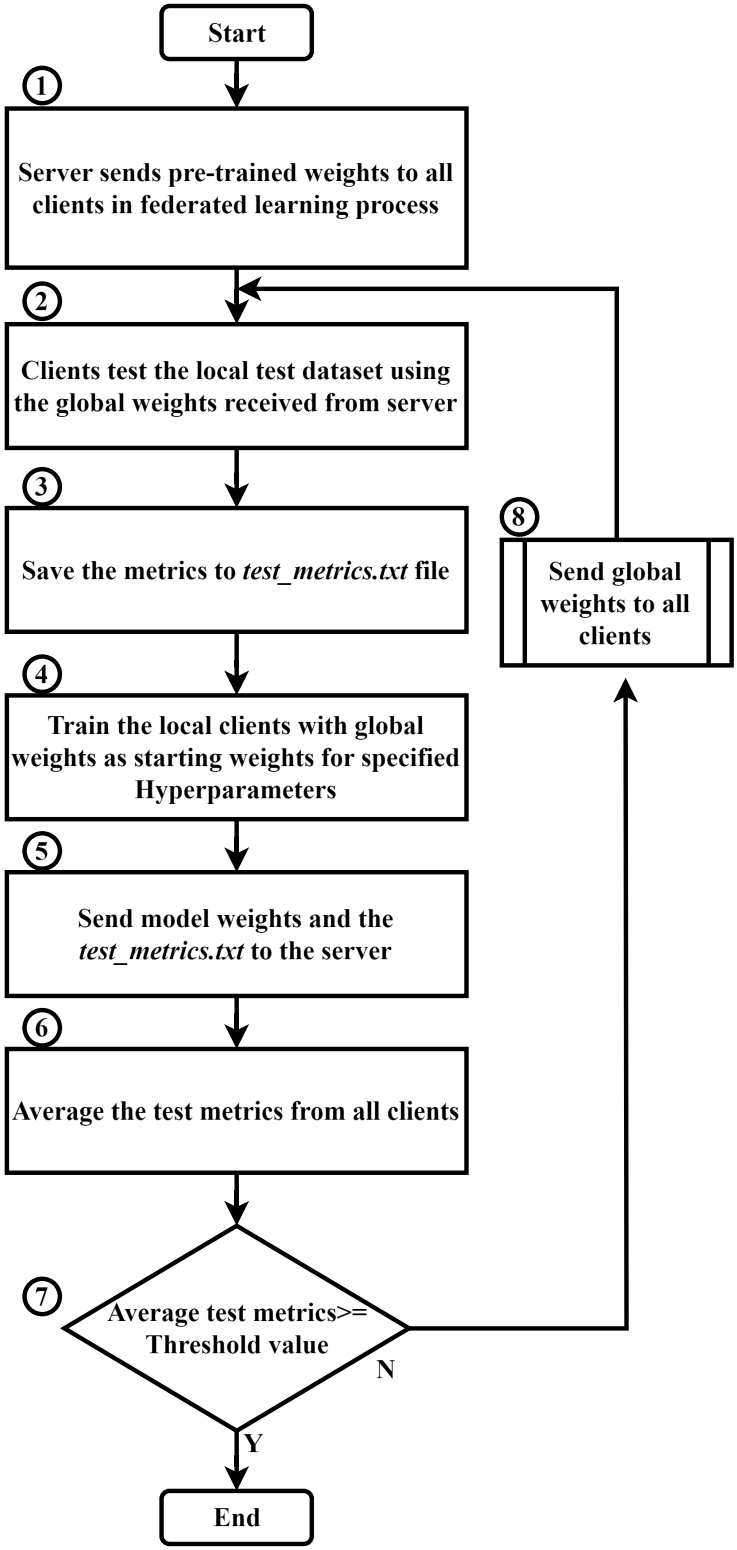

Figure 3.1: General Federated Training Process for Image Classification and Object Detection

clients. This aggregated model is then shared back with the clients, marking the start of a new CR. This iterative process continues until convergence is reached.

The global model's performance evaluated using the mean test accuracy across all clients, provides an objective metric of generalization across different data distributions. Moreover, an accuracy threshold can be predefined on the server side to terminate the FL process once the model achieves satisfactory performance across clients as shown in Figure 2.9. It is important to note that in real-world deployments, the successful implementation of FL necessitates standardized agreements among clients, including:

- Consensus on the same model architecture across all participants.
- Standardization of class labels and object detection indexing.
- Synchronization of CRs to determine the frequency of model aggregation.
- Agreement on a confidence threshold for validation metrics.
- Definition of a global accuracy threshold to establish a stopping criterion for training.

By adhering to these protocols, it is shown in this thesis thatFL can be effectively deployed in industrial scenarios while ensuring interoperability and model robustness.

Local Weight Selection Strategy

In practical applications of FL, particularly within industrial environments such as manufacturing, optimizing the aggregation of local models is essential to ensure the robust performance of the global model. Unlike large-scale consumer applications, where millions of clients can participate (e.g., Google's Gboard use case [Ha18]), FL in industrial contexts typically involves a limited number of clients who are willing to collaborate. Therefore, consistent participation of all clients in each CR is essential to maintain robust convergence and ensure effectiveness in the collaborative learning process.

In the default FL process, clients send the model weights from the final local epoch to the server for aggregation after training concludes [Mc17]. However, an alternative strategy is employed in this thesis, in which the ***best-performing weights*** are selected using a validation dataset instead of using the ***last weights*** from the local training epochs. This approach improves the accuracy and stability of the global model, particularly in scenarios involving non-IID client data or heterogeneous data distributions.

Given the heterogeneous nature of industrial data, adopting the best-weight selection strategy becomes especially beneficial. This is shown in details in Fig. 3.2, where the left side (a) shows the process flow of sending the Last weights, i.e., if the algorithm runs for 20 epochs on the local side, then sending the 20th epoch's model weights to the server. Whereas the right

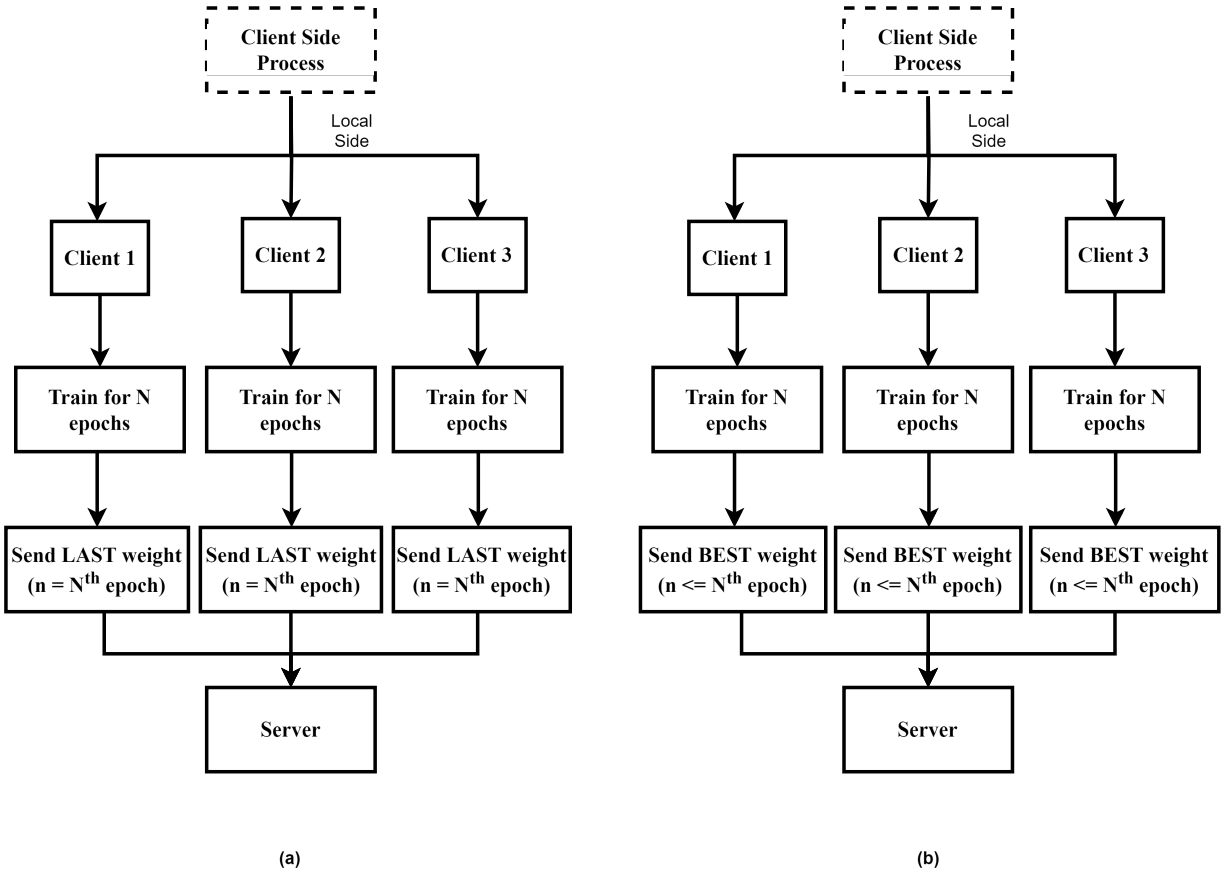

Figure 3.2: (a) Sending the LAST weights of specified epochs to the server. **(b)** Sending the BEST weights out of specified epochs to the server.

side of figure (b) shows the process flow of sending the Best weights, i.e., validation is done after each epoch and the best performing epoch's model weights are sent to the server. Consequently, this technique will serve as the default aggregation method in the FL architecture applied throughout the experiments discussed in this thesis.

3.2 Federated Image Classification for Quality Inspection

Although numerous studies have investigated federated image classification, most existing research primarily relies on benchmark datasets such as CIFAR-10 and MNIST [We23; Ya19]. While these datasets are useful for initial validation, they do not address the specific requirements of industrial quality inspection, which demands high-resolution imagery and complex feature detection. Until 2021, application-driven FL research in the manufacturing domain remained limited [Li21]. To address this gap, a federated image classification framework tailored to industrial use cases was developed. The USB Type-A quality inspection use case involves three independent clients contributing images from distinct production contexts. The first client represents Huawei OpenLab Munich, which manufactures USB sticks as part of its industrial testbed. The second client corresponds to the SmartFactory KL environment, where Lego style USB sticks are produced to simulate electronic assembly processes. To replicate a third collaborating par-

ticipant, an additional set of images featuring a red colored USB stick, visually similar to the SmartFactory variant, is labeled under WSKL as a client. This setup demonstrates a realistic multi-client FL scenario for detecting defective USB ports while preserving local data privacy.

To further evaluate the federated image classification framework within the SmartFactory KL demonstrator ecosystem for Industry 4.0, an additional use case involving cabin windshield type classification was developed. This scenario uses images of cabins captured from a model truck integrated into the SmartFactory KL testbed, which simulates a real-world production line for cabin assembly. In practice, this setup reflects a supply chain scenario, where a supplier and an Original Equipment Manufacturer (OEM) would jointly benefit from collaborative model training while keeping their respective data private. By reproducing such collaboration under realistic privacy restrictions, the SmartFactory KL demonstrator allows us to simulate both types of plausible industrial use cases: collaboration between competitors and collaboration within the supply chain. The objective in this experiment is to enable multiple collaborating clients to classify windshield types and detect assembly product type in a distributed manner, thereby demonstrating the practical applicability of FL in an industrial setting. Prior to deployment on custom industrial datasets, the proposed FL architecture was validated using a standard benchmark dataset.

3.2.1 Datasets

To validate the proposed FL architecture, initial experiments were conducted using the well-established **CIFAR-10** dataset, comprising 60,000 images across 10 classes. This dataset is widely used in the literature and serves as a standard benchmark for federated image classification tasks. As highlighted in Section 2.3, the majority of related works adopt CIFAR-10 as a baseline, making it an ideal candidate for comparative evaluation. The dataset was partitioned across five clients, each receiving a full set of 10 classes. The number of clients was deliberately limited to reflect realistic manufacturing scenarios, where the number of federated participants is typically small. Despite its popularity, the CIFAR-10 dataset poses limitations for industrial use cases, particularly in quality inspection. The image resolution of 32×32 pixels is insufficient for fine-grained visual inspection, as illustrated in Figure 3.3. Even with enlargement, these images remain challenging to interpret for both humans and Deep LearningDL models, underscoring the need for higher-resolution, domain-specific datasets.

USB Quality Inspection Dataset

To address this problem, a custom dataset was developed for a quality inspection use case of USB Type-A ports, with collaboration of three independent partners/clients, namely ‘Huawei’, ‘SmartFactory KL’ and ‘WSKL’. Each client provided images of their respective USB product for

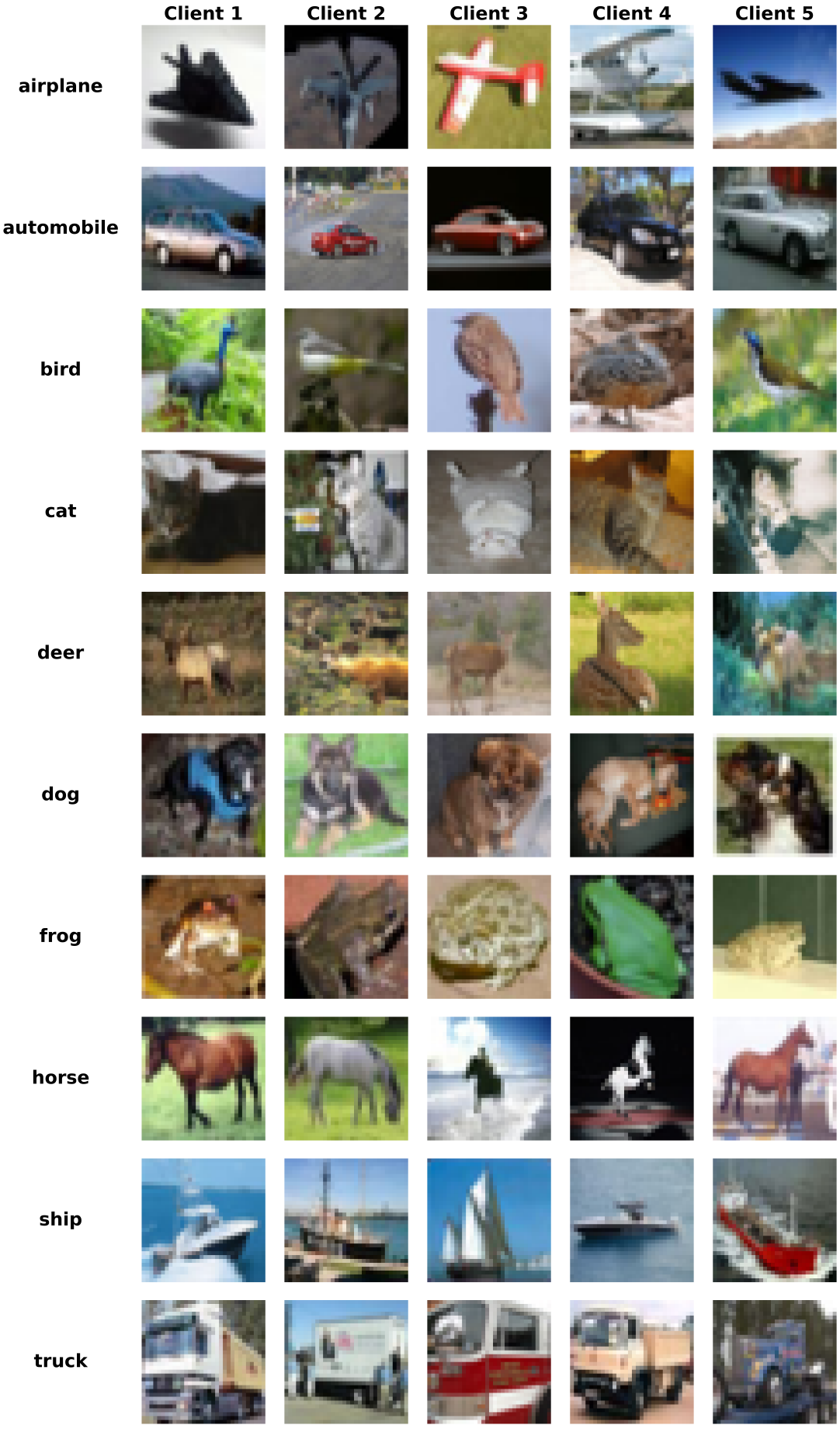

Figure 3.3: Illustration of the partitioning of the CIFAR-10 dataset across five clients [Kr09].

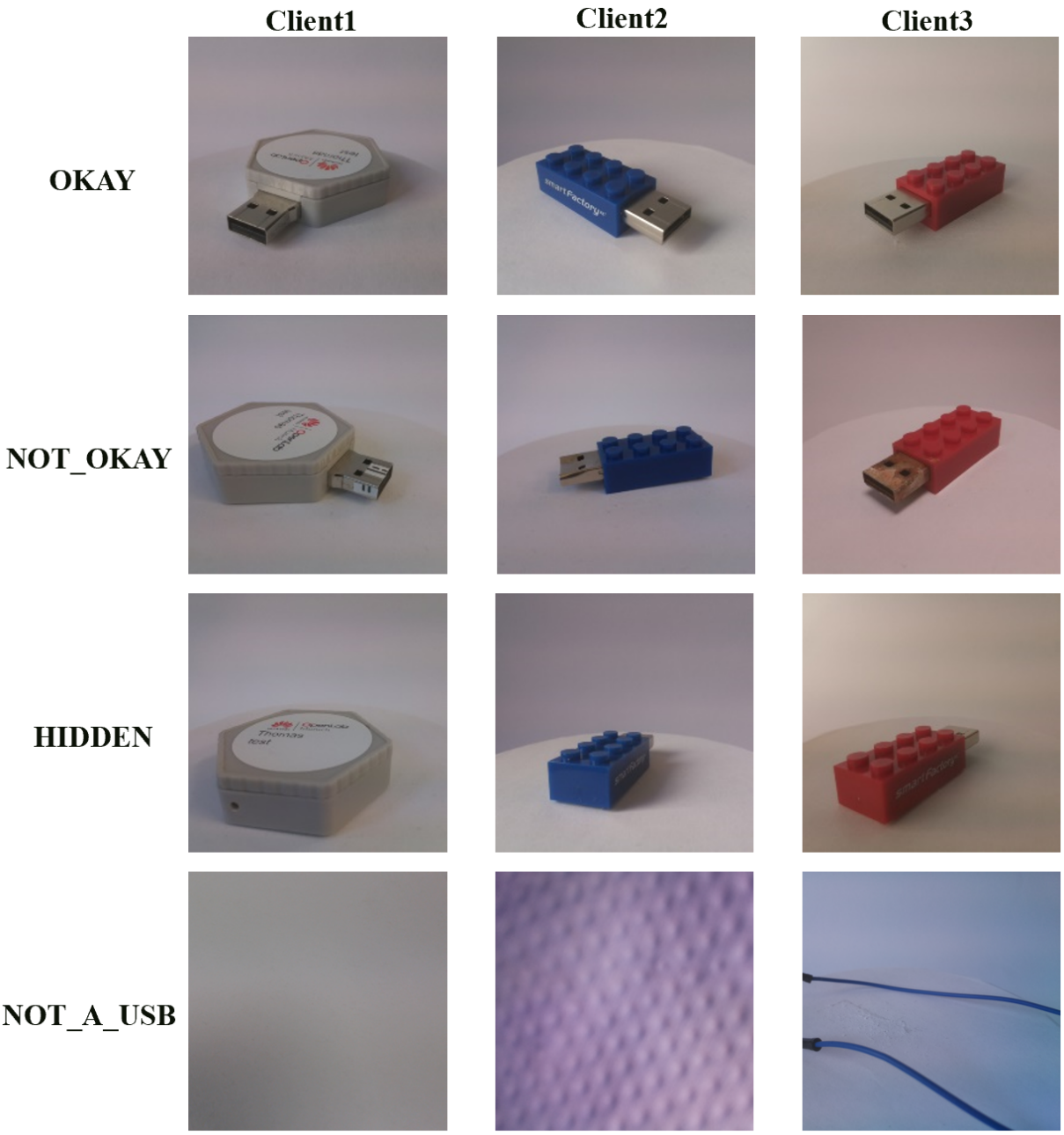

Figure 3.4: Federated learning dataset for USB Type-A port quality inspection across three clients, where each client holds images of its unique USB type. Although the class labels (*OKAY*, *NOT_OKAY*, *HIDDEN*, and *NOT_A_USB*) are common across clients, the physical errors and object appearances differ, reflecting client-specific visual heterogeneity for evaluating federated learning performance.

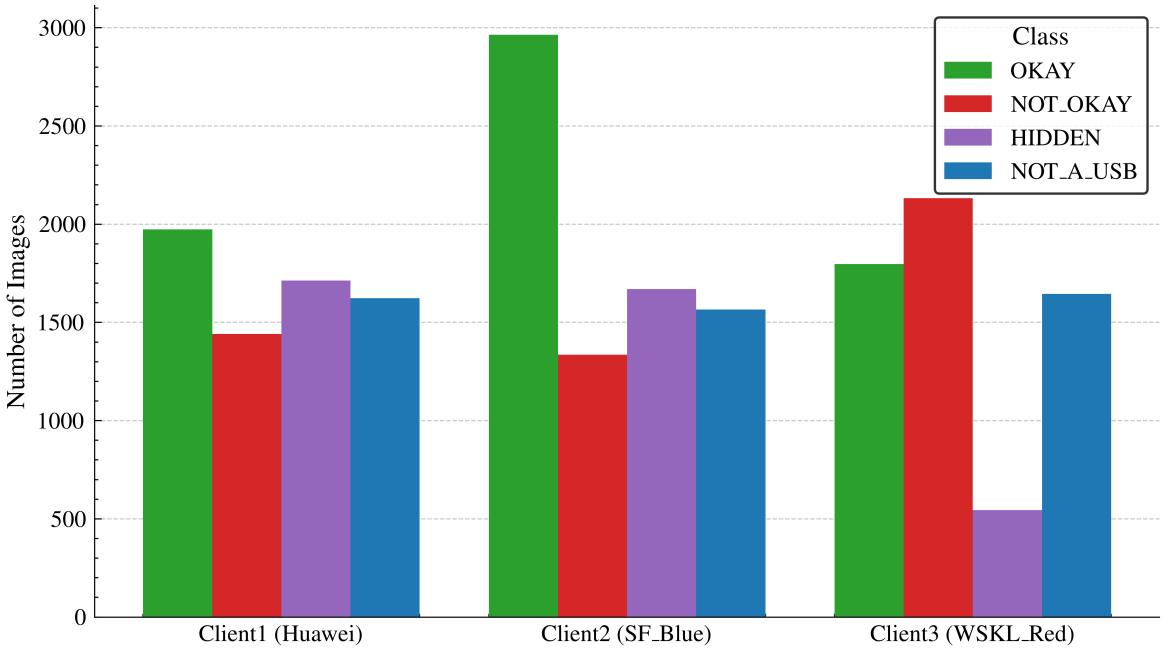

Figure 3.5: Class-wise distribution of the USB dataset across three clients, highlighting the non-IID nature of the data splits.

the classification task of determining the condition of the USB port. The dataset comprises four semantic classes: 'OKAY', 'NOT_OKAY', 'HIDDEN', and 'NOT_A_USB'. The inclusion of the NOT_A_USB class was crucial for enabling the model to distinguish between valid USB images and irrelevant or empty scenes. Without such a buffer class, the model would be forced to assign one of the remaining classes even when the USB port is not present in the image. Client 1 contributed data from a uniquely shaped hexagonal USB device manufactured by Huawei at OpenLab Munich. This device includes a rotating body mechanism that sometimes conceals the USB port, contributing to the HIDDEN class. The NOT_OKAY class was constructed by applying sticker-based artifacts onto the port surface, simulating manufacturing defects. The overall visual appearance of this client's USB is notably distinct from standard designs, introducing diversity to the FL process. Client 2 provided a blue LEGO-style USB stick. In contrast to Client 1, the NOT_OKAY class for this client was created by physically damaging the USB port using pliers, offering a different visual context for defects while retaining the same label semantics. Client 3 used a red LEGO-style USB stick similar to that of Client 2, but introduced rust as the defect mode in the NOT_OKAY class. All three clients adhered to a common label schema, enabling collaborative training while introducing heterogeneity in the data sources. Figure 3.5 shows the distribution of image classes across all three clients, highlighting the non-IID nature of the dataset. The imbalance in the number of samples per class and per client emulates real-world industrial conditions, where data availability is uneven and often constrained. Furthermore, the use of a unified NOT_OKAY label across diverse defect types strengthens the global model's ability to generalize over different failure modes, which is an essential requirement for scalable federated quality inspection systems. This makes the clients ready for an unseen error in their manufacturing and also benefits from being a part of the federated model training process.

Cabin Windshield Classification Dataset

To further evaluate the performance of federated image classification under more complex and realistic non-IID scenarios, a new dataset was created using 3D-printed cabin models with five semantic classes: '`No_windshield`', '`Type A`', '`Type B`', '`Type C`', and '`Type D`'. Fig. 3.6 shows a schematic view of the cabin design along with different windshield types available in total. Each client was assigned cabins of a distinct color and slightly varied design to introduce visual

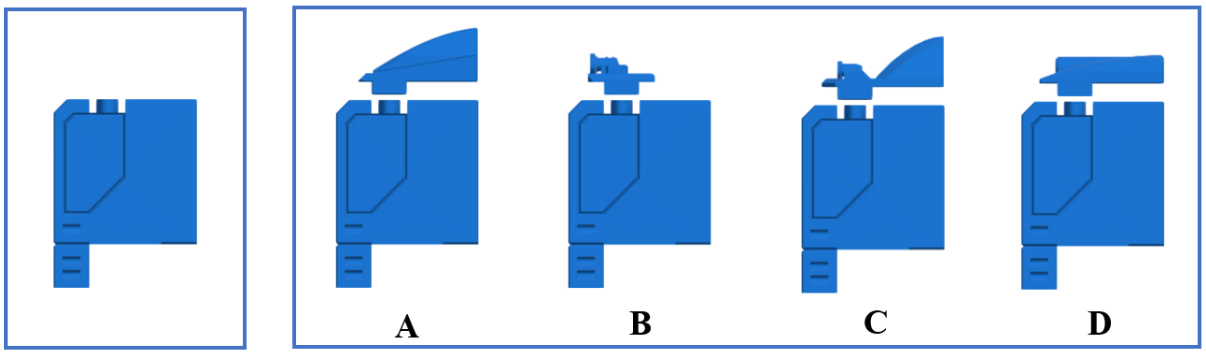

Figure 3.6: Schematic of Cabin Designs and Windshield Types in the SmartFactory-KL ecosystem.

domain shifts. Additionally, every client was deliberately excluded from one specific windshield type to create class-heterogeneous training sets. As shown in Fig. 3.7, Client 1 lacks samples for `Type A`, Client 2 for `Type D`, Client 3 for `Type C`, and Client 4 for `Type B`.

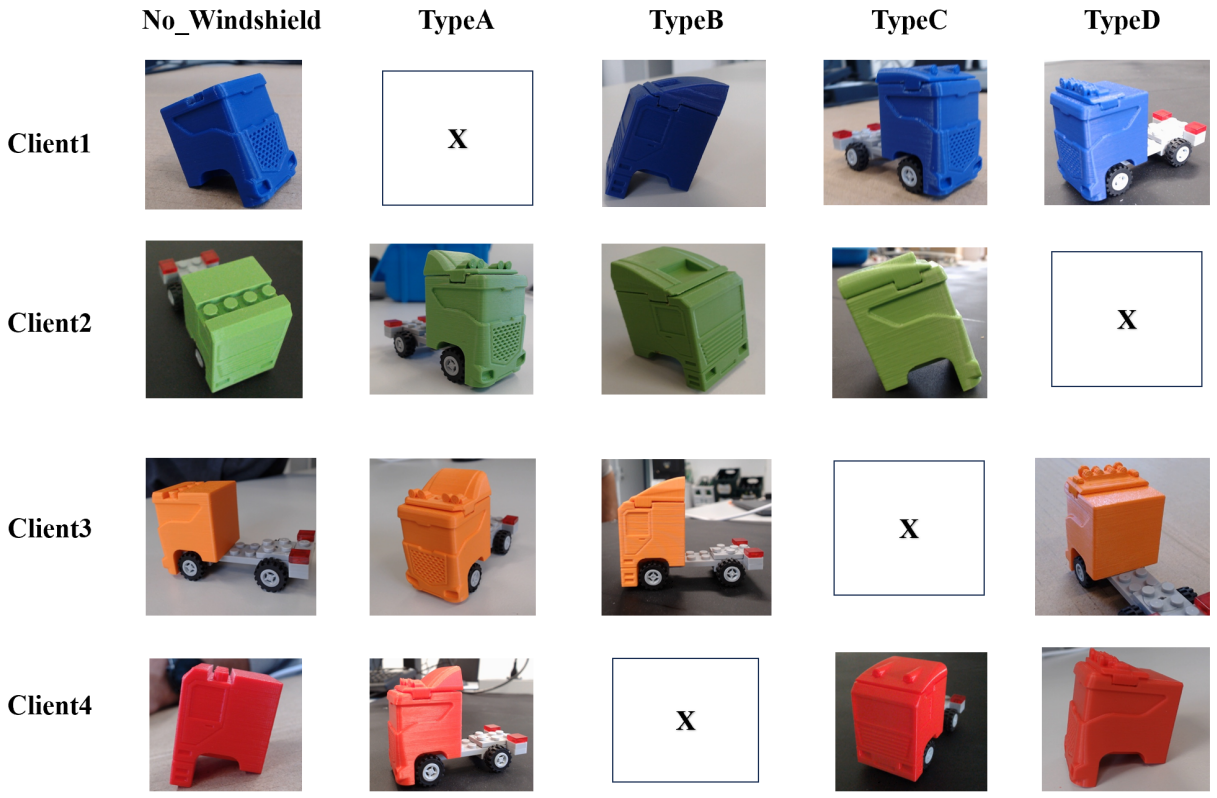

Figure 3.7: Federated dataset distribution across four clients, where each client is missing one windshield type, resulting in a class-heterogeneous non-IID training setup.

This setup simulates real-world industrial scenarios in which regional manufacturers or suppliers may not encounter the full range of product variants during training their model. Nevertheless, the federated model is expected to learn a generalizable classifier that can recognize all five classes across all clients, despite local data being incomplete. To evaluate model generalization in such cross-client data gaps, the test set for each client includes examples of the class absent during training. In addition, an independent external test dataset containing all five classes under unseen environmental conditions was created to assess model robustness in fully unfamiliar domains. The complete class-wise distribution across the training, validation, and test sets for all clients is presented in Table 3.1.

Table 3.1: Class distribution across training, validation, and test splits for each of the four clients in the cabin windshield dataset. Missing classes (highlighted in red) reflect the intentionally heterogeneous data partitions used to simulate non-IID conditions.

