Degree project

UNIVERSITY OF SKÖVDE

A Comparative Study of Deep Learning Techniques for Automated Anomaly Detection in Battery Assembly Inspection

Master Degree Project in Comparing Deep Learning Methods for Battery Assembly Anomaly Detection

Second Cycle 30 credits

Spring semester 2025

Student: Harsh Panchal

Supervisor: Richard Senington

Examiner: Masood Fathi

Abstract

This report studies automated visual inspection for battery assembly using three deep learning methods with different strengths: a supervised object detector (YOLOv7), a reconstruction-based autoencoder, and EfficientAD. A common pipeline was used to ensure a fair comparison. All images were resized to 256 × 256, normalised, and augmented using brightness changes, flips, colour variations, and mild geometric transforms. Performance was evaluated at the image level using precision, recall, F1 score, and ROC–AUC, supported by confusion matrices and visual heatmaps. Stress tests were also performed under changes in background, lighting, and cell orientation.

The results show that YOLOv7 is effective for enforcing clear assembly rules such as component presence, count, and left–right terminal polarity, making it suitable for stop or rework decisions. The autoencoder provides clear heatmaps that highlight missing or extra structures and is useful for monitoring data drift, but it is sensitive to background and pose changes. EfficientAD achieves the best overall anomaly detection performance and robustly highlights both texture anomalies and spatial deviations.

Although EfficientAD performs best in anomaly detection, it does not produce bounding-box outputs needed for explicit rule checking. Based on this, a layered inspection strategy is proposed: YOLOv7 is used for rule-based checks, EfficientAD acts as a high-sensitivity anomaly filter, and a lightweight autoencoder monitors data drift over time. This approach improves reliability and practical deployment in real battery production lines.

Certification
and
Declaration on the Use of Artificial Intelligence (AI) Tools during the degree project

This report has been submitted by *Harsh Panchal* to the University of Skövde as a requirement for the degree of Bachelor of Science in Intelligent Automation.

The undersigned certifies that all the material in this report that is not my own has been properly acknowledged using accepted referencing practices and, further, that the report includes no material for which I have previously received academic credit.

As part of my commitment to academic integrity and transparency, I hereby declare the extent to which I have used Artificial Intelligence (AI) tools during the preparation and writing of this report. This declaration does not apply to the use of basic tools for checking grammar, spelling, references etc.

Please check all that apply:

☐ **No use of AI tools**
I have not used any AI-based tools or services in any part of the research, writing, analysis, or presentation of this report.

☑ **Language editing assistance only**
I have used AI tools exclusively for language correction, grammar checking, or stylistic improvements. No substantive content was generated or modified by AI.

☐ **Idea generation / brainstorming**
I used AI tools to help generate initial ideas, brainstorm possible research questions, or explore conceptual frameworks. The final work and conclusions are my own.

☐ **Methodology design**
I used AI tools to assist in designing the research methodology (e.g., structuring experiments, surveys, or models). The methodology has been critically reviewed and adapted by me.

☐ **Programming or code assistance**
I used AI-based coding assistants to help write or debug code. All code has been reviewed, tested, and modified by me.

☐ **Data analysis / processing support**
I used AI tools to assist with data analysis, visualization, or statistical processing. Interpretations and conclusions were drawn independently.

☐ **Content generation (text, figures, tables)**
AI tools were used to generate portions of text, diagrams, or tables that appear in this report. Such content has been critically reviewed, fact-checked, and appropriately cited where applicable.

☐ **Other (please specify):**

Responsibility Statement
I confirm that all AI-assisted contributions have been used ethically and in accordance with institutional guidelines. I take full responsibility for the accuracy, originality, and academic integrity of the content presented in this report.

I understand that any misuse or misrepresentation of AI tool usage may be considered a breach of academic conduct and could result in disciplinary actions, including revocation of the report and/or the academic degree awarded.

Signature: 

Name: Harsh Panchal

Date: 19/12/2025

Table of Contents

Table of Figures

Table of Tables

1 Introduction

In modern manufacturing, quality control is essential to ensure that every product meets the required standards before it is shipped. This project focuses on developing and comparing different AI methods to perform visual anomaly detection. The aim is to create a system that can reliably spot mistakes in an automated assembly process. During the literature review, methods that show promising performance will be identified, along with those most suitable for conditions similar to those in this project. Each approach uses deep learning to learn what a correctly assembled battery cell looks like, and then detect deviations from that expected appearance.

Traditional automated quality control systems often rely on fixed rules or predefined thresholds, which can struggle with natural variation in production. Minor misalignments, lighting differences or subtle changes in component appearance can lead to defective products being missed or, conversely, cause false alarms. These limitations increase the risk of defective items passing through and slow down production due to unnecessary manual checks (Zubayer et al., 2024).

Automation in industry has long relied on visual inspection, historically performed by trained human operators. Manual inspection, however, is inconsistent and resource intensive, particularly at scale. As manufacturing systems become more connected and data driven, inspection is no longer limited to final quality checks. Real time monitoring, predictive maintenance and continuous process feedback all contribute to more reliable production, and all rely on effective anomaly detection to identify unexpected or undesirable behaviour within the system (Faust et al., 2020). When these elements function well, they improve reliability and reduce both downtime and waste, making anomaly detection a central component of modern production environments.

The goal of this project is to compare the strengths and weaknesses of three important classes of AI methods in the context of this specific quality control task. The comparison aims to identify which approach is most suitable for fast and accurate detection of assembly errors in battery cells. The results have the potential to support higher production quality and reduce the cost and interruption associated with manual inspection or faulty products.

1.1 Overview of Industrial Anomaly Detection and Inspection Challenges

Industrial anomaly detection focuses on identifying unusual patterns or defects in manufactured products using computer vision. These defects may include scratches, missing components or small misalignments, all of which can influence product performance and reliability. Effective detection is important because undetected

defects can increase production costs, reduce safety and lead to customer dissatisfaction.

Several factors make anomaly detection challenging in real manufacturing environments. Defects occur infrequently, which limits the amount of faulty data available for model development. Industrial images also contain variation in lighting, reflections and background structure, which makes it harder to separate normal variation from true defects. Some anomalies are subtle and resemble acceptable differences in appearance, and many production lines require real time inspection to avoid interruptions in workflow.

Traditional rule-based inspection methods often struggle with these conditions because they rely on fixed thresholds and handcrafted features. Deep learning approaches have gained importance as they can learn useful visual representations directly from data. This improves detection accuracy and reduces manual effort, making deep learning-based inspection a practical and effective choice for modern industrial quality control systems (Liu et al., 2024).

1.2 Risks Involved in Battery Manufacturing and the Need for Reliable Detection

Battery manufacturing carries significant safety risks, particularly when small defects are not detected during production. Even minor issues such as particle contamination, misaligned components or poor welds can develop into serious failures later in the battery's life. Research highlights that early and accurate detection is essential to prevent defective cells from progressing through the production chain, reducing the likelihood of latent faults (Choudhary et al., 2022). Other studies note that anomalies left unidentified during early stages can evolve into hazardous conditions, including thermal runaway, which may cause permanent damage or safety incidents (Sun et al., 2023).

AI based inspection methods are increasingly used to identify early-stage defects, but missed anomalies remain a significant concern. Micro level internal short circuits, for example, may begin with very subtle signs and later develop into heat generation, separator damage and electrolyte decomposition, potentially leading to thermal runaway if not detected in time (Wang et al., 2024).

Reviews comparing laboratory performance to real world deployment show that data scarcity, distribution shifts, sensor noise and class imbalance can all contribute to missed detections. These limitations reduce the safety margin and allow incipient faults to propagate into modules and packs (Stavropoulos et al., 2024). On manufacturing lines, failure of visual or signal based inspection systems can allow inclusions or weak welds

to pass through, embedding defects that increase the risk of field failures and costly product recalls (Stavropoulos et al., 2024; Grabow et al., 2023).

These risks operate at two levels. First, there is an immediate safety risk if defects lead to failure in individual cells or modules. Second, undetected faults can be replicated across many units, affecting entire product fleets and creating large scale reliability and safety issues. Reliable anomaly detection during assembly therefore plays a central role in ensuring safety and preventing long term failures.

1.3 Background of Deep Learning and Anomaly Detection

Deep learning refers to a family of machine learning techniques that use multilayer neural networks to learn patterns directly from data. Instead of relying on handcrafted rules, these models learn to recognise relevant features automatically, which makes them well suited to complex visual tasks.

Anomaly detection focuses on identifying cases that deviate from what is considered normal. In manufacturing, an anomaly is typically a defect, an unexpected configuration or a missing component. Deep learning-based anomaly detection systems learn what a correctly assembled object should look like, then highlight any differences that fall outside the learned pattern.

YOLO (You Only Look Once) represents a class of deep learning models designed for real time object detection. These models detect and classify objects by predicting bounding boxes and class probabilities in a single forward pass. YOLO is highly effective for tasks where the goal is to locate and label objects in an image.

Anomaly detection methods differ in purpose and structure. Instead of detecting known object classes, they attempt to model normality. The idea is that deviations from the learned normal pattern signal potential defects. While YOLO can be adapted for defect detection by training it to recognise specific types of error, anomaly detection approaches are often more flexible, particularly when defects are rare or vary widely.

This distinction underpins the comparison carried out in this project: a traditional object detection framework like YOLO, versus techniques designed specifically to identify abnormal visual patterns.

1.4 Problem Description

Batteries rely on correct electrical configuration to operate safely and efficiently. Each cell has two colour coded terminals that must be positioned on specific sides to ensure the battery functions as intended. If a terminal is misplaced, reversed or missing, the assembled battery can fail or become unsafe to use. In automated production, such mistakes can occur during placement, alignment or handling, which makes reliable inspection essential.

The task in this thesis is to design an AI based vision system that can detect these assembly errors. The system analyses images taken immediately after the cells are placed into the battery case and checks whether the terminals appear on the correct

Figure 1: Normal Battery Assembly Placement

sides. Any deviation from the expected configuration is treated as a defect, and the system must flag it accurately and consistently.

In the industrial setup used for this project, a robot assembles battery cases designed to hold four individual cells. After assembly, the vision system captures an image of the completed case, processes the positions of the white and red terminals and determines whether the assembly meets the required standard. If a defect is found, the system sends feedback indicating that a rework is needed. Although the vision system interacts with the robot, the project does not involve controlling or programming the robot.

The experiments were conducted at ASSAR Industrial Innovation Arena, a research and development centre in Skövde, Sweden, focused on smart manufacturing technologies. The real-world environment at ASSAR provides the conditions under which the proposed inspection system is evaluated, ensuring that the results reflect practical industrial use.

Figure 2: Misplaced Battery Cells

1.5 Aim and Research Questions

The aim of this project is to evaluate and compare different AI methods for visual anomaly detection in the battery cell assembly process at ASSAR. By determining which approach is most effective for detecting assembly errors, the project seeks to ensure that only products meeting strict quality standards reach the market. In doing so, the findings will provide valuable guidance for optimizing manufacturing processes, making them more reliable and cost-effective, and ultimately reducing downtime and manual inspection costs.

By comparing different AI methods, this project aims to answer the following research questions

- **Among YOLOv7 (supervised), Autoencoder and EfficientAD (unsupervised), which deep learning approach is most effective for this specific quality control task**?
- **What are the training requirements of each method, including training time and the level of effort needed to achieve reliable results?**
- **How can the best-performing approach be integrated into an automated quality control pipeline, and to what extent is this integration pattern generalizable to similar assembly lines beyond ASSAR?**

The findings will help guide future improvements in manufacturing processes, making them more reliable and cost-effective, and ultimately ensuring that only products meeting the strict quality standards reach the market.

1.6 Position in VPR (Virtual Product Realization)

This master's thesis is conducted within the research profile Virtual Product realization (VPR) at the University of Skövde. VPR focuses on the development of virtual methods,

models, and tools that enable efficient product and production realization through digitalization, simulation, and automation. The present study contributes to this profile by applying deep learning–based computer-vision techniques to quality assurance in automated battery assembly.

The work strengthens the link between intelligent automation and virtual validation, showing how AI-driven inspection can improve accuracy and reduce waste in production systems. By combining industrial case data from ASSAR with state-of-the-art machine-learning approaches, the research supports VPR's goal of advancing data-centric methods that bridge virtual modeling and real manufacturing. Consequently, the thesis reinforces the VPR vision of sustainable, knowledge-based industrial development through virtual product realization.

1.7 Structure of the Thesis

This thesis has six main chapters. Each chapter focuses on one part of the study.

Chapter 1 explains the background of the research and why automated anomaly detection in battery assembly is important. It describes problems in industry, safety risks in battery production, and how deep learning helps with visual inspection. It also states the problem, research goals, research questions, and explains how the work fits into the Virtual Product Realisation research area.

Chapter 2 reviews earlier research on anomaly detection used in industrial inspection. It first explains older rule-based and classical machine learning methods. Then it describes deep learning methods such as YOLO object detection, autoencoders, and EfficientAD. It also looks at research done specifically on battery manufacturing inspection and points out what is still missing in current studies.

Chapter 3 explains how the experiments were done. It describes how YOLOv7, an autoencoder, and EfficientAD were tested under the same conditions. The chapter explains how the dataset was prepared, how preprocessing was done, how the models were built, how they were trained, and which hyperparameters were used. This is to make the comparison fair and reproducible.

Chapter 4 presents the results of the experiments. It shows the performance of all three methods using measures such as precision, recall, F1 score, and ROC AUC, along with confusion matrices and score plots. It also shows visual examples like anomaly heatmaps and tests with different backgrounds, lighting, and orientations.

Chapter 5 discusses what the results mean. It compares what each method does well and where each one fails, especially in real production situations. It talks about threshold selection, calibration, and how these methods could be used in practice. It also considers ethical and sustainability issues when using AI for battery inspection.

Chapter 6 ends the thesis. It summarises the main results and contributions. It gives suggestions for how inspection systems can be used in real industry and proposes ideas for future work, such as hybrid models, better robustness, and real-time use in factories.

2 Literature Review

This section reviews a range of deep learning methods that have been applied to anomaly detection in industrial settings. Industrial anomaly detection plays an important role in maintaining product quality, ensuring safety and reducing downtime, but real-world datasets often contain very few examples of faults. Deep learning approaches respond to this challenge in different ways. Methods such as YOLO require labelled examples of defects and can be affected by limited training data, while reconstruction-based approaches like autoencoders and generative models can operate with fewer or even no labelled anomalies by learning a representation of normal data. The following subsections discuss these approaches in more detail and compare their strengths and limitations.

2.1 Traditional Approaches for Anomaly Detection

Before the development of deep learning, automated anomaly detection in industrial inspection depended largely on statistical models and classical machine-learning methods that used hand-crafted features. Early statistical approaches modelled the distribution of normal samples and flagged observations that deviated from that distribution. In image-based inspection, this typically involved extracting simple descriptors such as colour histograms, texture vectors or edge information, then fitting models such as Gaussian mixture models (GMMs) or kernel density estimators (KDEs). Images with features that had a low probability under the learned distribution were treated as potential defects. These methods were effective only when normal data followed predictable patterns and when defects produced large deviations in these manually designed features.

Classical machine-learning approaches relied on similar feature extraction followed by anomaly classifiers. Common choices included one-class Support Vector Machines (OC-SVM), k-means clustering and Local Outlier Factor (LOF). OC-SVMs attempted to learn a tight boundary around normal feature vectors and flagged points outside this boundary as anomalies. LOF compared the local density of a sample to that of its neighbours, identifying samples in sparse regions as outliers. These algorithms performed reasonably well when the feature design captured the relevant characteristics of the product. For example, surface inspection pipelines often used Laws filters or Gabor filters to highlight texture irregularities, colour histograms to detect stains or discolouration, and simple geometric descriptors to identify cracks or shape deformities.

Song et al. (2019) demonstrated that this strategy could work for wood defect detection using texture and colour descriptors, and Wang et al. (2020) applied Fourier-transform-based features to identify defects in magnetic materials. Although these pipelines

worked on narrow, well-defined problems, they often required extensive manual tuning to handle variations in lighting, material appearance or camera conditions.

A practical illustration of these limitations appears in lithium-battery electrode inspection. Xu et al. (2021) combined multi-feature image fusion with a particle-swarm-optimised SVM classifier to identify coating defects on electrode sheets. While the method detected certain surface flaws, it depended heavily on hand-engineered features and struggled to capture all the subtle variations that occur in real electrode manufacturing. As defect variability increased, the handcrafted descriptors failed to generalise, leading to missed detections and reduced robustness.

These issues were common across traditional methods. Their performance depended on the quality of the extracted features and on assumptions such as approximate Gaussian distributions or simple boundary shapes in feature space. As products became more visually complex and industrial images grew higher in resolution, these manual feature pipelines struggled to scale. They lacked the flexibility to capture intricate patterns, subtle texture changes or global layout irregularities, all of which are important in modern manufacturing.

