Master thesis

Automated Classification of bp-sWEEE X-Ray Images via Web Scraping and UNU-KEYS Taxonomy for Automatic Disassembly

carried out for the purpose of obtaining the academic degree of a

Diplom-Ingenieur

submitted to the

Vienna University of Technology
Faculty of Mechanical Engineering and Management Sciences

under the supervision of:
Univ.-Prof. Dr.-Ing Sebastian Schlund
(E330 Institute of Management Sciences, Department: Human-Machine Interaction)

and

Univ.-Ass. Dipl.-Ing. Sebastian Seisl B.Sc.
(E330 Institute of Management Sciences, Department: Industrial Engineering)

by

Anas Ben Hammad
[redacted]

Wien, 21.06.2025

Anas Ben Hammad

I have taken note that I am authorised to submit my thesis for printing under the title

Automated Classification of bp-sWEEE X-Ray Images via Web Scraping and UNU-KEYS Taxonomy for Automatic Disassembly

only with the authorisation of the examination board.

I further declare in lieu of oath that I have completed my thesis independently in accordance with the recognised principles of academic writing and that I have cited all the resources used, the literature on which it is based. Furthermore, I declare that I have not previously submitted this thesis topic in any form as an examination paper either in Austria or abroad (to an assessor for assessment) and that this thesis corresponds to the thesis assessed by the assessor.

Wien, 21.06.2025

Anas Ben Hammad

Acknowledgements

I express my profound gratitude to **EIT Manufacturing** for awarding me the prestigious scholarship that made this master's programme possible. This transformative opportunity not only provided essential financial support but also immersed me in a dynamic ecosystem of innovation, sustainability, and cross-European collaboration.

My deepest appreciation extends to the **Technical University of Vienna** and its Faculty of Mechanical Engineering and Management Sciences for their world-class academic environment. I am indebted to **Univ.-Prof. Dr.-Ing. Sebastian Schlund** for his visionary guidance. My sincere thanks also go to **Univ.-Ass. Dipl.-Ing. Sebastian Seisl, B.Sc.**, whose technical expertise, rigorous feedback, and steadfast encouragement were indispensable throughout this journey.

To **Mondragon University**, I convey heartfelt gratitude for enriching my perspective on robotics engineering. Special recognition is reserved for **Prof. Dr. Eñaut Muxika**, whose exceptional mentorship during my studies refined my systems-thinking approach.

My family—my parents, siblings, and loved ones—deserve immeasurable recognition. Your sacrifices, emotional resilience, and boundless belief in my ambitions sustained me through challenges. This milestone is a testament to your unwavering support.

Finally, I acknowledge peers at both institutions for fostering a collaborative spirit, and EIT Manufacturing's network for catalysing interdisciplinary growth. This achievement stands on the shoulders of your collective generosity.

Kurzfassung

Diese Arbeit entwickelt einen computergestützten Rahmen zur Verbesserung der automatisierten Demontage von batteriebetriebenem kleinem Elektro- und Elektronik-Altgeräten (bp-sWEEE). Unter Nutzung von Web Scraping und der UNU-KEYS-Taxonomie etabliert die Forschung eine Prozesskette zur Erfassung, Klassifizierung und Validierung von Röntgenbildern bp-sWEEE-Geräte. Die Studie adressiert zentrale Herausforderungen der robotergestützten Demontage, darunter Gerätevariabilität, Sicherheitsrisiken und Designinkonsistenzen. Sie bewertet technologische Fortschritte in maschinellem Sehen und adaptiver Robotik und identifiziert kritische Barrieren sowie Erfolgsfaktoren für skalierbare Automation. Ein kuratierter Datensatz von Röntgenbildern wird zusammengestellt und rigoros validiert, wodurch die Machbarkeit automatisierter Klassifizierungssysteme demonstriert wird. Die Arbeit unterstreicht die Notwendigkeit standardisierter Metadatenschemata und kooperativer Industrie-Hochschul-Anstrengungen zur Förderung kreislaufwirtschaftlicher Praktiken im Elektroschrott-Recycling.

Keywords: e-waste recycling, UNU-KEYS, bp-sWEEE, Web scraping, Automation, Automated disassembly.