Client	Set	No_windshield	TypeA	TypeB	TypeC	TypeD
Client1	Train	409	0	404	440	441
	Val	87	0	86	94	94
	Test	89	100	88	95	96
Client2	Train	382	398	375	394	0
	Val	82	85	80	84	0
	Test	83	86	81	85	77
Client3	Train	408	427	445	0	422
	Val	87	91	95	0	90
	Test	89	93	97	86	91
Client4	Train	362	352	0	360	352
	Val	77	75	0	77	75
	Test	79	76	77	78	76

3.2.2 Federated Image Classification Architecture and training process

Starting with the CIFAR10 dataset for federated image classification, different versions of VGG neural network and Residual Network (ResNet), like VGG-11, VGG-16, VGG-19, ResNet-34 and ResNet-50, were used. The goal here was to select the model with the highest accuracy on the test dataset, and hence VGG-19 was selected for the FL process. The fully connected neural network layer of VGG-19 was changed to a custom layer to reduce the number of trainable parameters. It can be referred in Fig. 3.8, where the fully connected layers of the architecture is changed. Federated image classification follows the training process already mentioned in Section 3.1, and the image classification model is replaced by the VGG-19 architecture for the USB quality inspection use case. All the clients have the custom VGG-19 architecture mentioned

in Fig. 3.8. The complete implementation, including scripts and configurations, is available on the public GitHub repository[1].

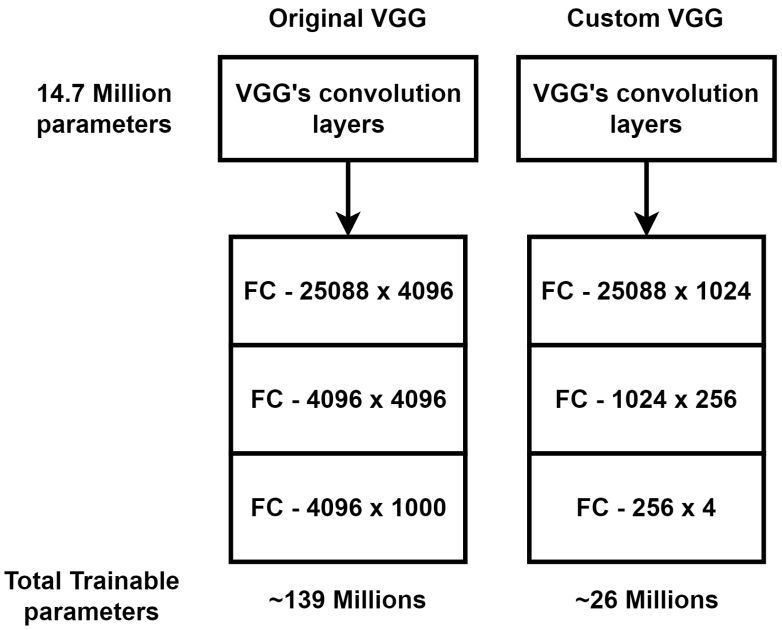

Figure 3.8: VGG 19's original fully connected (FC) layers vs Fine-tuned VGG's custom FC layers.

For the second use case of quality inspection of cabins with windshield types, the centralized model was trained with different architectures to find the best performing architecture using both federated and normal centralized dataset (centralized dataset = merging all train, validation, and test datasets of all clients into a centralized train, validation, and test dataset). Architectures such as VGG-19, ResNet-50, DenseNet-121, and EfficientNetv2 were trained and tested. This was done to also test how federated global model trained using various architectures perform against each other. All the architectures consist of variable trainable parameters, as shown in Table 3.2. Here implementation was done using the Flower framework, and the scripts and configurations are available on the public GitHub repository[2].

Table 3.2: Comparison of image classification architectures based on the number of trainable parameters (M = million).

Model	Parameters	Notes
DenseNet-121	8M	Balanced trade-off between accuracy and efficiency.
ResNet-50	23M	Higher accuracy but computationally expensive.
VGG-19	139M	Large model, impractical for FL on constrained devices.
MobileNetV2	3.4M	Compact but lower accuracy.
EfficientNetV2-S	21M	Efficient but requires higher computational power.

[1] https://github.com/vnt1537/Fed_USB_classification

[2] https://github.com/vnt1537/federated_CabinWindshield_flower

3.3 Federated Object Detection for Quality Inspection

The natural progression from image classification was the incorporation of object detection into the field of quality inspection. Object detection is explained in detail in Section 2.2. While classification determines the object class or label in an image, its drawback lies in the fact that it can be applied to only a single object within a given frame. Furthermore, image classification does not provide the spatial coordinates of a defect or component within the product. In contrast, object detection extends this task by identifying both the object class and its precise location within the image. This spatial awareness is particularly critical in industrial settings, for example in pinpointing the location of cracks, scratches, or missing parts on a product, or in enabling robotic arms to perform pick-and-place operations that require accurate localization.

3.3.1 Datasets

This subsection presents two FedOD use cases, building directly on the classification tasks from the previous sections: USB quality inspection and the SmartFactory-KL cabin models. Each subsection elaborates on dataset creation, annotation strategy, and architectural choices for FedOD. A core contribution of this study is the analysis of **bounding box variability** across clients, which is a unique challenge for federated object detection. Variability arises when the same object type appears in slightly different shapes, scales, or backgrounds across clients. This heterogeneity makes federated aggregation more difficult compared to image-level classification, as the global model must learn to generalize not only across label distributions but also across differences in object geometry and annotation consistency.

USB Quality Detection Dataset

The USB dataset introduced earlier for image classification (Fig. 3.4) was extended to object detection by manually annotating each image in the YOLO format. In this format, every object instance in an image is represented by a single line in a corresponding `.txt` file with the structure:

```
<class_id> <x_center> <y_center> <width> <height>
```

where the `class_id` identifies the object label, while `x_center`, `y_center`, `width`, and `height` specify the bounding box in normalized coordinates relative to the image size. A visual explanation of this format is shown in Fig. 3.9, where the purple rectangle denotes the bounding box and the red dot indicates its normalized center.

All images in the dataset were annotated using makesenseAI[3], an open-source annotation tool.

[3] www.makesense.ai

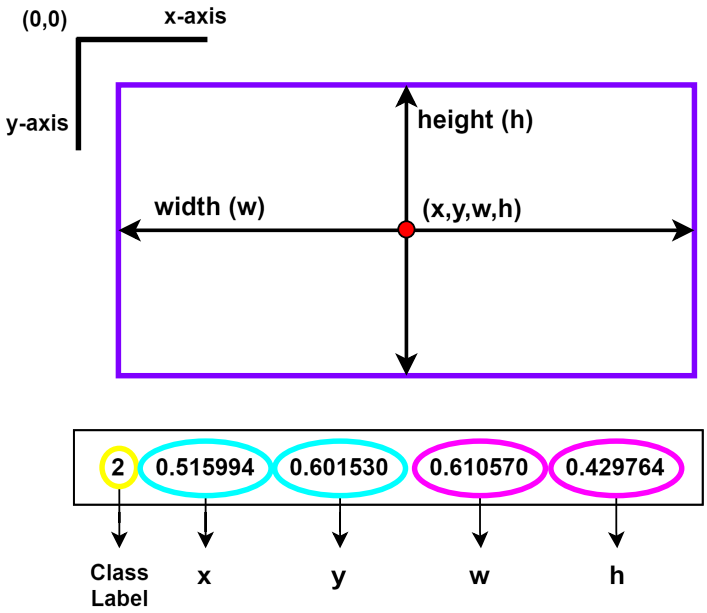

Figure 3.9: YOLO annotation format: The purple box denotes the bounding box. `(x, y)` is the normalized center, and `w, h` are width and height relative to the image size. The red dot marks the bounding box center.

Annotation is one of the most labor-intensive parts of any object detection pipeline, as it requires manually drawing bounding boxes around every object instance. The tool exports the annotations in YOLO-compatible `.txt` files, ensuring direct compatibility with the chosen detection architecture. The simplicity and compactness of the YOLO format make it particularly suitable for real-time object detection tasks, and its efficiency has contributed to its wide adoption in both academic research and industrial practice [Jo20; Re16; RF18].

For the detection experiments, the `EMPTY` class used in the classification task was excluded, since object detection focuses on identifying objects present in an image frame. The remaining three classes were indexed as follows: `OKAY` (class 0), `NOT_OKAY` (class 1), and `HIDDEN` (class 2). Figure 3.10 shows representative annotated samples for each client, illustrating the different conditions and defect variations that occur in real-world manufacturing setups.

To further enrich the dataset, additional samples were collected with variations in background textures and lighting conditions. This augmentation step reflects real-world deployment, where inspection cameras are often affected by changes in illumination or object positioning. The final dataset is characterized not only by the diversity of defect types but also by a non-uniform distribution of classes across clients. Figure 3.11 presents the class-wise distribution, showing that the number of samples varies both across clients and across classes. This imbalance is a key source of non-IID data distribution in the federated setup, and it poses additional challenges for training a global model that generalizes well across all clients.

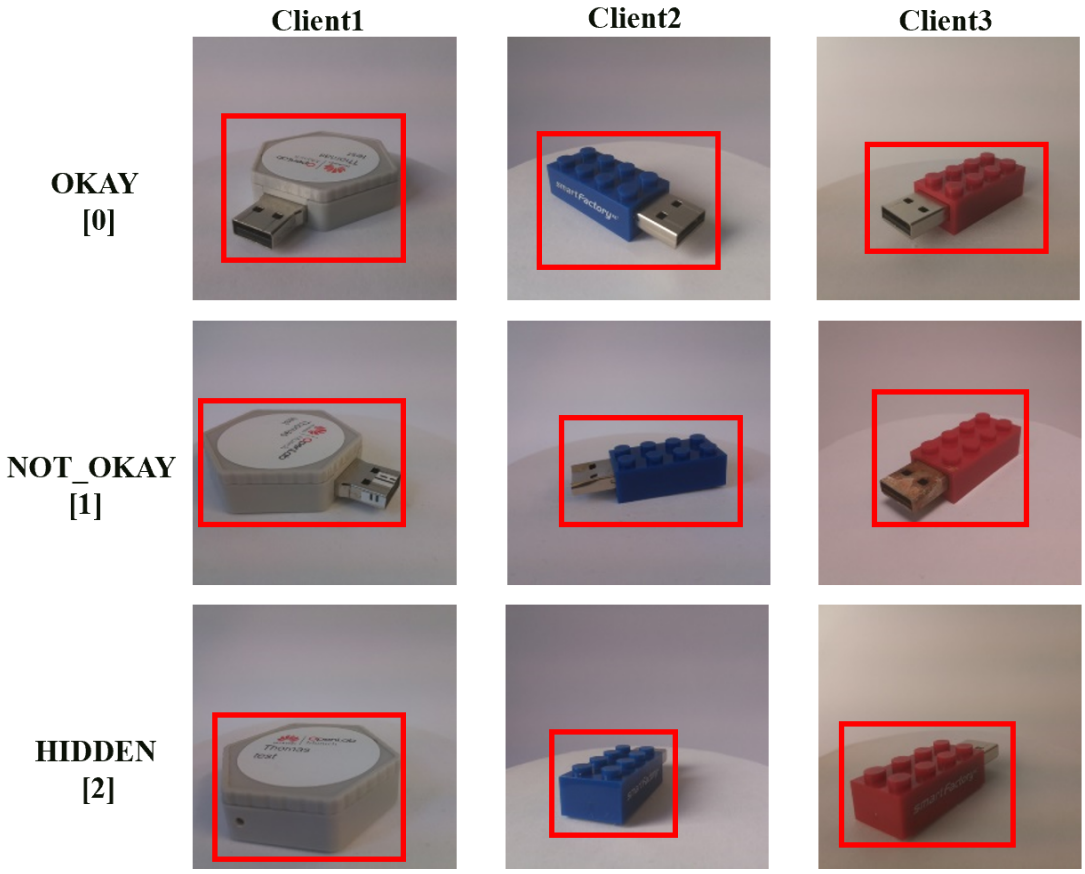

Figure 3.10: The USB classification dataset was extended with bounding box annotations (shown in red) to create the USB object detection dataset across three clients. Each client retains its original local data characteristics, enabling evaluation of federated object detection performance.

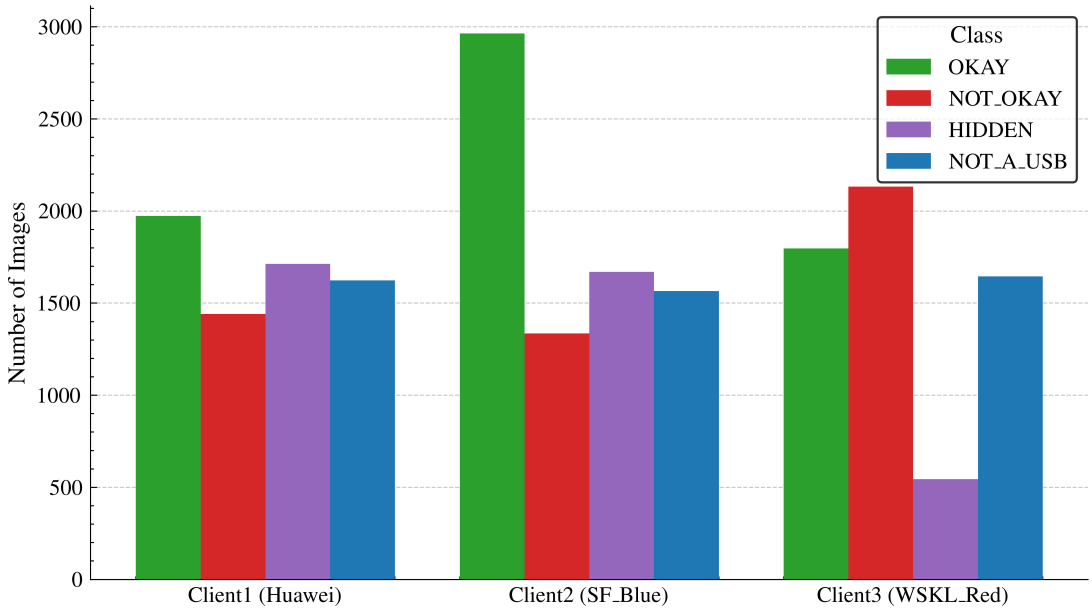

Figure 3.11: Class-wise distribution of the USB dataset across three clients, highlighting variations in both dataset size and class proportions. This reflects the non-IID nature of the federated setup.

SmartFactory Cabin Detection Dataset

To further evaluate the generalization capability of the proposed FedOD framework, a second dataset was created focusing on the quality inspection of toy cabin models with the objective of detecting the presence or absence of windshields. These 3D-printed cabin models were specifically developed within the SmartFactory KL truck ecosystem to simulate realistic industrial inspection scenarios. Fig. 3.6 shows a schematic view of the cabin design along with different windshield types available in total. In this use case, the setup simulates two distinct manufacturers: the first produces blue cabins equipped with windshield types A and B, while the second manufactures red cabins featuring windshield types C and D, as illustrated in Fig. 3.12. Multiple

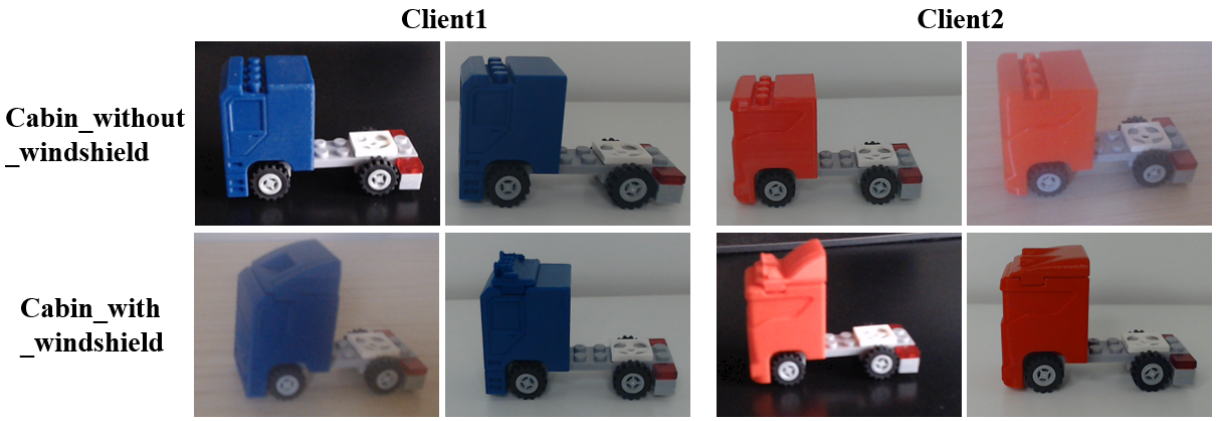

Figure 3.12: Sample images from Client 1 and Client 2 datasets, Class 0: *Cabin_without_windshield*, Class 1: *Cabin_with_windshield*. Each client holds an exclusive set of windshield types, with Client 1 containing types A and B, and Client 2 containing types C and D in their training datasets.

images were captured for each cabin–windshield combination, resulting in a structured dataset representing both clients. To introduce domain variability, data were collected under three different background settings, i.e., plain black, brown, and white surfaces and included both artificial lighting and natural daylight exposure. Furthermore, a controlled level of motion blur was deliberately introduced in selected samples to simulate practical imperfections commonly encountered in industrial environments. These domain-specific variations are exemplified in Fig. 3.12. This controlled diversity ensures that the trained model is exposed to a wide range of visual contexts, improving its robustness against real-world variability encountered in industrial production lines.

Both clients aim to expand their product lines by manufacturing additional cabin–windshield combinations previously produced by the other. However, the exchange of raw image data and annotations is constrained by privacy regulations and competitive concerns. Consequently, collaborative model development must be achieved without sharing locally stored visual data. The proposed FedOD framework addresses this challenge by enabling decentralized model training, allowing both clients to contribute to a globally improved model while preserving full ownership and privacy of their respective datasets. The heterogeneity in client data extends beyond class imbalance, encompassing differences in lighting setups, backgrounds, product colors, and

windshield types. These domain shifts introduce client-specific biases that impact generalization and model robustness. These differences are illustrated in Fig. 3.13.

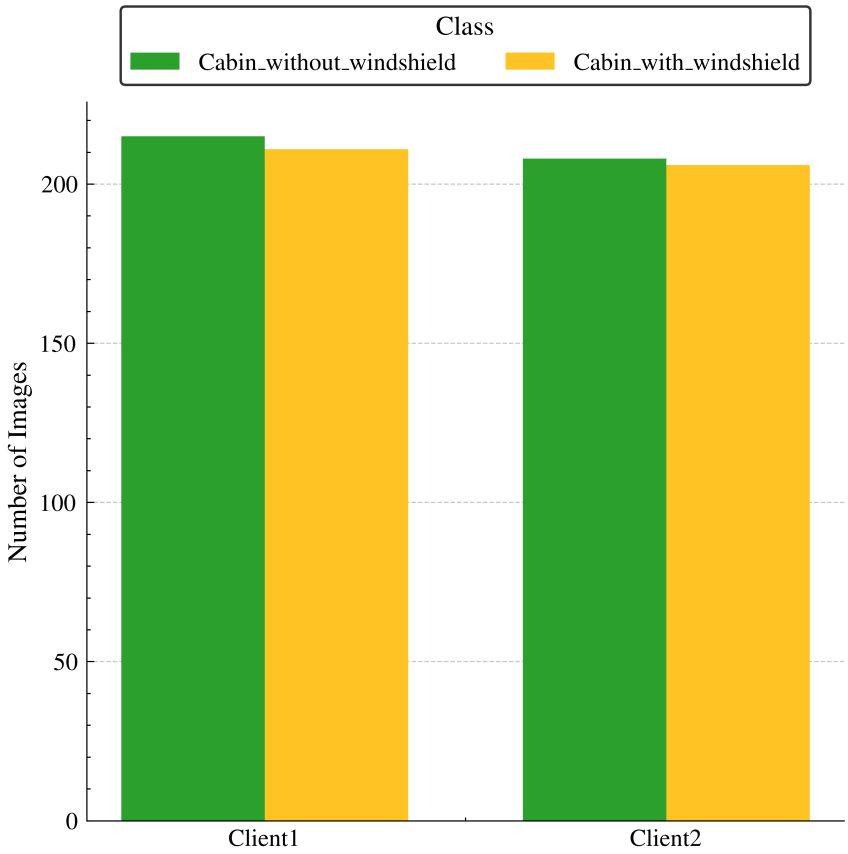

Figure 3.13: Distribution of classes for object detection across Client 1 and Client 2.

The final objective is to train a federated global model capable of accurately detecting whether the object in the frame is a ‘`Cabin_without_windshield`' or ‘`Cabin_with_windshield`’, irrespective of the specific windshield type. To evaluate this, a cross-client generalization test was conducted in which each client was exposed to windshield types not present during their local training (i.e., Types C and D for Client 1, and Types A and B for Client 2). Such cross-domain testing conditions are critical for evaluating whether the learned representations in FL are transferable beyond the local domain-specific biases. This setup enables the assessment of FedOD’s ability to generalize beyond locally observed classes, reflecting real-world scalability requirements in collaborative industrial deployments.

3.3.2 Federated Object Detection Architecture and Training Process

For this research, the YOLOv5 architecture was selected due to its balance between inference speed and accuracy. While recent versions like YOLOv8 and YOLOv11 were not available during the time of this research, YOLOv5 was the SOTA and was the primary architecture for most experiments due to its extensive community support and modularity at the time of development. Implementing FL with YOLOv5 required fully customized coding. Popular frameworks like Flower

and PySyft were not suitable as they lacked native support for object detection pipelines and constrained flexibility in handling YOLO-specific data flows. As a result, the federated training process, including model initialization, weight selection, evaluation, and aggregation, was built independently and integrated directly with the YOLOv5 codebase [Jo20].

The overall training process follows the FL architecture described earlier (Fig. 3.1), with the YOLOv5 network acting as the base detection model deployed at each client. During each CR, the central server first distributes the current global model to all participating clients. Each client then performs local training on its respective annotated dataset for a defined number of epochs. After local training concludes, the client selects the best-performing model based on validation metrics, rather than simply using the final epoch weights. These best weights, along with the associated evaluation metrics, are transmitted back to the server. Upon receiving updates from all clients, the server performs model aggregation using FedAvg strategy to compute the updated global model. This new global model is then broadcast to all clients, initiating the next CR in the training cycle. In terms of metrics, mAP metrics is used as accuracy for the threshold. A visual overview of the deployment setup across two clients is provided in Fig. 3.14. The server and clients are synchronized in real-time through a central scheduler that coordinates the CRs and weight updates.

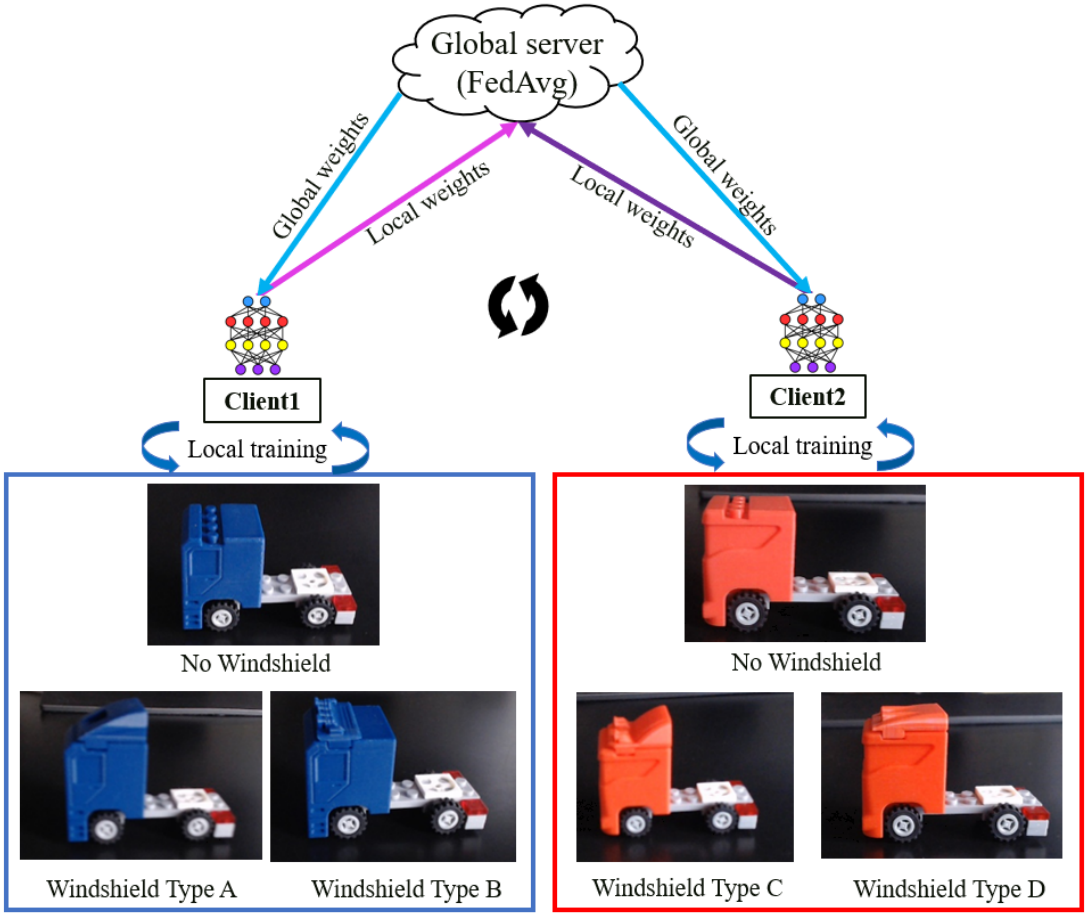

Figure 3.14: Visualization of federated object detection between Client 1 (Blue Cabin) and Client 2 (Red Cabin), each trained on distinct windshield types and aggregated using the FedAvg algorithm.

A key technical contribution of this work is the integration of FL into the YOLOv5 architecture. The implementation is available on public GitHub repositories for the USB Type-A detection use case[4] and for the Cabin-Windshield use case[5]. A unified codebase supports both use cases, with configurable dataset paths and parameters defined in a `config.yaml` file. This setup enables consistent client training and server aggregation across tasks. The objective is to train a federated global model capable of accurately detecting and localizing objects across clients. Unlike classification, the model must predict bounding boxes, which may vary due to differences in anchor box distributions across local datasets.

3.4 Federated Ensemble Learning

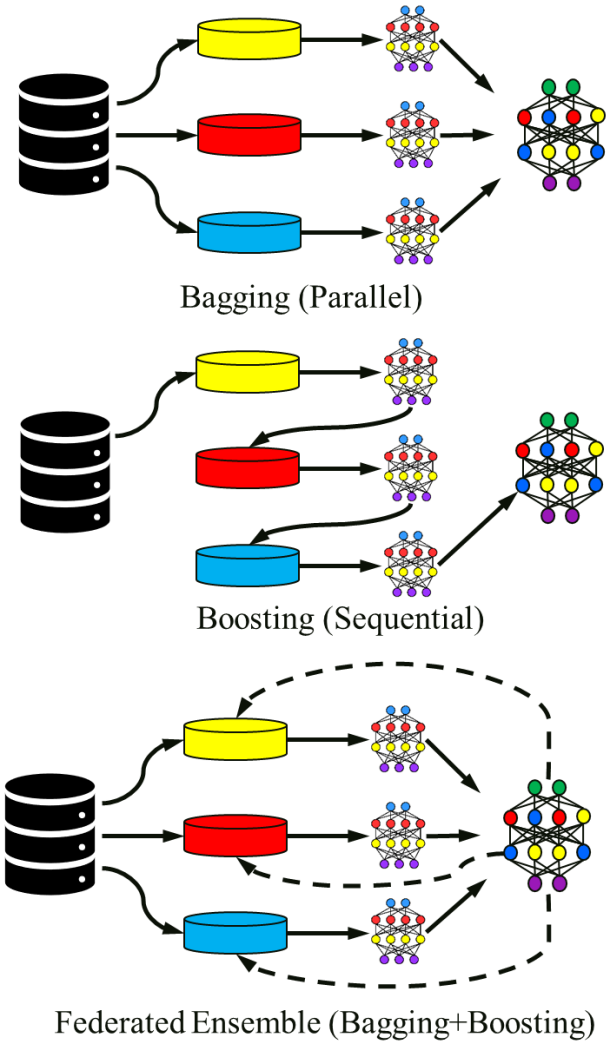

Figure 3.15: Bagging, Boosting and Federated Ensemble Learning Algorithms

Ensemble learning refers to the technique of combining multiple base models to produce a more robust and accurate model than any individual constituent (Fig. 3.15). Traditional ensem-

[4] https://github.com/vnt1537/FL_USB_YOLOv5
[5] https://github.com/vnt1537/federatedlearning_YOLOv5_cabindetection

ble approaches include ***bagging*** and ***boosting***. Bagging (Bootstrap Aggregating) trains multiple models in parallel on different subsets of the dataset and then aggregates their predictions, often through averaging or majority voting. This method is particularly effective at reducing variance. Boosting, on the other hand, trains models sequentially, where each new model focuses on the errors made by the previous ones, thus aiming to reduce bias.

FL though primarily developed to ensure data privacy across decentralized clients, also inherently exhibits ensemble-like behavior. Each client trains a local model on its private dataset, and the server aggregates these models into a single global model. **FedEnsemble** shows that if a limited centralized dataset is available, how FL can be used as an ensemble method to achieve a more robust model. The ensemble approach also provides an inherent robustness to outliers and domain shifts, making it suitable for FedOD tasks. Further subsections showcases the dataset creation for FedEnsemble learning use case, architecture, and training process.

3.4.1 Datasets

The FedEnsemble framework proposed in this research is based on partitioning a centralized dataset into multiple clients, drawing inspiration from ensemble learning techniques such as bagging and boosting. This approach enables the training of multiple weak classifiers (clients) independently, which are subsequently aggregated within the FedOD process to enhance the overall model generalization and performance. By combining the diversity of independently trained client models, the FedEnsemble framework achieves improved robustness and mitigates overfitting, particularly in heterogeneous and limited-data manufacturing scenarios.

Trailer Quality Inspection Dataset

For the first use case, a centralized trailer dataset containing three distinct trailer classes was utilized. The classes include ‘`trailer_body_blue`’, ‘`trailer_body_white`’, and ‘`trailer_body_white_penholder`’, where the third class represents a toy trailer with a milled structure, distinct from the other two classes produced using 3D printing technique. The dataset was divided randomly across three clients, ensuring that each client had access to all three classes within their local dataset, as illustrated in Fig. 3.16. The images were captured under varying real-world conditions, including three distinct backgrounds (plain black, brown, and white), multiple lighting environments (natural and artificial), and different camera perspectives to introduce variability. Each trailer class comprised a total of 300 images, out of which 240 images were allocated for training, 30 images for validation, and 30 images for testing per client.

Figure 3.16: Federated Ensemble Trailer Dataset: Centralized trailer dataset divided randomly across 3 clients for FL experiments.