Given these limitations, feature-learning approaches became increasingly attractive. Deep learning methods do not rely on manual feature design. Instead, they learn hierarchical representations directly from data, enabling more robust modelling of complex visual patterns and laying the foundation for the more advanced anomaly-detection techniques reviewed in the following sections.

2.2 Deep Learning Approaches for Anomaly Detection

2.2.1 YOLO-Based Models for Visual Anomaly Detection

Object detection models in the YOLO (You Only Look Once) family have been widely adopted for detecting visible defects in manufacturing. These models frame anomaly detection as an object detection problem, identifying defects by bounding boxes and class labels in real time.

YOLO Network Architecture

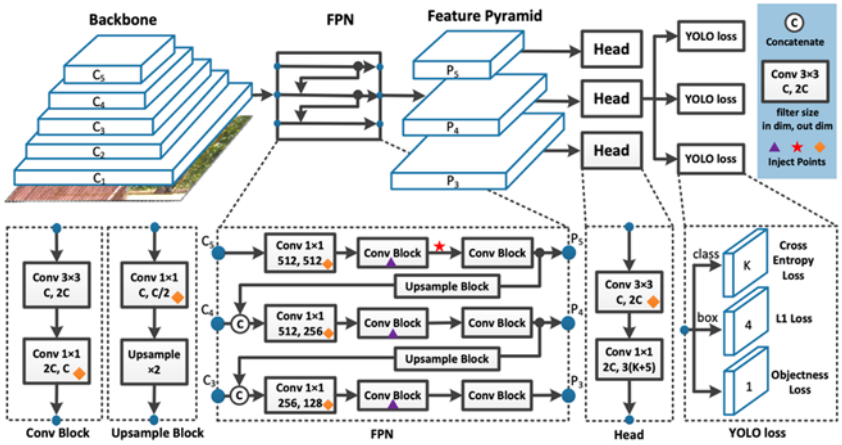

Figure 3: YOLOv7 Network Architecture (Source: Solawetz, 2024. Roboflow Blog)

The YOLOv7 architecture used in this study follows a standard three-stage design: backbone, neck, and detection head, tailored for real-time multi-scale object detection. This configuration corresponds to the canonical YOLOv7 model variant and is well-matched to the 256 × 256 input size.

YOLOv3, introduced by Redmon and Farhadi (2018), became a strong baseline due to its multi-scale predictions and impressive speed. In recent years, modified YOLOv3 models have been applied in industrial inspection contexts. For instance, Kou et al. (2021) developed an improved YOLOv3 model using dense feature extraction to detect steel surface defects, achieving 71–72% mean Average Precision (mAP) on datasets like NEU-DET and GC10-DET.

With the release of YOLOv4, Bochkovskiy, Wang, and Liao (2020) introduced improvements such as CSPDarknet as the backbone and techniques like mosaic data augmentation. These enhancements led to better accuracy without a significant drop in inference speed. YOLOv4 variants have been customized for more refined detection. For example, Wang, Zhou, and Chen (2022) incorporated attention-based feature fusion into YOLOv4 to enhance small defect detection on metal surfaces.

Ultralytics released YOLOv5 in 2020, which quickly became popular in industrial applications due to its modular design and excellent performance. Mehta and Klarmann (2023) demonstrated that YOLOv5 could achieve over 90% precision in surface inspection tasks at high frame rates. Researchers have also enhanced YOLOv5 by integrating BiFPN feature pyramids, significantly improving fault detection in printed circuit boards (PCBs), as reported by various authors e.g., (Zhang, Jia, & Yu, 2024).

YOLOv6, designed specifically for industrial applications, was presented by Nie et al. (2022). The model offered enhancements for single-stage detection frameworks,

contributing to better trade-offs between speed and accuracy. Meanwhile, YOLOv7, introduced by Wang, Bochkovskiy, and Liao (2023), achieved state-of-the-art performance for real-time detectors. Zhang et al. (2024) customized a lightweight YOLOv7-tiny version for fabric defect detection, which maintained high precision despite reduced computational requirements.

More recently, YOLOv8, also from Ultralytics, introduced several architectural improvements such as the C2f module and transformer-based blocks. Liu et al. (2025) demonstrated that their lightweight "SLF-YOLO" model, based on YOLOv8, outperformed the standard YOLOv8 by boosting mAP from 75.9% to 80.0% on the NEU-DET dataset. In another study, Ren, Zhang, and Yue (2023) reported that their attention-augmented YOLOv8 model achieved more than 97% mAP for detecting weld defects in automotive components, outperforming the YOLOv8-L baseline.

These findings indicate that YOLO models, particularly the recent versions, offer a powerful and scalable solution for visual anomaly detection in industrial quality control systems. Their high detection accuracy, fast inference, and adaptability to lightweight deployment make them ideal for manufacturing environments where defects must be located in real time with limited hardware.

2.2.2 Autoencoder-Based Approaches for Anomaly Detection

An autoencoder is a deep learning model designed to learn compact and meaningful representations of data by reconstructing its input. It consists of two main components: an encoder that compresses the input into a lower dimensional latent space, and a decoder that attempts to reconstruct the original data from this compressed representation. During training, the model minimizes the reconstruction error, which encourages the encoder to capture the most informative patterns in the data. In anomaly detection, autoencoders are useful because they learn the normal structure of the data; when an abnormal or unseen pattern is presented, the reconstruction error typically increases, allowing the system to identify deviations from expected behaviour.

Autoencoder Network Architecture

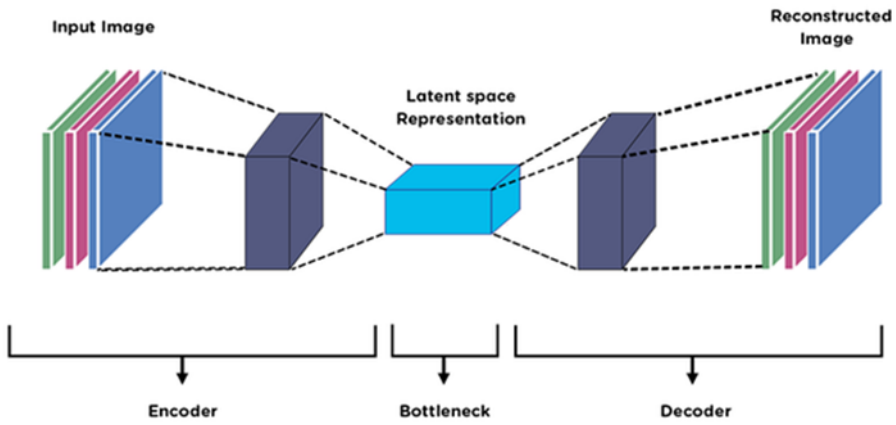

Figure 4: Autoencoder Network Architecture (Source: Medium, 2019)

A compact convolutional autoencoder was employed to balance capacity and overfitting risk given the dataset size and 256 × 256 resolution. The encoder consisted of four convolutional blocks:

Conv(3→32, 3×3, stride 2) + BatchNorm + ReLU
Conv(32→64, 3×3, stride 2) + BatchNorm + ReLU
Conv(64→128, 3×3, stride 2) + BatchNorm + ReLU
Conv(128→256, 3×3, stride 2) + BatchNorm + ReLU

This produced a 256 × 16 × 16 latent feature map. A final Conv(256→256, 3×3, stride 1) with ReLU formed the bottleneck. The decoder mirrored the encoder using ConvTranspose layers:

ConvT(256→128, 4×4, stride 2) + BatchNorm + ReLU
ConvT(128→64, 4×4, stride 2) + BatchNorm + ReLU
ConvT(64→32, 4×4, stride 2) + BatchNorm + ReLU
ConvT(32→3, 4×4, stride 2)

followed by Sigmoid to reconstruct the [0, 1] image. Skip connections were not used to avoid trivially passing through fine-grained defect cues. The model contained approximately 2.8 million parameters, which fit comfortably in 6 GB VRAM with the chosen batch size.

Autoencoders (AEs) provide an unsupervised learning approach to anomaly detection by learning to reconstruct normal data patterns. In manufacturing inspection, an autoencoder is typically trained on defect-free images so that it can accurately reproduce normal appearances; anomalies are then flagged when the reconstruction error is high in a region, indicating something is missing or distorted in the autoencoder's output (Mehta & Klarmann, 2023). This strategy is popular for detecting subtle flaws on

product surfaces where anomalous regions cause visible reconstruction discrepancies. Many studies report that autoencoder-based methods can reliably detect and even localize unknown defects without requiring annotated anomalies (Zavrtanik et al., 2021; Mujkić et al., 2022).

Over the years, numerous autoencoder variants have been explored to improve anomaly detection performance:

Variational Autoencoders (VAEs): VAEs introduce a probabilistic latent space, training the model to encode inputs into a distribution (often Gaussian) and then decode from samples of that distribution. This encourages the model to learn a compact distribution of normal features, making it easier to identify out-of-distribution anomalies. Yapp et al. (2024) applied a two-level vector-quantized VAE (VQ-VAE-2) to the MVTec AD manufacturing dataset, achieving high accuracy in detecting defects by better reconstructing fine details of normal images. Other extensions like memory-augmented VAEs and Bayesian VAEs have also emerged post-2020, aiming to enhance the model's capacity to represent normal variability while still amplifying reconstruction errors for novel defects (Neloy & Turgeon, 2024).

Sparse Autoencoders: Sparse AEs impose a sparsity penalty on the hidden units or features, forcing the model to learn a compressed representation that uses as few active neurons as possible. This helps ensure the autoencoder captures only key normal patterns. Wei et al. (2023) combined a sparse autoencoder with a one-class SVM, encoding data into a low-dimensional space and separating anomalies in that space. This hybrid method improved detection precision on complex high-dimensional datasets.

Denoising Autoencoders (DAEs): DAEs are trained to remove noise or occlusions from input images, which can make them especially robust for anomaly detection. Kascenas et al. (2021) showed that a simple DAE with carefully chosen noise injection can better localize subtle anomalies than more complex VAEs. Mehta and Klarmann (2023) reported that using DAEs with structural similarity (SSIM) loss proved effective, helping the model better preserve normal texture and flag even small defects. Replacing a basic convolutional autoencoder with a denoising variant yielded improved anomaly localization in a recent manufacturing study. Zavrtanik et al. (2021) developed the DRAEM approach, which couples a denoising reconstruction network with a discriminative network. DRAEM achieved near fully-supervised anomaly detection accuracy on the MVTec AD benchmark by training on simulated defects and was able to precisely segment industrial surface anomalies.

Notably, autoencoder methods are often used in unsupervised or semi-supervised settings, making them attractive when only normal samples are available for training. Researchers have reported that autoencoder-based detectors can rival the performance of supervised detection in some cases. For example, Mujkić et al. (2022) found that a

denoising autoencoder and a variational autoencoder could detect agricultural equipment faults nearly as well as a YOLOv5 detector, while operating without labeled anomaly examples. These findings demonstrate that autoencoders, especially when enhanced with domain-specific constraints or augmentations, can effectively model normal appearance and thereby highlight abnormal regions on parts and products.

2.2.3 EfficientAD:

EfficientAD is a recent visual anomaly-detection method proposed by Batzner et al. (2024). It builds on the student–teacher framework that has become common in industrial anomaly detection but introduces several changes that improve both accuracy and speed. The student–teacher principle is based on feature distillation: a large or well-trained "teacher" network produces feature maps for normal images, and a smaller "student" network is trained to reproduce these features. During deployment, if an input image differs from the normal training distribution, the student's predicted features will diverge from the teacher's features, creating an anomaly signal. This strategy has been used in several earlier works, including the asymmetric student–teacher model of Rudolph et al. (2023), but EfficientAD refines the idea with a focus on latency and robustness.

Network Architecture

EfficientAD uses a student-teacher framework with an additional lightweight autoencoder branch. The teacher is a frozen EfficientNet-based backbone that emits mid-level feature maps. A compact student network with matched channel dimensions is trained to regress the teacher features on normal data. The anomaly signal is the distance between normalized teacher and student features, aggregated into an anomaly map. The autoencoder branch captures global structural irregularities that might not be fully reflected in local feature distances. After quantile-based normalization, the student-teacher map and the autoencoder signal are fused to produce a robust image-level score. The reference design exposes practical knobs such as model size to trade accuracy and latency, channel widths that typically reach a few hundred channels in the deepest block, and padding for feature alignment to keep map dimensions consistent. This architecture achieves millisecond-level latency on modern GPUs while maintaining high accuracy on industrial anomaly benchmarks (Batzner, Heckler, and König, 2024).

EfficientAD employs a student–teacher pathway for local feature discrepancies together with a lightweight autoencoder for global structural regularities. The overall flow and fusion are summarized in below Figure (adapted from Batzner, Heckler, and König, 2024).

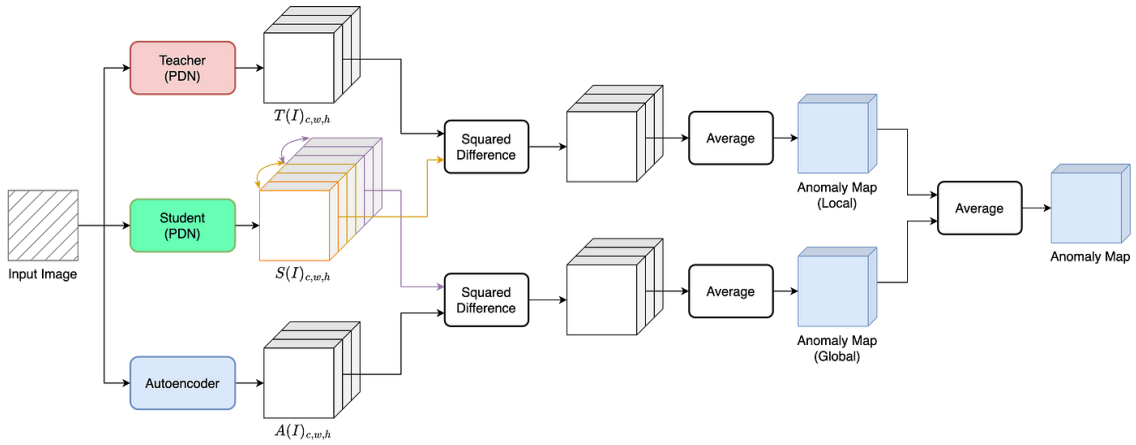

Figure 6: EfficientAD Network Architecture (Source: Medium, 2024)

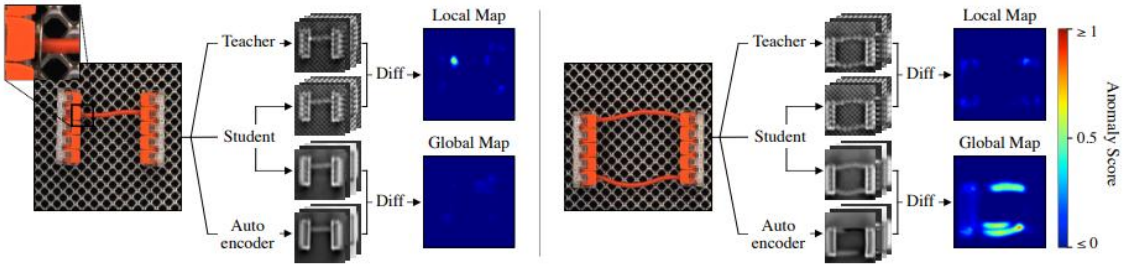

Figure 5: Illustration of local and global anomaly detection (Batzner, Heckler, and König, 2024)

To illustrate the distinction between local and global anomaly cues, Figure 6 shows an example product with heatmaps highlighting spatially confined defects (local) and broader structural deviations (global), following the presentation in EfficientAD (Batzner et al., 2024).

- **Lightweight Teacher Network:** In many prior feature-distillation methods, the teacher was a deep backbone such as ResNet or WideResNet, which produced accurate features but was slow to evaluate. EfficientAD instead uses a lightweight convolutional network as the teacher. The purpose of the teacher is to provide stable feature representations of normal images, not to perform the anomaly decision itself. By designing the teacher to be compact, Batzner et al. (2024) reduced inference time to under one millisecond per image on modern GPUs. This design follows developments in efficient CNN architectures such as MobileNetV3 (Howard et al., 2019) and EfficientNet-V2 (Tan & Le, 2021), which demonstrate that small models can still extract useful features while keeping latency very low.
- **Student Network with Loss-Induced Asymmetry:** The student network is trained to match the teacher's feature maps using only normal data. Earlier approaches

typically encouraged the student to reproduce the teacher as closely as possible. EfficientAD modifies this by introducing a loss term that prevents perfect imitation outside the normal data distribution. In practical terms, the student becomes a reliable predictor of normal patterns but remains sensitive to unfamiliar ones. When an anomalous image is given, the mismatch between student and teacher features increases sharply, resulting in a strong anomaly signal. Unlike Rudolph et al. (2023), who enforced asymmetry by structural differences between teacher and student, EfficientAD achieves it through the training loss, which allows both networks to remain lightweight.