Abstract

This thesis develops a computational framework to enhance the automated disassembly of battery-powered small waste electrical and electronic equipment (bp-sWEEE). Leveraging web scraping and the UNU-KEYS taxonomy, the research establishes a pipeline for acquiring, classifying, and validating X-ray images of bp-sWEEE devices. The study addresses key challenges in robotic disassembly, including device variability, safety risks, and design inconsistencies. It evaluates technological advancements in machine vision and adaptive robotics while identifying critical barriers and enablers for scalable automation. A curated dataset of X-ray images is compiled and rigorously validated, demonstrating the feasibility of automated classification systems. The work underscores the need for standardized metadata schemas and collaborative industry academia efforts to advance circular economy practices in e-waste recycling.

Keywords: e-waste recycling, UNU-KEYS, bp-sWEEE, Web scraping, Automation, Automated disassembly.

Contents

1 Introduction **3**
1.1 Background and Motivation 3
1.2 Problem Statement 4
1.3 Research Objectives and Questions 5
1.3.1 Research Objectives: 5
1.3.2 Research Questions: 5

2 Literature Review **7**
2.1 Current State of the Art in Automated Screw Opening 7
2.1.1 Machine Vision and AI-Driven Detection: 7
2.1.2 Torque-Adaptive Robotic Systems: 8
2.1.3 Modular and Multi-Functional End-Effectors: 8
2.2 Barriers and Enablers for Automated Disassembly 9
2.3 Connection Elements in bp-sWEEE: 15
2.3.1 Screw Types and Usage: 15
2.3.2 Adhesive Technologies: 18
2.3.3 Snap-Fit and Modular Connection Systems: 20
2.4 Design Principles in bp-sWEEE: 22

3 Methodology: **25**
3.1 Research Design: 25
3.2 Systematic Literature Review Method: 26

4 Technical Implementation: **30**
4.1 Web Scraping and Classification System: 30
4.2 Dataset Validation Framework: 40
4.3 Automated Evaluation Suite: 49

5 Results: **62**
5.1 Findings from Systematic Literature Review: 62
5.2 Observations from Web Scraping and X-ray Analysis: 64

6 Discussion **67**
6.1 Interpretation of Results: 67

7 Conclusion **70**

1 Introduction

1.1 Background and Motivation

The exponential growth of battery-powered small waste electrical and electronic equipment (bp-sWEEE), including smartphones, wireless earbuds, and wearable devices, has become a critical environmental and economic challenge. Globally, over 53 million metric tons of e-waste were generated in 2023, with bp-sWEEE accounting for 30% of this volume due to short product lifecycles and rapid technological obsolescence [1]. These devices contain valuable materials such as cobalt, lithium, and rare earth metals, yet less than 20 % are recycled due to inefficiencies in disassembly processes [2].

Automated disassembly is pivotal for enabling circular economy practices, as manual disassembly is labor-intensive, costly, and hazardous. For instance, lithium-ion batteries integrated into 90% of bp-sWEEE pose significant fire and explosion risks during manual handling due to thermal runaway, releasing toxic gases such as hydrogen fluoride [3]. Furthermore, the high variability in design (e.g., proprietary screws, adhesives, and nonmodular structures) complicates the development of universal automated systems, perpetuating the reliance on unsustainable shredding methods that degrade material quality [4].

The European Union's Circular Economy Action Plan and the WEEE Directive emphasize the urgent need for scalable, automated recycling solutions to meet 2030 sustainability targets [5]. However, current research disproportionately focuses on large appliances (e.g., washing machines), leaving bp-sWEEE a fast-growing waste stream understudied [6]. This gap hinders progress toward efficient material recovery and underscores the necessity of this research.

1.2 Problem Statement

The automation of disassembly processes for bp-sWEEE is impeded by three interrelated challenges:

- High Device Variability and Uncertainty:
  The lack of standardized design principles across manufacturers results in diverse connection elements (e.g., Torx, Phillips, and Pentalobe screws; epoxy adhesives; snap-fit clips). For example, a teardown analysis of 50 smartphones revealed 12 distinct screw types and 8 adhesive patterns, necessitating adaptive robotic systems [7]. This variability complicates pre-programmed automation and increases operational costs.