Cabin Quality Inspection Dataset

To further evaluate the robustness of the FedEnsemble framework, the previously developed cabin dataset (Fig. 3.12) was centralized into a single dataset. This combined dataset was then randomly shuffled and partitioned into three clients, ensuring a balanced distribution of samples across participants. The data distribution and structure of this federated cabin dataset are shown in Fig. 3.17.

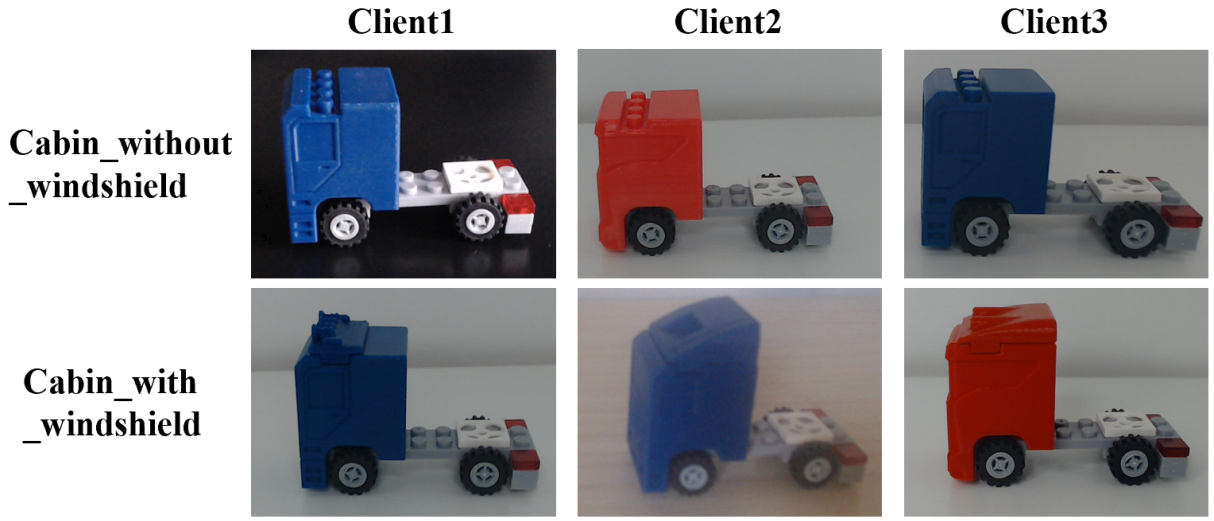

Figure 3.17: Federated Ensemble Cabin Dataset: Centralized cabin dataset divided into 3 clients for FL experiments.

3.4.2 Federated Ensemble Algorithm and Architecture

The proposed FedEnsemble architecture integrates the core principles of bagging and boosting into the FL paradigm to enhance model generalization and robustness, especially for object detection tasks in industrial environments. A centralized dataset $\mathcal{D}$ is partitioned into K non-overlapping subsets $\{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_K\}$, where each client k receives a distinct dataset $\mathcal{D}_k$. Training proceeds locally per client, resulting in a local model $f_k^{(t)}$ for CR t.

Following each round, the global model $f_G^{(t)}$ is computed using weighted averaging

$$f_G^{(t)} = \sum_{k=1}^{K} \frac{n_k}{n} f_k^{(t)}, \quad \text{where} \quad n = \sum_{k=1}^{K} n_k$$

Here, n_k denotes the number of samples on client k. This weighted aggregation captures the essence of **bagging**, where independently trained models on different data partitions are combined to reduce variance and improve stability. By leveraging diverse data distributions from heterogeneous clients, FL implicitly introduces model diversity, which one of the cornerstones of bagging.

Algorithm 6: Federated Ensemble Learning (FedEnsemble)

Require: Centralized dataset $\mathcal{D}$, number of clients K, total rounds T, local epochs E

1: Partition $\mathcal{D}$ into $\{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_K\}$ (without replacement)
2: Initialize global model $f_G^{(0)}$ (e.g., using pretrained weights)
3: **for** each round $t = 1, 2, \dots, T$ **do**
4: **for** each client $k \in \{1, 2, \dots, K\}$ **in parallel do**
5: Download $f_G^{(t-1)}$
6: Train local model $f_k^{(t)}$ on $\mathcal{D}_k$ for E epochs
7: Select best-performing local weights (e.g., via validation)
8: **end for**
9: Aggregate weights:

$$f_G^{(t)} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} f_k^{(t)}$$

10: **end for**
11: **return** Final global model $f_G^{(T)}$

In subsequent rounds, each client begins local training using the newly aggregated global model $f_G^{(t)}$. This resembles the core idea of **boosting**, where knowledge is progressively refined across iterations by building upon prior model states. Clients benefit from the collective learning information encoded in the global model, which helps correct local biases and enhances generalization. Thus, over T CRs, the final model $f_G^{(T)}$ synthesizes the strengths of both ensemble strategies: it leverages cross-client diversity like in bagging, and incorporates iterative refinement akin to boosting. This hybrid mechanism enables FedEnsemble to act as a natural regularizer, mitigating overfitting and enhancing robustness, particularly in non-IID industrial scenarios.

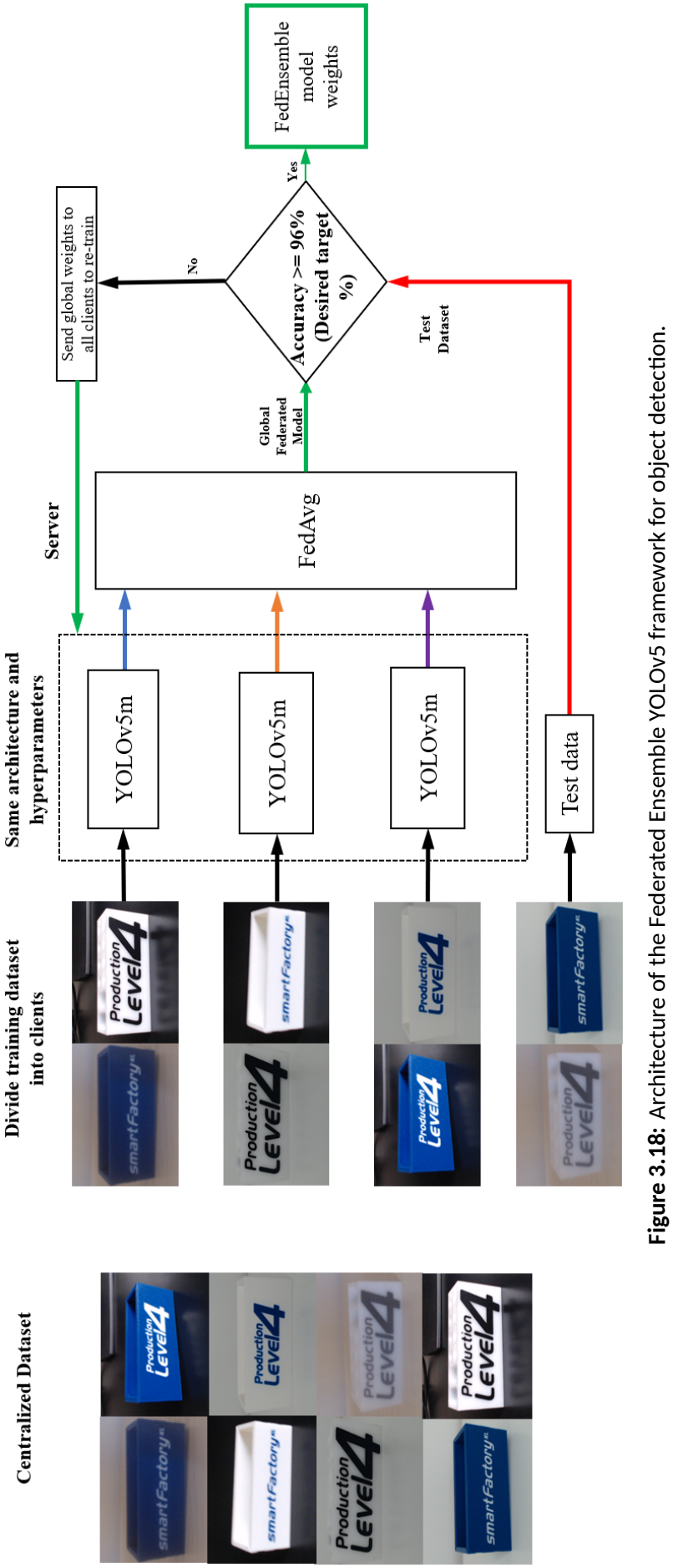

Figure 3.18: Architecture of the Federated Ensemble YOLOv5 framework for object detection.

Figure 3.18 illustrates the training pipeline for the trailer dataset. A centralized dataset containing three trailer classes is split into three clients without replacement. Each client trains a YOLOv5m model under varied imaging conditions. After local training, clients send their best-performing weights to the server, which aggregates them using FedAvg. This process iterates until a defined accuracy threshold (e.g., in this case 96%) is met. The framework implementation along with scripts and configurations are available on the public GitHub repository[6]. This highlights FedEnsemble architecture's applicability for quality inspection in industrial scenarios with limited amount of dataset.

3.5 Federated Learning with Hybrid Dataset

In industrial quality inspection, a frequent challenge is the scarcity of annotated data, particularly for underrepresented or defective classes. In many scenarios, clients may possess very few annotated samples, either due to operational constraints or inherent dataset bias, such as a lower occurrence of faulty products. This scarcity can significantly hinder the performance and generalizability of ML models trained locally in a federated setting.

Recent studies have shown that augmenting real datasets with high-quality synthetic images can address these limitations by improving class balance and enhancing feature diversity [SK19]. It has been shown that incorporating an equivalent quantity of synthetically generated images, even when the synthetic data does not exhibit perfect visual realism, can significantly improve model accuracy compared to training solely on small real datasets [An24a]. The combined training dataset, comprising real images and their synthetic counterparts, is referred to as a ***hybrid dataset*** in this thesis. An important advantage of synthetic data is its cost-efficiency. Unlike real-world data collection, which requires manual image annotation and physical manipulation of product conditions, synthetic data generation enables automated labeling. Additionally, textures, lighting conditions, backgrounds, occlusions, and camera angles can be programmatically varied without additional labor, thereby reducing both time and cost. This level of control also enables targeted data generation to simulate rare defect scenarios that may be difficult to capture in physical settings.

In this section, the integration of synthetic datasets into the FL process by assigning synthetic data to an entire client is investigated. This hybrid approach evaluates whether the presence of synthetic-only clients can enhance the accuracy and robustness of the federated global model in quality inspection tasks. This setting closely mirrors real-world collaborations, where certain partners may rely entirely on simulated data due to limited on-site defect occurrence or lack of labeled samples.

[6] https://github.com/vnt1537/federated_ensemble_objectdetection

3.5.1 Dataset

The dataset used in this section is distinct from previous experiments and was specifically selected due to its inclusion of multiple small object classes. Small objects, defined as those with dimensions smaller than 32×32 pixels, pose significant challenges for object detection models due to limited pixel representation, reduced feature richness, and lower contextual information [Li21].

To simulate a practical industrial scenario, a hybrid FL setup was designed in which one client contains real images while the other client contains only synthetic images of the same object classes. The real dataset comprises 300 high-resolution images (1080×1080), with each image containing multiple annotated objects, as shown in Fig. 3.19. Bounding boxes for objects are drawn in red. The dataset contains five object classes, summarized in Table 3.3, with the `LED`, `Resistor`, and `Button` classes classified as small objects due to their bounding box dimensions. The remaining classes, `Buzzer` and `Arduino` fall under the medium or large object category.

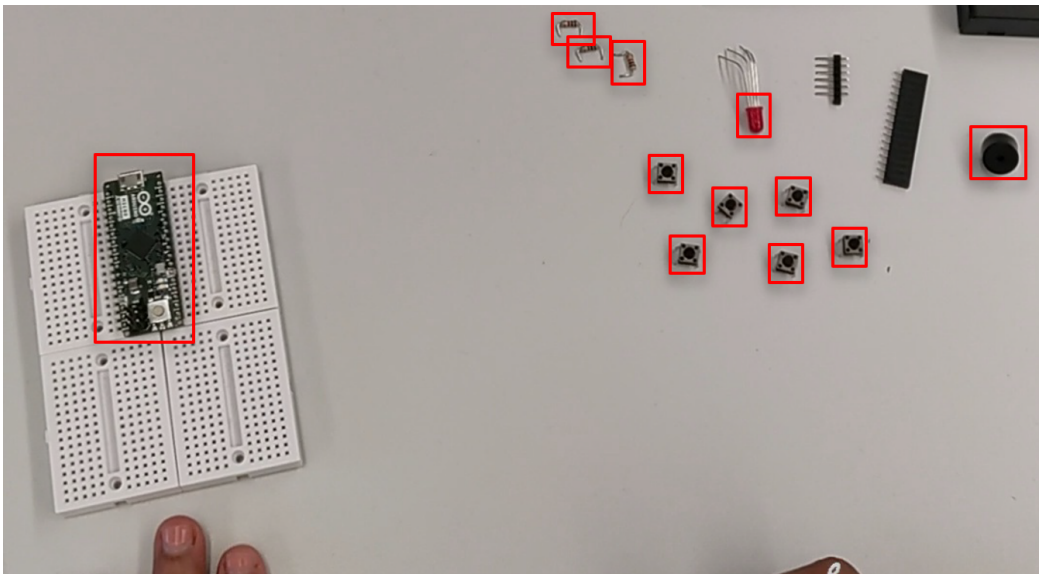

Figure 3.19: Image sample from the Real Images dataset showing all 5 classes along with their annotated bounding boxes.

Client 1 was assigned the real dataset, which contains a total of 3001 bounding boxes across 300 images. Client 2 was provided a fully synthetic dataset, generated using 3D open-source CAD models of equivalent components in Unity. The synthetic dataset also contains 300 images with 2700 bounding boxes. To address class imbalance, the number of Button annotations was deliberately reduced in the synthetic data. A sample from the synthetic dataset is shown in Fig. 3.20, where object locations are again marked with red bounding boxes.

To evaluate the generalization capabilities of all models under distribution shift, an independent test dataset was created in a separate environment. This setting introduced variations in lighting conditions by combining both natural and artificial illumination as well as mild motion blur to simulate realistic industrial inspection scenarios. The dataset comprises 116 images containing

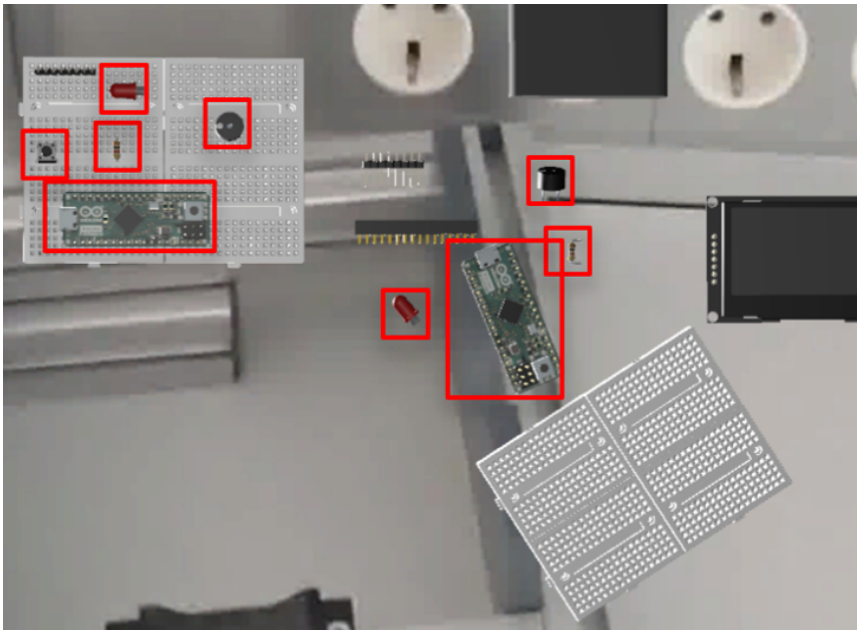

Figure 3.20: Image sample from the Synthetic Images dataset showing all 5 classes along with their annotated bounding boxes. The numbers of Resistor and Button objects were reduced, while other component classes were increased to achieve better class balance.

a total of 1101 annotated bounding boxes spanning the same five object classes. Importantly, 16 of these images do not contain any of the target classes and were intentionally included to assess model robustness in terms of false positive suppression. All the real images were captured using a HOLO-Lens2 camera, which is situated on the user's head. All datasets adhere to the YOLO annotation format to ensure consistency and interoperability within the FL training pipeline.

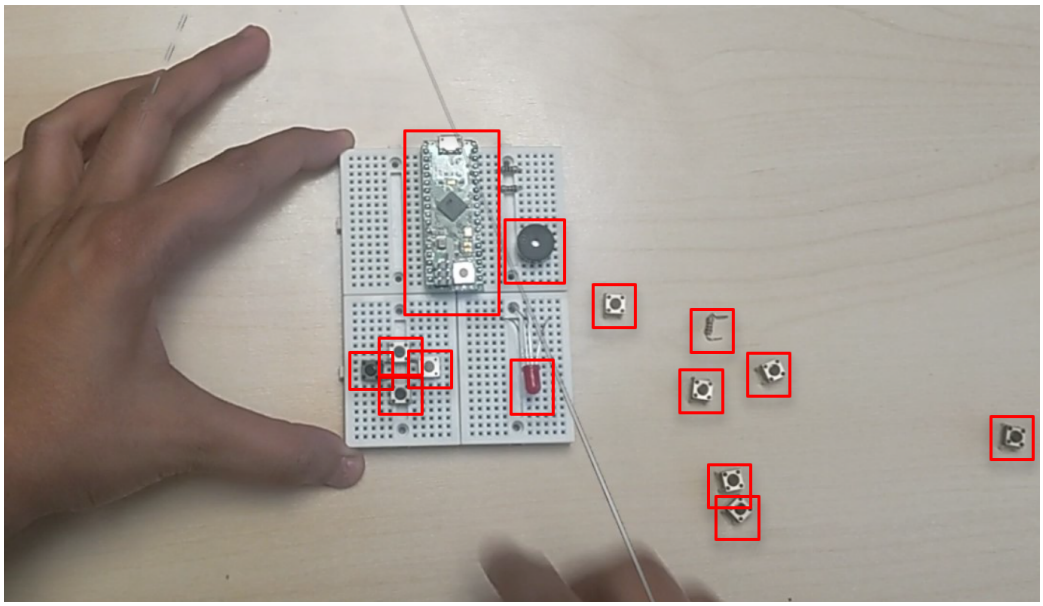

Figure 3.21: Image sample from the External Test dataset showing all 5 classes with annotated bounding boxes under varied background, lighting, and blur conditions.

Table 3.3: Class Labels and Size Categorization for Hybrid Object Detection Dataset

Class ID	Object Class	Size Category
0	LED	Small Object
1	Resistor	Small Object
2	Button	Small Object
3	Buzzer	Medium/Large Object
4	Arduino	Medium/Large Object

3.5.2 Architecture and Training Process

The training process followed the FedOD framework previously introduced in Section 3.3.2 and illustrated in Fig. 3.1, with YOLOv5l employed as the backbone object detection model. Two clients participated in this setup: Client 1 contained real images and Client 2 held only synthetic images. Each client independently trained a local model and transmitted its best-performing weights (selected based on local validation accuracy) to the server. These weights were aggregated using the FedAvg algorithm to form a global model, which was then redistributed to the clients. This procedure was repeated over several CRs to iteratively refine the global model.

The training pipeline remained identical across both clients, demonstrating that synthetic datasets can be seamlessly incorporated into the FL workflow. This interoperability allows data-scarce manufacturing facilities or newly onboarded industrial partners to contribute meaningfully to a federated ecosystem without requiring expensive manual data collection or annotation. While the initial experiments were conducted using YOLOv5l due to its maturity and stability at the time, the pipeline was subsequently extended to support YOLOv8 as the architecture became available during the final stages of this work. This transition further validated the flexibility and forward compatibility of the proposed FedOD framework.

The FedEnsemble strategy, the hybrid dataset was split across three clients, each receiving a unique subset containing both real and synthetic images. A visual comparison of the FedAvg-based hybrid setup and the FedEnsemble configuration is shown in Fig. 3.22 and Fig. 3.23, respectively. To benchmark the federated approaches, multiple centralized training strategies were implemented for comparison. These included: (1) training solely on synthetic data followed by transfer learning and fine-tuning on real data, and (2) direct training on a merged hybrid dataset composed of real and synthetic images. All models (federated and centralized) were evaluated on an independent test dataset collected under previously unseen environmental conditions. The implementation of all the experiments along with scripts and configurations are available on the public GitHub repository[7].

[7] https://github.com/vnt1537/federatedlearning_hybriddataset

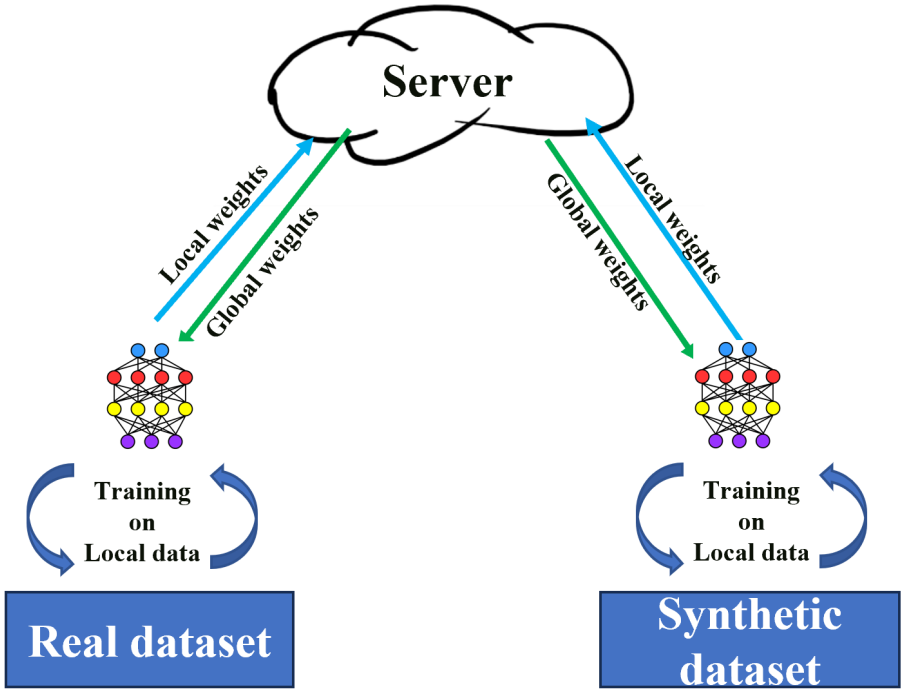

Figure 3.22: Federated learning with hybrid dataset setup, where Client 1 is trained on the Real dataset and Client 2 on the Synthetic generated dataset. This configuration enables performance evaluation between real and synthetic data domains under a federated setting.

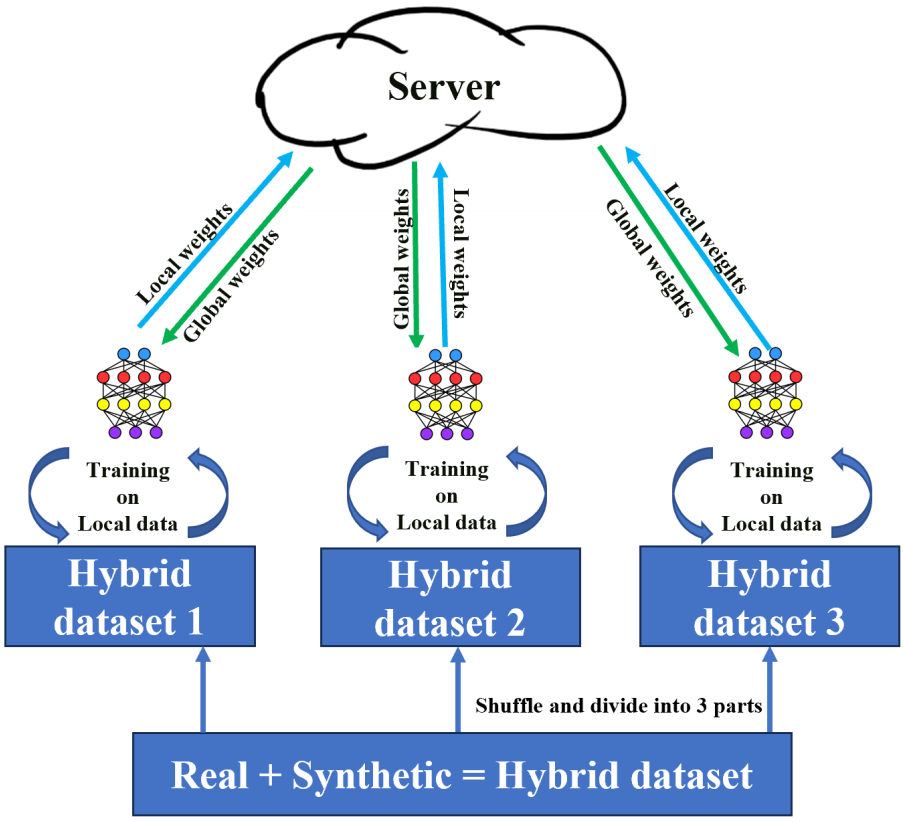

Figure 3.23: Federated ensemble learning with hybrid dataset, where the combined Real and Synthetic data are shuffled and divided into three hybrid clients such that the class distribution across clients remains balanced.

3.6 Summary

This chapter presented a comprehensive methodology and implementation for designing and implementing FL pipelines for industrial computer vision tasks, with a strong emphasis on quality inspection under privacy constraints. The primary objective was to enable collaborative learning across decentralized clients in realistic manufacturing scenarios, where data heterogeneity, limited annotations, and non-IID distributions are prevalent. The chapter first established the foundational FL setup, outlining the client-server architecture, CRs, and privacy-preserving model weight exchange mechanisms. A custom weight selection strategy, based on the best-performing local epoch rather than the last, was uniformly applied to enhance convergence and generalization across all experiments. The framework assumes a secure environment, such as Gaia-X, where all clients are trusted and operate under agreed-upon training protocols.

Building on this foundation, the methodology addressed two core computer vision tasks: image classification and object detection. Federated image classification experiments began with CIFAR-10 as a benchmark and transitioned to a real-world USB quality inspection dataset, involving three clients with diverse defects on their respective product. The architecture was standardized across clients using a fine-tuned VGG-19 network to enable fair model aggregation. To further evaluate FL in a more complex and realistic non-IID scenario, a novel 4-client dataset was developed using 3D-printed cabin models from the SmartFactory KL ecosystem, each equipped with different windshield types. Each client was assigned a unique color and was intentionally missing one windshield class, simulating class-heterogeneous data distribution. This setup introduced stronger domain and label shifts, pushing the FL system to learn generalizable representations even when local data was incomplete. An external test set with all windshield types was used to assess the model's robustness under unseen conditions.

FedOD extended the classification task by incorporating spatial localization requirements using YOLOv5. A fully customized FL training loop was developed for object detection due to the lack of native support in existing frameworks like Flower and PySyft. A key technical focus was placed on anchor box variability and its impact on collaborative detection performance to test the global models' ability to generalize across clients with dissimilar object shapes. The chapter then introduced FedEnsemble, a novel FL approach that combines principles of bagging and boosting within the FL paradigm. By partitioning a centralized dataset among clients and iteratively aggregating their best local models, FedEnsemble enhances model robustness and cross-client generalization.

Lastly, the integration of synthetic data into FL was explored through a hybrid setup, where one client used real images and another used CAD-generated synthetic images. This implementation of FedOD was conducted using YOLOv5 as the base model architecture (further implemented with YOLOv8 as well), requiring a tailored integration into the federated training loop. Despite

the inherent differences in visual fidelity, the synthetic-only client contributed meaningfully to the global model's performance. This approach demonstrated a viable pathway for engaging data-scarce or newly on-boarded clients in FL systems without incurring high annotation costs. Additionally, the same hybrid dataset was evaluated under the FedEnsemble strategy, highlighting its scalability and modularity for more complex FL configurations.

The methodology described in this chapter tries to address the core research questions posed in Section 1.2. **Question 1**, concerning the feasibility of FL in manufacturing use cases, is investigated through systematic application of FL to image classification and object detection using custom industrial datasets under non-IID conditions and limited client counts. These include USB classification, cabin quality inspection, and robust architecture design strategies. **Question 2**, which focuses on extending FL to spatially aware computer vision tasks such as object detection, is addressed through the implementation of the FedOD framework with YOLO algorithm and the evaluation of bounding box precision and generalization across varied domains. Furthermore, the challenges of small client number, dataset heterogeneity, and domain shifts are explored through architecture comparisons, cross-client testing, and integration of synthetic data.

In summary, this chapter outlined a modular and scalable FL methodology tailored to the unique constraints of industrial visual inspection. By combining principled aggregation strategies with application-specific innovations such as ensemble learning, complex non-IID setups, and synthetic data integration, the proposed framework paves the way for robust, collaborative quality control systems. This methodology also addresses the research gaps identified in Section 2.5, by delivering a unified and application-driven FL framework capable of supporting both classification and object detection under non-IID, privacy-preserving conditions typical of industrial manufacturing environments. The next chapter evaluates the effectiveness of these methods across various datasets, model architectures, and training configurations, and analyses their performance under real-world testing conditions.

4 Experimental Validation and Results

This chapter presents the experiments and results of FL use cases described in Chapter 3. Each section corresponds to a specific computer vision tasks, such as image classification, object detection, federated ensemble learning, and hybrid training with synthetic clients and evaluates the proposed FL pipelines under realistic manufacturing scenarios. All experiments follow a client-server architecture based on the FedAvg algorithm with best-weight selection, as introduced in Section 3.1. The design choices and evaluation metrics directly address the research objectives and questions posed in Section 1.2.

The first research question investigates the feasibility of deploying FL in industrial quality inspection with non-IID data, limited clients, and high-resolution inputs. This is evaluated through federated image classification tasks using both benchmark (CIFAR-10) and real-world datasets (USB and Cabin-Windshield classification). The second research question extends this investigation to more complex tasks, such as object detection, where spatial localization of features is critical. The results in Sections 4.2, 4.3, and 4.4 demonstrate the effectiveness of FL in handling bounding box variability, unseen object domains, and synthetic data integration. In most of the cases, the performance of FL model is compared against locally trained and centralized training baselines to assess generalization, robustness, and real-world deployability of the federated models.

4.1 Federated Image Classification for Quality Inspection in Manufacturing

This section evaluates the proposed federated image classification framework across three use cases: a benchmark CIFAR-10 dataset (Section 4.1.1), a USB quality inspection dataset with three clients (Section 4.1.2), and a cabin windshield classification dataset involving four clients (Section 4.1.3). The model architectures for each use case are described in detail in Section 3.2.2 of the Methodology.