- **Global Autoencoder for Logical Anomalies:** Some defects do not appear as local texture changes but as global layout problems, known as logical anomalies (Bergmann et al., 2022). For instance, an object may be assembled from correct parts placed in the wrong order. Local feature-based student–teacher methods often struggle with these cases because each individual region looks normal. EfficientAD adds a lightweight autoencoder that reconstructs the full image or a downsampled version of it. Since the autoencoder is trained only on correct assemblies, violations of global structure produce high reconstruction error. EfficientAD then fuses this global signal with the local student–teacher anomaly map, enabling it to detect both small, local defects and broader layout inconsistencies.

- **Performance and Advantages**

  EfficientAD demonstrates state-of-the-art performance across major industrial anomaly-detection benchmarks, including MVTec AD, MVTec LOCO and VisA (Batzner et al., 2024). It improves detection of subtle texture defects, global layout mistakes and logically inconsistent assemblies. A key advantage is its speed: the method processes an image in roughly 2 milliseconds, corresponding to approximately 600 images per second on an NVIDIA RTX A6000. Competing methods such as PatchCore or AST typically require 10–50 milliseconds per image, which makes EfficientAD an order of magnitude faster. This makes it suitable for real-time manufacturing inspection where each item has a strict time budget.

  Unlike many fast models, EfficientAD does not sacrifice accuracy for speed. It was the first method on MVTec AD to exceed a mean ROC-AUC of 95 percent while also achieving best-in-class segmentation performance. This combination of accuracy, robustness and low latency explains its relevance to industrial applications. In contexts such as battery assembly, where defects may be subtle and inspection systems must operate continuously at high throughput, EfficientAD provides a practical balance between performance and deployment cost.

2.3 Anomaly Detection in Battery Manufacturing

Manufacturing of lithium-ion batteries including cell production and battery pack assembly is a complex process where quality control is critical for safety. Failures in battery assembly (like internal cell defects or misassembled packs) can lead to overheating, fire, or reduced performance. Thus, researchers have actively applied deep learning to detect anomalies in battery production, from microscopic electrode flaws to pack-level faults.

The paper by Xing et al. (2024) focuses on detecting minimal defects on the end face of lithium battery shells using a YOLO-based approach named YOLO-MDD. The authors address the limitations of conventional inspection techniques, which often fail to identify small but critical defects such as scratches, bumps, troughs, and rust spots. Their method integrates deformable convolution and attention modules into YOLO's backbone, combined with a four-scale detection head and a hybrid loss function that includes normalized Wasserstein distance. Experimental results demonstrate that YOLO-MDD achieves a mean average precision of 80%, with particularly strong performance in identifying rust spots (74.1% mAP, 71.5% recall), outperforming mainstream algorithms like YOLOv5, YOLOv7, and Faster R-CNN.

This work is highly relevant to anomaly detection in batteries since early identification of such defects directly impacts product safety, reliability, and longevity. In the context of automated battery inspection, it provides valuable insights into designing deep-learning systems capable of handling small-scale anomalies with industrial applicability (Xing et al., 2024).

Battery Electrode and Cell Production: One stream of work focuses on detecting defects during battery cell manufacturing (e.g., coating electrodes with active material, assembling electrodes into cells). (Choudhary et al., 2022) developed an autonomous vision inspection system for electrode production. Using a YOLOv5 model, they detected common electrode defects such as material agglomerates, bubbles in the coating, foil exposure, and scratches. The system was trained on 882 images of electrodes (some containing these defects) and achieved an 88% mAP in identifying all defect types, with each image processed in ~9.5 milliseconds. This real-world result shows the feasibility of deploying object detection for inline inspection in battery factories, where speed and accuracy are needed to catch defects on a moving production line. The YOLO-based approach effectively replaced a purely manual visual check and could even be integrated into a web platform for remote monitoring.

Badmos et al. (2020) took a classification approach for battery electrode defects. They collected 3,286 images from a lithium-ion electrode production line, labeled for four defect categories. Various CNN models were trained and compared: a standard baseline CNN, versions with different output layers (Sigmoid vs. Softmax), and three deep

architectures (VGG19, InceptionV3, Xception). The VGG19 network performed best, with an F1-score of 0.99, indicating near-perfect detection of defective electrodes. InceptionV3 was second best (F1 ~0.97), while simpler models underperformed. This comparison highlights that in battery cell manufacturing, deeper CNNs pre-trained on large datasets (ImageNet) can be fine-tuned to achieve extremely high accuracy in defect detection. The authors suggested that VGG19's strong performance could make it a viable solution for automated optical inspection of electrodes. The trade-off is that such deep models may require significant computational resources, but given the high stakes (preventing faulty cells), the investment is justified.

Vu et al. (2024) took a more integrative approach in a battery component assembly context. They targeted the detection of scratches on microfasteners (tiny screws) used in battery packs. Scratches on fasteners might seem minor, but they can indicate tooling issues or potential assembly problems. Vu et al. applied and compared two state-of-the-art object detection models, EfficientDet and YOLOv8, under two inference strategies ("direct" vs. "indirect" prediction). By incorporating statistical quality control (SQC) techniques into their deep learning pipeline, they continually improved detection accuracy during deployment. On a dataset of 1,150 real images from a battery factory, their best model (using YOLOv8 with an indirect prediction method) achieved about 91% accuracy in classifying scratch defects. This study not only shows the applicability of advanced detectors like YOLOv8 to battery assembly components, but also emphasizes integrating AI with quality control processes. The indirect method (which involved an intermediate step, perhaps aggregating model outputs over time) significantly outperformed a direct one (mAP 0.84 vs 0.29 in one metric), illustrating how combining domain knowledge (SQC) with deep learning can yield robust results in manufacturing.

Battery Pack Assembly and Operation: At the battery pack level (where many cells are combined, e.g., in an electric vehicle battery pack), anomalies may not be visible externally but can be inferred from sensor data (voltage, temperature, etc.). Deep learning has been used to detect pack assembly faults or early failures during battery operation. Sun et al. (2023) focused on early fault detection in EV battery packs using a GRU-VAE model. They treated the battery management system (BMS) sensor readings over time as multivariate time series and trained a GRU-based variational autoencoder on normal operation data. During use, if the pack's sensor pattern deviated significantly (high reconstruction error), the model would flag an anomaly. This method proved capable of detecting subtle issues before they led to thermal runaway or fires, which is crucial because once a battery cell goes into thermal failure, it's often too late to prevent disaster. By applying extreme value theory to set thresholds, Sun et al. ensured a low false-alarm rate while still catching real faults early. They reported that GRU-VAE outperformed several state-of-the-art baselines in accuracy on real EV battery data. This kind of anomaly detection supports predictive maintenance identifying packs that need

service or have assembly flaws (like an improperly welded connection or an out-of-balance module) before they cause accidents.

In pack assembly plants, another type of anomaly is structural: e.g., cells or modules misaligned, or weld defects. X-ray or ultrasonic inspection combined with CNN analysis has been explored in industry (though specific recent papers are sparse due to proprietary data). Cai et al. (2022) as noted in a survey, used a gated recurrent unit-based sequence model for battery pack inconsistency detection, which is essentially anomaly detection on pack voltages. Similarly, researchers are investigating one-class classifiers and outlier detectors (like isolation forests or one-class SVMs) alongside deep neural networks for battery packs. Kongbantabam (2022) compared a GRU-VAE against classical algorithms on battery fleet data; interestingly, a non-deep method (k-NN outlier detection) performed nearly as well as the GRU-VAE, both surpassing other methods in detecting days with abnormal battery behaviour. This suggests that while deep learning generally improves detection, in some cases a well-tuned simpler algorithm can compete, especially when data are limited a reminder that algorithm selection should consider the specific use case and dataset.

2.4 Comparative Studies and Benchmarks

Raj and Prabadevi (2025) conducted one of the most comprehensive investigations into real-time object detectors for surface defect detection, focusing on steel strip datasets (NEU-DET and GC10-DET). By enhancing YOLOv5 with efficient channel attention (ECA) and coordinated attention (CoordAtt), they achieved a significant uplift in mean average precision 76.3% on NEU-DET, compared to 58.7% for YOLOv8 and demonstrated inference speeds 2–4 × faster than prior YOLO versions; the CoordAtt integration alone reduced parameter counts by 25% without sacrificing accuracy . Their hybrid YOLOv5 model also outperformed subsequent YOLOv6–v8 variants in head-to-head comparisons, underscoring the value of lightweight attention modules in constrained industrial settings .

In parallel, purely autoencoder-based approaches have long been explored for unsupervised anomaly detection, where models learn to reconstruct “normal” images so that defects manifest as high reconstruction error. Early work on GAN-augmented reconstruction, such as Skip-GANomaly (Akçay, Atapour-Abarghouei, & Breckon, 2019), improved upon AnoGAN by incorporating skip connections into the encoder–decoder backbone, though it still suffered from unstable convergence . Schlegl et al. (2019) introduced f-AnoGAN, which added a training stage to guide the generator toward high-fidelity reconstructions, yielding faster inference but limited performance on complex textures. In the automotive domain, Chen et al. (2025) applied a denoising reconstruction autoencoder (DRAEM) to glue-line defects, reporting substantial

reductions in false positives by leveraging reconstruction residuals to pinpoint subtle anomalies.

Generative adversarial networks (GANs) that combine autoencoder and discriminator architectures have further advanced industrial inspection. Tang et al. (2020) proposed DAGAN, a dual autoencoder GAN that stabilizes training via symmetric reconstruction pathways. Evaluated on MVTec AD and two real production-line datasets (mobile phone glass and wood), DAGAN outperformed AnoGAN, GANomaly, and Skip-GANomaly, achieving AUC improvements of up to 0.250 and higher anomaly localization accuracy even with sparse training data.

Even with all this progress, there are still very few studies that put YOLO, simple autoencoders, and GAN-based models head-to-head in the same experiment especially when it comes to spotting problems in battery assembly. For example, Mattern et al. (2025) only compared a transformer-based model (DINO) and YOLOv8 for inspecting Li-ion battery electrodes. So far, nobody has tested YOLO, autoencoders, and GANs side by side under the same conditions for battery assembly inspections. This leaves a clear gap, almost no research directly compares all three methods for detecting anomalies in battery assembly.

2.5 Gaps and Research Opportunity

Future research in industrial anomaly detection should focus on improving model generalization with limited labelled data. Since anomalies are rare, developing techniques that can learn effectively from small datasets is crucial. One approach is to enhance synthetic data generation, making it more realistic and diverse to better train machine learning models.

Another promising direction is the integration of different deep learning techniques. Combining attention mechanisms with CNNs or transformers could improve feature extraction, making anomaly detection more accurate. Hybrid methods that leverage the strengths of multiple architectures may also provide more reliable results across various industrial settings.

Additionally, optimizing training methods to reduce computational costs while maintaining high detection accuracy is an important research goal. Efficient models will enable real-time detection without requiring extensive hardware resources. Continued comparative evaluations and advancements in hybrid approaches will be key to meeting the evolving demands of industrial manufacturing (Kharitonov et al., 2022).

2.6 Method Selection Rationale

Method	Key Strengths	Why Relevant for Battery Assembly?
YOLO	Supervised detector with strong localization, robust to background and lighting variation when trained with standard augmentation, real-time friendly, box-level outputs are easy to act on	Enforces assembly rules directly: checks presence, count, and placement of cells and terminals, flags polarity or position violations, produces class labels and boxes that map to rework instructions on the line
Autoencoder	Unsupervised, needs only normal images, transparent anomaly maps from reconstruction error, sensitive to subtle texture or appearance shifts, simple to train and tune	Useful when defect taxonomy is incomplete, detects small visual deviations like terminal wear, glare, minor dents, or surface contamination without needing labels, heatmaps help explain why an image is abnormal
EfficientAD	Student–teacher features capture local defects, lightweight AE captures global structure, calibrated image scores, fast inference, minimal external preprocessing	Covers both local faults (misplaced terminal, missing terminal) and global layout violations (wrong cell arrangement), calibrated scoring is stable across batches and products, heatmaps support root-cause checks

Table 1: Method Selection Rationale

Table 1 summarizes why YOLOv7, Autoencoder, and EfficientAD were selected for comparison. The three methods span the main paradigms needed on a factory line. YOLOv7 represents supervised detection that directly enforces assembly rules through bounding boxes and class labels, which is vital for presence, count, and placement checks. The Autoencoder provides an unsupervised baseline that requires only normal data and highlights subtle appearance shifts through reconstruction error maps. EfficientAD combines student–teacher feature comparison for local defects with an auxiliary autoencoder for global structure, yielding calibrated scores and informative heatmaps. Together they offer complementary strengths while keeping scope focused and reproducible.

3 Methodology

This chapter describes the experimental method used to compare YOLOv7, a reconstruction-based autoencoder and EfficientAD for anomaly detection in battery assembly images. The study is experimental rather than developmental: the objective is not to build a single system, but to carry out a controlled comparison of three deep learning approaches under identical conditions. To ensure fairness, all models were evaluated on the same dataset, preprocessed with the same image resolution, split using the same proportions of training, validation and test sets, and assessed using the same evaluation metrics.

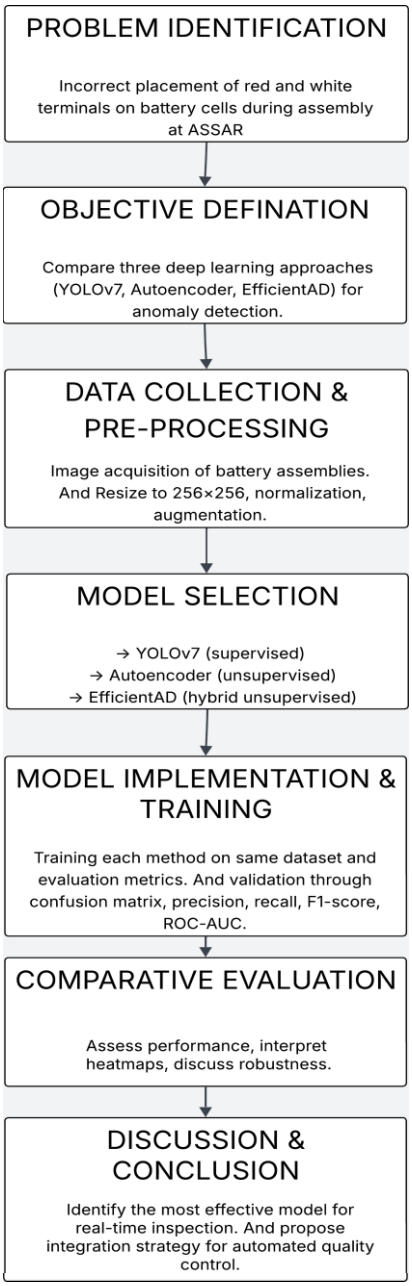

Figure 7: Overview of the experimental methodology

The figure 7 summarises the experimental workflow, including data collection and preprocessing, model selection, training, and unified evaluation of YOLOv7, a reconstruction-based autoencoder, and EfficientAD under standardised conditions for battery assembly anomaly detection.

The experimental process consisted of four main stages:

1. **Dataset preparation**
   All images were collected from the same battery assembly setup at ASSAR. The dataset contained normal (defect-free) images and anomalous images. For comparability, every image was resized to 256 × 256 and converted to RGB. Each method received the same dataset split (70 percent training, 20 percent validation and 10 percent testing), with anomaly images excluded from training for the unsupervised methods.

2. **Model-specific training under standardised constraints**
   Each model was trained according to its methodological requirements, but with controlled preprocessing and consistent hyperparameter budgets. YOLOv7 was trained in a supervised manner using full annotations; the autoencoder and EfficientAD were trained only on normal images. Preprocessing steps and augmentation choices were kept minimal and aligned across methods so that differences in outcome reflect model behaviour rather than data handling.

3. **Unified evaluation protocol**
   All models were evaluated using the same image-level metrics: accuracy, precision, recall, F1-score and ROC-AUC. Confusion matrices and anomaly heatmaps were generated for qualitative comparison. In addition, stress tests examined robustness to background variation, lighting differences and cell orientation changes. This provided insight into each method's failure modes and suitability for real industrial use.

4. **Comparative analysis under identical experimental conditions**
   The results were compared to identify the strengths, weaknesses and practical implications of each approach. Since every model used the same input format, same dataset split and same evaluation metrics, the comparison isolates model-specific performance rather than differences in experimental setup. This ensures that the conclusions reflect the capabilities of the algorithms rather than differences in training conditions.

The overall aim of this methodology is to create a fair, controlled and replicable framework for assessing the three approaches, enabling meaningful conclusions about their suitability for battery assembly inspection.

3.1 Rationale for selecting YOLOv7 in Thesis

YOLOv7 was chosen for this thesis because, at the time of project initiation in January 2025, it represented a mature, well-documented, and high-performing real-time object detection framework. Introduced in 2022 by Wang, Bochkovskiy, and Liao, it delivered 56.8 % average precision at 30 frames per second on the MS COCO dataset, surpassing both convolutional and transformer-based competitors (Wang, Bochkovskiy, & Liao, 2022). This level of performance, combined with stable support in PyTorch and CUDA, made it particularly suitable for experimentation on consumer-grade hardware such as an RTX 4050 laptop. Newer versions of YOLO, including v8 to v11, were available, but they were still undergoing active revisions and had not yet reached the same level of maturity for PyTorch-based workflows, which created compatibility risks and potential reproducibility issues.

Beyond general detection capabilities, YOLOv7 has been successfully applied in several anomaly detection tasks, confirming its relevance to industrial inspection contexts. Zhang et al. (2024) applied an enhanced YOLOv7 framework for the detection of anomalies in photovoltaic cells and reported strong gains in detection robustness and speed. Zou, Wang, Wang, Xu, and He (2025) presented YOLO-SRSA, an improved YOLOv7 variant for power equipment anomaly detection in adverse weather, and achieved increases in mAP@0.5 from 89.2 % to 93.5 %, precision from 95.9 % to 97.1 %, and recall from 95 % to 97 %. These studies highlight that YOLOv7 has not only remained relevant but has continued to provide reliable performance in anomaly detection applications well after its initial release.

By the time this research was initiated, newer YOLO versions had not yet established stable documentation, long-term validation, or proven reliability in anomaly detection scenarios. YOLOv7, in contrast, had extensive pretrained weights, strong community adoption, and wide open-source support, which ensured reproducibility and consistency in academic experimentation. For these reasons, YOLOv7 was selected as the most practical and academically credible choice for the comparative anomaly detection study presented in this thesis.

3.2 Implementation Details of YOLO V7

The object detection model was implemented using the YOLOv7 framework in PyTorch. A total of approximately 250 images were annotated using the Roboflow platform, consisting of 170 images representing good battery assemblies and the remaining images representing anomalies.

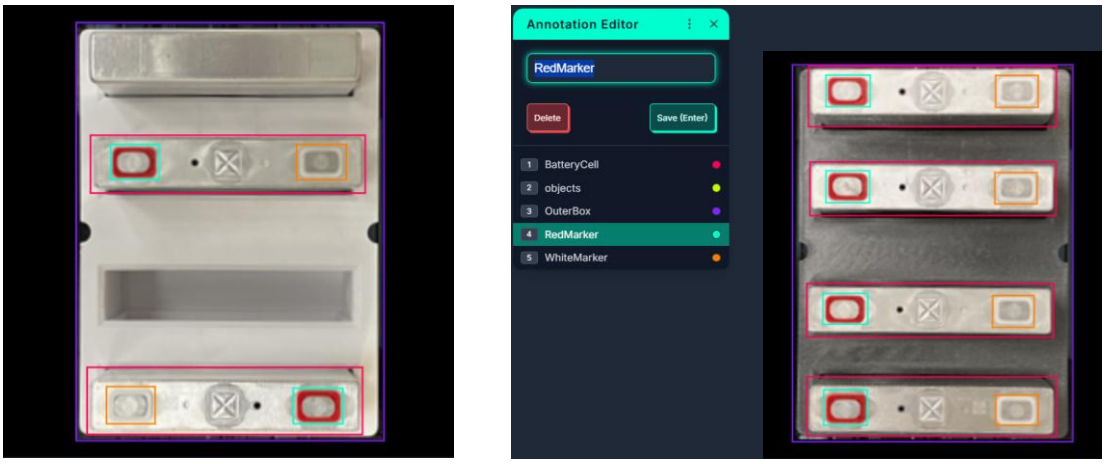

Figure 8: Labelling of Images in Roboflow for YOLOv7

Each image was annotated at four different levels: the outer box, the four battery cells, and the red and white terminal terminals on each cell. In anomaly images, only the components that physically existed were annotated. For example, if a terminal or a cell was missing, its corresponding annotation was deliberately omitted to reflect the anomaly condition. This annotation strategy ensured the dataset captured both correct and faulty assembly scenarios in a consistent manner.

3.2.1 Data Pre-processing Steps

For YOLOv7, common pre-processing options include letterbox resizing that preserves aspect ratio with padding, direct resizing to a fixed square shape, standard pixel normalization, and a configurable set of augmentation “bag-of-freebies” such as HSV jitter, random flips, random scaling and translation, random perspective or affine transforms, Mosaic and MixUp. These are exposed through the YOLOv7 training hyperparameters and scripts and are widely used to improve generalization without increasing inference cost (Wang, Bochkovskiy, & Liao, 2023; WongKinYiu, 2025).

In this thesis the pipeline was kept focused and task driven. Images were resized to 256 × 256 with aspect-ratio stretching and normalized. Stretching was chosen to keep the input identical to the other two methods in the

comparison and to avoid padding artifacts that could interact with the small terminal terminals in the battery cells. Augmentations were limited to brightness and exposure variation, horizontal flips, HSV color-space jitter, random scaling and translations for boxes, and Mosaic. These choices target the dominant nuisances in the dataset, namely illumination shifts, small in-plane pose changes, and modest object placement differences, while remaining consistent with well-established augmentation practices for YOLO-style detectors (Bochkovskiy, Wang, & Liao, 2020; Xu, Yoon, Fuentes, & Park, 2023).

Input image options for YOLOv7 are configurable via the -img flag, with commonly used square sizes such as 640 or 1280, and any consistent square size can be trained if model and anchors are configured accordingly. YOLOv7 supports model and resolution scaling across device classes, and the repository examples document training at both 640 and 1280. For this work, 256 × 256 was selected to maintain a fair, like-for-like comparison with the Autoencoder and EfficientAD pipelines, and to fit GPU memory while enabling larger batch sizes, accepting that YOLOv7's default recipes often use higher resolutions (Wang et al., 2023).

3.2.2 Techniques Used

The training approach followed a supervised learning paradigm since the annotations were fully labeled. A training-validation-test split was used to evaluate model performance:

- 70% training set
- 20% validation set
- 10% test set

This ensured sufficient data for training while preserving unseen samples for validation and final performance evaluation. Anomalies were included proportionally across all splits to ensure balanced evaluation.

3.2.3 Network Architecture

- **Backbone (Feature Extraction)**
  - Input: 256 × 256 × 3 RGB.
  - Stem: A sequence of Conv-BN-SiLU layers that downsample the input and expand channels.

- E-ELAN blocks: These extended efficient layer aggregation network blocks widen the feature space while preserving gradient flow. They encourage multi-branch aggregation and controlled depth growth, improving learning stability on small datasets.
- SPPCSPC: A split-and-concat spatial pyramid pooling layer is used near the end of the backbone to enlarge the effective receptive field using parallel max-pooling kernels.
- Output Feature Maps: The backbone produces multi-resolution features at strides 8, 16, and 32, which are then passed to the neck.

- **Neck (Multi-Scale Fusion)**
  - FPN + PAN: A combination of top-down (Feature Pyramid Network) and bottom-up (Path Aggregation Network) paths fuses semantic and spatial information across different scales.
  - Lateral Connections and Concatenations: These are used to align channel dimensions and merge features from adjacent resolutions.
  - Result: Three fused feature maps at strides 8, 16, and 32 that are richer in both context and detail.
- **Detection Head (Anchor-Based Prediction)**
  - Predictions: For each of the three scales, the head predicts bounding box offsets, an objectness score, and class probabilities per anchor.
  - Losses: The model uses an IoU-based box regression loss combined with objectness and classification losses. Non-maximum suppression with an IoU threshold of 0.45 is applied at inference.
  - Strides and Output Sizes: With a 256 × 256 input, the output sizes are 32 × 32 (stride 8), 16 × 16 (stride 16), and 8 × 8 (stride 32).
- **Model Size and Efficiency**
  - The model has approximately 37 million trainable parameters and over 400 layers.
  - RepConv reparameterization is applied during the export phase, which fuses multi-branch blocks into single-branch convolutions for faster inference without changing the predictions.
- **Task-Specific Suitability**

- The multi-scale head at strides 8, 16, and 32 effectively supports the detection of very small parts, such as terminal terminals, while maintaining accuracy on larger structures like the outer box.
- The SPPCSPC and E-ELAN blocks improve feature reuse and gradient propagation, which is particularly helpful for limited data and the class imbalance often found between small terminals and larger components.

3.2.4 Hyperparameters

The YOLOv7 model was trained using the following hyperparameter settings. These values follow the default recommendations of the YOLOv7 framework where appropriate and were only minimally adjusted to balance detection accuracy with hardware constraints, while preserving strict comparability with the other methods evaluated in this study.

- **Epochs:** 100 (empirically selected to ensure convergence without overfitting)
- **Batch size:** 16 (adjusted based on GPU memory availability)
- **Learning rate:** 0.01 with cosine decay (starting at 0.01 and reducing to 0.001)
- **Optimizer:** Stochastic Gradient Descent (SGD) with momentum 0.937 and weight decay 0.0005
- **Image size:** 256 × 256 pixels
- **IoU threshold:** 0.45 for non-maximum suppression
- **Confidence threshold:** 0.40 to filter low-confidence predictions

The final model checkpoint was selected based on the best validation mean Average Precision (mAP) on the held-out validation set.

3.3 Implementation Details of Autoencoder

3.3.1 Pre-processing Steps

The dataset comprised 170 normal images and 50 anomaly images. To maintain strict comparability with the other methods in this study, every image was resized to 256 × 256 and converted to RGB, then scaled to [0, 1]. Channel-wise normalization used the mean and standard deviation computed from the training subset of normal images. This combination of fixed input size plus dataset-driven normalization is standard in reconstruction-based anomaly pipelines and is supported by mature anomaly-detection toolchains that expose transform hooks for consistent training and evaluation. Validation and

test sets received only resize and normalization to preserve fidelity. (Berahmand et al., 2024).

To mitigate illumination nonuniformity and mild contrast shifts observed in the battery assembly images, Contrast Limited Adaptive Histogram Equalization (CLAHE) was applied during training with probability 0.5 (clip limit 2.0, 8 × 8 tiles). CLAHE improves local contrast while limiting noise amplification and has shown measurable benefits in dental radiography image-quality assessments relative to plain histogram equalization, which is consistent with its use as a low-risk enhancement step before learning-based models (Mehdizadeh et al., 2023).

Augmentation was intentionally low distortion to avoid inventing structures that an autoencoder might learn to reconstruct. We used brightness/contrast jitter within ±10 percent, horizontal flip with probability 0.5, and small affine perturbations limited to rotation ±3 degrees, translation up to 4 percent, and scale 0.95–1.05. Contemporary augmentation surveys describe these operations as effective, model-agnostic ways to improve robustness while preserving semantic content, and reconstruction-based studies emphasize training on normal data with restrained perturbations so that reconstruction error remains a reliable anomaly signal (Givnan et al., 2022; Xu et al., 2023). Validation and test images were not augmented beyond the common resize and normalization.

3.3.2 Techniques Used

The autoencoder was trained in a one-class manner using only normal (good) images. From the available good images, a 70/20/10 split was applied to form the training, validation, and test subsets. All anomaly images were held out from training and were used only in validation and test to set the decision threshold and evaluate performance. The anomaly score for an image was computed as the mean absolute reconstruction error per pixel, optionally combined with a structural similarity (SSIM) deficit term for heatmap visualization. The decision threshold was selected on the validation set by maximizing Youden's J statistic on the image-level ROC curve. The same threshold was then used to report test metrics. A fixed random seed ensured reproducibility of data splits, weight initialization, and data loader shuffling.

3.3.3 Hyperparameters

Training ran for 100 epochs with a batch size of 32 on an NVIDIA RTX 4050 (6 The autoencoder was trained for 100 epochs using a batch size of 32 with automatic mixed precision. All experiments were conducted using a fixed

random seed to ensure reproducibility, and early stopping was not applied in order to keep the training budget identical across methods.

The optimizer used was AdamW with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of 1×10^{-4}. The initial learning rate was set to 1×10^{-3} and followed a cosine decay schedule down to 1×10^{-5} over the full training duration.

The training loss combined L1 reconstruction loss (weight 0.8) with a Structural Similarity Index Measure (SSIM) loss term (weight 0.2) to encourage both pixel-level fidelity and preservation of structural information. Gradient clipping with a maximum norm of 1.0 was enabled to stabilise early training.

The final model checkpoint was selected based on the lowest validation loss. Inference used the same 256 × 256 preprocessing pipeline as training, and anomaly heatmaps were computed from per-pixel absolute reconstruction error, optionally smoothed using a 3 × 3 mean filter for visualisation purposes only.

3.4 Implementation Details of EfficientAD

EfficientAD was selected in this study as one of the deep learning approaches for anomaly detection due to its balance of efficiency and detection capability. Unlike purely reconstruction-based methods, it integrates both feature distillation and reconstruction error to highlight deviations between normal and abnormal samples. This dual mechanism allows the framework to capture fine local irregularities as well as larger structural defects without requiring pixel-level annotations. The following subsections describe the pre-processing pipeline, the techniques employed, the architecture of the networks, and the hyperparameters used during training.

3.4.1 Pre-processing Steps

The dataset contained 170 normal images and 50 anomaly images. All images were resized to 256 × 256 to keep a strict like-for-like comparison with the YOLOv7 and autoencoder pipelines and to maintain stable batch scheduling on the RTX 4050. In the commonly used implementation of EfficientAD, the external transform is intentionally minimal. Inputs are resized only, while normalization is handled internally using ImageNet statistics. Keeping normalization inside the model avoids conflicts between teacher feature statistics and input scaling, which is consistent with guidance from the official implementation notes and examples in practice. This setup also matches the recommendation in EfficientAD's original description that emphasizes simplicity in pre-processing for robustness across product types (Batzner,

Heckler, and König, 2024). Validation and test images used the same resize-only policy to preserve fidelity.

3.4.2 Techniques Used

EfficientAD was trained in a one-class setting using only normal images. From the 170 normal images, 70 percent were used for training, 20 percent for validation, and 10 percent for test. The 50 anomaly images were held out from training and split evenly between validation and test to tune the operating threshold and report unbiased performance. Following the approach described in the paper, EfficientAD produces a per-pixel anomaly map and an image-level score. Map scaling uses quantile calibration computed on normal validation data, where lower and upper quantiles are chosen to stabilize scores across different appearances of the same product. The image-level decision threshold was selected on the validation set by maximizing Youden's J statistic and then fixed for the test set (Batzner, Heckler, and König, 2024). For broader anomaly detection context and dataset protocol consistency, we follow the MVTec AD evaluation practice of training on normal data and validating thresholding on held-out sets.

3.4.3 Hyperparameters

EfficientAD was trained for 100 epochs using automatic mixed precision. The optimizer was Adam with an initial learning rate of 1.0×10^{-4} and a weight decay of 1.0×10^{-5}. A step decay schedule reduced the learning rate by a factor of 0.1 near 95 percent of the training budget.

Consistent with the standard EfficientAD setup, the batch size was fixed to 1 so that per-image feature statistics and quantile calibration remain well defined. The input size was fixed at 256 × 256 for both training and evaluation to preserve strict comparability with the other methods. All runs used a fixed random seed to ensure reproducibility.

The final operating threshold for image-level anomaly decisions was selected on the validation set using Youden's J statistic and then left unchanged for the test set, following standard practice in industrial anomaly detection (Batzner et al., 2024; Bergmann et al., 2021).

4 Experiments and Results

This section evaluates three approaches for battery assembly anomaly detection: YOLOv7 as a supervised detector, and Autoencoder and EfficientAD as unsupervised methods. All models use 256 × 256 inputs on the same dataset split to ensure fair comparison. Metrics include accuracy, precision, recall, F1, ROC-AUC, and qualitative heatmaps.

Table 2 summarizes configurations and thresholds, and the subsections report per-method results and error analysis.