To ensure a consistent evaluation, the following performance metrics are employed throughout this section. For multi-class image classification tasks, the primary metric reported is **accuracy**, which quantifies the proportion of correctly classified samples. Formally, accuracy is defined as

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(y_i = \hat{y}_i)$$

where N denotes the total number of test samples, y_i is the ground-truth label of the i^{th} sample, $\hat{y}_i$ is the predicted label, and $\mathbb{1}(\cdot)$ is the indicator function that equals 1 when the condition is true and 0 otherwise.

In addition, for live classification scenarios, the predicted **confidence score** is reported alongside the label prediction. The confidence score reflects the probability assigned by the model to its most likely class and is given by

$$\text{Confidence}(x_i) = \max_{c \in \mathcal{C}} \; p_\theta(y = c \mid x_i)$$

where x_i is the input sample, $\mathcal{C}$ denotes the set of possible classes, and $p_\theta(y = c \mid x_i)$ is the model's softmax probability (parameterized by θ) that the sample belongs to class c. A higher confidence score indicates greater certainty in the predicted label, which is particularly relevant in quality inspection tasks where misclassifications can have critical implications.

4.1.1 Benchmark Dataset: CIFAR-10

The CIFAR-10 experiment served as an initial benchmark to validate the FL framework under ideal Independent and Identically Distributed (IID) conditions. As described in Section 3.2.1, the dataset was evenly partitioned among five clients, and federated training was conducted using several popular architectures including ResNet-34, ResNet-50, VGG-11, VGG-16, and VGG-19. All models were initialized with pretrained ImageNet weights to reduce training time and improve convergence, a common strategy in industrial scenarios where resources are limited.

Table 4.1: Accuracy of federated global models on the CIFAR-10 test set after 20 communication rounds across different architectures.

Architecture	Test Accuracy (%)
ResNet-34	83.15
ResNet-50	84.22
VGG-11	86.25
VGG-16	86.42
VGG-19	**87.19**

The CIFAR-10 test data consist of 10,000 images, with each class consisting of 1000 images each. Table 4.1 summarizes the final accuracy scores of global federated models for each architecture after 20 CRs on this test data. VGG-19 achieved the highest accuracy (87.19 %), outperforming

both lighter and deeper variants of ResNet and other VGG architectures. Consequently, VGG-19 was chosen as the base model for the subsequent USB quality inspection tasks, the use case discussed in Section 4.1.2.

Table 4.2: Impact of using pretrained ImageNet weights on VGG-19 global model accuracy for CIFAR-10 classification after 20 communication rounds.

VGG-19 Configuration	Test Accuracy (%)
With Pretrained Weights	**87.19**
Without Pretrained Weights	72.15

To further assess the benefits of using pre-trained weights as starting weights, a comparison is done between FL using pre-trained VGG-19 as starting weights for all clients and FL using VGG-19 trained from scratch, i.e., without pretrained weights. As shown in Table 4.2, the model trained from scratch reached only 72.15 % accuracy after 20 CRs, which is significantly lower than the pretrained version. This clearly demonstrates the practical advantage of initializing client models with pretrained weights, which not only improves accuracy but also speeds up convergence, which is a critical factor in federated settings with limited communication budgets.

These results motivated the adoption of VGG-19 with pretrained ImageNet weights as the classification architecture for the USB quality inspection task, ensuring efficient convergence and competitive accuracy under industrial constraints.

4.1.2 Federated USB Quality Inspection between 3 Clients

This experiment evaluates the classification of USB Type-A ports across three clients, each possessing unique product geometries and fault types. All clients share a common annotation schema comprising four classes: `OKAY`, `NOT_OKAY`, `HIDDEN`, and `NOT_A_USB`, as introduced in Section 3.2.1. The results and methodology presented in this section have been previously published in an IEEE conference paper [HLR22], where a practical deployment of FL was demonstrated for the first time in the context of industrial quality inspection using real-world manufacturing use case. This also led to winning the Best paper award at the conference.

Following initial experimentation with the CIFAR-10 dataset, the VGG-19 architecture was selected for its strong performance. However, due to the large number of trainable parameters (approximately 139 million), the fully connected layer was customized to reduce computational overhead. All clients employ the same modified VGG-19 architecture, as depicted in Fig. 3.8. Training was conducted under multiple configurations of local epochs and CRs, using pretrained ImageNet weights for initialization. The best performing configuration was 15 CRs with 10 local epochs. Each client was also independently trained on its respective local dataset for a total of 100 epochs, and the best performing weights based on local validation accuracy were selected. These are referred to as the **locally trained models** throughout this thesis. To evaluate

global performance, a merged test dataset was created by aggregating the test sets from all clients. The per-class distribution of samples across clients and the global test set is presented in Table 4.3. Table 4.4 presents the number of correctly predicted samples for each class by the centralized model, global federated model, and the three local models when evaluated on the global test dataset.

Table 4.3: USB image classification: Distribution of test samples across clients and the global test dataset.

	OKAY	NOT_OKAY	HIDDEN	NOT_A_USB
Client1	296	216	257	243
Client2	445	200	250	235
Client3	270	320	81	247
Total (Global)	**1011**	**736**	**588**	**725**

Table 4.4: Number of correctly predicted samples per class by each model on the global test dataset.

Model	OKAY	NOT_OKAY	HIDDEN	NOT_A_USB	**Overall Acc. (%)**
Centralized Model	**1009**	735	**588**	725	**99.90**
Global Model	1008	**736**	586	725	99.84
Client1 Model	368	240	280	725	52.71
Client2 Model	750	245	292	725	64.77
Client3 Model	680	340	203	725	66.99

From Table 4.4, it is evident that the global federated model achieves performance almost equivalent to that of the centralized model, with both attaining approximately 99.90 % accuracy on the global test set. It is just a matter of 1 or 2 samples and hence having a difference of just 0.06 %. In contrast, while the local models perform well on their respective test datasets, they fail to generalize to unseen client's test data. Notably, Clients 2 and 3, which share similar LEGO-style USB stick geometries but differ in color and defect types, achieve better cross-client accuracy than Client 1. For instance, the Client 2 model often misclassifies unfamiliar USB variants as `OKAY`, while the Client 1 model sometimes predicts the USB sticks from Clients 2 and 3 as belonging to the `NOT_A_USB` class, as for that model it has never seen that a USB stick of that shape and size. However, all clients achieve 100 % correct classification for `NOT_A_USB` images, as these typically depict empty or irrelevant scenes.

To further assess robustness, live inference experiments were conducted using a custom interface where a camera stream was fed simultaneously to the global federated model along with all three local models. Each frame's **predicted class** and associated **confidence score** (i.e., model probability for the most likely class [Gu17]) were displayed in real time. Representative snapshots from these tests are shown in Figures 4.1 through 4.4.

In Fig. 4.1, the test object is Client 1's USB stick with ground truth `OKAY`. It is correctly classified by the global model with higher confidence than even Client 1's local model. Client 2 incorrectly classifies it as `NOT_OKAY`, having never seen this USB geometry, while Client 3 predicts it as `OKAY`, which is correct but it was seen that Client3 model tends to predict most of objects as `OKAY` with a very high confidence which can be seen in further test images as well.

Figure 4.1: Live inference on a Client1 USB stick using the global federated model and client models trained on their respective local datasets. **Ground truth: `OKAY`**.

Fig. 4.2 shows a rusted USB stick from Client 3 (`NOT_OKAY`). Both the global and Client 3 models classify it correctly with high confidence. In contrast, Client 2 classifies it as `OKAY`, as it has never seen such an error type in its own local dataset. Client 1 classifies the USB as `NOT_A_USB`, as the local dataset consists of USB stick which is very different compared to the Client 3's USB present in the test frame.

An interesting scenario is presented in Fig. 4.3, where a Client 2 USB stick is corrupted using Client 1's error type. Although this defect and USB combination was never a part of training dataset as the error were specific to each client, yet the global model correctly classifies it as `NOT_OKAY` with a high confidence of 99 %. Client 1 model also classifies it correctly as the defect is part of its own error type, but with lower confidence of 53 %. While both Client 2 and Client 3 misclassify it as `OKAY`, as the error was never seen to those models before. To evaluate generalization under domain shift, an additional test was conducted using an unseen third-party USB stick, as illustrated in

As shown in Fig. 4.4, the test sample is a third-party USB stick unseen by any client, with its port

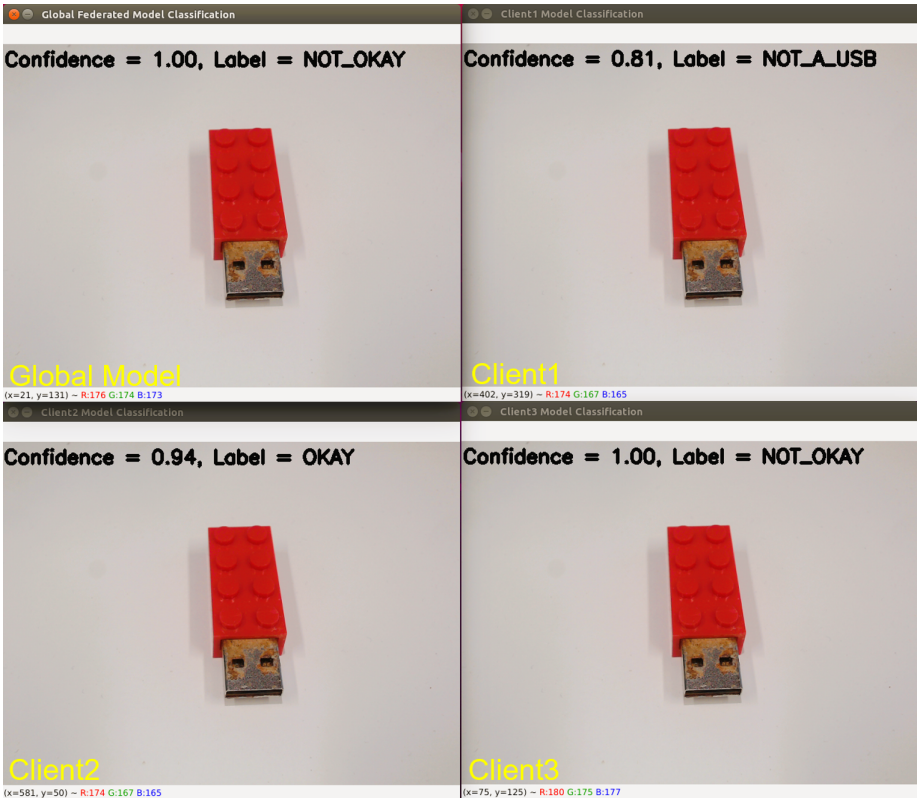

Figure 4.2: Live inference on a Client3 USB stick using the global federated model and client models trained on their respective local datasets. **Ground truth:** `NOT_OKAY`.

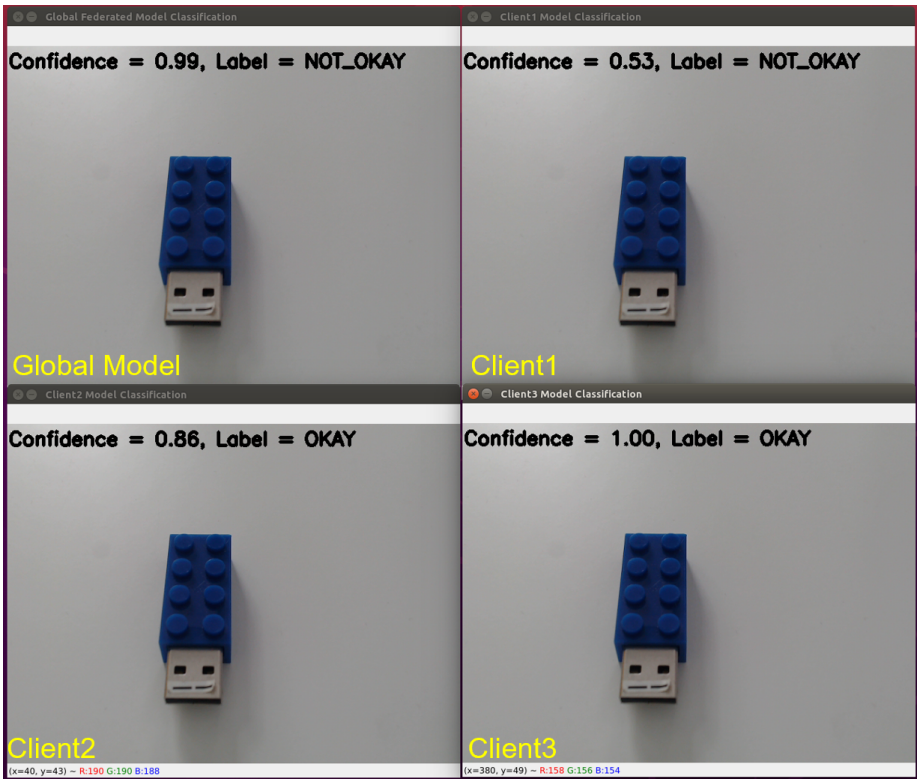

Figure 4.3: Live inference on a Client2 USB stick showing a Client1 error type using the global federated and client models. **Ground truth:** `NOT_OKAY`. Although this error type was unseen by Client2, the federated model correctly classifies the USB.

hidden by a protective cover. Although the class `HIDDEN` was present in training, only the global federated model correctly classified the sample, while all local models failed. This demonstrates the superior robustness and generalization of FL to external samples beyond the training distribution. Moreover, new clients joining the federated process at a later stage can initialize from the global weights rather than training from scratch, thereby accelerating convergence and benefiting immediately from the shared knowledge of the federation.

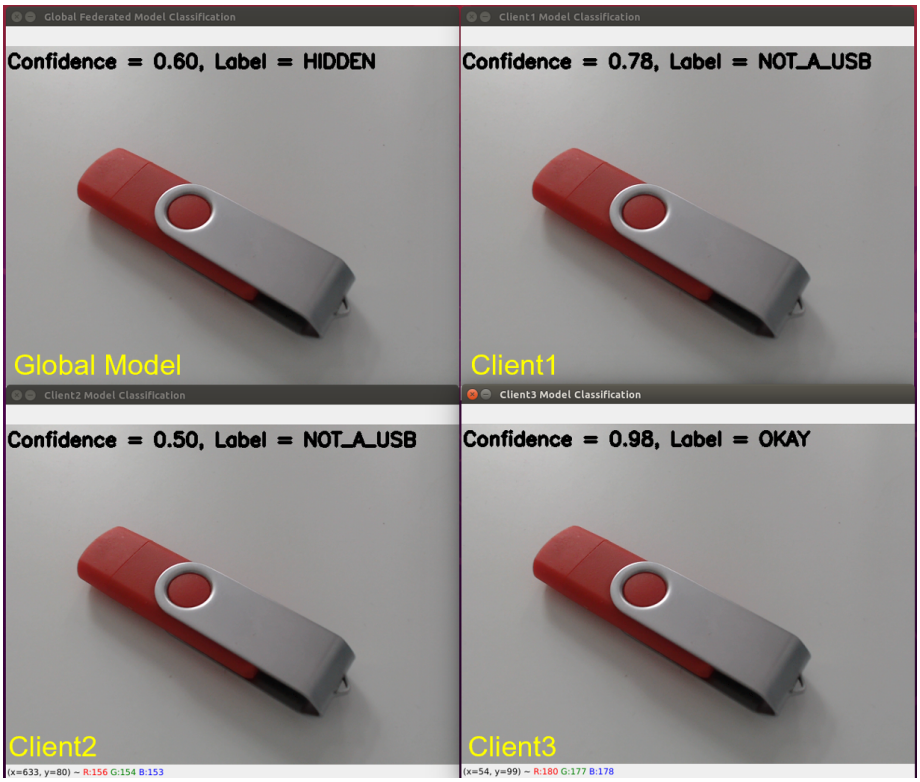

Figure 4.4: Live inference on a third-party USB stick unseen by any client, using the Global Federated, Client1, Client2, and Client3 models. Only the Federated model is able to correctly classify this new USB.

To further evaluate robustness, the federated global model was directly compared with a centralized VGG-19 model. Both models were simultaneously tested in live environments, similar to the earlier setup with client-specific models. As shown in Figures 4.5 and 4.6, both models predict the correct label, but the federated model consistently does so with higher confidence. In Fig. 4.7, a Client2 USB stick with a defect pattern originally present only in Client1 is tested. Although such a combination was never part of the training dataset, both models classify it correctly; however, the federated model achieves higher confidence, illustrating its stronger reliability under deployment conditions when confronted with unfamiliar combinations.

These results demonstrate that the global federated model not only matches the centralized model in accuracy but also surpasses the centralized model in confidence and generalization under real-world variability. This is particularly important in industrial quality inspection, where models must remain reliable across unseen lighting, background, and geometric configurations. The results firmly establish the capability of FL to deliver highly accurate, generalizable, and deployment-ready models without requiring direct data sharing across sites.

Figure 4.5: Live inference on a Client1 USB stick using the global federated model and a centralized VGG-19 model. The ground truth is `NOT_OKAY`. Both models classify correctly, but the federated model achieves higher confidence, illustrating its stronger reliability under deployment conditions.

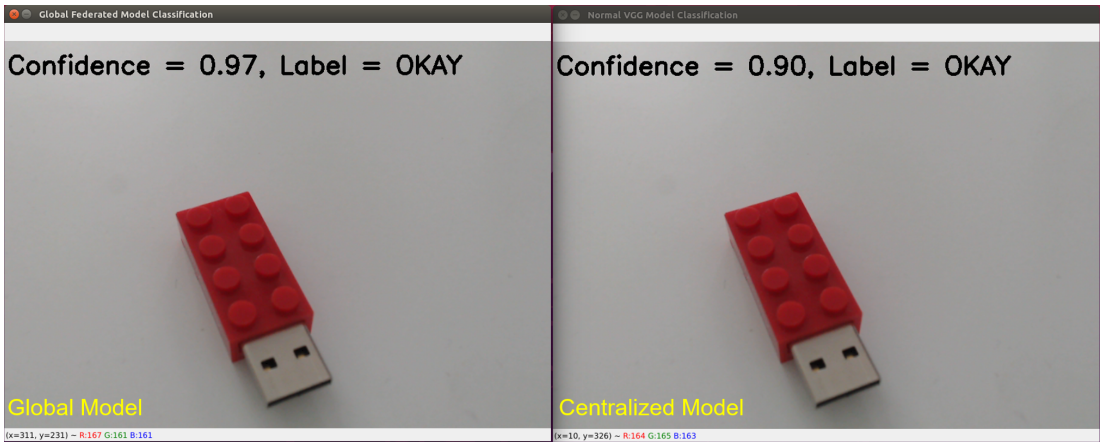

Figure 4.6: Live inference on a Client3 USB stick using the global federated model and a centralized VGG-19 model. The ground truth is `OKAY`. Both models classify correctly, but the federated model achieves higher confidence, indicating stronger reliability under deployment conditions.

Figure 4.7: Live inference on a Client2 USB stick using the global federated model and a centralized VGG-19 model. The ground truth is `NOT_OKAY`. Both models classify correctly, but the federated model achieves higher confidence, highlighting its stronger reliability under deployment conditions.

4.1.3 Federated Cabin Windshield Quality Inspection with 4 Clients

As introduced in Section 3.2.1, this experiment evaluates the performance of FL in a challenging non-IID setup involving four clients, each performing image classification of cabin models with distinct windshield types. To simulate practical industrial constraints, each client was intentionally excluded from training on one of the four windshield classes: `TypeA`, `TypeB`, `TypeC`, or `TypeD`, while all the clients had the `No_windshield` class. This omission led to a class heterogeneous distribution across the training data, closely reflecting decentralized manufacturing environments where clients may only observe partial class coverage. The experimental design and core results presented in this section have been peer-reviewed and published at an IEEE conference dedicated to FL research [HLR24].

The classification capabilities lie with the DL architecture and hence for this use case different architectures such as VGG-19, ResNet-50, DenseNet-121 and EfficientNetv2 were trained and tested. Federated training was conducted with several configurations of local epochs and CRs, with the best results observed for 15 CRs and 5 local epochs per round. For comparison, centralized models were trained using a merged dataset combining training and validation splits from all four clients. These models were trained for 100 epochs, with early stopping after 30 epochs to match the effective training time of the FL setup (15 CRs × 5 epochs). In addition, local client models were trained independently using their own data to simulate performance in the absence of federated collaboration.

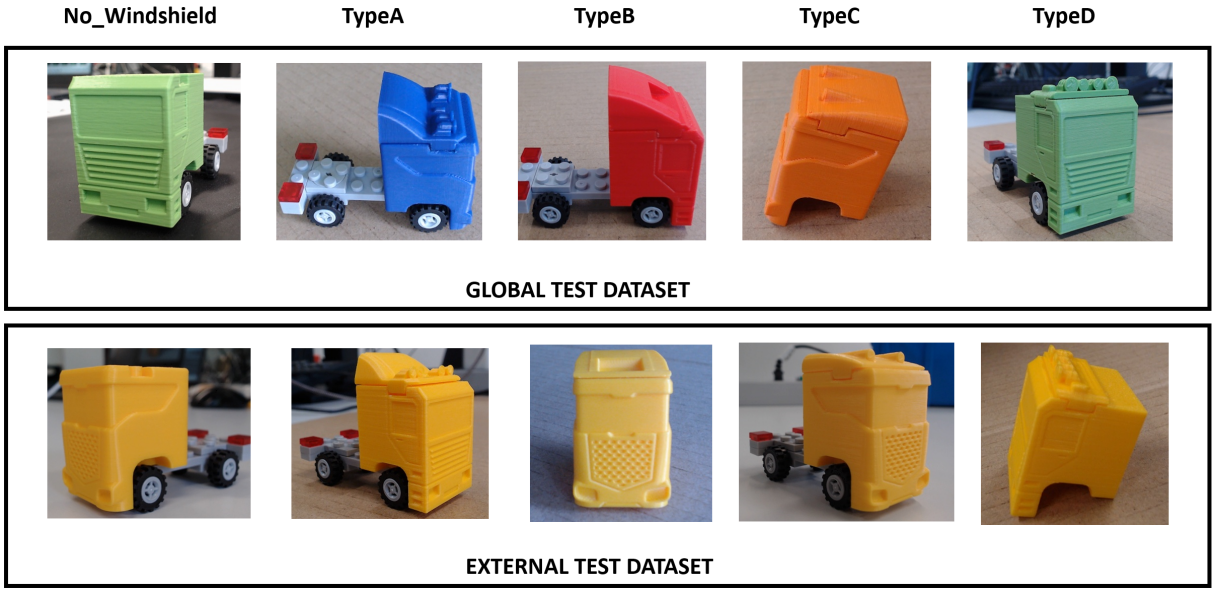

Figure 4.8: Test datasets used for cabin windshield quality inspection. The global test dataset contains samples from all windshield types, while the external test dataset was captured in a different environment to evaluate model generalization under domain shift.

To assess generalization, two separate test datasets were constructed. The first, referred to as the global test set, consists of merged test samples from all four clients, including windshield classes absent during local training. The second is an external client dataset captured under

unseen environmental conditions to evaluate cross-domain robustness. The class-wise distributions for both datasets are shown in Table 4.5, and representative samples are illustrated in Fig. 4.8.

Table 4.5: Distribution of images per class in the global test set and external client test set used for evaluating model generalization.

Dataset	No_windshield	TypeA	TypeB	TypeC	TypeD
Global Test Set	340	355	343	344	340
External Client Test Set	96	100	88	95	89

Table 4.6: Performance comparison across architectures for clients' local models, the global federated model, and the centralized model. DenseNet-121 is included as the most efficient and effective architecture based on prior analysis.

Architecture	Client	Test Acc	Precision	Recall	F1
ResNet-50	Client1	0.2892	0.2497	0.2914	0.2192
	Client2	0.2660	0.2166	0.2664	0.2227
	Client3	0.2631	0.3363	0.2644	0.1734
	Client4	0.2811	0.2681	0.2832	0.2188
	FL Model	**0.9890**	**0.9890**	**0.9890**	**0.9890**
	Centralized Model	0.4826	0.5864	0.4845	0.4524
VGG-19	Client1	0.4187	0.4746	0.4212	0.3871
	Client2	0.4686	0.3877	0.4675	0.4167
	Client3	0.5331	0.4263	0.5338	0.4730
	Client4	0.4210	0.3864	0.4242	0.3389
	FL Model	**0.9669**	**0.9671**	**0.9672**	**0.9669**
	Centralized Model	0.8095	0.8359	0.8101	0.7895
DenseNet-121	Client1	0.3101	0.2759	0.3130	0.2675
	Client2	0.3171	0.2880	0.3180	0.2594
	Client3	0.3601	0.3286	0.3578	0.3102
	Client4	0.3089	0.2745	0.3070	0.2444
	FL Model	**0.9901**	**0.9902**	**0.9903**	**0.9902**
	Centralized Model	0.6440	0.6559	0.6451	0.6461
EfficientNetv2	Client1	0.2282	0.2821	0.2302	0.2036
	Client2	0.2178	0.2379	0.2174	0.1850
	Client3	0.2346	0.1929	0.2342	0.2049
	Client4	0.2404	0.2070	0.2407	0.2107
	FL Model	**0.9866**	**0.9867**	**0.9868**	**0.9867**
	Centralized Model	0.3351	0.3432	0.3357	0.3271

Model performance across ResNet-50, VGG-19, DenseNet-121, and EfficientNetv2 was evaluated on the global test dataset using accuracy, precision, recall, and F1-score, with results summarized in Table 4.6. Across all architectures, the locally trained models exhibited weak generalization, with accuracy consistently below 54 % due to missing classes in their training data. By contrast, the FL models consistently outperformed both local and centralized baselines, achieving near-perfect performance even on classes absent from some clients. Centralized models trained on the merged dataset generalized better than local models but still lagged behind, particularly under class imbalance. Among the backbones, DenseNet-121 provided the most favorable balance of accuracy and computational efficiency.

While the global test results highlight the strength of the federated approach, it is equally important to assess performance under domain shift. For this purpose, only FL and centralized models were evaluated on the external client dataset (Table 4.7). Here too, FL consistently outperformed centralized training across all architectures. For example, federated DenseNet-121 achieved 99.0 % accuracy on the global test set versus 64.4 % for centralized, and 95.0 % versus 56.4 % on the external dataset. EfficientNet showed the most dramatic gain, improving from 27.6 % (centralized) to 97.3 % (federated). These results underscore the robustness of FL across both in-distribution and cross-domain evaluation.

Among the federated models, DenseNet-121 again emerged as the most practical backbone, reaching 95 % accuracy on the external dataset with near-perfect precision, recall, and F1 scores. This confirms its suitability for industrial FL deployments, where both accuracy and efficiency are critical. Overall, this experiment demonstrates that FL not only overcomes extreme class heterogeneity but also consistently outperforms centralized learning under domain shift, making it a highly reliable approach for real-world quality inspection tasks.

Finally, qualitative evaluation was performed on the external dataset to illustrate prediction behavior. Figures 4.9 and 4.10 show representative cases where both models predicted correctly, yet the federated model produced much higher confidence (0.95 vs. 0.28 for `TypeD`, 0.99 vs. 0.32 for `TypeB`). This pattern was consistent across the dataset: centralized predictions often remained uncertain even when correct, whereas federated predictions were both accurate and confident. These observations further emphasize the superior reliability of the proposed FL framework under domain shift, highlighting its suitability for deployment in privacy-preserving industrial inspection scenarios. Importantly, they demonstrate how FL can provide manufacturers with models that are not only accurate, but also trustworthy in critical decision-making contexts, thereby bridging the gap between controlled research settings and deployment in real production environments.

Table 4.7: Comparison of performance metrics for Centralized and Federated models on external test dataset across multiple architectures

Model	Training Approach	Accuracy	Precision	Recall	F1 Score
ResNet-50	Centralized Model	0.4214	0.5190	0.4214	0.3511
	Federated Model	**0.9457**	**0.9527**	**0.9457**	**0.9458**
VGG-19	Centralized Model	0.8557	0.8613	0.8557	0.8509
	Federated Model	**0.9329**	**0.9392**	**0.9329**	**0.9334**
DenseNet-121	Centralized Model	0.5643	0.5199	0.5643	0.5297
	Federated Model	**0.9500**	**0.9546**	**0.9500**	**0.9501**
EfficientNetv2	Centralized Model	0.2757	0.2748	0.2757	0.2301
	Federated Model	**0.9729**	**0.9733**	**0.9729**	**0.9729**

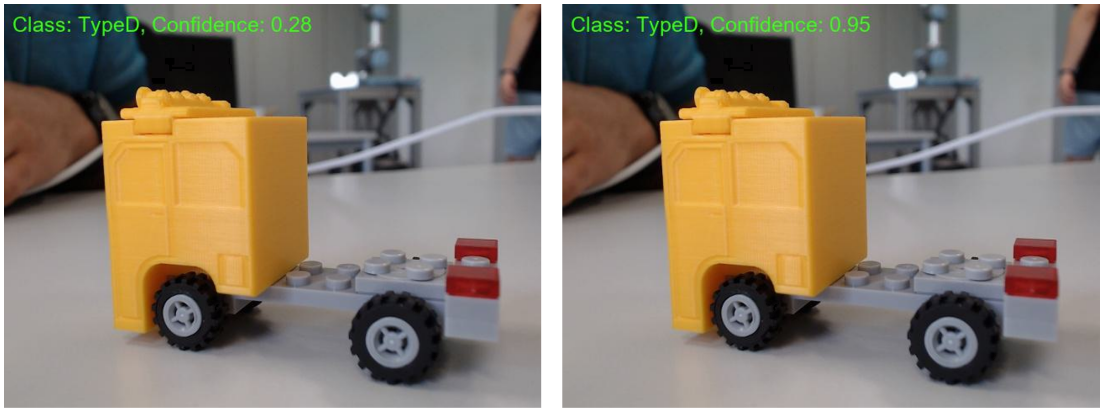

Figure 4.9: Qualitative comparison between centralized and federated DenseNet-121 models on the external test dataset. Both models correctly classify the object as `TypeD`, but the federated model achieves much higher confidence (0.95 vs. 0.28).

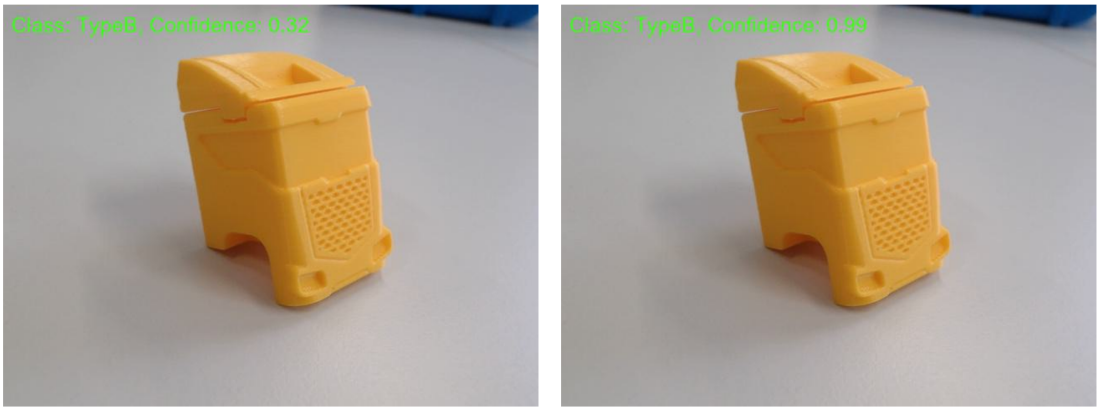

Figure 4.10: Qualitative comparison between centralized and federated DenseNet-121 models on the external test dataset. Both models correctly classify the object as `TypeB`, but the federated model achieves higher confidence (0.99 vs. 0.32).

4.1.4 Discussion

Across the three evaluated use cases, the experimental results clearly establish the viability of FL for industrial image classification tasks. Starting with the benchmark CIFAR-10 dataset, it was observed that VGG-19 consistently outperformed other architectures such as ResNet-34, ResNet-50, VGG-11, and VGG-16 when used with pretrained weights. This motivated its adoption for the USB quality inspection experiments. Furthermore, models initialized with pre-trained ImageNet weights achieved significantly higher accuracy with fewer CRs, making them particularly attractive for resource-constrained industrial deployments. These findings underscore the importance of starting with pre-trained weights in FL settings, where computational and data efficiency are critical.

In the USB classification task, the global federated model achieved performance nearly equivalent to the centralized model, achieving around 99.84 % accuracy, while providing a critical advantage in preserving data privacy. In contrast, models trained locally on individual client datasets exhibited limited generalization to unseen data from other clients. This discrepancy is attributed to the inherent differences in product geometry, defect types, and imaging conditions between clients. The global model, by aggregating knowledge across all participants, effectively captured a broader representation of the problem space. Live testing with real-time inference further validated these findings. The federated model consistently predicted the correct class with higher confidence scores, even in cases involving unknown USB sticks or defect types not present in any client's local training data. In particular, the model correctly identified classes under varying lighting conditions, partially blurred objects, and previously unseen physical defects on specific USBs. Such capabilities are vital in real-world manufacturing environments, where input conditions can change unpredictably and defects may manifest in novel forms. These results suggest that FL based models are not only accurate but also robust and deployment ready for industrial settings.

This robustness was further affirmed in the cabin windshield classification task, which introduced a more complex five class problem and extreme non-IID conditions by deliberately omitting one class from each client's training set. Despite these constraints, the federated models achieved near-perfect classification performance across all architectures. DenseNet-121 in particular delivered the highest accuracy while maintaining a lower training parameter count, making it well suited for practical deployment. Centralized models trained on the merged dataset underperformed relative to federated models, especially in generalizing to missing or underrepresented classes. Evaluation on an external test dataset containing samples collected under unseen environmental conditions such as different lighting, angles, and backgrounds, further demonstrated the generalization capability of the federated models. Even without having been trained on this data, the FL models maintained high classification accuracy and confidence, whereas centralized models exhibited reduced performance. This domain robustness is especially important for manufacturers joining the federated ecosystem at later stages or those con-

tributing novel product variants.

Importantly, FL enables these performance benefits while ensuring that no raw data is exchanged between clients. This is a crucial consideration in industrial setup where data sensitivity, intellectual property protection, and regulatory compliance (e.g., GDPR) are paramount. The federated setup ensures collaborative model improvement without compromising proprietary data. Moreover, by participating in the FL process, each client gains access to a significantly more robust and generalizable global model than what would be achievable in isolation. In summary, the results affirm that FL offers a scalable, privacy-preserving, and industrially viable alternative to centralized training for image classification tasks. The combination of high accuracy, strong cross-client generalization, and resilience to environmental variability makes FL a compelling solution for distributed quality inspection in manufacturing ecosystems.

4.2 Federated Object Detection for Quality Inspection in Manufacturing

This section presents experimental results on applying FL to object detection tasks relevant to industrial quality inspection. The evaluation builds upon the FedOD framework detailed in Section 3.3.2, utilizing the YOLOv5 architecture as the core object detector. Two representative use cases are investigated: USB Type-A port detection across three clients, and cabin with or without windshield detection involving two clients. Both scenarios simulate non-IID data partitions and domain variability, such as variations in lighting, background, and object geometry, without requiring raw data to be shared across clients. The results and contributions of these use cases were peer-reviewed and published in the inaugural IEEE FL Conference [He23], marking one of the first demonstrations of end-to-end YOLOv5 based FedOD on custom use case.

Before presenting the experimental results, the evaluation metrics used for object detection are formally introduced. These metrics are used consistently throughout the thesis in all subsequent object detection use cases. The primary metric reported is the **mean Average Precision (mAP)** at a fixed Intersection-over-Union (IoU) threshold of 0.75, denoted as mAP@0.75 (IoU is explained in details in Section 2.2, Fig. 2.4). Given a class $c \in \mathcal{C}$, the Average Precision (AP) is defined as the area under the precision–recall curve

$$AP(c) = \int_0^1 p_c(r)\, dr$$

where $p_c(r)$ is the precision as a function of recall r. The mAP is then computed as the mean over all classes

$$mAP = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} AP(c)$$

The IoU threshold determines whether a predicted bounding box is considered a correct detection. For two boxes, the predicted $\hat{B}$ and the ground truth B, the IoU is defined as

$$IoU(B, \hat{B}) = \frac{|B \cap \hat{B}|}{|B \cup \hat{B}|}$$

A detection is classified as a true positive if $IoU(B, \hat{B}) \geq \tau$, where τ is the IoU threshold. In this work, $\tau = 0.75$ is used for mAP@0.75, enforcing stricter localization accuracy in line with industrial inspection requirements.

In addition, the **COCO-style metric** AP@[.50:.05:.95] is reported [Li14]. This metric averages the AP across ten IoU thresholds ranging from 0.50 to 0.95 in steps of 0.05:

$$\mathsf{AP}_{\mathsf{COCO}} = \mathsf{AP@}[0.50 : 0.05 : 0.95] = \frac{1}{10} \sum_{i=0}^{9} \mathsf{AP}\big(0.50 + 0.05\, i\big)$$

providing a comprehensive evaluation of model performance under varying localization tolerances. Compared to a single-threshold mAP, this metric better reflects robustness across different application contexts, from coarse to precise detection. Together, mAP@0.75 and AP@[.50:.05:.95] allow for a balanced assessment of detection accuracy and generalization in federated object detection experiments.

4.2.1 USB Quality Inspection with Object Detection

The USB dataset originally developed for image classification was extended with bounding box annotations in YOLO format to support object detection, as described in Section 3.3.1. Three semantic classes:`OKAY`, `NOT_OKAY`, and `HIDDEN` were annotated across datasets collected from three clients, each with heterogeneous USB stick geometries and visual domains. Data from each client was captured under distinct conditions (e.g., lighting, background) and included client-specific defects such as stickers, physical damage, or rust. The dataset was split into 70 % training, 15 % validation, and 15 % test subsets for each client.

This work introduces the novel FedOD architecture and setup (Section 3.3.2), using YOLOv5m as the base model initialized with pretrained weights. Multiple configurations of CRs and local epochs were evaluated. The best performing setup, based on highest validation mAP, was achieved with 5 CRs and 15 local epochs. Local models were also trained for 100 epochs, and the best performing weights on the validation set were selected for testing. The test set distribution across the three clients is summarized in Table 4.8. Each client represents a distinct domain, contributing to a diverse and challenging evaluation scenario. The global test set merges samples from all clients to evaluate the generalization capability of both federated and local

Table 4.8: Class-wise sample distribution for USB object detection across three clients and the aggregated global test set.

	OKAY	NOT_OKAY	HIDDEN
Client 1	327	236	290
Client 2	474	236	262
Client 3	297	250	93
Total (Global Test)	**1098**	**722**	**645**

models.

Before presenting the evaluation results, the object detection metrics used are briefly clarified. The primary metric reported is the mAP at an IoU threshold of 0.75 (mAP@0.75), which evaluates model performance when the predicted bounding boxes overlap with the ground truth by at least 75 %. This stricter threshold emphasizes precise localization. Additionally, AP@[.50:.05:.95] is reported as a COCO-style metric that averages the Average Precision (AP) across ten IoU thresholds from 0.50 to 0.95 in increments of 0.05, offering a more comprehensive assessment of detection performance under varying levels of localization tolerance.

The performance of each model is summarized in Table 4.9. The global federated model significantly outperforms all local models, achieving a perfect mAP@0.75 score of 1.00 and a high COCO-style AP@[.50:.05:.95] of 0.94. These results indicate that the federated model not only learns class discrimination but also achieves highly accurate spatial localization across all domains. In contrast, local models exhibit overfitting to their respective training domains and struggle to generalize across unseen USB geometries and defect types. All client models perform poorly on the global test set, particularly in the detection of defects from other domains.

Table 4.9: Performance of local and federated models on the global USB test dataset.

Model	**mAP@0.75**	**AP@[.50:.05:.95]**
Client 1	0.70	0.62
Client 2	0.73	0.68
Client 3	0.69	0.65
Federated Global Model	**1.00**	**0.94**

To further validate these findings, a live inference setup was developed to evaluate all four models simultaneously on an unseen frame containing mixed USB objects. The test frame included the following: a Huawei USB device from Client 1 with rust (a defect seen only in Client 3), intact LEGO-style USB sticks from Clients 2 and 3, a damaged blue USB stick from Client 2, and a rusted red USB stick from Client 3. A confidence threshold of 0.2 was applied during inference to even show false positives detections, i.e., any predicted bounding box with a confidence score below

0.2 was excluded from the output.

Figure 4.11 presents the results. Visually one can also spot the difference of different color bounding boxes assigned to different labels by the YOLO renderer. The top-left quadrant shows the output from the global federated model, followed clockwise by Client 1, Client 3, and Client 2 models. The federated model successfully detects and classifies all objects with high confidence, producing accurate bounding boxes, even for the rusted Client 1 USB, which is not a part of any local training data. In contrast, Client 1's own model misclassifies the rusted USB as `OKAY`. The Client 2 model correctly identifies its own damaged blue USB but misclassifies the red LEGO stick as `NOT_OKAY`, and Client 3's model only identifies its own USBs correctly, failing to detect the foreign geometries. These discrepancies highlight the brittleness of models trained in isolation and emphasize the generalization benefit of federated training across heterogeneous domains.

Figure 4.11: Live inference comparison of the global federated model (top left) against local client models: Client1 (top right), Client2 (bottom left), and Client3 (bottom right). The federated model successfully detects and classifies all USBs, including previously unseen variants, whereas local models show misclassifications and limited generalization. Confidence threshold = 0.20.

This experiment demonstrates that the proposed FedOD framework achieves robust object detection performance even in the presence of domain shifts, varying lighting conditions, and heterogeneous object geometries. The federated global model not only matches but surpasses

local models in terms of detection accuracy and bounding box precision, confirming its suitability for industrial deployment in quality inspection pipelines.

4.2.2 Cabin Quality Inspection Using Federated Object Detection

Following the initial USB use case, the second experiment evaluates the proposed FedOD architecture on a more challenging task involving detection of cabins, whether they are with or without a windshield. As detailed in Section 3.3.1, this dataset comprises images of custom 3D-printed cabins developed at SmartFactory-KL. Two clients contributed to the dataset: Client 1 provided blue cabins with windshields of Type A and B, while Client 2 supplied red cabins with windshields of Type C and D. Each image was annotated for binary detection, i.e., either `Cabin_with_windshield` or `Cabin_without_windshield`.

For evaluation, the global and local models were tested on windshield types absent from their respective training datasets, i.e., Client 1 was tested on Type C and D windshields and Client 2 on Type A and B. The objective was to assess both detection accuracy and the quality of bounding box localization, ensuring precise object coverage and not just overlapping over the object.

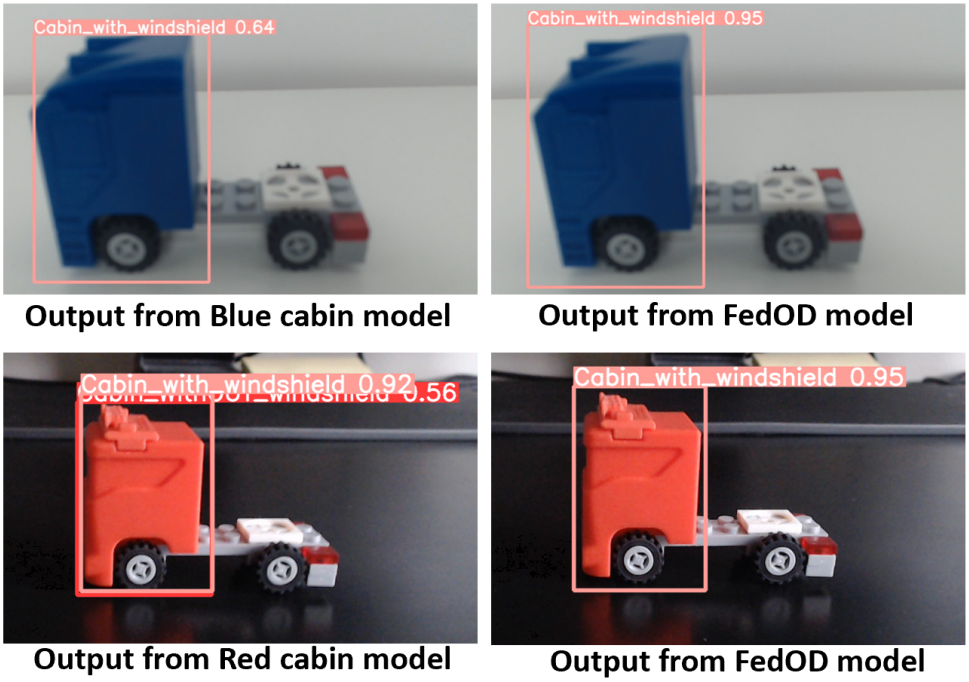

Figure 4.12: Bounding box predictions from Client1, Client2, and the global FedOD model on unseen windshield types. Client1 was tested with Type D and Client2 with Type B, both unseen during local training, demonstrating the superior generalization of the federated model.

Multiple FL training configurations were tested, with the best performance obtained at 10 CRs and 15 local epochs. The test dataset was constructed by combining unseen windshields: Type C and D from Client 1 and Type A and B from Client 2, thereby simulating a realistic scenario where manufacturers aim to expand product lines without retraining models or sharing raw data from their quality inspection modules. This demonstrates the relevance of FL in enabling

high-performance model training under strict data privacy constraints.

A quantitative comparison is provided in Table 4.10, where each client's local model was evaluated on its own unseen windshield types. The Client 1 model achieved a mAP@0.75 of 0.83 and an AP@[.50:.05:.95] of 0.70 when tested on Type C and Type D windshields, while the Client 2 model performed slightly better, reaching 0.97 mAP and 0.83 AP when tested on Type A and Type B windshields. In both cases, however, the global FedOD model surpassed the client-specific models, achieving perfect mAP (1.00) and over 90 % AP across the unseen windshield types. This demonstrates that the federated model not only matches but improves upon the performance of individual client models, even when evaluated on previously unseen combinations of their own cabin types.

Table 4.10: Performance of local and federated models on unseen windshield combinations

Model	**Training Dataset**	**Test Dataset**	**mAP @0.75**	**AP [.50:.05:.95]**
Client 1 (Blue cabin)	Blue cabins with Windshield Types A and B	Blue cabins with Windshield Types C and D	0.83	0.70
Global Federated Model	—	Blue cabins with Windshield Types C and D	**1.00**	**0.96**
Client 2 (Red cabin)	Red cabins with Windshield Types C and D	Red cabins with Windshield Types A and B	0.97	0.83
Global Federated Model	—	Red cabins with Windshield Types A and B	**1.00**	**0.91**

To further stress generalization, both client models and the global model were tested on a merged global test dataset containing unseen windshield classes from both the blue and red cabins. The results, summarized in Table 4.11, reveal a sharp decline in performance for the local models. Client 1 and Client 2 both fell below 50 % AP, reflecting their inability to detect windshield types belonging to the other client's domain. In other words, the blue cabin model failed to recognize red cabin windshields, and vice versa. This limitation highlights the fundamental weakness of isolated local training under non-IID data splits, where each client has access to only a subset of object categories. In contrast, the global federated model maintained high performance, achieving perfect mAP@0.75 and 0.93 AP on the merged dataset. These results clearly demonstrate that federated training enables cross-client knowledge transfer, al-

lowing the global model to generalize to all windshield types despite no single client having seen the full class distribution during training. The accuracy drop observed in the local models thus reinforces the advantage of FL in handling class-heterogeneous industrial datasets, where collaboration between clients is essential for reliable deployment.

Table 4.11: Comparison of mAP and AP metrics for Client 1, Client 2, and Global FedOD model on merged global test dataset

Model	**Training Dataset**	**Test Dataset**	**mAP @0.75**	**AP [.50:.05:.95]**
Client 1 (Blue cabin)	Blue cabins without windshield and with windshield types A and B	Blue cabins with windshield type C and D and Red cabins with windshield type A and B (unseen during training)	0.42	0.35
Client 2 (Red cabin)	Red cabins without windshield and with windshield types C and D		0.49	0.42
Global Federated Model	—		**1.00**	**0.93**

To assess real-time performance, a live inference setup was configured in which all three models, i.e., Client 1, Client 2, and the global FedOD model, received identical input from a shared camera stream. This setup ensured a fair and synchronized comparison of predictions across models under the same visual conditions, closely mimicking real deployment scenarios in industrial inspection where multiple models may be applied to a common production line feed. To avoid clutter and confusion in the visual overlays, the class labels were numerically encoded as '0' (`Cabin_with_windshield`) and '1' (`Cabin_without_windshield`). For consistency in visualization, **red bounding boxes were used to denote class '0'** and **pink bounding boxes to denote class '1'**, with the accompanying numbers representing the predicted confidence scores of each detection. A confidence threshold of 0.20 was applied, lower than typical operational values, to ensure that not only high-confidence detections but also weaker predictions were visualized. This setting enabled the comparison of correct classifications and potential false positives across models, highlighting differences in uncertainty between local and federated approaches. In doing so, the live inference experiment provided a clear qualitative view of how well the federated model generalized in comparison to client-specific models, both in detection accuracy and bounding box precision.

Figure 4.13 presents an inference case containing both blue and red cabins placed side by side, with instances of both `Cabin_with_windshield` and `Cabin_without_windshield` visible in

a single frame. The global FedOD model (top-left) correctly detects all four cabin instances, assigning the right class labels and drawing precise bounding boxes that fully capture the cabins and their windshields. In contrast, the local models fail to generalize beyond their own training distributions. The Client 1 model (top-right), trained only on blue cabins, detects both blue

Figure 4.13: Live inference on mixed cabin types using Client 1, Client 2, and Global FedOD models (Confidence threshold = 0.20). The federated model detects both cabins with precise bounding boxes and high confidence, while local models detect only their respective cabin types and fail on unseen geometries.

cabins correctly but completely fails to recognize the red ones. Conversely, the Client 2 model (bottom-left), trained only on red cabins, detects both red cabins with accurate bounding boxes but entirely misses the blue ones. These observations clearly illustrate the overfitting of local models to their own client-specific domains, whereas the federated model learns a shared representation that generalizes across both cabin types.

Figure 4.14 evaluates robustness under a more challenging scenario, where cabins are presented with **swapped windshield colors** (e.g., a blue cabin fitted with a red windshield and vice versa). The federated model (top-left) again achieves reliable detection, correctly identifying both cabins with high confidence (0.89–0.95) and well-localized bounding boxes. By contrast, Client 1 (top-right) detects the blue cabin but fails to localize the red windshield, producing an

incomplete bounding box and missing the red cabin entirely. Client 2 (bottom-left) correctly recognizes the red cabin but mislocalizes the bounding box around the blue cabin. Once again, the local models fail on unfamiliar combinations, while the federated model maintains accurate classification and localization even under domain variation.

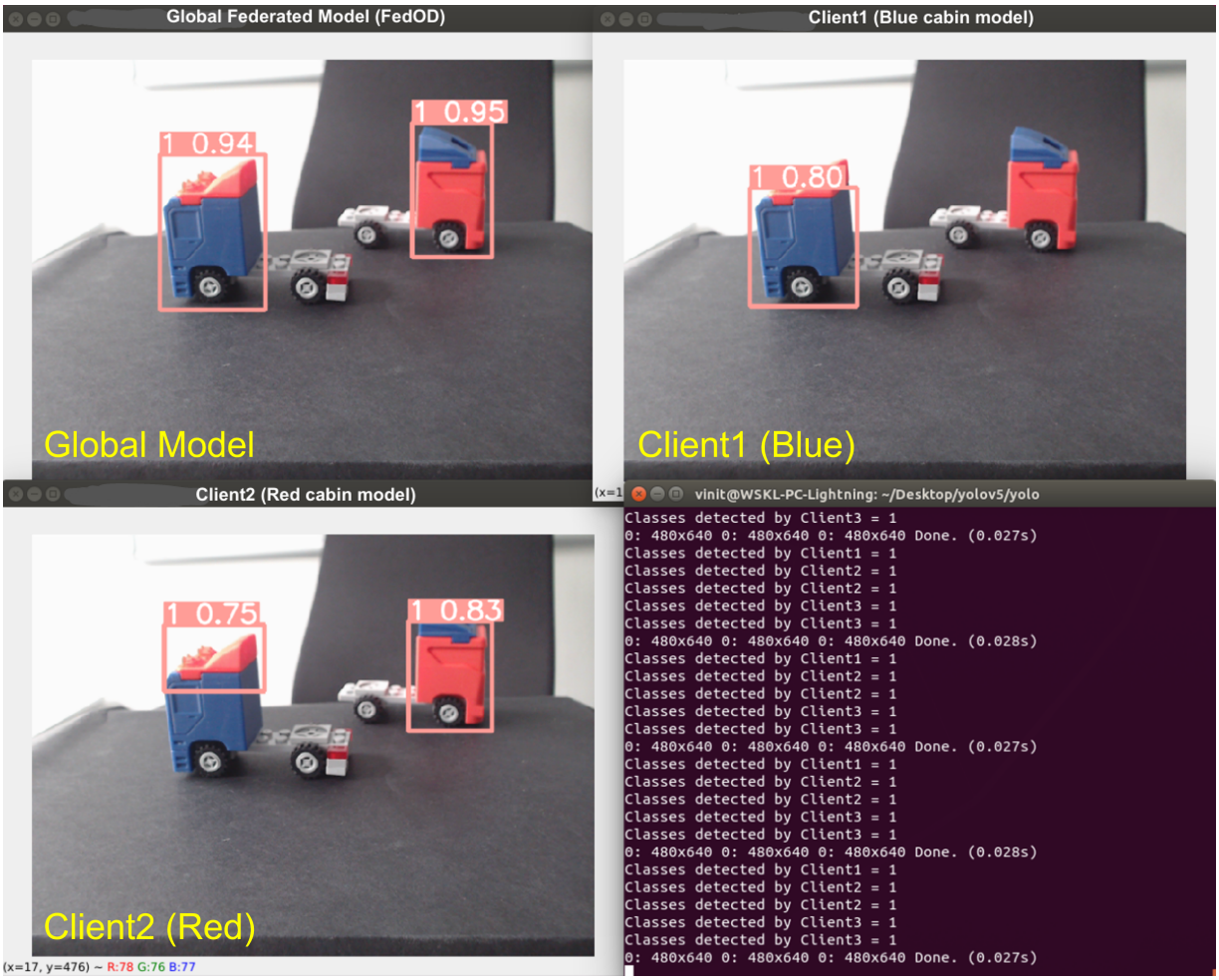

Figure 4.14: Live inference on cabins with swapped windshield colors using Client 1, Client 2, and Global FedOD models (Confidence threshold = 0.20). The federated model generalizes correctly across color-swapped cabins, while local models either miss detections or produce incomplete bounding boxes.

Finally, all models were evaluated on real-world images from the SmartFactory-KL demonstrator's quality inspection module. This setup introduced significant domain shifts, including diverse lighting conditions, reflections, printed company logos, and structural background noise, none of which were present in the training data. To further test robustness, trailer objects—also part of the demonstrator ecosystem but outside the target classes—were included to examine whether the models produce false detections on non-relevant components. As shown in Fig. 4.15, the Client 1 (Blue Cabin) model demonstrates strong overfitting to its training distribution. It detects only blue cabins with windshields and even misclassifies instances of `Cabin_without_windshield` as `Cabin_with_windshield` (pink bounding box). Red cabins are missed entirely, and the model generates multiple false positives by assigning bounding

boxes to logos, lighting grids, and background textures. False activations are also observed on the blue and white trailers, which were never part of its training data.

A similar pattern is observed in Fig. 4.16 for the Client 2 (Red Cabin) model. This model not only fails to detect blue cabins but also generates false positives on its own red cabins, misclassifying or incompletely localizing them. In one case, a blue cabin with a red windshield was incorrectly localized, with the model drawing a bounding box only around the windshield and labeling it as `Cabin_with_windshield`. While it avoids false detections on the trailers, it consistently produces spurious bounding boxes over the lighting grid. These outcomes underline the brittleness of local models: their predictions collapse when confronted with unseen geometries, colors, or distractors, reflecting their strong bias toward narrowly defined training distributions.

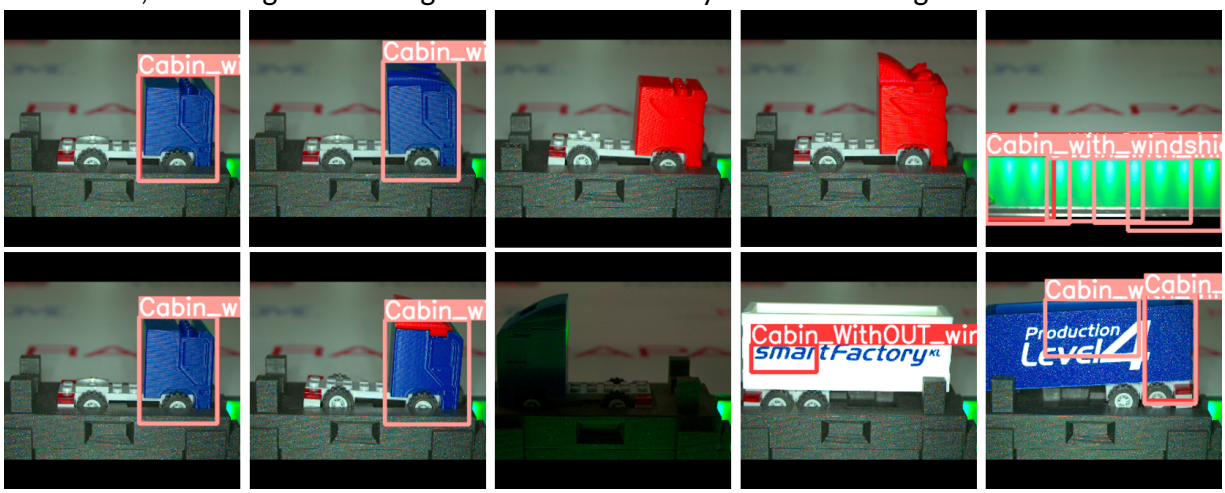

Figure 4.15: Client 1 (Blue Cabin) model inference on SmartFactory-KL demonstrator images (confidence threshold = 0.20). The model fails to detect red cabins and generates false positives on logos and background patterns.

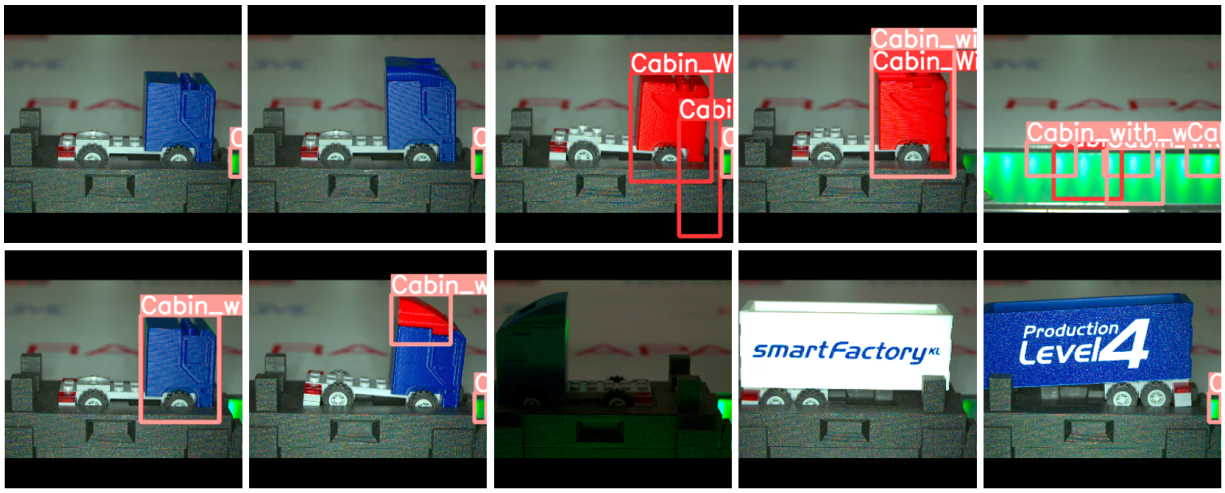

Figure 4.16: Client 2 (Red Cabin) model inference on SmartFactory-KL demonstrator images (confidence threshold = 0.20). The model detects only red cabins and produces spurious detections on irrelevant regions and objects.

In contrast, the global FedOD model (Fig. 4.17) demonstrates strong generalization across both cabin types. It accurately detects red and blue cabins with and without windshields, drawing

precise bounding boxes that align well with the true object geometry. Crucially, it suppresses false activations on irrelevant background elements such as trailers, company logos, and the lighting grid, which had triggered spurious predictions in the local models. This outcome highlights the practical strength of FL, that even without direct exposure to data from the demonstrator environment, the global federated model achieves reliable detection performance under challenging deployment conditions. Compared to the inconsistent and error-prone predictions of the local models, the FedOD approach delivers accurate, robust, and privacy-preserving inference suitable for real-world industrial inspection. The global model was further deployed within the SmartFactory-KL quality inspection demonstrator, validating its applicability in an operational manufacturing setting.

Figure 4.17: Global FedOD model inference on SmartFactory-KL demonstrator images (confidence threshold = 0.20). Unlike the local models, it detects both cabin types accurately while avoiding false positives.

4.2.3 Discussion

The object detection experiments conducted on the USB and Cabin datasets collectively highlight the practical effectiveness of the proposed FedOD framework for collaborative quality inspection in manufacturing environments. Across both use cases, the federated global model consistently outperformed individual client models in terms of both detection accuracy and localization robustness, even when evaluated on entirely unseen object geometries, lighting conditions, and background domains.

In the USB scenario, the federated model demonstrated strong generalization across highly heterogeneous object shapes and defect modalities, while the local models struggled when exposed to unfamiliar domains. Despite achieving reasonable performance within their own data distributions, client-specific models failed to correctly detect foreign geometries or defect types, confirming their tendency to overfit. The global model, in contrast, synthesized knowledge across all clients and correctly detected all test objects, including those belonging to classes never observed by individual clients.