Criterion	**YOLOv7 (supervised detector)**	**Autoencoder (unsupervised, reconstruction)**	**EfficientAD (unsupervised, student–teacher + AE)**
Goal	Detect objects and layout errors via bounding boxes	Learn to reconstruct normal images, flag high reconstruction error	Match student features to frozen teacher, fuse with lightweight AE for global cues
Supervision & labels	Supervised, requires annotated boxes for classes	Unsupervised, trained on normal only	Unsupervised, trained on normal only
Dataset used	170 normal + 50 anomaly images; split 70% train, 20% val, 10% test for parity	170 normal for training/val/test (70/20/10). 50 anomaly held out for val/test	170 normal for training/val/test (70/20/10). 50 anomaly held out for val/test
Input size	256 × 256 (chosen to match other methods)	256 × 256	256 × 256
Pre-processing	Resize to 256 × 256 with aspect ratio stretching; standard normalization; augmentations: brightness ±10%, horizontal flip	Resize, RGB, scale to [0,1], channel-wise normalization from training set; CLAHE p=0.5; light aug: brightness/contrast ±10%, small affine, horizontal flip	Minimal external transforms. Resize only. Normalization handled inside model with ImageNet stats per reference implementation

Framework / implementation	PyTorch (YOLOv7 training code)	PyTorch	PyTorch via Anomalib (EfficientAD)
Acceleration & precision	CUDA on NVIDIA RTX 4050 6 GB; AMP enabled	CUDA on NVIDIA RTX 4050 6 GB; AMP enabled	CUDA on NVIDIA RTX 4050 6 GB; AMP enabled
Key tooling	Native YOLOv7 repo, auto-anchor check, EMA	Albumentations for preprocessing/augs	Anomalib reference pipeline; batch size 1 for per-image stats
Output	Bounding boxes with class and confidence	Per-pixel error heatmap and image anomaly score	Per-pixel anomaly map and image anomaly score
Thresholding	Confidence threshold 0.40 and NMS IoU 0.45	Threshold chosen on validation to maximize Youden's J on ROC	Quantile calibration on normal val data, threshold chosen on validation via Youden's J
Optimizer	SGD, momentum 0.937, weight decay 0.0005	AdamW, weight decay 1e-4	Adam, weight decay 1e-5
Learning rate schedule	Cosine decay from 0.01 toward 0.001	Cosine decay from 1e-3 to 1e-5	Step decay by 0.1 near 95% of epochs
Batch size	16 (AMP enabled)	32 (AMP enabled)	1 as per reference to keep per-image stats well defined (AMP enabled)
Epochs	100	100	100
Training time on RTX 4050 6 GB	~23 minutes	~11 minutes	~17 minutes

Table 2: Comparison Parameter Results of Algorithms

4.1 YOLOv7 results

4.1.1 Confusion Matrix

All results in this subsection are produced by the YOLOv7 detector. The model was trained to detect four classes: BatteryCell, OuterBox, RedTerminal, and WhiteTerminal.

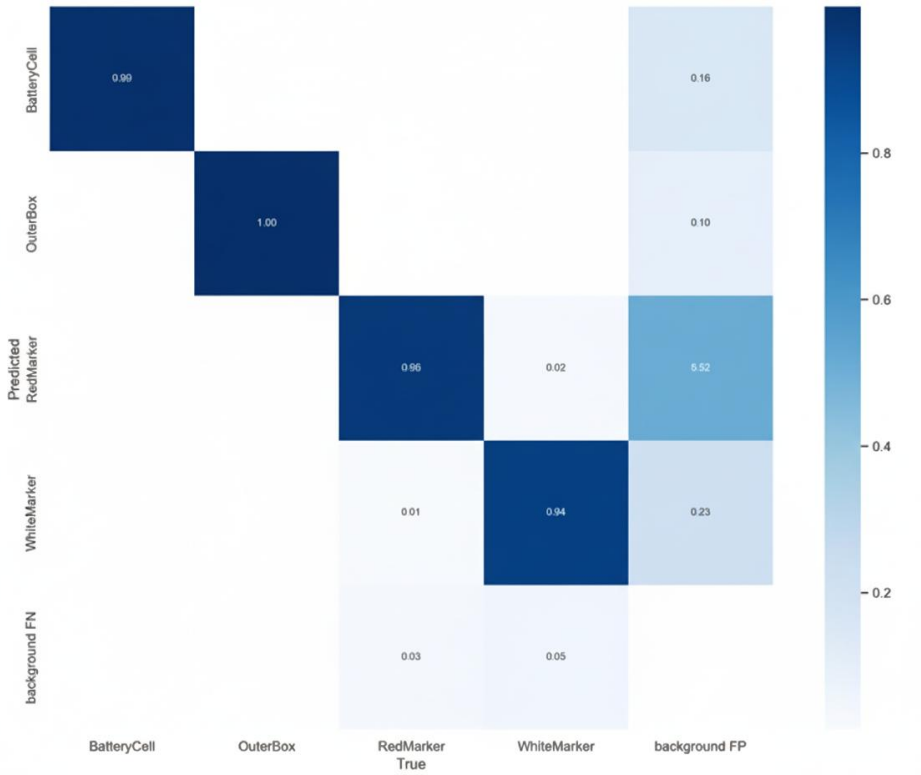

Figure 9: YOLOv7 Confusion Matrix

Figure 9 shows normalized confusion matrix indicates strong per-class recall on the diagonal:

- **BatteryCell:** 0.99
- **OuterBox:** 1.00
- **RedTerminal:** 0.96
- **WhiteTerminal:** 0.94

The last column ("background FP") reflects false positives where background regions were misclassified as one of the classes. Rates are lowest for OuterBox (0.10) and BatteryCell (0.16), but higher for RedTerminal (0.52) and WhiteTerminal (0.23), indicating that terminal-like textures in the background are occasionally over-predicted. The bottom row ("background FN") captures missed detections; values are small overall, with slightly higher misses for the terminal classes.

4.1.2 F1-Score Analysis

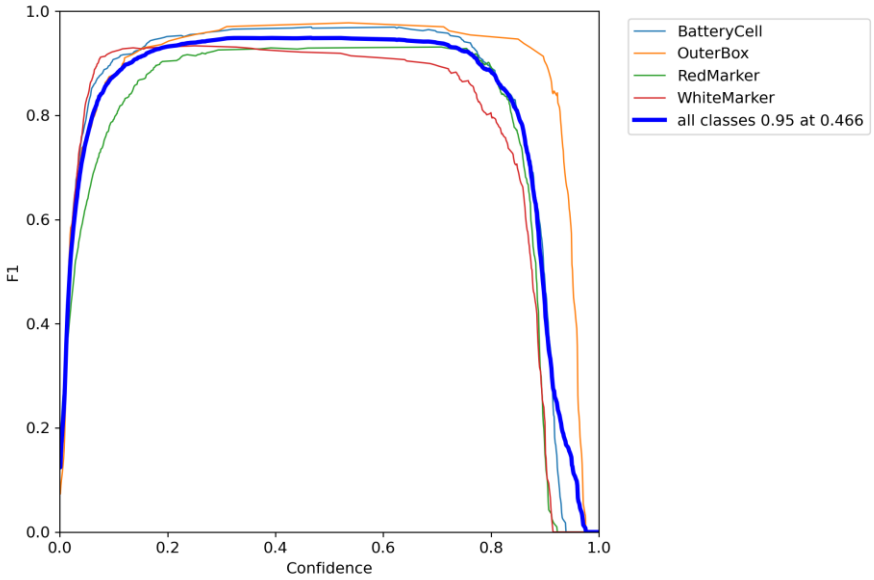

Figure 10: YOLOv7 F1-score

Figure 10 represents the F1-score curve of the YOLOv7 model, which shows the balance between precision and recall across varying confidence thresholds. The maximum F1-score reached approximately 0.95 at a threshold of 0.466, indicating strong overall performance in detecting battery assembly components. OuterBox detection remained the most stable across thresholds, while RedTerminal and WhiteTerminal showed sharper declines at higher thresholds due to their smaller size and greater sensitivity to background variation.

4.1.3 Precision Curve Evaluation

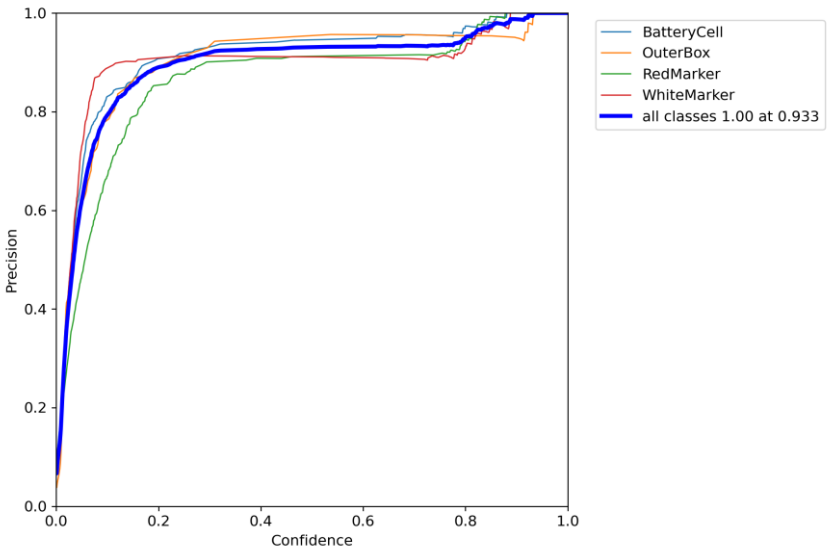

Figure 11: YOLOv7 Precision curve

Figure 11 represents the precision curve of the YOLOv7 model, which demonstrates the ability to avoid false positives across different confidence thresholds. At a threshold of 0.933, the precision for all classes approached 1.00, confirming that the model produces highly reliable detections under stricter operating conditions. Among the four classes, OuterBox maintained the highest consistency, while the terminal terminals (RedTerminal and WhiteTerminal) showed more fluctuation, reflecting the added challenge of detecting small and visually similar objects.

4.1.4 Precision–Recall Curve Analysis

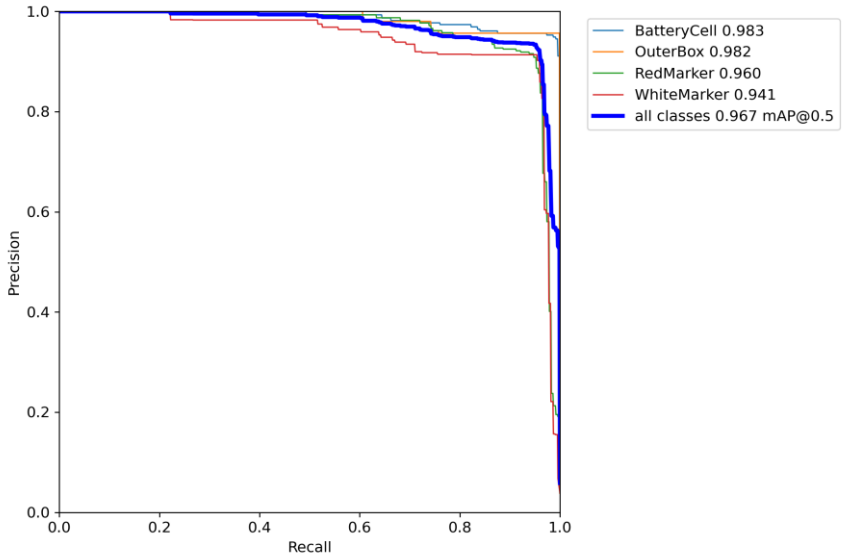

Figure 12: YOLOv7 Precision-Recall Curve

Figure 12 represents the precision–recall (PR) curve of the YOLOv7 model, which highlights the trade-off between precision and recall. The mean average precision (mAP@0.5) reached 0.967 across all classes, with individual results of 0.983 for BatteryCell, 0.982 for OuterBox, 0.960 for RedTerminal, and 0.941 for WhiteTerminal. These results confirm the robustness of YOLOv7 in detecting all battery components, though terminal detection proved slightly less consistent compared to larger structures such as BatteryCells and OuterBoxes.

4.1.5 Recall Curve Evaluation

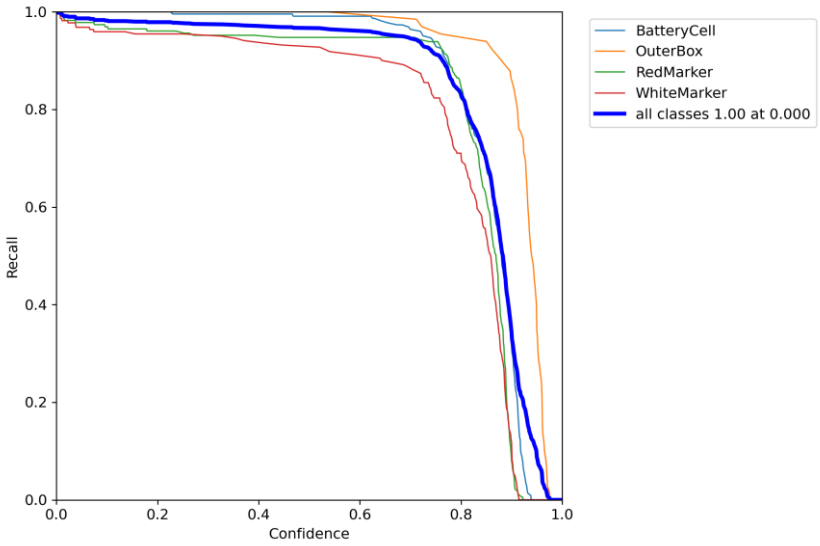

Figure 13: YOLOv7 Recall Curve

Figure 13 represents the recall curve of the YOLOv7 model, which measures the ability to correctly identify true positives at varying confidence thresholds. At very low thresholds, recall values approached 1.00, but declined gradually beyond 0.7, as stricter thresholds prioritized precision at the expense of recall. OuterBox and BatteryCell achieved the most stable recall values, while RedTerminal and WhiteTerminal showed reduced recall sensitivity, further highlighting the detection difficulty of smaller features.

4.1.6 Anomaly Visualization with Heatmaps

- Detection of Misplaced Red Terminals

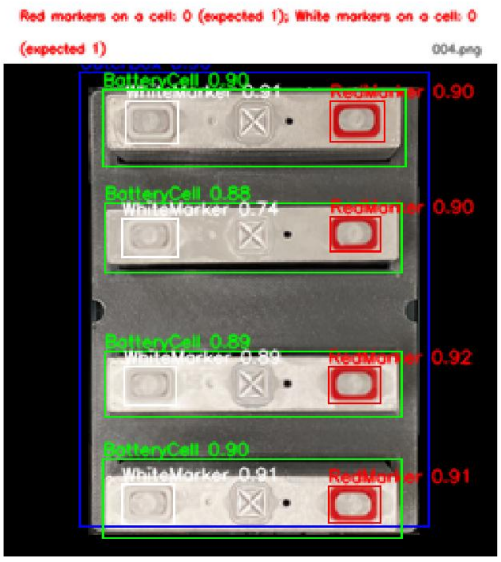

Figure 14: YOLOv7 Detection of Misplaced Terminal

Figure 14 represents an anomaly where red terminals are positioned on the right side of the battery cells instead of the left. The YOLOv7 model successfully identified this misplacement, highlighting its ability to correctly localize and classify terminal orientation. Even though the cells appear visually similar, the detection pipeline flagged the incorrect terminal alignment as an anomaly, demonstrating robustness against subtle placement errors.

- **Hidden Defect between Terminal Cells**

Figure 15: YOLOv7 Dectecting Hidden Defect in Battery Cells

Figure 15 represents a more challenging case where all visible red and white terminals are correctly positioned. However, the battery cells located between these terminals were intentionally defected. Since the anomaly is not directly linked to the terminal placement, the model did not classify it as defective. This result highlights a limitation, as the framework primarily relies on terminal placement consistency rather than internal cell surface abnormalities.

- **Detection of Double Red Terminals**

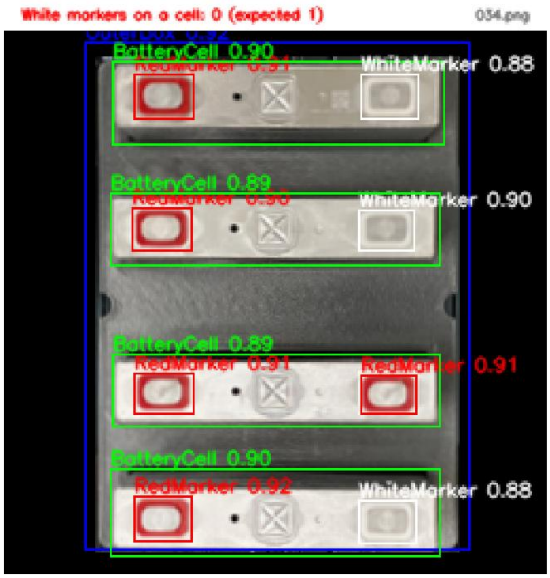

Figure 16: YOLOv7 Dectecting Double Red Terminal in Battery Cell

Figure 16 represents an anomaly where a single battery cell contains two red terminals instead of the expected one. The YOLOv7 model detected this error accurately, classifying the cell as anomalous. The strong detection confidence indicates the model's sensitivity to structural irregularities beyond simple left–right terminal placement, capturing both missing and duplicated components. This demonstrates its reliability in handling uncommon but critical assembly defects.