The cabin use case introduced even greater variation due to colored backgrounds, geometric nuances, and domain-specific windshields. Once again, the global FedOD model proved more resilient, accurately detecting both known and previously unseen cabin types. It exhibited not only strong classification performance, as reflected in high mAP scores, but also precise bounding box localization; an essential requirement for visual quality inspection where incorrect spatial coverage can lead to missed or falsely identified defects.

Live inference tests reinforced these findings, showcasing the superior generalization of the global model under real-world conditions. When deployed in a shared camera stream setting, the local models showed a clear preference for their training distributions, failing to recognize unfamiliar patterns or producing false positives. The federated model maintained high performance even on complex test scenarios, including objects with swapped windshields and hybrid characteristics. Notably, it remained robust when deployed on the physical quality inspection module at SmartFactory-KL demonstrator, which presented lighting noise, distractors, and domain-specific artifacts not present in any training data. This successful transfer validates the federated model's adaptability for real industrial use.

Overall, these results confirm the advantages of FL for object detection tasks in collaborative manufacturing setups. The federated model generalized better across domains, learned richer feature representations through aggregated supervision, and preserved spatial precision necessary for detecting manufacturing defects. All of this was achieved without any need to share raw training data, making the framework especially relevant in industry settings where data sensitivity and regulatory constraints prohibit centralized training. Furthermore, the modular nature of the FedOD framework makes it scalable and deployable across distributed production lines, enabling collaborative model development without compromising data ownership or privacy. This demonstrates the potential of FL to serve as a robust backbone for future vision-based quality inspection systems across decentralized industrial ecosystems.

4.3 Federated Ensemble Learning in Object Detection

Following the development of the FedOD framework for cabin object detection, a baseline centralized YOLOv5 model was first trained to detect trailers within the SmartFactory-KL quality inspection module. The training dataset for this model was constructed under controlled laboratory conditions, where trailers were imaged against plain, uncluttered backgrounds to ensure clean annotations, as illustrated in Fig. 3.16. While this controlled setup simplified the training process, it also introduced a strong bias: the model was exposed only to idealized conditions and lacked robustness to real-world deployment scenarios. When evaluated in the live SmartFactory-KL test bed, which is characterized by dynamic lighting variations, reflections, textured floors, complex structural backgrounds, and environmental noise, the centralized model failed to generalize. Specifically, it exhibited poor localization accuracy, frequent false positives,

and missed detections, underscoring the challenge of domain shift between controlled training data and realistic deployment environments.

To address these limitations, the proposed FedEnsemble framework was applied to the trailer detection task. Unlike conventional centralized training, FedEnsemble partitions the same centralized dataset into multiple artificial clients and performs federated training using FedAvg aggregation. Although privacy is not a constraint in this setup, such partitioning introduces client diversity and implicit regularization effects that improve robustness under domain shift. In addition to trailer detection, the same methodology was validated on a reorganized version of the cabin dataset, where client partitions were created to mimic heterogeneous data availability across sites, as illustrated in Fig. 3.17. Both use cases were formally peer-reviewed and presented at the IEEE Federated Learning Conference [HLR23], marking one of the first demonstrations of FedEnsemble for enhancing object detection generalization in industrial inspection tasks.

4.3.1 FedEnsemble Trailer Detection

The trailer detection dataset consisted of three object categories: `trailer_body_blue` (**class '0'**), `trailer_body_white` (**class '1'**), and `trailer_body_white_penholder` (**class '2'**). These samples were randomly partitioned across three clients to simulate distributed training. Each client independently trained a YOLOv5m model, and their updates were aggregated into a global model using the FedAvg algorithm after each communication round. The most effective configuration was achieved with 10 local training epochs per round and 5 CRs, balancing local adaptation with global synchronization. For baseline comparison, a centralized YOLOv5 model was trained for 100 epochs with early stopping set to 30 epochs, ensuring similar effective training time. In both setups, the best-performing weights based on validation accuracy were retained for final evaluation.

Table 4.12: Performance comparison of centralized and federated models on the trailer test dataset.

Model	**mAP@0.75**	**AP@[.50:.05:.95]**
Global Federated Model	1.00	0.94
Centralized Model	**1.00**	**0.98**

As shown in Table 4.12, both the centralized and global federated models reached a perfect mAP@0.75 of 1.00, confirming that both approaches successfully learned precise bounding box localization under the controlled test distribution. However, the centralized model reported a slightly higher COCO-style AP@[.50:.05:.95] score of 0.98 compared to 0.94 for the federated counterpart. This marginal advantage is explained by the fact that both models were evaluated on test data drawn from the same distribution as the training set. Consequently, this evaluation primarily reflects in-distribution performance, without providing a reliable measure of robust-

ness to domain shift. The limitations of this controlled evaluation motivated further testing under live deployment conditions, where the benefits of the proposed FedEnsemble approach became more evident.

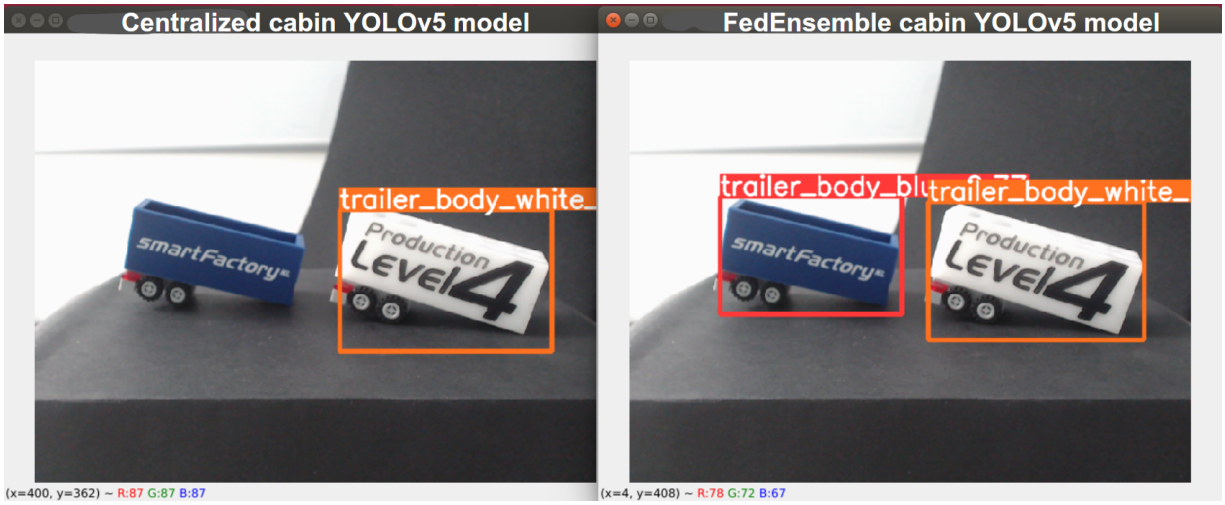

Figure 4.18: Live inference comparison between centralized YOLOv5 (left) and FedEnsemble YOLOv5 (right) on trailer detection (Confidence Threshold = 0.20). The centralized model fails to detect class 1 and produces an imprecise bounding box for class 2, whereas the FedEnsemble model correctly identifies and localizes all trailer types with precise bounding boxes.

A live inference experiment was carried out to emulate deployment-level complexity, including multiple trailer instances within a single frame, orientation variations introduced by chassis attachments, and environmental noise from lighting and background clutter. Figure 4.18 compares the detection outputs of the centralized and FedEnsemble models. The centralized YOLOv5 model exhibited clear shortcomings such as, it completely missed the trailer of class 1 and generated an imprecise bounding box for class 2, truncating significant portions of the object. In contrast, the FedEnsemble model successfully detected and classified all trailer types present in the scene, with bounding boxes tightly aligned to the actual object boundaries. For clarity, bounding boxes are color-coded by class: **red for class 0**, **pink for class 1**, and **orange for class 2**. These qualitative results demonstrate that while the centralized model overfits to controlled training conditions, the FedEnsemble approach provides superior robustness and localization accuracy under realistic deployment scenarios.

The models were further evaluated on real-world images from the SmartFactory-KL demonstrator's quality inspection module, which introduced additional challenges such as diverse lighting conditions, reflections, printed company logos, trailers with varied color schemes, and background noise from the light grid. Importantly, no images from this environment were included in the training data, making this a strict test of generalization. As shown in Fig. 4.19, the centralized YOLOv5 model was able to detect blue and white trailers but produced imprecise bounding boxes, often covering irrelevant regions such as the chassis or cutting off parts of the trailer. Moreover, it generated false positives on the background light grid, inaccurately classifying it as a trailer. In contrast, the FedEnsemble model (Fig. 4.20) demonstrated strong robustness, it suc-

cessfully detected all trailer types with accurate bounding boxes tightly enclosing the objects, while suppressing false activations on unrelated background structures. These results highlight the superior reliability of FedEnsemble under deployment-level conditions, confirming that partitioning a centralized dataset across clients and applying federated training yields models with stronger generalization than conventional centralized training.

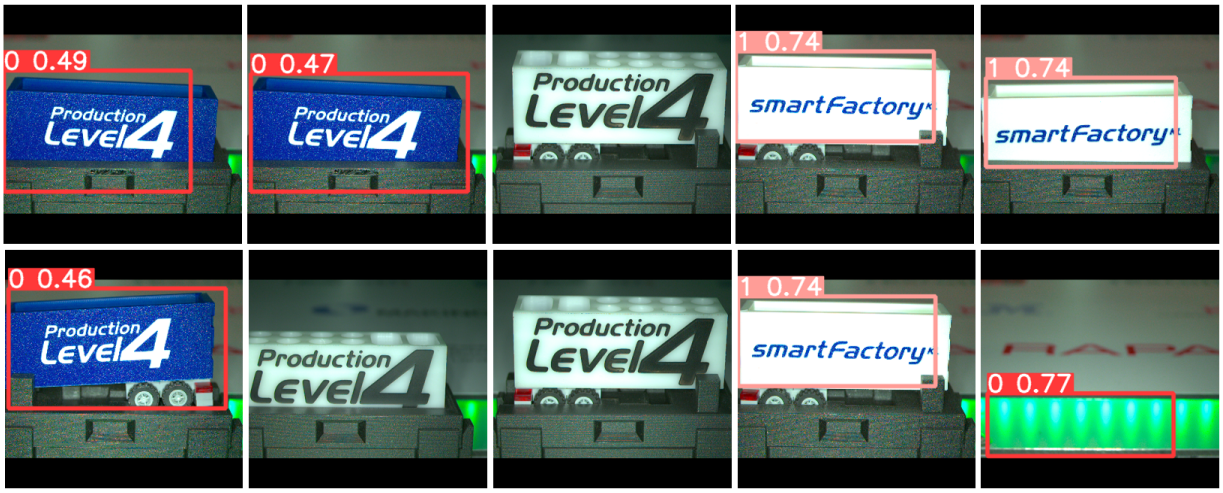

Figure 4.19: Centralized YOLOv5 model inference on SmartFactory-KL demonstrator images. While trailers are detected, bounding boxes are imprecise and the light grid is misclassified as a trailer (Confidence Threshold = 0.20).

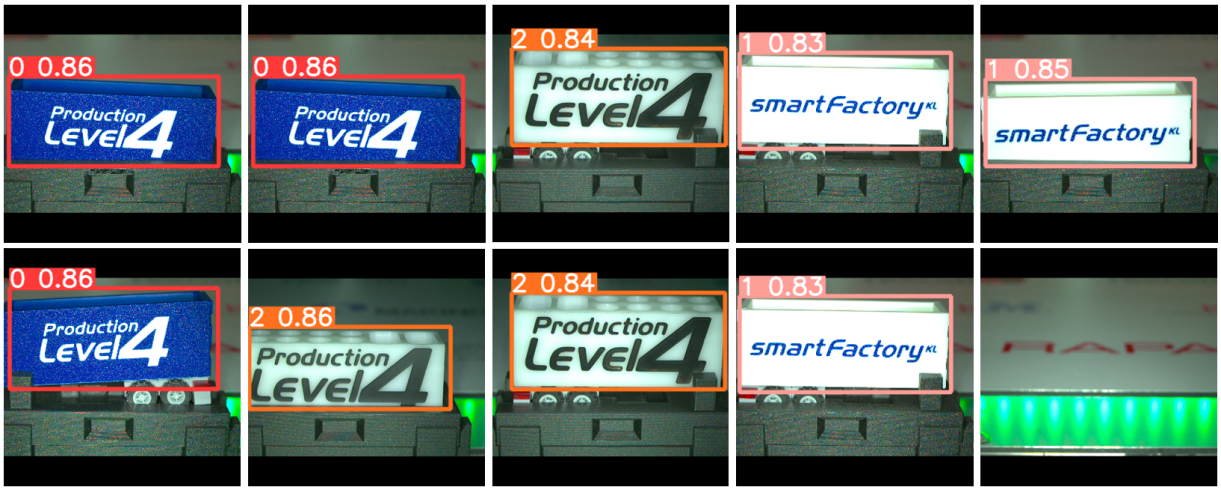

Figure 4.20: FedEnsemble YOLOv5 model inference on SmartFactory-KL demonstrator images. All trailers are detected with accurate bounding boxes and no false positives on background structures (Confidence Threshold = 0.20).

4.3.2 FedEnsemble Cabin Detection

A second experiment extended the FedEnsemble strategy to the cabin detection dataset. The dataset was shuffled and randomly partitioned into three client subsets, ensuring balanced yet distinct distributions across clients. Each client trained a local YOLOv5m model, and the global model was aggregated using FedAvg. The best configuration used 5 CRs and 15 local epochs per client, achieving a stable trade-off between convergence and communication efficiency. For

benchmarking, a centralized YOLOv5 model was trained for 150 epochs with early stopping at 30 epochs, and the best weights based on validation accuracy were selected for evaluation.

Table 4.13 summarizes the results of both models on a test set drawn from the same distribution as the training data. Similar to the trailer detection experiment, both the centralized and federated models achieved a perfect **mAP@0.75 of 1.00**, confirming that the task can be solved effectively under in-distribution conditions. However, the centralized model attained a slightly higher COCO-style AP@[.50:.05:.95] score (0.99 versus 0.95), indicating marginally better localization precision. This pattern mirrors the trend observed in the trailer use case, where centralized training showed a slight advantage under a similar test dataset, while FedEnsemble proved more resilient in real-world deployments involving domain shift and unseen environments. For this further deployment test were conducted on the SmartFactory-KL demonstrator.

Table 4.13: Comparison of global federated and centralized models on the cabin test dataset.

Model	mAP@0.75	AP@[.50:.05:.95]
Global Federated Model	1.00	0.95
Centralized Model	1.00	**0.99**

Similar to the trailer evaluation, Fig. 4.21 illustrates a challenging test scenario with two cabin objects presented without chassis, oriented in a novel configuration not seen during training, and with swapped windshield colors. Under these conditions, the centralized YOLOv5 model is able to correctly classify the objects as cabins with windshields, but the bounding boxes are poorly localized, either truncated or extending beyond the true object boundaries. In contrast, the FedEnsemble model not only classifies both cabins correctly but also produces precise bounding boxes that closely match the object contours. This outcome highlights the superior robustness of the FedEnsemble approach, which generalizes effectively to variations in geometry, orientation, and color combinations that were absent from the training distribution.

Figure 4.21: Live inference comparison on cabins with swapped windshield colors and novel orientations. The centralized YOLOv5 model (left) produces imprecise bounding boxes, whereas the FedEnsemble model (right) correctly classifies both cabins (Confidence Threshold = 0.20).

Finally, both models were evaluated on real-world images captured from the SmartFactory-KL quality inspection demonstrator. These test scenes were deliberately chosen to match those used earlier in the FedOD evaluation (Fig. 4.17), ensuring a fair basis for comparison between them. The test set included diverse challenges such as varying lighting conditions, background clutter, printed logos, trailers (which are part of the entire truck), and even configurations unseen during training (e.g., blue cabins with red windshields). As shown in Fig. 4.22, the centralized YOLOv5 model successfully detects cabin types present in its training distribution but fails to generalize to unique configurations. Specifically, it misses the blue cabin with a red windshield and frequently produces false positives by misclassifying parts of trailers as cabins. Although it does not generate false detections on empty background frames, its predictions are inconsistent and prone to error when the deployment environment differs from training conditions.

In contrast, the FedEnsemble model (Fig. 4.23) demonstrates robust performance across all test scenarios. It correctly detects all cabin types, including those with unseen color combinations, and produces precise bounding boxes that tightly align with the objects. Notably, it avoids false activations on trailers, logos, or empty backgrounds, showcasing its robustness against false positive detections. Even under low-light conditions, where only the cabin silhouette is visible, the FedEnsemble model correctly identifies and localizes the object (which was not possible even by the FL model as shown in Fig. 4.17). These results confirm that, despite being trained on the same limited dataset as the centralized baseline, the federated ensemble approach achieves stronger robustness and superior real-world generalization. Furthermore, these results reinforce that ensemble aggregation within the federated framework leverages the complementary strengths of local models, enabling more consistent and confident detections across varying visual domains.

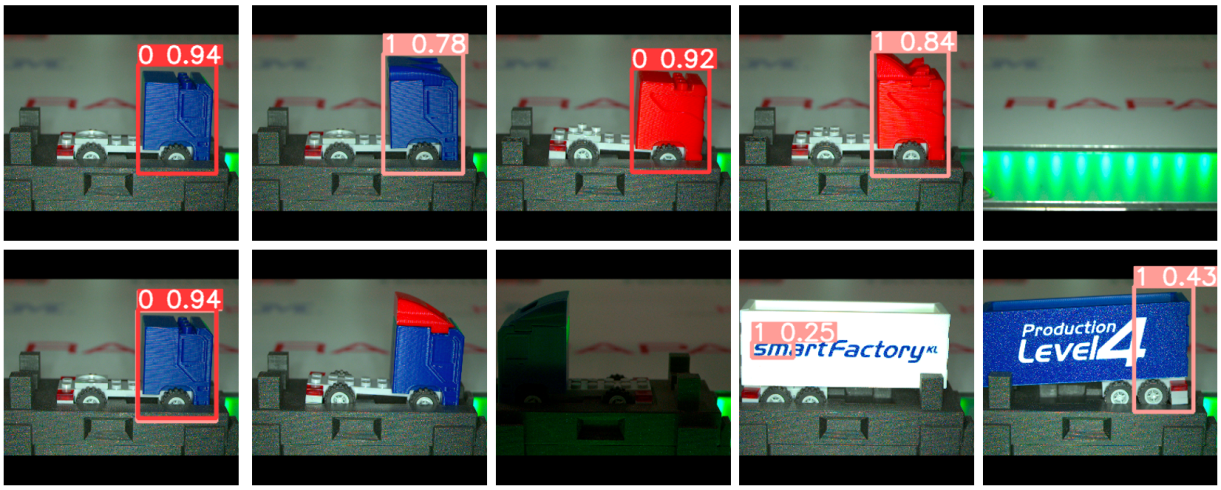

Figure 4.22: Centralized YOLOv5 model inference on SmartFactory-KL demonstrator images. The model detects cabins seen in training but fails on unseen configurations (e.g., blue cabin with red windshield) and generates false positives on trailers (Confidence Threshold = 0.20).

Figure 4.23: FedEnsemble YOLOv5 model inference on SmartFactory-KL demonstrator images. All cabin types, including unseen configurations and low-light silhouettes, are correctly detected with precise bounding boxes and no false positives (Confidence Threshold = 0.20).

4.3.3 Discussion

The experimental results from both trailer and cabin detection use cases demonstrate that the proposed FedEnsemble architecture provides a compelling alternative to conventional centralized training, particularly under domain shift and unseen object configurations. While the centralized YOLOv5 models trained on controlled datasets achieved near-perfect performance on in-distribution test data, their generalization collapsed when deployed on real-world images from the SmartFactory-KL quality inspection module. The primary sources of this degradation were complex backgrounds, variable illumination, novel orientations, and object–color combinations absent from the training distribution.

In contrast, the FedEnsemble models consistently delivered robust and accurate predictions across both test and deployment-level scenarios. These included multi-object frames, orientation changes caused by chassis attachments, cluttered industrial backdrops, and lighting variability. In the trailer detection use case, the centralized model failed to detect the penholder milled trailer and also produced imprecise bounding boxes for the white trailer, truncating the object geometry. The FedEnsemble model, however, successfully identified all trailer types and localized them with precise bounding boxes. A similar trend was observed in the cabin detection task, where the centralized model failed on critical cases such as the blue cabin with a red windshield and generated false positives on trailers. Conversely, the FedEnsemble model correctly classified all cabins, maintained high localization accuracy, and even performed reliably on low-light images where only silhouettes were visible.

These findings validate the hypothesis that FedEnsemble enhances generalization by leveraging client-specific specialization and iterative aggregation. Treating each client as a weak learner and combining their specialized representations enables the global model to capture a broader feature space, thereby becoming resilient to distributional variances not explicitly encountered

during local training. From this perspective, FL can also be interpreted as a form of ensemble learning, where the aggregation process acts as an implicit regularizer for DL models. A key distinction, however, is that FedEnsemble assumes full access to a centralized dataset that can be partitioned across clients, whereas traditional FL is motivated by privacy and data locality constraints.

Overall, the FedEnsemble framework improves detection accuracy, bounding-box precision, and robustness under deployment-level variability, outperforming centralized baselines even when trained on identical data. These results highlight its practical value for real-time industrial quality inspection, where reliability and cross-domain generalization are essential for deployment in decentralized manufacturing environments.

4.4 Federated Learning with Synthetic Dataset

This section evaluates the performance of FL on a hybrid dataset comprising both real and synthetic images for the task of object detection. The motivation for incorporating synthetic data stems from the limited number of available real training images (300 training images), which may be insufficient to train generalizable object detection models. The experimental setup simulates a realistic industrial use case in which one client holds real-world data, while another participates in the federated process using only synthetic images. This hybrid configuration is used to evaluate the robustness and generalization capability of the resulting global model, particularly in detecting small objects. Detailed descriptions of the dataset, experimental design, and architecture are provided in Section 3.5.

The object detection dataset comprises five classes: `LED`, `Resistor`, `Button`, `Buzzer`, and `Arduino`, as previously introduced in Section 3.5.1. To benchmark the effectiveness of the hybrid FL setup, a total of seven training strategies were evaluated. After conducting multiple trials with varying training configurations, the optimal setting for the setup was found to be 15 local epochs and 10 CRs, which consistently yielded the best global performance in terms of detection accuracy and generalization. The FedEnsemble variant utilized the same hybrid dataset, composed of 300 real and 300 synthetic images, randomly partitioned across three clients. Each client trained its local model using the same configuration of 15 local epochs and 10 CRs, and the server then ensembled the best-performing models from each client to construct the final global model.

A centralized baseline was trained on the complete hybrid dataset using YOLOv5l for 200 epochs. The model weights corresponding to the best validation accuracy were selected for evaluation. To assess the efficacy of knowledge transfer, a model was first trained on the synthetic dataset for 200 epochs. The core 10 layers were then frozen, and the remaining layers were fine-tuned on the real dataset for an additional 200 epochs. The best-performing weights across both

phases were used for final evaluation. For comparison, a fine-tuning variant was also tested by retraining the entire model on the real dataset using the same synthetic pretraining weights, without freezing any layers. Additional baselines included models trained exclusively on either the real or the synthetic dataset. These models used pretrained YOLOv5l weights and were trained for 200 epochs, and the best weights, determined by validation accuracy, were used for inference. All models were evaluated on an external test dataset designed to assess cross-domain generalization and robustness under unseen environmental conditions. The detailed comparative analysis and results are presented in the following subsections.

4.4.1 Evaluation Results

The evaluation metrics in this experiment follow the COCO protocol [Li14], introduced earlier in Section 3.3. In contrast to the previous object detection use cases, which primarily reported mAP@0.75 to emphasize precise localization, this section uses mAP@0.50 as the main criterion. At this IoU threshold, a prediction is correct if its overlap with the ground-truth box is at least 50 %. This choice reflects the challenges of detecting small, densely packed components in manufacturing, where minor misalignments are tolerable and recall of critical parts is prioritized over pixel-level precision.

To analyze scale-dependent behavior, the COCO size-specific metrics $\texttt{AP}_{\texttt{small}}$, $\texttt{AP}_{\texttt{medium}}$, and $\texttt{AP}_{\texttt{large}}$ are additionally reported. These are mean AP values averaged over IoU thresholds from 0.50 to 0.95 in steps of 0.05, but restricted to ground-truth objects within specific bounding-box area ranges:

$$AP_{\text{small}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} AP(c \mid A < 32^2)$$

$$AP_{\text{medium}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} AP(c \mid 32^2 \leq A < 96^2)$$

$$AP_{\text{large}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} AP(c \mid A \geq 96^2)$$

where A is the ground-truth bounding-box area in pixels and $\mathcal{C}$ is the set of classes. In the COCO convention, “small” means $A < 32^2$ pixels, “medium” means $32^2 \leq A < 96^2$, and “large” means $A \geq 96^2$. These categories are particularly relevant in manufacturing, where many safety critical parts (e.g., LEDs, resistors) fall into the `small` regime.

The models were evaluated on an external test dataset collected under significantly different conditions from the training distribution, including mixed lighting (natural and artificial), motion blur, cluttered backgrounds, which also included 16 empty frames without any target objects. As shown in Table 4.14, the FL model using FedAvg achieved the highest overall performance, with a **mAP@0.50 of 0.8638**. It also led across $\texttt{AP}_{\texttt{small}}$ and $\texttt{AP}_{\texttt{medium}}$, demonstrating strong generalization on small and mid-sized objects such as LEDs and resistors. The FedEnsemble model

followed closely with a mAP@0.50 of 0.8212, particularly excelling on large objects. Centralized models trained solely on real or synthetic data struggled to generalize, with the synthetic-only model performing worst (mAP@0.50 = 0.4066). Fine-tuning achieved the highest AP_{large}, but lagged on small components, which are most critical for inspection tasks.

Table 4.14: Comparison of all models based on object size-specific AP metrics and mAP@0.50 on the external test dataset captured under domain shift (blur, lighting, and noise)

Algorithm/Models	**AP_{small}**	**AP_{medium}**	**AP_{large}**	**mAP@0.50**
FedEnsemble Model	0.2882	0.4188	0.7480	0.8212
Federated Learning Model	**0.2969**	**0.4555**	0.7580	**0.8638**
Transfer Learning Model	0.2373	0.3272	0.7245	0.7681
Fine-Tuning Model	0.2912	0.3606	**0.7837**	0.7856
Hybrid Centralized Model	0.2843	0.3801	0.7531	0.7865
Real Centralized Model	0.2556	0.3559	0.7322	0.7638
Synthetic Centralized Model	0.0875	0.2445	0.3828	0.4066

To complement the quantitative results, the following figures show uncropped HOLO-Lens2 frames with outputs from top four representative models (from Table 4.14): Fine-tuning, Hybrid centralized, Federated global, and FedEnsemble. The first set shows detection performance on a complex test image containing all object classes; the second set evaluates robustness on an empty background frame.

Figures 4.24, 4.25, 4.26 and 4.27 present qualitative outputs from the four representative models on a representative external test image containing multiple instances of all five object classes. The fine-tune model, while able to detect large components such as the Arduino and buzzer with high confidence, consistently struggled on small objects such as LEDs and resistors, either missing them entirely or predicting with low confidence. The hybrid centralized model showed some improvement over fine-tuning by combining real and synthetic data, yet it still failed to consistently detect small and densely packed components, producing weaker confidence scores and occasional misclassifications. In contrast, the global federated model demonstrated robust generalization across all object classes, including small LEDs and resistors, and produced bounding boxes with confidence above 80 % even for the small object detections. The FedEnsemble model further confirmed this trend, delivering performance comparable to the global federated model, with accurate detections across scales and minimal errors. These observations closely align with the quantitative findings in Table 4.14, where federated models consistently achieved the highest mAP scores, particularly on small and medium objects.

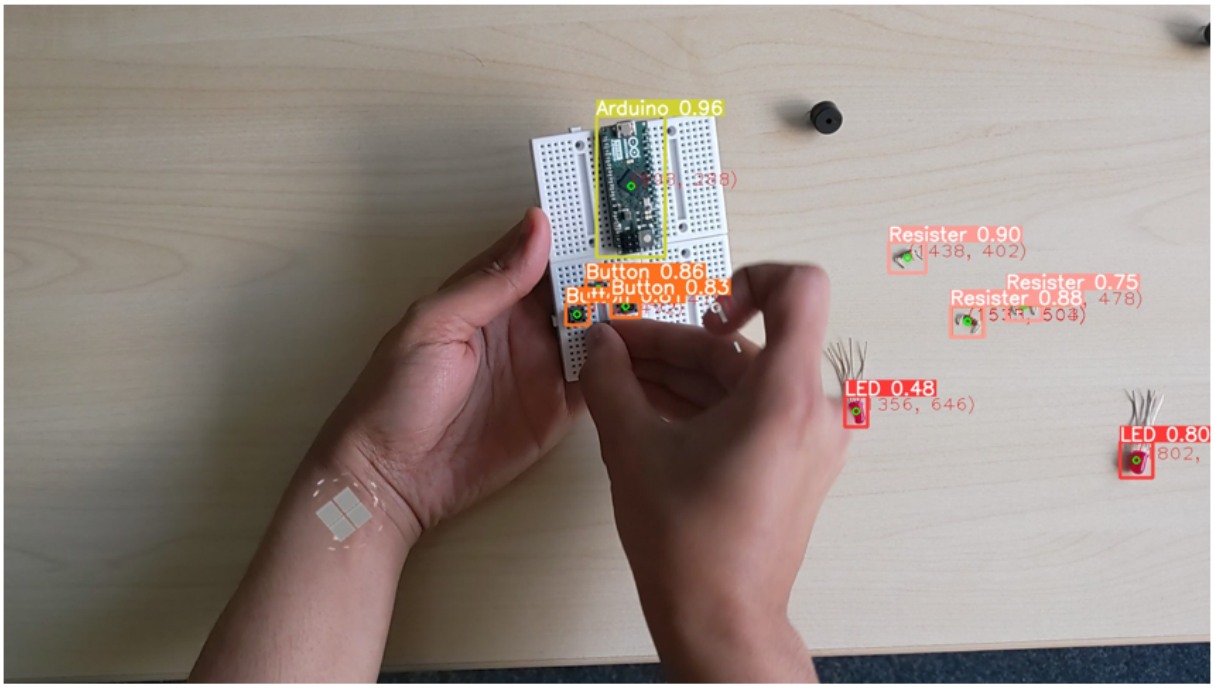

Figure 4.24: Fine-tune model inference on an external test image. Most large objects (Arduino, buzzer) are detected correctly, but several small resistors and LEDs are missed or predicted with low confidence. This illustrates the weakness of fine-tuning for small-object generalization.