- **Detection under Background Variation**

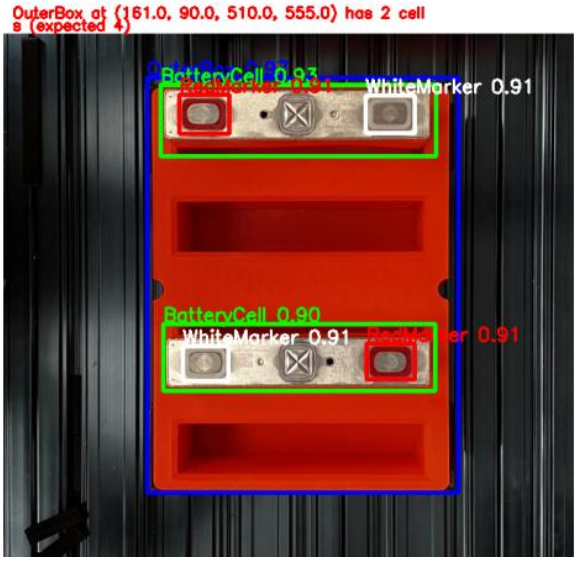

Figure 17: YOLOv7 Detecting in Different Background Variation

Figure 17 represents a case where the outer box and cells were placed on a background different from the training dataset. Despite this variation, the YOLOv7 model successfully detected all four cells and their respective terminals with high accuracy. The correct classification under these conditions demonstrates the robustness and generalization capacity of the model, confirming that background textures and color changes do not affect anomaly detection performance.

4.2 Autoencoder results

4.2.1 Confusion Matrix

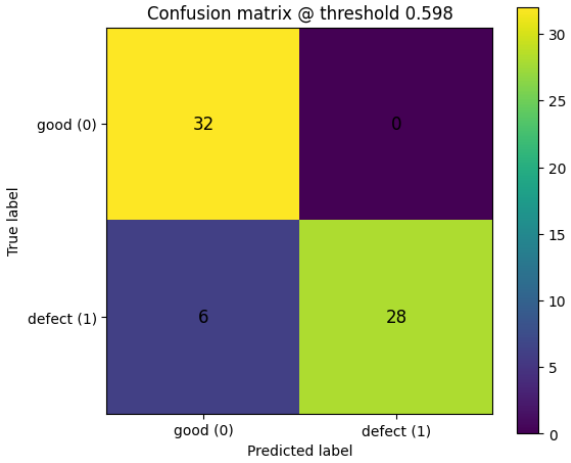

Figure 18: Autoencoder Confusion Matrix

Figure 18 represents the confusion matrix of the Autoencoder at a threshold of 0.598. The model correctly classified 32 good samples and 28 defective ones, but 6 defective samples were misclassified as good. This imbalance shows that while the Autoencoder performed well in detecting normal images, it had a tendency to miss certain anomalies, indicating lower sensitivity compared to other approaches.

4.2.2 ROC Curve

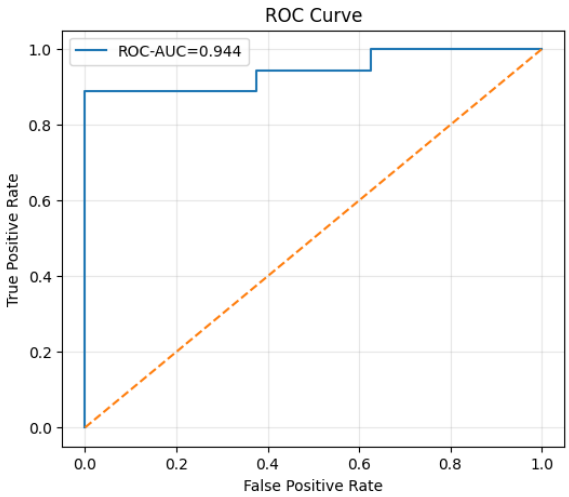

Figure 19: Autoencoder ROC Curve

Figure 19 represents the ROC curve of the Autoencoder, which evaluates the trade-off between true positive and false positive rates. The model achieved an AUC score of 0.944, suggesting good discriminative ability between normal and anomalous samples. However, the curve is not as close to the top-left corner as in other models like EfficientAD, showing that the Autoencoder is relatively weaker in perfectly separating the two classes.

4.2.3 Precision–Recall Curve

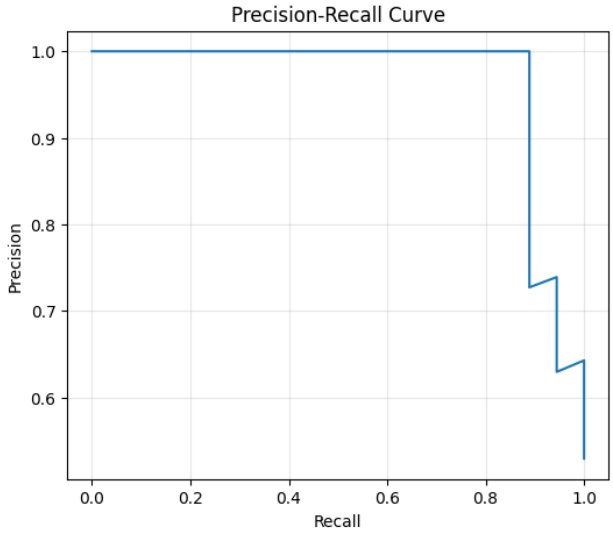

Figure 20: Autoencoder Precision-Recall Curve

Figure 20 represents the precision–recall curve of the Autoencoder. The model maintained near-perfect precision for a wide range of recall values, but performance dropped sharply after 0.8 recall. This shows that while the

Autoencoder is very accurate in the anomalies it detects, it struggles to capture all possible defects, leading to lower recall performance.

4.2.4 Score Distribution

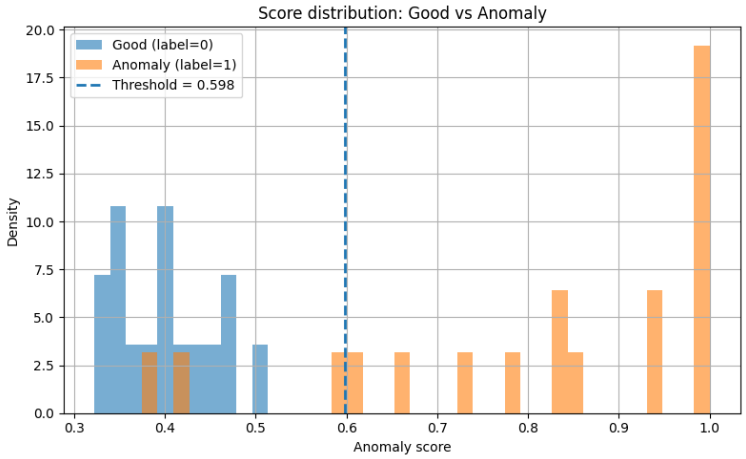

Figure 21: Autoencoder Score Distribution

Figure 21 represents the anomaly score distribution for good and defective samples produced by the Autoencoder. Good samples (blue) are tightly clustered around lower scores, while defective samples (orange) spread across higher scores. The threshold of 0.598 provides a reasonable separation between the two classes, but some overlap is visible, which explains the misclassifications observed in the confusion matrix.

4.2.5 Anomaly Visualization with Heatmaps

- **Detection under Different Backgrounds**

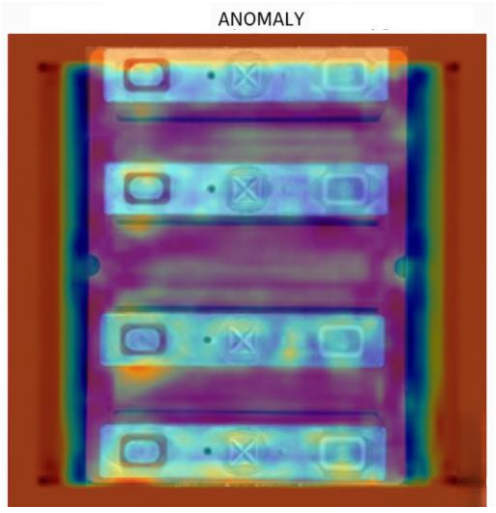

Figure 22: Autoencoder Detection in Different Background

Figure 22 represents an anomaly visualization where the Autoencoder processed a battery assembly placed on a background different from its training

distribution. The heatmap highlights large activation areas across the image, incorrectly flagging the entire sample as defective. This indicates that the Autoencoder is sensitive to background variation, which can cause false positives and reduce its robustness in real-world inspection environments where background textures may change.

- **Missing Battery Cell Detection**

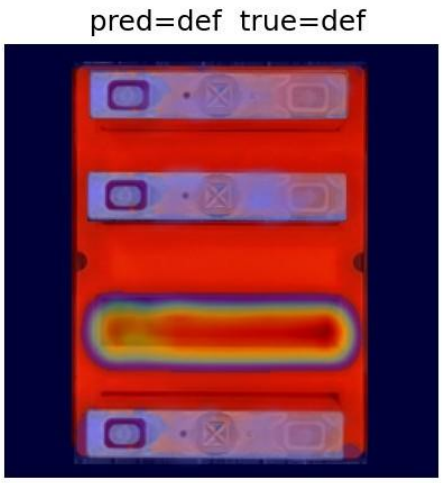

Figure 23: Autoencoder Defect Detection

Figure 23 represents an anomaly where the third battery cell is missing. The Autoencoder clearly detected this irregularity, producing strong heatmap activation in the empty region. This visualization shows the strength of the Autoencoder in recognizing structural defects such as missing components. By highlighting the absence of expected features, the model demonstrates its capacity to flag critical assembly failures effectively.

- **Misclassification in Flipped Cells**

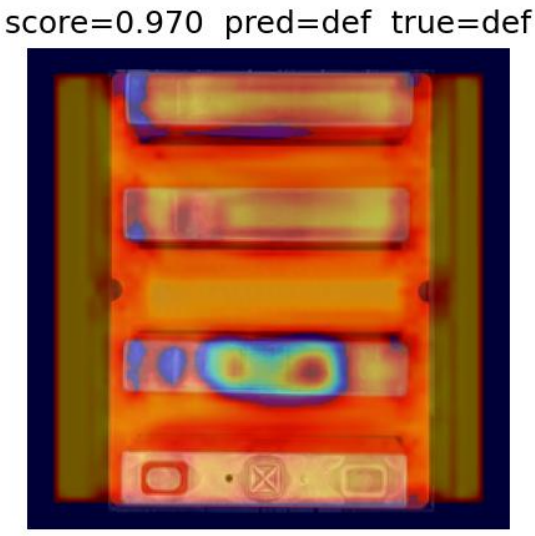

Figure 24: Autoencoder Fails Detection in Flipped Cells

Figure 24 represents a challenging case where the first three battery cells are flipped upside down. The Autoencoder struggled in this scenario, as the heatmap produced strong activations only around the third cell, while the rest of the image was broadly highlighted in red. This suggests that the model is less sensitive to rotational anomalies, showing limitations in capturing orientation-based defects consistently.

4.3 EfficientAD results

4.3.1 Confusion Matrix

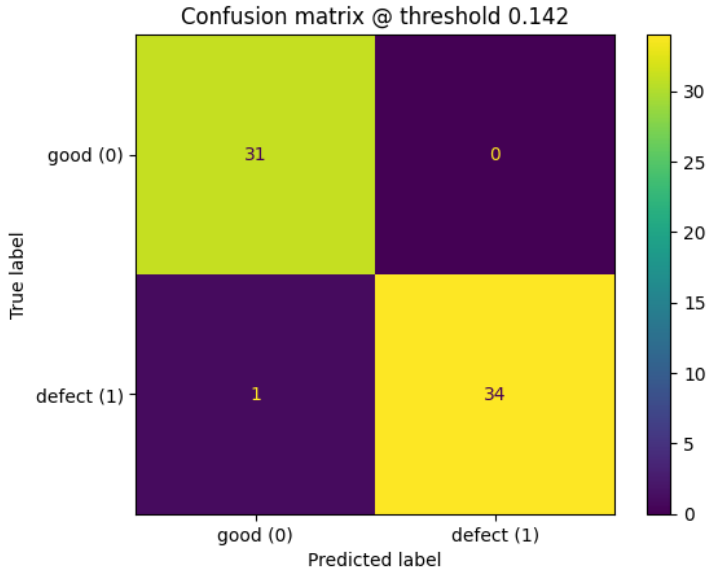

Figure 25: EfficienctAD Confusion Matrix

Figure 25 represents the confusion matrix of EfficientAD at a threshold of 0.142, which compares predicted and true labels of good and defective samples. The model correctly classified 31 good images and 34 defective images, with only a single misclassification. This demonstrates highly reliable discrimination between normal and anomalous samples, reflecting EfficientAD's ability to generalize well to unseen data.

4.3.2 ROC Curve

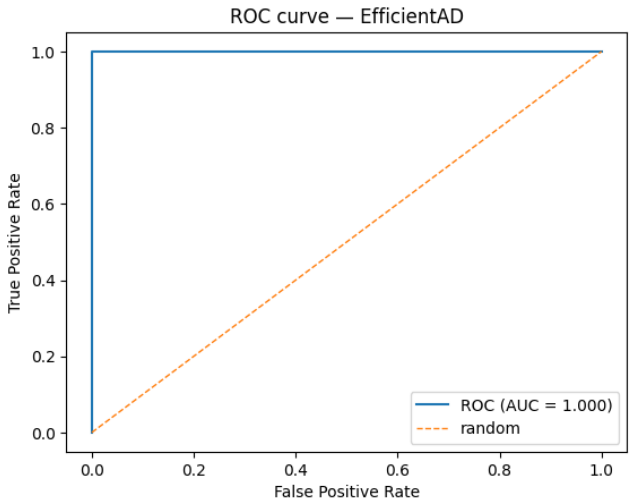

Figure 26: EfficienctAD ROC Curve

Figure 26 represents the ROC curve of EfficientAD, which plots the true positive rate against the false positive rate at different thresholds. The model achieved an AUC score of 1.000, indicating perfect separability between good and defective samples. The curve's shape shows that EfficientAD maintains both high sensitivity and specificity across all thresholds, confirming its robustness in anomaly detection.

4.3.3 Precision–Recall Curve

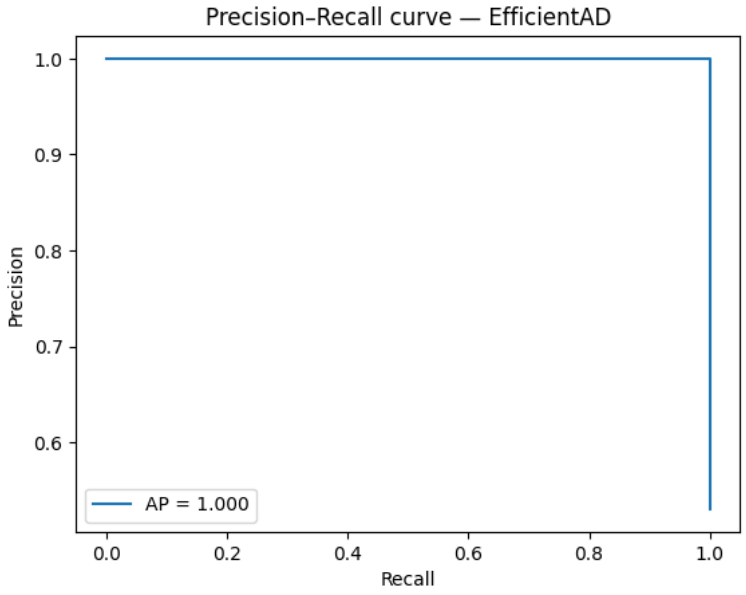

Figure 27: EfficienctAD Precision-Recall Curve

Figure 27 represents the precision–recall curve of EfficientAD, which highlights the trade-off between precision and recall. The model achieved an average

precision (AP) of 1.000, showing that it can detect all anomalies without producing false positives. The flat curve near the top of the graph confirms consistent high performance regardless of the threshold applied.

4.3.4 Score Distribution

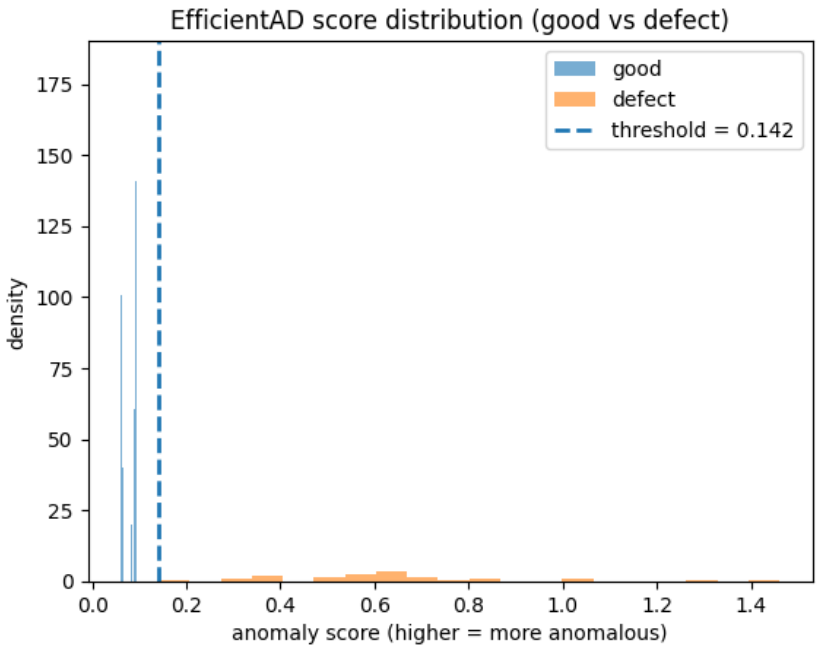

Figure 28: EfficienctAD Score Distribution

Figure 28 represents the anomaly score distribution produced by EfficientAD for good and defective samples. The results show a clear separation between good samples (clustered at lower scores) and defective samples (distributed at higher scores). The threshold at 0.142 effectively divides the two groups, ensuring minimal overlap and enabling accurate anomaly classification.

4.3.5 Anomaly Visualization with Heatmaps

- **Detection of Misplaced Red Terminals**

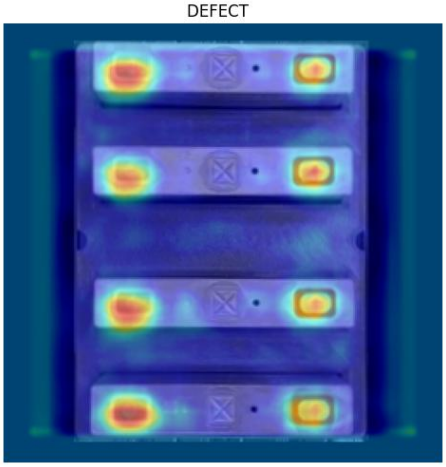

Figure 29: EfficienctAD Detection of Misplaced Cells

Figure 29 represents an anomaly where the heatmap highlights both terminal regions, indicating incorrect placement of red and white terminals. The EfficientAD model successfully localized these deviations, with strong activations on the misplaced terminals. This demonstrates the ability of EfficientAD to not only flag an image as defective but also provide visual evidence of where the structural misalignment occurs, improving interpretability of the detection.

- **Hidden Defect in Terminal-Free Cells**

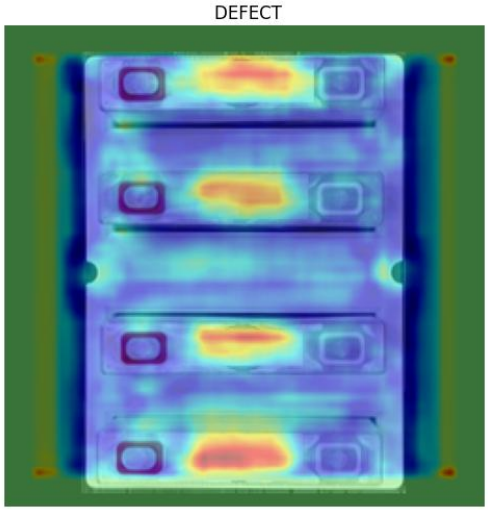

Figure 30: EfficienctAD Detects Hidden Defect in Cells

Figure 30 represents an anomaly where the terminals appear correctly positioned, but the heatmap activation is concentrated in the middle region of the cells. This indicates that although terminal alignment was correct, the underlying cell texture appeared abnormal. EfficientAD effectively captured this

subtle defect, proving that its anomaly scoring extends beyond terminal placement and can detect irregularities in the cell surface itself.

- **Detection of Double Red Terminals**

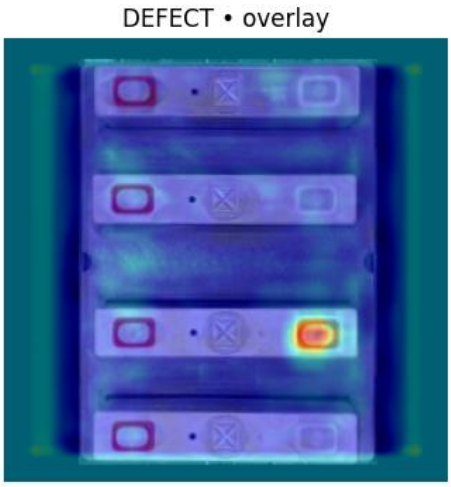

Figure 31: EfficienctAD Detects Double Red Terminal

Figure 31 represents an anomaly where two red terminals were detected in a single battery cell. The heatmap clearly highlights the duplicated terminal region, showing high activation in the area of error. EfficientAD accurately flagged this structural abnormality with strong localization, confirming its effectiveness in identifying uncommon but highly critical defects such as duplicate terminal placement.

- **Detection under Background Variation**

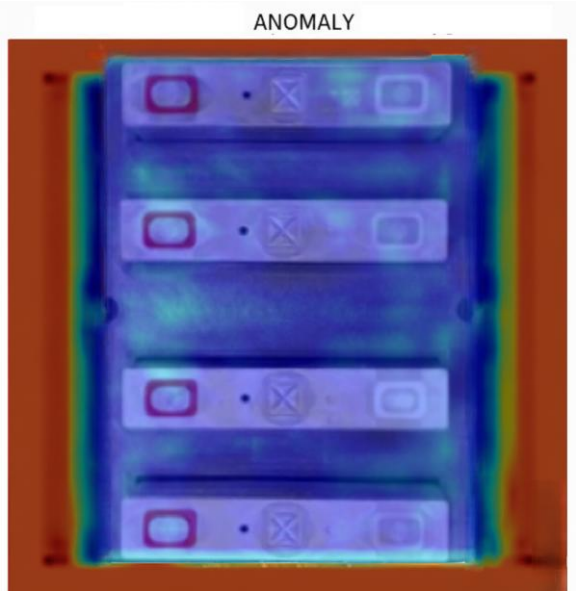

Figure 32: EfficienctAD Detects Under Background Variation

Figure 32 represents a case where the battery assembly is placed against a background different from the training data. The heatmap activation covers a wide region in red, leading to the image being flagged as anomalous. This demonstrates that EfficientAD can generalize beyond the trained environment and maintain anomaly detection performance even under background variation, thereby increasing its robustness in real-world industrial inspection.

4.4 Comparison of results

Metric	YOLOv7	Autoencoder	EfficientAD
mAP / AP	mAP@0.5 = 0.967 BatteryCell = 0.983 OuterBox = 0.982 RedTerminal = 0.960 WhiteTerminal = 0.941	AP ≈ 0.79–0.80	AP = 1.000
ROC–AUC	≈ 0.95–0.97 (estimated from PR and mAP curves)	0.944	1.000
Confusion Matrix (Correct/Total)	~65/66 (only 1 defect misclassified)	60/66 (32 good, 28 defect)	~65/66 (near perfect)
Strengths	Excellent at detecting structured objects (cells, boxes, terminals). Robust under background variation.	Simple and interpretable. Good for missing components. Provides anomaly heatmaps.	Outstanding accuracy. Detects missing, misplaced, and duplicated terminals. Strong heatmaps.
Limitations	Needs annotated data. Slight sensitivity to very small terminal misplacements.	Struggles with orientation-based errors and background changes. Lower recall.	May over-flag anomalies in unfamiliar conditions. Higher computational cost.
Best Use Case	Real-time industrial inspection where	Quick anomaly screening without	High-precision inspection where

	speed and structured detection are critical.	much labeled data. Pre-checks in production.	missing even small defects is unacceptable.

Table 3: Comparison of Results with Strength and Limitations

Table 3 represents the comparative results of YOLOv7, Autoencoder, and EfficientAD across multiple performance metrics. From the comparison, it is evident that EfficientAD achieved the highest accuracy and ROC–AUC, making it the most reliable method for detecting anomalies. YOLOv7 also performed strongly with high mAP values and robust detection of structured objects, making it suitable for real-time inspection tasks. The Autoencoder, while interpretable and effective at detecting missing components, showed weaker recall and struggled under background variations.

5 Discussion and Analysis

This study compared three complementary strategies for automated visual inspection of battery assembly: a supervised object detector (YOLOv7), a reconstruction-based autoencoder, and EfficientAD, which combines student-teacher features with an auxiliary reconstruction branch. The intention was not only to rank methods by headline metrics, but to understand what types of defects each method catches, what they miss, and how they behave under realistic change in background, lighting, and part orientation. The analysis below synthesizes quantitative outcomes with qualitative failure patterns to guide deployment decisions.

5.1 What each method is really good at

YOLOv7 translates explicit assembly rules into detections that are easy to act on. Presence, count, and polarity of terminals map naturally to classes and bounding boxes. When a red terminal is missing or swapped left-right, the model provides a direct, localized signal that a station can use for stop, rework, or robot correction. This supervised framing is powerful when the rules are known and the defect types are discrete.

Autoencoder excels at revealing absence or addition of structure relative to learned “normal.” Missing a cell or introducing an unexpected shape produces high residual energy and clean heatmaps. It is well suited when labeled anomalies are scarce, because it trains only on normal data and still yields interpretable overlays that help operators see what changed.

EfficientAD covers both local defects and global layout inconsistencies. Its feature comparison captures subtle texture changes in terminals and cell faces, while the decoder branch highlights misplaced or rotated components. In practice it behaves like a conservative gate that raises the alarm when the frame contains unfamiliar content, including genuine defects and certain environmental shifts.

5.2 Failure modes that matter on a Production line

Small parts and fine structures. Terminal terminals are small relative to 256 × 256 inputs. YOLOv7 sometimes trades recall for precision at stricter thresholds on these tiny regions, especially if contrast is poor. The autoencoder can under-react to slight terminal displacement because the global reconstruction is still plausible. EfficientAD keeps sensitivity high, but that sensitivity can also elevate scores when terminals are visually noisy due to glare.

Orientation and pose changes. Flipped cells created a clear split. YOLOv7, if trained with class labels that encode orientation or with sufficient augmentation, remains robust and flags orientation violations explicitly. The autoencoder often highlights only part of the flipped area, producing diffuse heat that is less actionable. EfficientAD responds strongly to pose inconsistency due to feature mismatch, which is desirable for safety.

Background and lighting shifts. A detector grounded in objects and anchors is less distracted by background texture, so YOLOv7 holds up as long as the parts remain visible. The autoencoder is the most sensitive to global background changes. It can paint broad red regions even when the assembly itself is correct. EfficientAD treats unfamiliar backgrounds as anomalies unless they have been seen during calibration. This is safe but may require downstream logic to whitelist benign changes.

5.3 Thresholding and calibration are as important as the model

Performance depends on where the operating point is set. For the unsupervised models, thresholds selected from validation ROC curves produced reasonable trade-offs between false alarms and misses. Practical deployment should include a standing procedure for recalibration after any upstream change in optics, lighting, or fixtures. Quantile-based recalibration on a short run of known-good images is a lightweight way to realign scores. For YOLOv7, per-class confidence tuning is essential, with slightly lower thresholds for small terminals and stricter thresholds for larger, visually distinct objects such as cell bodies and the outer box.

5.4 Summary of research findings

- **Which approach is most effective for this task.**

  On this dataset, EfficientAD yielded the strongest image-level separability and the most consistent localization when the anomaly had texture or layout components. YOLOv7 was the most reliable at enforcing deterministic rules such as presence, count, and left-right polarity, producing outputs that are directly usable for automation. The autoencoder added value as a transparent baseline and as a drift monitor, but it was the most fragile under background change and certain orientation anomalies. The most effective approach is not a single model, but a layered configuration that exploits complementary strengths.

- **How integration of the best method enhances quality and efficiency, and whether this generalizes.**

A two-stage pipeline is recommended. Stage one runs YOLOv7 to enforce explicit rules and provide fine-grained, class-level feedback to machines and operators. Stage two runs EfficientAD as a high-sensitivity safety net that catches texture defects, subtle misalignments, and unexpected content that bounding boxes might miss. The autoencoder remains in the loop as a lightweight drift sentinel and as an explainability tool for incident review. This architecture generalizes well to other assembly lines that combine rule-based constraints with subtle surface quality requirements, because it separates concerns between symbolic rule checking and distributional novelty detection.

5.5 Cost, latency, and maintainability

Labeling cost is non-trivial for YOLOv7, but the payoff is crisp supervision and straightforward maintainability once classes and rules stabilize. Unsupervised methods avoid defect labels and can be retrained quickly when a product variant arrives. Inference on an RTX 4050-class edge GPU is compatible with line speeds for all three methods. For real use, the models should be wrapped with simple services that log per-image scores, predicted labels, and heatmaps for traceability across shifts and batches.

5.6 Why results transfer beyond the lab

Although the dataset is modest and uses a fixed input resolution, several choices push toward external validity. First, all methods were trained under the same preprocessing and augmentation regime, which isolates model behavior rather than data trickery. Second, evaluation included deliberate background changes and pose variants to probe robustness. Third, operating points were selected by standard procedures rather than hand-tuned on test images. These practices reduce the risk of over-fitting to a narrow slice of conditions and make the conclusions more portable to similar stations.

5.7 Limitations of the Study

Although the results demonstrate promising performance across all three methods, several limitations should be acknowledged. First, the dataset used for training and evaluation was relatively small, consisting of images collected under controlled laboratory conditions. While this ensured data consistency, it may have limited the models' exposure to real-world variability such as illumination changes, camera noise, or mechanical misalignment during production. Second, due to time and hardware constraints, hyperparameter tuning was performed within a fixed search range, which may have prevented further performance optimization. Third, the evaluation focused primarily on

visual anomaly detection rather than full system integration with the robotic control loop, meaning that the feedback latency and operational response of the entire automation system were not empirically assessed. Lastly, the comparison was limited to three representative methods like YOLOv7, Autoencoder, and EfficientAD, while other recent architectures, such as Vision Transformers or hybrid diffusion-based models, were excluded from scope. These limitations define opportunities for future research, where larger datasets, expanded model sets, and full robotic integration could provide a more comprehensive understanding of real-time industrial deployment.

5.8 Ethical and Sustainability Considerations

The use of artificial intelligence in battery manufacturing raises important ethical and sustainability considerations because lithium-ion batteries are safety-critical products. Small defects introduced during assembly may not cause immediate failure but can later develop into serious problems such as internal short circuits, overheating, or thermal runaway. Research shows that many battery safety incidents originate from manufacturing defects that were not detected early enough (Grabow et al., 2023; Zhao et al., 2024). For this reason, ethical responsibility in this project is closely linked to improving inspection reliability and preventing unsafe products from reaching the market.

The main ethical concern addressed by this work is safety and harm prevention. Early detection of assembly errors reduces the risk that defective cells continue through production and are later used in real products. Studies on battery systems highlight that defects caused during manufacturing are difficult to identify and mitigate once the battery is in operation (Sun et al., 2023; Wang et al., 2024). By improving inspection at the assembly stage, AI-based visual inspection helps reduce foreseeable risks to users, production equipment, and the surrounding environment.

This project does not involve personal data or human subjects. All images were captured from an automated industrial setup and contain only mechanical battery components. No information related to people, identity, or behaviour is processed. As a result, common ethical issues related to privacy, surveillance, or data misuse do not apply. The system is therefore compliant with data-protection regulations such as GDPR by design.

Another ethical aspect is transparency and accountability in automated decision-making. In industrial environments, operators must be able to understand why a product is flagged as defective. Fully black-box systems can reduce trust and make fault investigation difficult. In this study, all evaluated methods provide interpretable outputs. YOLOv7 produces bounding boxes and

class labels that clearly indicate which assembly rule is violated, while the autoencoder and EfficientAD generate heatmaps that highlight abnormal image regions. Explainable outputs are widely recognised as essential for responsible industrial AI, as they support human oversight and informed intervention (Bergmann et al., 2022; Liu et al., 2024).

From a sustainability perspective, early anomaly detection helps reduce waste, rework, and energy consumption. Defects discovered late in production often lead to scrapping or energy-intensive reprocessing. In battery manufacturing, this has a significant environmental impact due to the materials and energy required to produce cells. Studies show that reliable early inspection can reduce scrap rates and improve resource efficiency in manufacturing systems (Stavropoulos et al., 2024; Zubayer et al., 2024).

Computational efficiency is also relevant to sustainability. The models used in this study were selected to operate in real time on moderate edge hardware rather than relying on energy-intensive cloud infrastructure. EfficientAD, in particular, was designed to achieve high accuracy with low computational cost (Batzner et al., 2024). Efficient models reduce operational energy use and support long-term sustainable deployment.