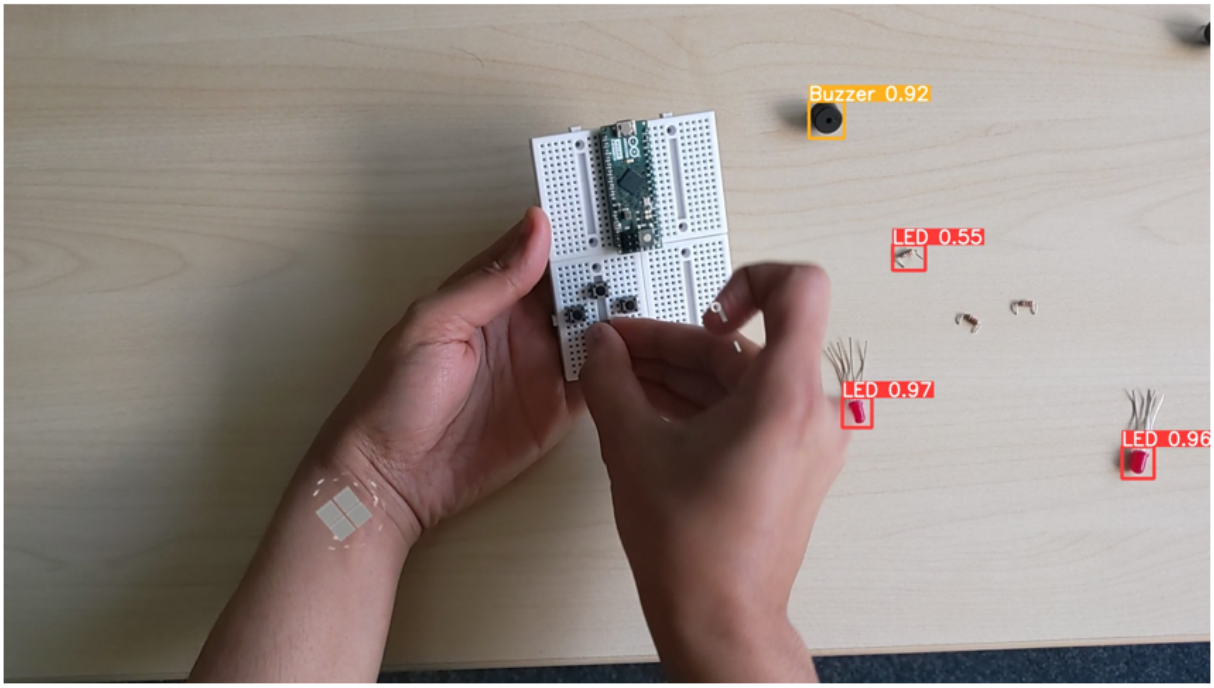

Figure 4.25: Hybrid centralized model inference on the same test image. While combining real and synthetic data improves detection compared to fine-tuning, small components such as buttons and LEDs remain inconsistently detected, with lower confidence or missing bounding boxes.

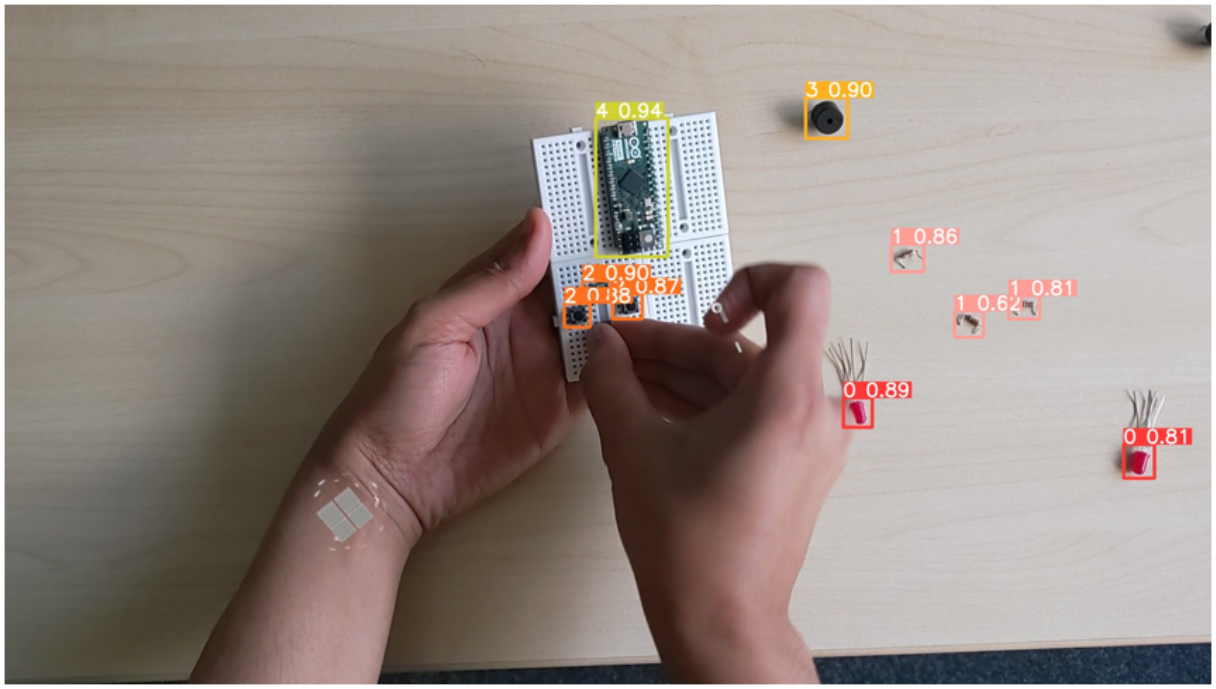

Figure 4.26: Global federated model inference on the same test image. Nearly all objects are detected correctly, including small LEDs and resistors, with confidence above 0.80. This demonstrates the robustness of federated training in capturing small and densely packed components.

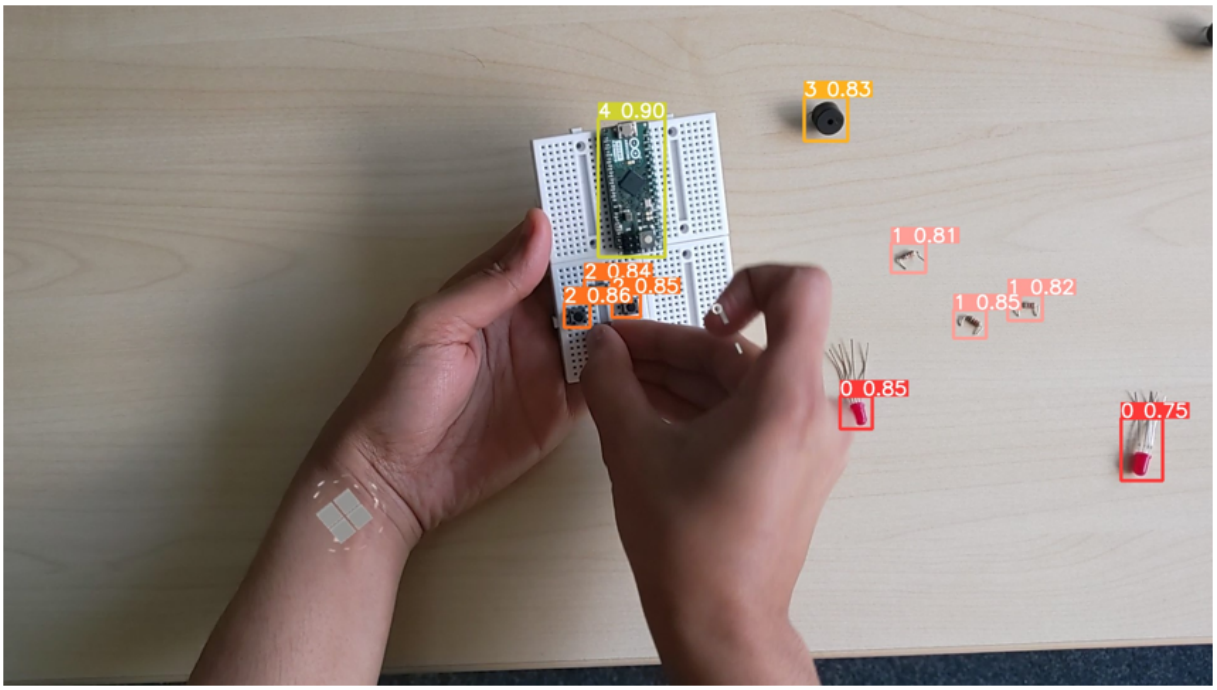

Figure 4.27: FedEnsemble model inference on the same test image. Performance is comparable to the global federated model, with accurate detection across all classes and high confidence levels, further validating ensemble aggregation as a viable alternative in hybrid client setups.

Figures 4.28, 4.29, 4.30, and 4.31 present the outputs of the four evaluated models on an empty background frame containing no target object classes. This test case was deliberately chosen to examine model robustness against false positives, a critical requirement in real-world deployment where unnecessary detections can lead to wasted resources and operational inefficiencies. The image itself shows the laboratory background, including a metallic grid or mesh structure with small box-like patterns. The fine-tune model misclassified these patterns as `Buttons` or `Buzzers`, producing several false detections. The hybrid centralized model reduced the number of such false positives but still assigned incorrect labels to parts of the background, showing its limited robustness under unseen conditions. By contrast, the global federated model and the FedEnsemble model correctly suppressed all predictions, demonstrating complete robustness by recognizing the absence of objects. This outcome is particularly significant in manufacturing contexts, where even a small number of false alarms can interrupt workflows or trigger unnecessary quality checks. Taken together, these qualitative observations strongly reinforce the quantitative evidence reported earlier: federated approaches not only deliver higher accuracy on small and medium components but also provide superior reliability under domain shift, making them especially well-suited for deployment in privacy-preserving industrial inspection pipelines.

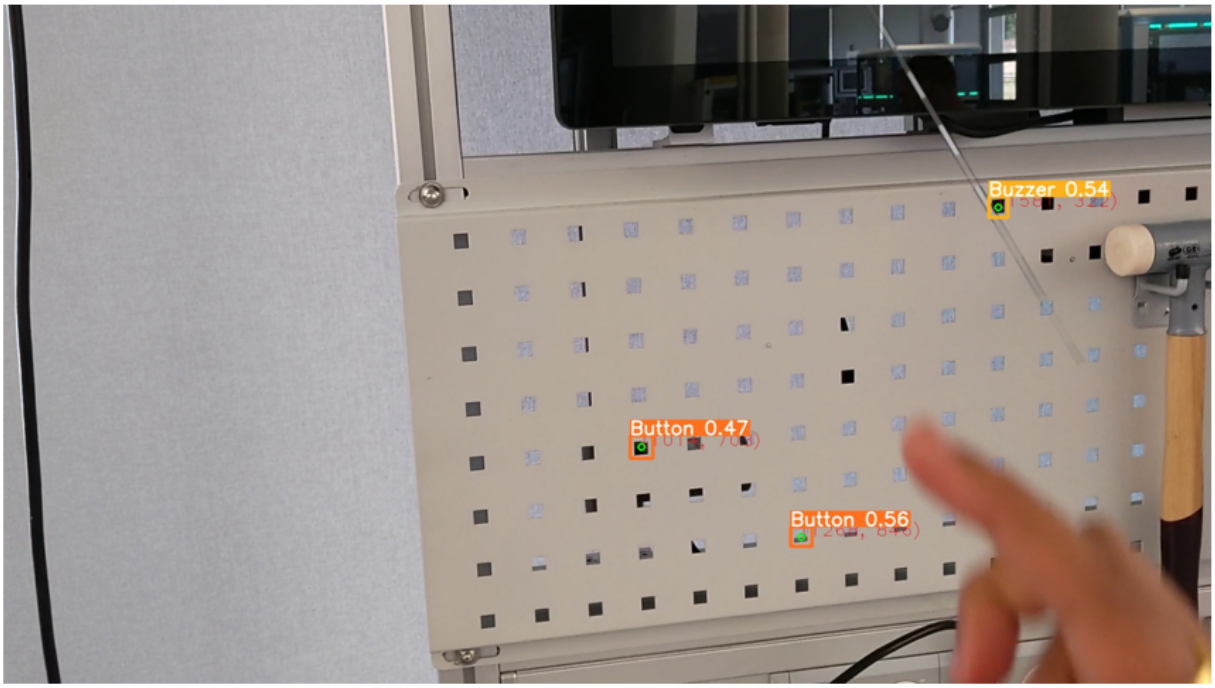

Figure 4.28: Fine-tune model inference on an empty background frame. The model produces false positives, mistakenly detecting nonexistent buttons and buzzers in the grid pattern.

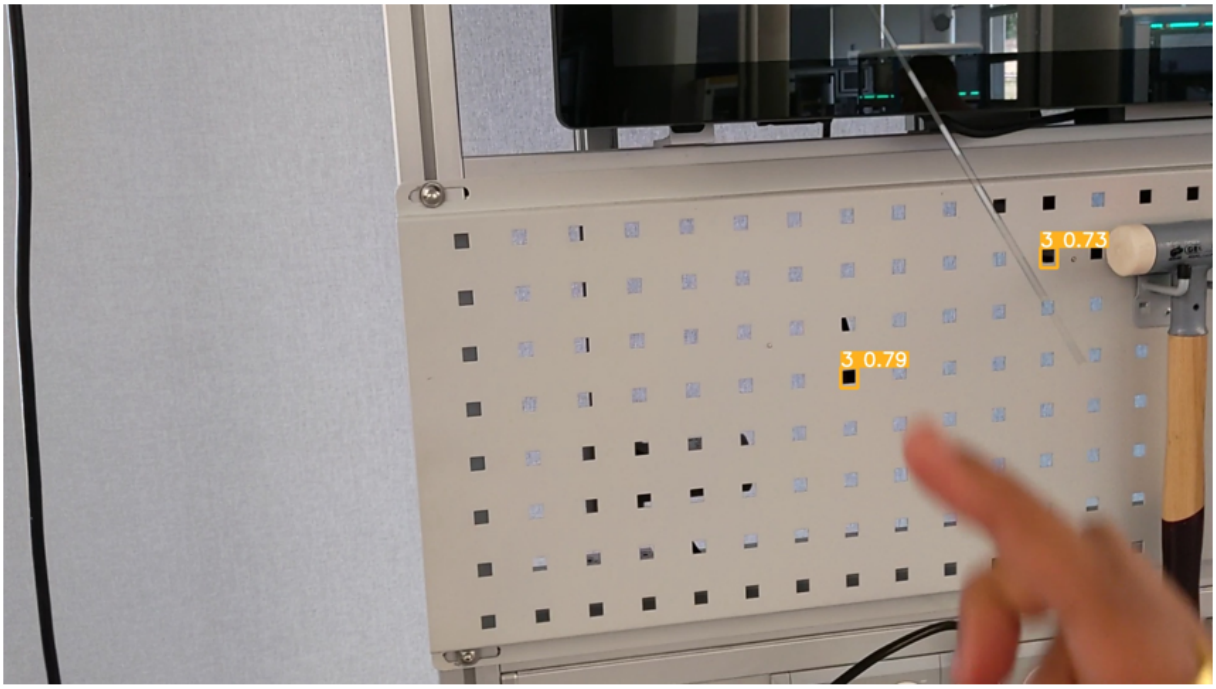

Figure 4.29: Hybrid centralized model inference on the empty frame. False positives are reduced compared to fine-tuning, but some spurious detections (e.g., Buzzer) still occur.

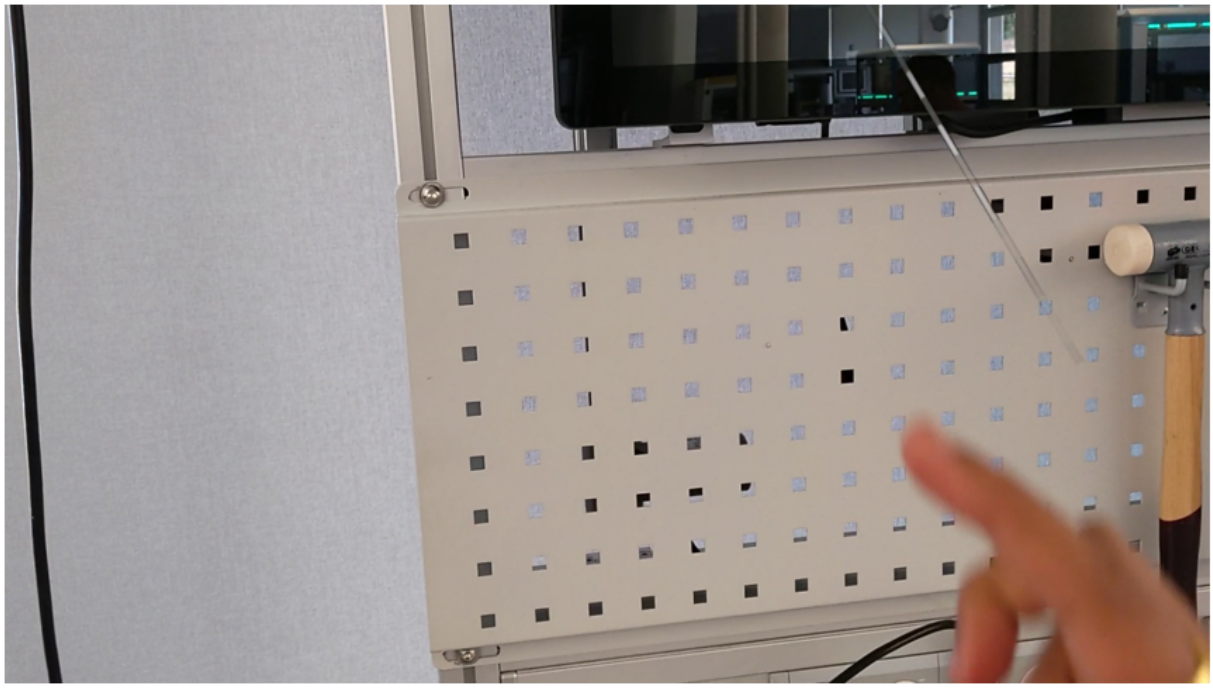

Figure 4.30: Global federated model inference on the empty frame. No false positives are produced, confirming the reliability of the federated model under unseen backgrounds.

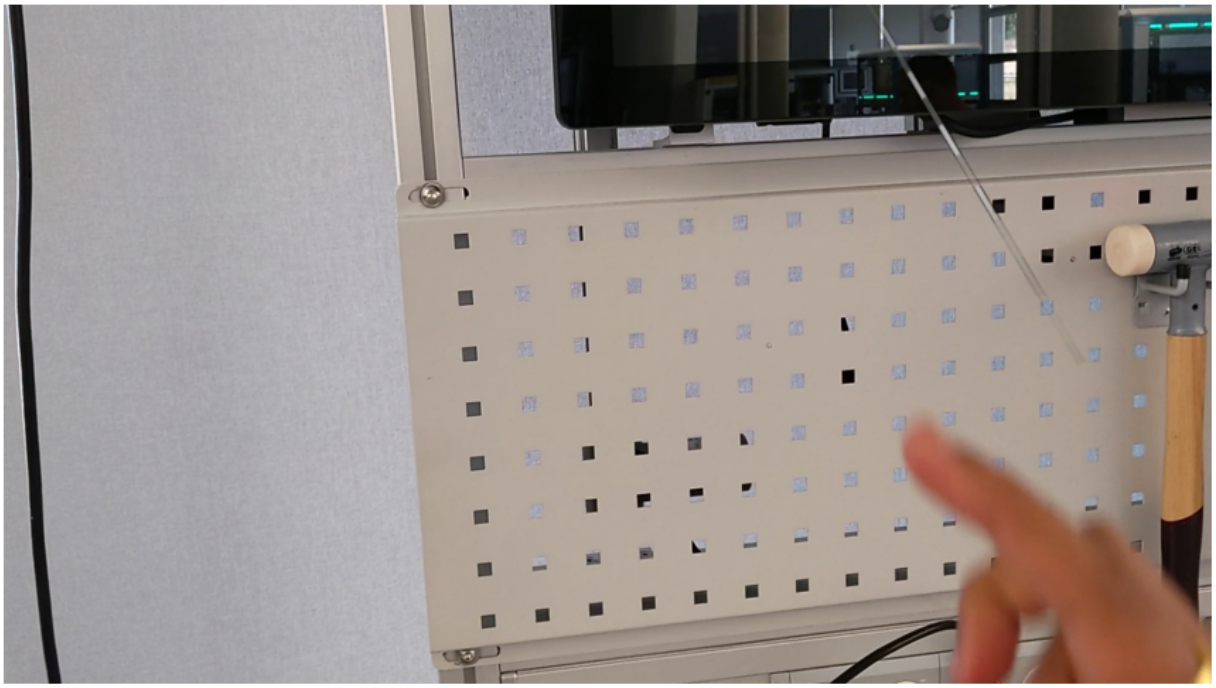

Figure 4.31: FedEnsemble model inference on the empty frame. Like the global federated model, it correctly suppresses all predictions, showing strong robustness against false detections in cluttered environments.

4.4.2 Discussion

The experiments on the hybrid real–synthetic dataset highlight the versatility and robustness of the proposed FL framework under conditions of data scarcity and domain shift. In this setup, one client contributed only synthetic CAD-generated samples while the other provided real images, thereby simulating a practical federation where data modalities differ substantially. Despite this imbalance, the federated training process produced global models that generalized strongly to external test datasets captured under unseen conditions.

Across all configurations, the global model trained with FedAvg consistently outperformed centralized and transfer learning baselines in terms of mAP@0.50 and size-specific metrics such as AP_{small}. Its superior performance on small and medium objects such as LEDs and resistors, demonstrates the ability of FL to learn transferable representations that remain robust across heterogeneous data sources. This is particularly relevant in manufacturing scenarios where the reliable detection of small components is crucial for avoiding costly quality failures.

The FedEnsemble model, while slightly behind FedAvg in mAP@0.50, delivered comparable robustness by aggregating diverse local learners. It was particularly effective on larger objects, confirming that ensembling in a federated setting captures complementary features from distinct data distributions. These results indicate that federated ensembles can provide an additional safeguard against domain-specific biases, even when one client contributes only synthetic data.

By contrast, centralized approaches struggled to match federated performance. The synthetic-only model performed the worst, suffering from poor generalization and high false positives, highlighting the domain gap between synthetic and real images. The hybrid centralized model, although trained on both real and synthetic samples, failed to reach the same level of robustness as the federated models. This gap can be attributed to centralized overfitting and the absence of the implicit regularization effect induced by client diversity in FL. Similarly, transfer learning and fine-tuning strategies showed inconsistent behavior: while fine-tuning achieved strong $\mathrm{AP}_{\mathrm{large}}$ scores, it degraded on small objects and exhibited false detections on empty frames, limiting its applicability in practice.

Overall, these findings validate the hypothesis that FL not only remains feasible but is advantageous when combining heterogeneous data modalities such as synthetic and real samples. The federated models consistently demonstrated higher robustness, reduced false positives, and stronger adaptability under domain shift. Importantly, this establishes FL as a practical solution for data-scarce industrial domains, where annotated real data may be limited or expensive to obtain, while synthetic approximations can act as effective surrogates when integrated through a federated pipeline.

4.5 Summary

This chapter evaluated the proposed FL framework across a broad range of computer vision tasks relevant to industrial quality inspection. The experiments were designed under realistic manufacturing constraints, including non-IID data distributions, limited client participation, domain shift, and scarce annotated data. Across all scenarios, the global federated models consistently outperformed or matched centralized and local baselines, not only in benchmark classification but also in complex object detection tasks under heterogeneous conditions.

The image classification experiments highlighted the generalization capability of FL on real datasets such as USB and cabin classification. In the USB case with three clients, the global model achieved accuracy comparable to the centralized model on the global test dataset (around 99.90 %) while surpassing local models on unseen defect types and geometries. Moreover, in live tests, the federated model delivered higher confidence scores than the centralized baseline, demonstrating greater reliability under deployment conditions. The cabin classification task, simulating extreme class-heterogeneous non-IID splits, further emphasized the resilience of the FL approach. Here, federated DenseNet-121 models achieved near-perfect accuracy and generalized better than centralized models to external datasets containing noise, lighting variations, and unseen windshield types.

The object detection experiments extended the framework to tasks requiring spatial localization. In USB detection, the global FedOD model attained perfect mAP@0.75 and high COCO-

style AP scores, outperforming all local models that struggled with unfamiliar geometries. In the more challenging cabin detection case, federated models again surpassed both local and centralized counterparts in detection accuracy and bounding box precision, including on unseen windshield classes and external deployment data. Live demonstrator tests confirmed these findings, with federated models showing greater robustness to visual noise and distractors.

The FedEnsemble architecture was then introduced as a novel extension of FL for object detection. Applied to trailer and cabin datasets, it delivered superior robustness and localization accuracy compared to centralized YOLOv5 models, particularly under domain shift. Whereas centralized models failed to generalize to real deployment frames with multi-object scenes, unseen color patterns, and low visibility, FedEnsemble consistently provided accurate detections.

Finally, the hybrid training experiments combining real and synthetic datasets offered further evidence of FL's flexibility. Despite the domain gap, federated models, outperformed all centralized strategies across mAP and object-size-specific AP metrics. The FedEnsemble variant maintained high accuracy for large objects, and both federated models demonstrated resilience to false positives on empty background images. In contrast, synthetic and real-only models and conventional transfer learning and centralized hybrid models failed to generalize effectively under deployment-like conditions.

In summary, the results across all use cases demonstrate that the proposed FL framework enables generalizable, robust, and privacy-preserving models for industrial quality inspection. The empirical evidence confirms that FL supports collaborative intelligence across distributed manufacturing nodes while preserving data ownership, addressing heterogeneity, and reducing annotation and deployment costs. These findings position FL as a key enabler of intelligent, decentralized, and privacy-conscious manufacturing systems of the future.

5 Conclusion and Outlook

This chapter brings together the main findings of this thesis and answers the research questions defined at the beginning. It shows how the proposed Federated Learning (FL) framework contributes to industrial quality inspection and related fields. The chapter reflects on the methods developed, the experiments carried out, and the practical deployments achieved during the research. Finally, it looks ahead to how these results can guide the development of the next generation of FL systems.

5.1 Summary and Conclusion

The main objective of this thesis was to investigate whether FL can serve as a practically feasible and technically robust paradigm for industrial quality inspection, where data is inherently distributed, privacy-sensitive, and heterogeneous. Over the span of four years, this research advanced from foundational experiments on federated image classification to the development of novel federated object detection pipelines, ensemble strategies, and hybrid synthetic–real training paradigms. The key contributions and insights are summarized below with respect to the guiding research questions.

In doing so, this section provides a concise summary of the research carried out in the thesis, covering the datasets developed, the methods proposed, and the experiments conducted, before drawing the main conclusions with respect to the research questions defined in Chapter 1.2.

Addressing Question 1: Feasibility of FL for custom manufacturing use cases

The first research question examined whether FL is a viable and effective approach for real-world quality inspection scenarios, particularly where client data is independently collected, heterogeneous, and non-IID. This thesis demonstrated feasibility across both benchmark and industrial datasets. On the CIFAR-10 benchmark under IID conditions, federated models reached accuracies comparable to centralized baselines, thereby validating the implementation of the framework. More importantly, when applied to custom datasets created in the context of SmartFactory-KL, the framework consistently outperformed local baselines and matched or ex-

ceeded centralized models. In the USB classification use case with three clients, the federated global model not only achieved competitive accuracy on the global test set but also proved more robust in live deployment, where unseen defect geometries and lighting variations challenged the models. The cabin classification experiments extended this analysis to four clients under extreme class-heterogeneous conditions. Here, the federated DenseNet-121 global model surpassed centralized trained model in terms of generalizing effectively to the external dataset that contained previously unseen windshield types, noise, and environmental shifts.

These results provide conclusive evidence that FL is both technically feasible and practically advantageous in industrial manufacturing scenarios. It enables inter-organizational collaboration while preserving data ownership and privacy, addressing both regulatory and competitive concerns that make centralized data collection impractical.

Addressing Question 2: Extending FL to complex vision tasks such as object detection

The second research question investigated whether FL can be extended from classification to object detection, which requires accurate spatial localization of components and defects. To address this, a Federated Object Detection (FedOD) pipeline was implemented based on the YOLO architecture. This represented the first federated adaptation of YOLO for industrial quality inspection tasks. The results from USB and cabin object detection confirmed that the federated global models consistently outperformed local-only models across key metrics such as mAP, bounding box precision, and robustness under non-IID distributions. Live inference on SmartFactory-KL demonstrators further validated the generalization capability of the federated models: the global detectors were able to handle visual distractors, unseen geometries, and noisy deployment conditions significantly better than both centralized and local baselines. Importantly, images from the demonstrators were never included in the training data, highlighting the strength of federated learning in producing robust models under domain shift.

These findings confirm that FL is not restricted to image-level classification but can be effectively extended to advanced, spatially aware computer vision tasks. This establishes its scalability to more complex manufacturing inspection applications.

Addressing Sub-questions: Design strategies and practical benefits for stakeholders

The first sub-question focused on architectural and system-level design considerations for FL in industrial environments, where the number of clients is small and data distributions are highly heterogeneous. The thesis proposed a modular FL architecture that incorporated several critical strategies: lightweight but expressive backbone models, a best-weight selection mechanism

to improve generalization, centralized evaluation for stable aggregation, and adaptive training configurations tuned for few-client setups. These innovations were empirically validated to stabilize convergence and enhance accuracy even under limited data availability.

The second sub-question addressed the practical value of FL for manufacturing stakeholders. Experimental evidence confirmed that federated training yielded global models that outperformed isolated client models, while ensuring that raw data never left local premises. This dual benefit of maintaining competitive accuracy with centralized training while fully preserving data privacy, positions FL as a compelling and practical solution for manufacturers who wish to collaborate without exposing proprietary information.

Novel contributions of this thesis

Beyond addressing the central research questions, this thesis introduced two notable extensions to the FL paradigm. First, the Federated Ensemble (FedEnsemble) approach was proposed, where datasets were deliberately partitioned across clients and aggregated with FedAvg. Applied to trailer and cabin datasets, this strategy consistently outperformed centralized training and achieved stronger generalization on external test data under domain shift. Second, a hybrid federated setup with one client using only real data and another only synthetic data demonstrated that even synthetic-only clients can contribute meaningfully to the global model. The resulting federated models surpassed centralized baselines trained on merged hybrid datasets, particularly on external tests captured in new environments.

Together, these contributions show that FL can deliver robustness under domain shift and enable simulation-to-real transfer, broadening its applicability beyond conventional classification and detection tasks.

Positioning within the State of the Art

A structured comparison with related work, as summarized in Table 5.1, highlights how this thesis bridges critical research gaps in FL for industrial applications. Unlike prior studies that focused on synthetic benchmarks or lacked external validation, this research utilized real manufacturing datasets, integrated real and synthetic clients, and validated models through live deployment at SmartFactory-KL. The final column of Table 5.1, clearly demonstrates that this thesis uniquely integrates all of the identified features, from using real industrial datasets, to hybrid real–synthetic setups, to open implementation and evaluation on external test data. By doing so, it positions the proposed framework as both technically rigorous and practically deployable, advancing the state of the art in FL for manufacturing.

Table 5.1: Feature Coverage in Selected FL Studies (Updated with Contributions from This Thesis)

Feature	Dib et al.	Ge et al.	Kanagavelu et al.	Pruckovskaja et al.	Becker et al.	Siemens & Katulu	Chen et al.	Ahn et al.	This Thesis
Vision Dataset	✗	✗	✗	✓	✗	✓	✓	✗	✓
Real Clients	✓	✓	✓	✓	✓	✓	✗	✓	✓
Mixed Clients (Real + Synthetic)	✗	✗	✗	✗	✗	✗	✗	✗	✓
Open Implementation	✗	✗	✗	✓	✓	✗	✓	✓	✓
Industrial Use Case	✓	✓	✓	✓	✓	✓	✗	✓	✓
FL Image Classification	✓	✗	✗	✓	✗	✓	✗	✗	✓
FL Object Detection	✗	✗	✗	✗	✗	✗	✓	✗	✓
External Test Data	✗	✓	✗	✓	✓	✗	✗	✗	✓
Comparison with Local Models	✗	✓	✗	✓	✓	✗	✗	✗	✓

5.2 Outlook

While the results presented in this work are promising, several areas remain open for exploration. One limitation is the assumption that all clients participate in the FL process from the very first communication round. In real-world federations, new clients may join after a global model has already converged or during intermediate training phases. Future research should explore incremental client integration, warm-start strategies, and fine-tuning mechanisms that support late joiners without degrading global performance.

Another extension involves relaxing the assumption of homogeneous model architectures. In this thesis, all clients used the same neural network backbone for simplicity and consistency. However, practical deployments may require clients to operate with diverse architectures based on their hardware constraints or task-specific needs. Supporting such heterogeneity, possibly through knowledge distillation, intermediate representation alignment, or federated meta-learning remains a critical challenge for next-generation FL frameworks.

The work also assumed a secure and trustworthy environment. In many industrial ecosystems, this may not hold true. Malicious clients could launch model poisoning or data inference attacks. Future work should investigate defense mechanisms such as anomaly detection, Byzantine-resilient aggregation, and differential privacy. These enhancements are essential for FL systems intended for open, multi-party industrial collaborations.

Lastly, the foundational architectures and training strategies proposed here are applicable beyond quality inspection. The FedOD and FedEnsemble frameworks could be adapted to other vision-based domains such as autonomous navigation, industrial robotics, or federated reinforcement learning for path planning and process optimization. Similarly, hybrid real-synthetic training strategies have potential applications in simulation-to-real transfer, enabling faster prototyping of AI models in low-data settings.

In conclusion, this thesis has presented a comprehensive study of FL for manufacturing quality inspection, addressing the central objectives defined at the outset. It demonstrated that FL is both technically feasible and practically deployable in decentralized industrial contexts, enabling collaborative intelligence without compromising data privacy. Across classification and detection tasks, the global federated models consistently matched or outperformed centralized baselines and proved robust under highly non-IID conditions. Novel contributions such as the FedOD pipeline, the FedEnsemble strategy, and hybrid real–synthetic integration extended the applicability of federated learning beyond conventional settings and tackled challenges of domain shift and data scarcity.

By creating new FL datasets, validating the framework in live demonstrators, and bridging theoretical advances with practical deployment, this work provides not only methodological innovations but also practical evidence of real-world applications. As manufacturing systems con-

tinue to evolve toward increasingly distributed and collaborative paradigms, the findings of this thesis highlight FL as a key enabler of intelligent, privacy-preserving, and scalable solutions shaping the future of Industry 4.0 and beyond.

Bibliography

Articles and Monographs

[Ab16] Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; Zhang, L.: Deep Learning with Differential Privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. Association for Computing Machinery, New York, NY, USA, pp. 308–318, 2016, doi: 10.1145/2976749.2978318, url: https://doi.org/10.1145/2976749.2978318.

[Ah23] Ahn, J.; Lee, Y.; Kim, N.; Park, C.; Jeong, J.: Federated Learning for Predictive Maintenance and Anomaly Detection Using Time Series Data Distribution Shifts in Manufacturing Processes. Sensors 23 (17), p. 7331, 2023.

[An24a] Antony, J.; Hegiste, V.; Nazeri, A.; Tavakoli, H.; Walunj, S.; Plociennik, C.; Ruskowski, M.: Enhancing Object Detection Performance for Small Objects through Synthetic Data Generation and Proportional Class-Balancing Technique: A Comparative Study in Industrial Scenarios. In: Advanced Machine Learning Applications in Industrial Settings. Springer, pp. 155–172, 2024, doi: 10.1007/978-3-031-57496-2_10.

[An24b] Anwar, A.; Moser, B.; Herurkar, D.; Raue, F.; Hegiste, V.; Legler, T.; Dengel, A.: FedAD-Bench: A Unified Benchmark for Federated Unsupervised Anomaly Detection in Tabular Data. In: 2024 2nd International Conference on Federated Learning Technologies and Applications (FLTA). Pp. 115–122, 2024, doi: 10.1109/FLTA63145.2024.10839838.

[Ao17] Aono, Y.; Hayashi, T.; Wang, L.; Moriai, S.: Privacy-preserving deep learning via additively homomorphic encryption. IEEE Transactions on Information Forensics and Security 13 (5), pp. 1333–1345, 2017, doi: 10.1109/TIFS.2017.2787987.

[Be22] Becker, S.; Styp-Rekowski, K.; Stoll, O. V. L.; Kao, O.: Federated Learning for Autoencoder-based Condition Monitoring in the Industrial Internet of Things. arXiv preprint arXiv:2211.07619, 2022.

[Bi06] Bishop, C. M.: Pattern Recognition and Machine Learning. Springer, 2006, isbn: 9780387310732.

[Bo15] Bost, R.; Popa, R. A.; Tu, S. E.; Goldwasser, S.: Machine learning classification over encrypted data. In: Proceedings of the Network and Distributed System Security Symposium (NDSS). The Internet Society, 2015, doi: 10.14722/ndss.2015.23241.

[Bo17] Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H. B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K.: Practical Secure Aggregation for Privacy-Preserving Machine Learning. In:

Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS). Pp. 1175–1191, 2017.

[Br20] Bressem, K. K.; Adams, L. C.; Erxleben, C.; Hamm, B.; Niehues, S. M.; Makowski, M. R.: Comparing different deep learning architectures for classification of chest radiographs. Scientific Reports 10 (1), pp. 1–10, 2020.

[Ca18] California State Legislature: California Consumer Privacy Act of 2018, 2018, url: https://oag.ca.gov/privacy/ccpa.

[CC21] Chen, H.-Y.; Chao, W.-L.: FedBE: Making Bayesian Model Ensemble Applicable to Federated Learning. In: International Conference on Learning Representations (ICLR). 2021.

[Ch19] Chen, M. et al.: Real-World Image Datasets for Federated Learning. arXiv preprint arXiv:1910.11089, 2019.

[Ch25] Cheng, H.: Advancements in Image Classification: From Machine Learning to Deep Learning. In: ITM Web of Conferences. Vol. 51, p. 02016, 2025, doi: 10.1051/itmconf/20235102016.

[De09] Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. Ieee, pp. 248–255, 2009.

[DR14] Dwork, C.; Roth, A.: The Algorithmic Foundations of Differential Privacy. Cambridge University Press, 2014.

[DRP21] Dib, M. A. d. S.; Ribeiro, B.; Prates, P.: Federated Learning as a Privacy-Providing Machine Learning for Defect Predictions in Smart Manufacturing. Smart and Sustainable Manufacturing Systems 5 (1), pp. 1–17, 2021.

[Ev10] Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; Zisserman, A.: The pascal visual object classes (voc) challenge. In: International journal of computer vision. Springer, pp. 303–338, 2010.

[Fr24] Fridman, K.; Grotepass, J.; Legler, T.; Hegiste, V.: Federated Learning for Shared Production Scenarios. In (Azevedo, A.; Gonçalves, R.; Stark, R., eds.): Advances in Production Management Systems. Resilient and Sustainable Industry 5.0 Systems. CRC Press, Taylor & Francis Group, 2024, doi: 10.1201/9781032690711-3.

[Ga22] Gaia-X European Association for Data and Cloud AISBL: Gaia-X Architecture Document, Accessed: 2025-02-06, 2022, url: https://docs.gaia-x.eu/technical-committee/architecture-document/22.10/.

[GBC16] Goodfellow, I.; Bengio, Y.; Courville, A.: Deep Learning. MIT Press, 2016.

[Ge21] Ge, N.; Li, G.; Zhang, L.; Liu, Y.: Failure Prediction in Production Line Based on Federated Learning: An Empirical Study. arXiv preprint arXiv:2101.11715, 2021.

[Gu17] Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K. Q.: On Calibration of Modern Neural Networks. In: Proceedings of the 34th International Conference on Machine Learning (ICML). Vol. 70, pp. 1321–1330, 2017.

[Ha18] Hard, A.; Rao, K.; Mathews, R.; Ramaswamy, S.; Beaufays, F.; Augenstein, S.; Eichner, H.; Kiddon, C.; Ramage, D.: Federated learning for mobile keyboard prediction. In: arXiv preprint arXiv:1811.03604. 2018.

[HAA20] He, C.; Annavaram, M.; Avestimehr, S.: FedMA: Federated learning via model agnostic matching. In: Proceedings of the 37th International Conference on Machine Learning (ICML). Vol. 119, PMLR, pp. 4381–4391, 2020.

[He16] He, K.; Zhang, X.; Ren, S.; Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. Pp. 770–778, 2016.

[He23] Hegiste, V.; Legler, T.; Fridman, K.; Ruskowski, M.: Federated Object Detection for Quality Inspection in Shared Production. In: 2023 Eighth International Conference on Fog and Mobile Edge Computing (FMEC). Pp. 151–158, 2023, doi: 10.1109/FMEC59375.2023.10305969.

[He24] Hegiste, V.; Walunj, S.; Antony, J.; Legler, T.; Ruskowski, M.: Enhancing Object Detection with Hybrid dataset in Manufacturing Environments: Comparing Federated Learning to Conventional Techniques. In: 2024 1st International Conference on Innovative Engineering Sciences and Technological Research (ICIESTR). Pp. 1–6, 2024, doi: 10.1109/ICIESTR60916.2024.10798269.

[HLR22] Hegiste, V.; Legler, T.; Ruskowski, M.: Application of Federated Machine Learning in Manufacturing. In: 2022 International Conference on Industry 4.0 Technology (I4Tech). Pp. 1–8, 2022, doi: 10.1109/I4Tech55392.2022.9952385.

[HLR23] Hegiste, V.; Legler, T.; Ruskowski, M.: Federated Ensemble YOLOv5 – A Better Generalized Object Detection Algorithm. In: 2023 Eighth International Conference on Fog and Mobile Edge Computing (FMEC). Pp. 7–14, 2023, doi: 10.1109/FMEC59375.2023.10305958.

[HLR24] Hegiste, V.; Legler, T.; Ruskowski, M.: Towards Robust Federated Image Classification: An Empirical Study of Weight Selection Strategies in Manufacturing. In: 2024 2nd International Conference on Federated Learning Technologies and Applications (FLTA). Pp. 55–62, 2024, doi: 10.1109/FLTA63145.2024.10839986.

[HLR25a] Hegiste, V.; Legler, T.; Ruskowski, M.: Collaborative Learning in Shared Production Environment Using Federated Image Classification. In (Alexopoulos, K.; Makris, S.; Stavropoulos, P., eds.): Advances in Artificial Intelligence in Manufacturing II. ESAIM 2024. Lecture Notes in Mechanical Engineering, Springer, Cham, pp. 98–106, 2025, doi: 10.1007/978-3-031-86489-6_11.

[HLR25b] Hegiste, V.; Legler, T.; Ruskowski, M.: Trade-Off Between Communication Rounds and Local Epochs in Federated Learning. In: Proceedings of the 45th IEEE International Conference on Distributed Computing Systems (ICDCS). Presented, IEEE, 2025.

[Hu17] Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K. Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. Pp. 4700–4708, 2017.

[JC23] Ju, R.-Y.; Cai, W.: Fracture detection in pediatric wrist trauma X-ray images using YOLOv8 algorithm. Scientific Reports 13, 2023, doi: 10.1038/s41598-023-47460-7.

[Jo20] Jocher, G.: Ultralytics YOLOv5, version 7.0, 2020, doi: 10.5281/zenodo.3908559, url: https://github.com/ultralytics/yolov5.

[Ka20] Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; Suresh, A. T.: SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In (III, H. D.; Singh, A., eds.): Proceedings of the 37th International Conference on Machine Learning. Vol. 119. Proceedings of Machine Learning Research, PMLR, pp. 5132–5143, 2020, url: https://proceedings.mlr.press/v119/karimireddy20a.html.

[Ka21a] Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; D'Oliveira, R. G. L.; Eichner, H.; El Rouayheb, S.; Evans, D.; Gardner, J.; Garrett, Z.; Gascón, A.; Ghazi, B.; Gibbons, P. B.; Gruteser, M.; Harchaoui, Z.; He, C.; He, L.; Huo, Z.; Hutchinson, B.; Hsu, J.; Jaggi, M.; Javidi, T.; Joshi, G.; Khodak, M.; Konecný, J.; Korolova, A.; Koushanfar, F.; Koyejo, S.; Lepoint, T.; Liu, Y.; Mittal, P.; Mohri, M.; Nock, R.; Özgür, A.; Pagh, R.; Qi, H.; Ramage, D.; Raskar, R.; Raykova, M.; Song, D.; Song, W.; Stich, S. U.; Sun, Z.; Suresh, A. T.; Tramèr, F.; Vepakomma, P.; Wang, J.; Xiong, L.; Xu, Z.; Yang, Q.; Yu, F. X.; Yu, H.; Zhao, S.: Advances and Open Problems in Federated Learning. Foundations and Trends® in Machine Learning 14 (1–2), pp. 1–210, 2021, doi: 10.1561/2200000083, url: http://dx.doi.org/10.1561/2200000083.

[Ka21b] Kanagavelu, R.; Li, Z.; Samsudin, J.; Hussain, S.; Yang, F.; Yang, Y.; Goh, R. S. M.; Cheah, M.: Federated Learning for Advanced Manufacturing Based on Industrial IoT Data Analytics. In: Implementing Industry 4.0. Springer, pp. 143–176, 2021.

[Ka22] Katsamenis, I.; Karolou, E.; Davradou, A.; Protopapadakis, E.; Doulamis, A.; Doulamis, N.; Kalogeras, D.: TraCon: A Novel Dataset for Real-Time Traffic Cones Detection Using Deep Learning. In. Pp. 382–391, 2022, isbn: 978-3-031-17600-5, doi: 10.1007/978-3-031-17601-2_37.

[Ke24] Keylabs: Image Classification Techniques for Different Tasks (e.g., Object Detection, Scene Recognition), https://keylabs.ai/blog/image-classification-techniques-for-different-tasks-e-g-object-detection-scene-recognition/, Accessed: 2025-04-24, 2024.

[Kh22] Khan, M. U.; Ali, S. H.; Khan, M. K.; Rehman, S. U.; Jan, S.; Kim, H. S.: FedGrid: A Secure Framework with Federated Learning for Energy Optimization in the Smart Grid. Energies 16 (24), p. 8097, 2022, doi: 10.3390/en16248097.

[Ko23] Koch, L.; Brandstetter, P.; Samarin, K.; Kastner, W.: Federated Learning for Predictive Maintenance and Quality Inspection in Manufacturing: A Comparison of Aggregation Methods. arXiv preprint arXiv:2304.11101, 2023, url: https://arxiv.org/abs/2304.11101.

[Kr09] Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images, tech. rep., University of Toronto, 2009, url: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[KS24] Katulu GmbH; Siemens AG: How Siemens Mastered Federated Learning with Katulu, tech. rep., White paper presented at Hannover Messe 2024., Katulu GmbH, 2024.

[KSH12] Krizhevsky, A.; Sutskever, I.; Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks. In (Pereira, F.; Burges, C.; Bottou, L.; Weinberger, K., eds.): Advances in Neural Information Processing Systems. Vol. 25, Curran Associates, Inc.,

2012, url: https : / / proceedings . neurips . cc / paper _ files / paper / 2012 / file / c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[Le25] Legler, T.; Hegiste, V.; Anwar, A.; Ruskowski, M.: Addressing Heterogeneity in Federated Learning: Challenges and Solutions for a Shared Production Environment. Procedia Computer Science 253, 6th International Conference on Industry 4.0 and Smart Manufacturing, pp. 2831–2840, 2025, issn: 1877-0509, doi: 10.1016/j.procs.2025.02.007.

[Le98] LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86 (11), pp. 2278–2324, 1998.

[LHR24] Legler, T.; Hegiste, V.; Ruskowski, M.: Mapping of Newcomer Clients in Federated Learning Based on Activation Strength. In: Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems. International Conference on Flexible Automation and Intelligent Manufacturing (FAIM). Springer Nature Switzerland, pp. 1139–1148, 2024, doi: 10.1007/978-3-031-38165-2_130.

[LHR25] Legler, T.; Hegiste, V.; Ruskowski, M.: Multifaceted Applications of Federated Learning: Beyond Neural Networks. In (Alexopoulos, K.; Makris, S.; Stavropoulos, P., eds.): Advances in Artificial Intelligence in Manufacturing II. ESAIM 2024. Lecture Notes in Mechanical Engineering, Springer, Cham, 2025, doi: 10.1007/978-3-031-86489-6_28.

[LHS21] Li, Q.; He, B.; Song, D.: Model-Contrastive Federated Learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Pp. 10096–10105, 2021.

[Li14] Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C. L.: Microsoft COCO: Common Objects in Context. In: European conference on computer vision. Springer, pp. 740–755, 2014.

[Li20] Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V.: Federated Optimization in Heterogeneous Networks. In: Proceedings of the 2nd MLSys Conference. MLSys, pp. 429–450, 2020.

[Li21] Liu, Y.; Sun, P.; Wergeles, N.; Shang, Y.: A survey and performance evaluation of deep learning methods for small object detection. Expert Systems with Applications 172, p. 114602, 2021.

[LWL19] Liu, B.; Wang, L.; Liu, M.: Lifelong Federated Reinforcement Learning: A Learning Architecture for Navigation in Cloud Robotic Systems. IEEE Robotics and Automation Letters 4 (4), pp. 4555–4562, 2019, doi: 10.1109/LRA.2019.2931769.

[Mc17] McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B. A. y.: Communication-Efficient Learning of Deep Networks from Decentralized Data. In (Singh, A.; Zhu, J., eds.): Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. Vol. 54. Proceedings of Machine Learning Research, PMLR, pp. 1273–1282, 2017, url: https://proceedings.mlr.press/v54/mcmahan17a.html.

[NSH18] Nasr, M.; Shokri, R.; Houmansadr, A.: Comprehensive Privacy Analysis of Deep Learning: Stand-alone and Federated Settings. Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, pp. 895–912, 2018, doi: 10.1145/3243734.3243837.

[Pa21] Parimala, M.; Priya, S. R.; Pham, Q.-V.; Dev, K.; Maddikunta, P. K. R.; Gadekallu, T. R.; Huynh-The, T.: Fusion of Federated Learning and Industrial Internet of Things: A Survey. arXiv preprint arXiv:2101.00798, 2021.

[Pr23] Pruckovskaja, V.; Weissenfeld, A.; Heistracher, C.; Graser, A.; Kafka, J.; Leputsch, P.; Schall, D.; Kemnitz, J.: Federated Learning for Predictive Maintenance and Quality Inspection in Industrial Applications. In: 2023 Prognostics and Health Management Conference (PHM). Pp. 312–317, 2023, doi: 10.1109/PHM58589.2023.00064.

[Ra20] Ratan, P.: What is the Convolutional Neural Network Architecture? Accessed: 2025-04-28, 2020, url: https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/.

[Re16] Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. Pp. 779–788, 2016.

[Re21] Reddi, S. J.; Charles, Z.; Zaheer, M.; Garrett, Z.; Rush, K.; Konečnỳ, J.; Kumar, S.; McMahan, H. B.: Adaptive federated optimization. In: International Conference on Learning Representations (ICLR). 2021.

[RF18] Redmon, J.; Farhadi, A.: YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767, 2018.

[Ri20] Rieke, N.; Hancox, J.; Li, W.; Milletari, F.; Roth, H. R.; Albarqouni, S.; Bakas, S.; Galtier, M. N.; Landman, B.; Maier-Hein, K. H., et al.: The future of digital health with federated learning. npj Digital Medicine 3 (1), pp. 1–7, 2020.

[SBD21] Samarakoon, S.; Bennis, M.; Debbah, M.: End-to-End Federated Learning for Autonomous Driving Vehicles. In: 2021 IEEE Global Communications Conference (GLOBECOM). IEEE, pp. 1–6, 2021, doi: 10.1109/GLOBECOM46510.2021.9685481.

[Se21] Senseye: The World's Largest Manufacturers Lose Almost $1Trillion a Year to Machine Failures$, Accessed: 2025-01-17, 2021, url: https://www.automation.com/en-us/articles/june-2021/world-largest-manufacturers-lose-almost-1-trillion.

[Sh17] Shokri, R.; Stronati, M.; Song, C.; Shmatikov, V.: Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on Security and Privacy (SP). IEEE, pp. 3–18, 2017, doi: 10.1109/SP.2017.41.

[SK19] Shorten, C.; Khoshgoftaar, T. M.: A survey on image data augmentation for deep learning. Journal of Big Data 6 (1), pp. 1–48, 2019, doi: 10.1186/s40537-019-0197-0, url: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0.

[SZ15] Simonyan, K.; Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations (ICLR). Computational and Biological Learning Society, pp. 1–14, 2015.

[Sz15] Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. Pp. 1–9, 2015.

[TL19] Tan, M.; Le, Q.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In (Chaudhuri, K.; Salakhutdinov, R., eds.): Proceedings of the 36th International Conference on Machine Learning. Vol. 97. Proceedings of Machine Learning Research, PMLR, pp. 6105–6114, 2019, url: https://proceedings.mlr.press/v97/tan19a.html.

[TZ25] Tian, Y.; Zhang, L.: YOLOv12: Object Detection Meets Attention. arXiv preprint arXiv:2502.12345, 2025.

[Ul23] Ultralytics: Ultralytics YOLOv8, 2023, url: https://github.com/ultralytics/ultralytics.

[Ul24] Ultralytics: Ultralytics YOLOv11 Documentation, https://docs.ultralytics.com/models/yolo11/, Accessed: 2025-04-30, 2024.

[VV17] Voigt, P.; Von dem Bussche, A.: The EU General Data Protection Regulation (GDPR): A Practical Guide. Springer International Publishing, 2017.

[Wa20a] Wang, H.; Yurochkin, M.; Sun, Y.; Papailiopoulos, D.; Khazaeni, Y.: Federated Learning with Matched Averaging. In: International Conference on Learning Representations. 2020, url: https://openreview.net/forum?id=BkluqlSFDS.

[Wa20b] Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; Poor, H. V.: Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. Vol. 34, 2020.

[Wa21] Wan, J.; Li, X.; Dai, H.-N.; Kusiak, A.; Martínez-García, M.; Li, D.: Artificial-Intelligence-Driven Customized Manufacturing Factory: Key Technologies, Applications, and Challenges. Proceedings of the IEEE 109 (4), pp. 377–394, 2021, doi: 10.1109/JPROC.2021.3052047, url: https://ieeexplore.ieee.org/document/9266587.

[WBL20] Wang, C.-Y.; Bochkovskiy, A.; Liao, H.-Y. M.: CSPNet: A New Backbone That Can Enhance Learning Capability of CNN. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020.

[We23] Wen, J.; Zhang, Z.; Lan, Y.; Yang, F.; Xiang, Y.; Zhou, W.: A survey on federated learning: challenges and applications. International Journal of Machine Learning and Cybernetics 14 (2), pp. 513–535, 2023, doi: 10.1007/s13042-022-01647-y, url: https://link.springer.com/article/10.1007/s13042-022-01647-y.

[Ya19] Yang, Q.; Liu, Y.; Chen, T.; Tong, Y.: Federated Machine Learning: Concept and Applications. ACM Transactions on Intelligent Systems and Technology (TIST) 10 (2), pp. 1–19, 2019, doi: 10.1145/3298981.

[Ya24] Yao, Y.; Zhang, J.; Wu, J.; Huang, C.; Xia, Y.; Yu, T.; Zhang, R.; Kim, S.; Rossi, R.; Li, A.; Yao, L.; McAuley, J.; Chen, Y.; Joe-Wong, C.: Federated Large Language Models: Current Progress and Future Directions. arXiv preprint arXiv:2409.15723, 2024, url: https://arxiv.org/abs/2409.15723.

[Zh21] Zhou, J.; Zhang, S.; Lu, Q.; Dai, W.; Chen, M.; Liu, X.; Pirttikangas, S.; Shi, Y.; Zhang, W.; Herrera-Viedma, E.: A Survey on Federated Learning and its Applications for Accelerating Industrial Internet of Things. arXiv preprint arXiv:2104.10501, 2021.

Own Publications

[An24a] Antony, J.; Hegiste, V.; Nazeri, A.; Tavakoli, H.; Walunj, S.; Plociennik, C.; Ruskowski, M.: Enhancing Object Detection Performance for Small Objects through Synthetic Data Generation and Proportional Class-Balancing Technique: A Comparative Study in Industrial Scenarios. In: Advanced Machine Learning Applications in Industrial Settings. Springer, pp. 155–172, 2024, doi: 10.1007/978-3-031-57496-2_10.

[An24b] Anwar, A.; Moser, B.; Herurkar, D.; Raue, F.; Hegiste, V.; Legler, T.; Dengel, A.: FedAD-Bench: A Unified Benchmark for Federated Unsupervised Anomaly Detection in Tabular Data. In: 2024 2nd International Conference on Federated Learning Technologies and Applications (FLTA). Pp. 115–122, 2024, doi: 10.1109/FLTA63145.2024.10839838.

[Fr24] Fridman, K.; Grotepass, J.; Legler, T.; Hegiste, V.: Federated Learning for Shared Production Scenarios. In (Azevedo, A.; Gonçalves, R.; Stark, R., eds.): Advances in Production Management Systems. Resilient and Sustainable Industry 5.0 Systems. CRC Press, Taylor & Francis Group, 2024, doi: 10.1201/9781032690711-3.

[He23] Hegiste, V.; Legler, T.; Fridman, K.; Ruskowski, M.: Federated Object Detection for Quality Inspection in Shared Production. In: 2023 Eighth International Conference on Fog and Mobile Edge Computing (FMEC). Pp. 151–158, 2023, doi: 10.1109/FMEC59375.2023.10305969.

[He24] Hegiste, V.; Walunj, S.; Antony, J.; Legler, T.; Ruskowski, M.: Enhancing Object Detection with Hybrid dataset in Manufacturing Environments: Comparing Federated Learning to Conventional Techniques. In: 2024 1st International Conference on Innovative Engineering Sciences and Technological Research (ICIESTR). Pp. 1–6, 2024, doi: 10.1109/ICIESTR60916.2024.10798269.

[HLR22] Hegiste, V.; Legler, T.; Ruskowski, M.: Application of Federated Machine Learning in Manufacturing. In: 2022 International Conference on Industry 4.0 Technology (I4Tech). Pp. 1–8, 2022, doi: 10.1109/I4Tech55392.2022.9952385.

[HLR23] Hegiste, V.; Legler, T.; Ruskowski, M.: Federated Ensemble YOLOv5 – A Better Generalized Object Detection Algorithm. In: 2023 Eighth International Conference on Fog and Mobile Edge Computing (FMEC). Pp. 7–14, 2023, doi: 10.1109/FMEC59375.2023.10305958.

[HLR24] Hegiste, V.; Legler, T.; Ruskowski, M.: Towards Robust Federated Image Classification: An Empirical Study of Weight Selection Strategies in Manufacturing. In: 2024 2nd International Conference on Federated Learning Technologies and Applications (FLTA). Pp. 55–62, 2024, doi: 10.1109/FLTA63145.2024.10839986.

[HLR25a] Hegiste, V.; Legler, T.; Ruskowski, M.: Collaborative Learning in Shared Production Environment Using Federated Image Classification. In (Alexopoulos, K.; Makris, S.; Stavropoulos, P., eds.): Advances in Artificial Intelligence in Manufacturing II. ESAIM 2024. Lecture Notes in Mechanical Engineering, Springer, Cham, pp. 98–106, 2025, doi: 10.1007/978-3-031-86489-6_11.

[HLR25b] Hegiste, V.; Legler, T.; Ruskowski, M.: Trade-Off Between Communication Rounds and Local Epochs in Federated Learning. In: Proceedings of the 45th IEEE International Conference on Distributed Computing Systems (ICDCS). Presented, IEEE, 2025.

[Le25] Legler, T.; Hegiste, V.; Anwar, A.; Ruskowski, M.: Addressing Heterogeneity in Federated Learning: Challenges and Solutions for a Shared Production Environment. Procedia Computer Science 253, 6th International Conference on Industry 4.0 and Smart Manufacturing, pp. 2831–2840, 2025, issn: 1877-0509, doi: 10.1016/j.procs.2025.02.007.

[LHR24] Legler, T.; Hegiste, V.; Ruskowski, M.: Mapping of Newcomer Clients in Federated Learning Based on Activation Strength. In: Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems. International Conference on Flexible Automation and Intelligent Manufacturing (FAIM). Springer Nature Switzerland, pp. 1139–1148, 2024, doi: 10.1007/978-3-031-38165-2_130.

[LHR25] Legler, T.; Hegiste, V.; Ruskowski, M.: Multifaceted Applications of Federated Learning: Beyond Neural Networks. In (Alexopoulos, K.; Makris, S.; Stavropoulos, P., eds.): Advances in Artificial Intelligence in Manufacturing II. ESAIM 2024. Lecture Notes in Mechanical Engineering, Springer, Cham, 2025, doi: 10.1007/978-3-031-86489-6_28.

Curriculum Vitae

Personal Information

Name:	Vinit Vikas Hegiste
Nationality:	German

Education

since 05.2021	Doctoral student, Chair of Machine Tools and Control Systems, Rheinland-Pfälzische Technische Universität (RPTU) Kaiserslautern-Landau
10.2018 - 03.2021	Masters in Computer Science, Saarland University
06.2014 - 05.2018	Bachelor in Electronics and Telecommunication, Mumbai University
06.2012 - 03.2014	Higher Secondary School Education, Mumbai

Work Experience

since 05.2021	Researcher, Chair of Machine Tools and Control Systems, RPTU Kaiserslautern-Landau
04.2019 - 03.2021	Student assistant at German Research Center for Artificial Intelligence (DFKI), Saarbrücken
06.2018 - 09.2018	Business Development Executive, Uber India
05.2017 - 07.2017	Student Intership, Bhabha Atomic Research Center, Trombay, India