Finally, this work recognises the limitations of automation. No anomaly-detection system is perfect, and over-reliance on a single model may introduce risk. For this reason, the thesis recommends a layered inspection approach and assumes continued human supervision, regular recalibration, and validation when production conditions change. This aligns with best practices for ethical deployment of AI in safety-critical industrial systems (Zhao et al., 2024).

6 Conclusion and Future Work

6.1 Conclusion

This thesis compared three complementary approaches for automated visual inspection of battery assembly: a supervised object detector (YOLOv7), a reconstruction-based autoencoder, and EfficientAD with student-teacher features and a reconstruction branch. The comparison was carried out in a controlled 256 × 256 regime to ensure fair training and evaluation. The findings are consistent and actionable.

First, the supervised detector is the most direct way to enforce explicit assembly rules. Presence, count, and left-right polarity of terminals align naturally with classes and bounding boxes, which makes the outputs easy for operators and machines to act on. When a rule is violated, YOLOv7 provides a precise, localized signal that maps to stop and rework decisions.

Second, the autoencoder offers an unsupervised baseline that is simple to train when defect labels are scarce. It provides intuitive heatmaps and is particularly effective for missing or added structure. However, it is the most sensitive to global appearance shifts and it can be unreliable for subtle orientation errors, which limits its use as a sole gatekeeper on a live line.

Third, EfficientAD consistently shows the strongest image-level separability across both local and global anomalies. By comparing features from a teacher network and supplementing them with reconstruction, it detects texture defects, misalignments, and unfamiliar layouts with high sensitivity. In practice it behaves like a conservative safety net that complements the explicit rule checking of YOLOv7.

Taken together, the results support a layered pipeline. Stage one uses YOLOv7 to assert deterministic assembly constraints and provide class-level feedback. Stage two applies EfficientAD to catch distributional oddities and fine-grained texture issues that bounding boxes might miss. The autoencoder is retained as a lightweight drift sentinel and as an explainable lens during incident review. This architecture balances false stops against missed defects, improves operator trust through clear visual evidence, and is maintainable on an edge GPU typical of factory deployments.

The study also highlights the importance of thresholding and calibration. Operating points selected on validation data transfer well, but any change in optics, lighting, or fixtures should trigger a quick recalibration on known-good images. Finally, while the dataset is modest and downsampled, the training

controls and evaluation design reduce the risk of over-fitting and make the conclusions portable to similar stations.

6.2 Future Work

1) Hybrid YOLO + EfficientAD with a shared backbone and dual heads that combine detection loss and anomaly loss, fusing anomaly scores with class confidences for a single calibrated decision.
2) cross-distillation between heads to transfer EfficientAD feature embeddings into the YOLO head for small-object recall and YOLO attention maps into the anomaly head for sharper localization.
3) uncertainty-aware gating using Monte Carlo dropout or small ensembles so rework is triggered by joint anomaly score and predictive uncertainty.
4) Label-efficient learning through semi-supervised fine-tuning on a small curated defect set and active learning that prioritizes high-uncertainty or high-disagreement frames.
5) Higher input resolution (384 to 512) with anchor and tiling retuning for terminal-scale objects while profiling edge latency and memory;
6) A score calibration protocol using quantile-based thresholds after any change in optics, lighting, or background, plus per-shift dashboards of score distributions.
7) Synthetic stress tests that generate controlled glare, occlusion, and background textures to probe failure boundaries and augment rare cases.
8) Multi-view fusion by adding a second camera and enforcing geometric consistency across views to mitigate occlusions and pose ambiguity;
9) Online drift monitoring with a lightweight autoencoder sentinel, statistical shift tests, and automatic triggers for quick recalibration or short retrains.
10) Operational KPIs that track accuracy, false-stop rate, rework precision, mean time to detection, and calibration stability, using KPI thresholds to decide when a model is promoted to production.

7 References

Badmos, O., Kopp, A., Bernthaler, T., & Schneider, G. (2020). Image-based defect detection in lithium-ion battery electrode using convolutional neural networks. *Journal of Intelligent Manufacturing*, *31*(4), 885–897. https://doi.org/10.1007/S10845-019-01484-X/FIGURES/6

Choudhary, N., Clever, H., Ludwigs, R., Rath, M., Gannouni, A., Schmetz, A., Hülsmann, T., Sawodny, J., Fischer, L., Kampker, A., Fleischer, J., & Stein, H. S. (2022). Autonomous Visual Detection of Defects from Battery Electrode Manufacturing. *Advanced Intelligent Systems*, *4*(12). https://doi.org/10.1002/AISY.202200142

Faust, O., Kareem, M., Shenfield, A., Ali, A., & Acharya, U. R. (2020). Validating the robustness of an internet of things based atrial fibrillation detection system. *Pattern Recognition Letters*, *133*, 55–61. https://doi.org/10.1016/J.PATREC.2020.02.005

Kharitonov, A., Nahhas, A., Pohl, M., & Turowski, K. (2022). Comparative analysis of machine learning models for anomaly detection in manufacturing. *Procedia Computer Science*, *200*, 1288–1297. https://doi.org/10.1016/j.procs.2022.01.330

Liu, J., Xie, G., Wang, J., Li, S., Wang, C., Zheng, F., & Jin, Y. (2024). Deep Industrial Image Anomaly Detection: A Survey. In *Machine Intelligence Research* (Vol. 21, Issue 1, pp. 104–135). Chinese Academy of Sciences. https://doi.org/10.1007/s11633-023-1459-z

Sun, C., He, Z., Lin, H., Cai, L., Cai, H., & Gao, M. (2023). Anomaly detection of power battery pack using gated recurrent units based variational autoencoder. *Applied Soft Computing*, *132*, 109903. https://doi.org/10.1016/J.ASOC.2022.109903

Vu, T. T. H., Chang, T. W., & Kim, H. (2024). Enhancing Quality Control in Battery Component Manufacturing: Deep Learning-Based Approaches for Defect Detection on Microfasteners. *Systems*, *12*(1). https://doi.org/10.3390/SYSTEMS12010024

Zubayer, M. H., Zhang, C., & Yafei, W. (2024). Comparative Analysis of Deep Learning-Based Defect Monitoring in Metal Additive Manufacturing. *2024 4th International Conference on Computer, Control and Robotics, ICCCR 2024*, 89–95. https://doi.org/10.1109/ICCCR61138.2024.10585345

Grabow, J., Klink, J., Benger, R., Hauer, I., & Beck, H.-P. (2023). Particle contamination in commercial lithium-ion cells: Risk assessment with focus on internal short circuits and replication by currently discussed trigger methods. *Batteries, 9*(1), 9. https://doi.org/10.3390/batteries9010009

Wang, J., Yang, Z., Wang, S., Yang, H., Du, M., & Song, J. (2024). An early micro internal short circuit fault diagnosis method based on accumulated correlation

coefficient for lithium-ion battery pack. *Energies, 17*(23), 6071. https://doi.org/10.3390/en17236071

Zhao, J., Feng, X., Tran, M.-K., Fowler, M., Ouyang, M., & Burke, A. F. (2024). Battery safety: Fault diagnosis from laboratory to real world. Journal of Power Sources, 598, 234111. https://doi.org/10.1016/j.jpowsour.2024.234111

Stavropoulos, P., Sabatakakis, K., & Bikas, H. (2024). Manufacturing challenges in electric vehicle battery pack and module welding: A review. *Batteries, 10*(5), 146. https://doi.org/10.3390/batteries10050146

Song, W., Chen, T., Gu, Z., Gai, W., Huang, W., & Wang, B. (2015). Wood materials defects detection using image block percentile color histogram and Eigenvector Texture feature. *Advances in Intelligent Systems Research/Advances in Intelligent Systems Research*. https://doi.org/10.2991/icismme-15.2015.163

Xing, Y., Liu, Q., Xing, Y., Liu, Z., & Wang, W. (2024). Minimal defect detection on end face of lithium battery shells. *Sustainability*, *16*(23), 10502. https://doi.org/10.3390/su162310502

Wang, J., Wang, Y., Wu, X., & Ci, W. (2023). Robust MPC for polytopic uncertain systems via a high-rate network with the round-robin scheduling. *PeerJ Computer Science*, *9*, e1269. https://doi.org/10.7717/peerj-cs.1269

Xu, C., Li, L., Li, J., & Wen, C. (2021). Surface Defects Detection and identification of lithium battery pole piece based on Multi-Feature Fusion and PSO-SVM. *IEEE Access*, *9*, 85232–85239. https://doi.org/10.1109/access.2021.3067641

Solawetz, J. (2024, January 4). *What is YOLOv7? A complete guide*. Roboflow Blog [Image]. https://blog.roboflow.com/yolov7-breakdown/

Medium. (2019, March 12). *Autoencoder network architecture* [Image]. https://miro.medium.com/v2/resize:fit:750/format:webp/1*nqzWupxC60iAH2dYrFT78Q.png

Medium. (2024, May 8). *EfficientAD network architecture* [Image]. https://miro.medium.com/v2/resize:fit:1400/1*UPL4Z2101Q8ZJtZ4kbenPQ.png

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696.* https://doi.org/10.48550/arXiv.2207.02696

Zhang, J., Yang, W., Chen, Y., Ding, M., Huang, H., Wang, B., Gao, K., Chen, S., & Du, R. (2024). Fast object detection of anomaly photovoltaic (PV) cells using deep neural networks. *Applied Energy, 372(C)*, 123759. https://doi.org/10.1016/j.apenergy.2024.123759

Zou, W., Wang, Q., Wang, Z., Xu, Y., & He, Z. (2025). YOLO-SRSA: An improved YOLOv7 network for the abnormal detection of power equipment. *Information, 16*(5), 407. https://doi.org/10.3390/info16050407

Kou, X., Liu, S., Cheng, K., & Qian, Y. (2021). Development of a YOLO-V3-based model for detecting defects on steel strip surface. *Measurement*, 182, 109454. https://doi.org/10.1016/j.measurement.2021.109454

Wang, C., Zhou, Z., & Chen, Z. (2022). An enhanced YOLOv4 model with self-dependent attentive fusion and component randomized mosaic augmentation for metal surface defect detection. *IEEE Access*, 10, 97758–97766. https://doi.org/10.1109/ACCESS.2022.3203198

Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., & Wei, X. (2022). YOLOv6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*. https://doi.org/10.48550/arXiv.2209.02976

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7464–7475). IEEE. https://doi.org/10.1109/CVPR52729.2023.00721

Zhang, Y., Jia, R., & Yu, L. (2024). Yolov7-Tinier: Towards high-precision and lightweight detection of fabric defects in textile plant. *Fibers and Polymers*, 25(5–6), 3549–3562. https://doi.org/10.1007/s12221-024-00662-w

Liu, Y., Liu, Y., Guo, X., Ling, X., & Geng, Q. (2025). Metal surface defect detection using SLF-YOLO enhanced YOLOv8 model. *Scientific Reports*, *15*(1). https://doi.org/10.1038/s41598-025-94936-9

Ren, J., Zhang, H., & Yue, M. (2025). YOLOv8-WD: Deep learning-based detection of defects in automotive brake joint laser welds. *Applied Sciences*, 15(3), 1184. https://doi.org/10.3390/app15031184

Xu, M., Yoon, S., Fuentes, A., & Park, D. S. (2023). A comprehensive survey of image augmentation techniques for deep learning. Pattern Recognition, 137, 109347. https://doi.org/10.1016/j.patcog.2023.109347.

Berahmand, K., Daneshfar, F., Salehi, E. S., Li, Y., & Xu, Y. (2024). Autoencoders and their applications in machine learning: A survey. Artificial Intelligence Review, 57, 28. https://doi.org/10.1007/s10462-023-10662-6

Givnan, S., Chalmers, C., Fergus, P., Ortega-Martorell, S., & Whalley, T. (2022). Anomaly detection using autoencoder reconstruction upon industrial motors. Sensors, 22(9), 3166. https://doi.org/10.3390/s22093166

Mehdizadeh, M., Tavakoli Tafti, K., & Soltani, P. (2023). Evaluation of histogram equalization and contrast limited adaptive histogram equalization effect on image quality and fractal dimensions of digital periapical radiographs. Oral Radiology, 39(2), 418–424. https://doi.org/10.1007/s11282-022-00654-7

Xu, M., Yoon, S., Fuentes, A., & Park, D. S. (2023). A comprehensive survey of image augmentation techniques for deep learning. Pattern Recognition, 137, 109347. https://doi.org/10.1016/j.patcog.2023.109347

Baur, C., Denner, S., Wiestler, B., Navab, N., & Albarqouni, S. (2021). Autoencoders for unsupervised anomaly segmentation in brain MR images: A comparative study. *Medical Image Analysis*, *69*, 101952. https://doi.org/10.1016/j.media.2020.101952

Mujkic, E., Philipsen, M. P., Moeslund, T. B., Christiansen, M. P., & Ravn, O. (2022b). Anomaly detection for agricultural vehicles using autoencoders. *Sensors*, *22*(10), 3608. https://doi.org/10.3390/s22103608

Neloy, A. A., & Turgeon, M. (2024). A comprehensive study of auto-encoders for anomaly detection: Efficiency and trade-offs. *Machine Learning With Applications*, *17*, 100572. https://doi.org/10.1016/j.mlwa.2024.100572

Wang, J., Wang, Y., Wu, X., & Ci, W. (2023). Robust MPC for polytopic uncertain systems via a high-rate network with the round-robin scheduling. *PeerJ Computer Science*, *9*, e1269. https://doi.org/10.7717/peerj-cs.1269

Yapp, E. K., & Doan, N. C. (2024). Anomaly detection on MVTec AD using VQ-VAE-2. *Procedia CIRP*, *130*, 1809–1814. https://doi.org/10.1016/j.procir.2024.10.320

Kim, J. M., Choe, J., Akata, Z., & Oh, S. J. (2021). Keep CALM and improve visual feature attribution. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 8330–8340. https://doi.org/10.1109/iccv48922.2021.00824

Akcay, S., Atapour-Abarghouei, A., & Breckon, T. P. (2019). GANoMaly: Semi-supervised anomaly detection via adversarial training. In *Lecture notes in computer science* (pp. 622–637). https://doi.org/10.1007/978-3-030-20893-6_39

Carrara, F., Amato, G., Brombin, L., Falchi, F., & Gennaro, C. (2021). Combining GANs and AutoEncoders for efficient anomaly detection. *2022 26th International Conference on Pattern Recognition (ICPR)*. https://doi.org/10.1109/icpr48806.2021.9412253

Zhang, L., Dai, Y., Fan, F., & He, C. (2022). Anomaly detection of GAN industrial image based on attention feature fusion. *Sensors*, *23*(1), 355. https://doi.org/10.3390/s23010355

Batzner, K., Heckler, L., & König, R. (2023). EfficientAD: Accurate Visual Anomaly Detection at Millisecond-Level Latencies. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2303.14535

Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., & Steger, C. (2022). Beyond dents and scratches: logical constraints in unsupervised anomaly detection and localization. *International Journal of Computer Vision*, *130*(4), 947–969. https://doi.org/10.1007/s11263-022-01578-9

Tan, M., & Le, Q. (2021). EfficientNetV2: Smaller models and faster training. *International Conference on Machine Learning*, 10096–10106. http://proceedings.mlr.press/v139/tan21a/tan21a.pdf

Rudolph, M., Wehrbein, T., Rosenhahn, B., & Wandt, B. (2022). Asymmetric Student-Teacher networks for industrial anomaly detection. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2210.07829

Schlegl, T., Seeböck, P., Waldstein, S. M., Langs, G., & Schmidt-Erfurth, U. (2019). f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis*, *54*, 30–44. https://doi.org/10.1016/j.media.2019.01.010

Zavrtanik, V., Kristan, M., & Skočaj, D. (2021). DRAEM -- A discriminatively trained reconstruction embedding for surface anomaly detection. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2108.07610

Chen, S., Bandaru, S., Marti, S., Bekar, E. T., & Skoogh, A. (2025). Comparison of unsupervised image anomaly detection models for sheet metal glue lines. *Engineering Applications of Artificial Intelligence*, *153*, 110740. https://doi.org/10.1016/j.engappai.2025.110740

Raj, G. D., & Prabadevi, B. (2025). ENHANCING SURFACE DETECTION : A COMPREHENSIVE ANALYSIS OF VARIOUS YOLO MODELS. *Heliyon*, *11*(3), e42433. https://doi.org/10.1016/j.heliyon.2025.e42433

Tang, T., Kuo, W., Lan, J., Ding, C., Hsu, H., & Young, H. (2020). Anomaly Detection Neural Network with Dual Auto-Encoders GAN and Its Industrial Inspection Applications. *Sensors*, *20*(12), 3336. https://doi.org/10.3390/s20123336