- Unknown Internal Structures:
  Prior to disassembly, internal connection elements and battery placements are often undocumented, requiring real-time detection systems. Current technologies, such as X-ray imaging, achieve only 75% accuracy in identifying internal screws, delaying workflows [8].

- Safety and Economic Barriers:
  Lithium-ion batteries, present in 90% of bp-sWEEE, pose fire hazards if damaged during disassembly. A study documented incidents of thermal runaway in recycling facilities, causing equipment damage and worker injuries [9]. Additionally, the upfront cost of robotic disassembly systems (e.g., $ 500,000 for a modular robotic cell) limits adoption among small-scale recyclers [10].

Existing classification systems, such as UNU-KEYS, prioritize functional categories (e.g., “personal care devices”) over disassembly-oriented criteria, further hindering automation efforts [11].

1.3 Research Objectives and Questions

This thesis establishes a computational framework to acquire, classify, and evaluate X-ray images of battery-powered small waste electrical and electronic equipment (bp-sWEEE) through automated web scraping, UNU-KEYS classification, and quality validation protocols. The objectives and research questions are structured as follows:

1.3.1 Research Objectives:

- Map the current state of automated screw opening technologies and their limitations.
- Create a dataset of X-ray images representing diverse bp-sWEEE device categories through automated web scraping.
- Develop a metadata schema to classify scraped images by UNU-Keys

1.3.2 Research Questions:

- RQ1: What are the technological advancements and limitations in automated screw opening for bp-sWEEE?
- RQ2: What are the key barriers (e.g., device variability, battery hazards) and enablers (e.g., modular design) for automating screw opening?
- RQ3: What criteria ensure the relevance and quality of X-ray images for analyzing connection elements?
- RQ4: What critical and supplementary metadata are essential for optimizing automated disassembly processes for bp-sWEEE?

This study employs a three-phase mixed-methods design to address the research objectives, focusing on web scraping, database creation, and validation/evaluation. The workflow is structured as follows:

- Phase 1: Web Scraping and Data Collection
  - Objective: Build a database of X-ray images of small WEEE (bp-sWEEE) using automated web scraping.
  - Output: Raw dataset of X-ray images with metadata.

- Phase 2: Database Creation and UNU-KEYS Classification
  - Objective: Organize images into a structured database grouped by UNU-KEYS categories.
  - Output: Tagged dataset with metadata.
- Phase 3: Validation and Evaluation
  - Objective: Assess data quality, relevance, and usability for disassembly automation research.
  - Output: Cleaned dataset, performance metrics, and evaluation report.

2 Literature Review

2.1 Current State of the Art in Automated Screw Opening

Automated screw opening technologies have witnessed significant evolution over the past decade, propelled by rapid advancements in robotics, machine vision, and artificial intelligence (AI). These sophisticated systems are primarily designed to overcome the inherent challenges presented by the high variability in screw types, sizes, and their pervasive miniaturization within battery-powered small waste electrical and electronic equipment (bp-sWEEE). The ultimate goal is to enable efficient and damage-free disassembly, crucial for material recovery and circular economy initiatives.

2.1.1 Machine Vision and AI-Driven Detection:

A cornerstone of modern robotic disassembly systems is their reliance on advanced machine vision capabilities to accurately identify and locate screws in real time. Chuangchuang Zhou et al. [12] have notably demonstrated the efficacy of deep learning models, particularly architectures like YOLOv5, in achieving high detection accuracies. Their research indicated a remarkable 95% accuracy in identifying various screw types, including Torx, Phillips, and hex screws, across a diverse range of bp-sWEEE devices such as smartphones and tablets. This precision is vital for the robot to select the correct tool and apply the appropriate force. However, real-world complexities introduce significant hurdles. Occlusion caused by elements like adhesives, labels, or intricate device structures remains a persistent challenge, often reducing detection accuracy to approximately 75% in complex devices such as waterproof fitness trackers [13]. To mitigate this, innovative approaches have emerged. Zhang et al. [15] proposed integrating thermal imaging with convolutional neural networks (CNNs), ingeniously leveraging the distinct heat signatures emitted by metallic screws. This method proved particularly effective in improving detection rates by an additional 15% in devices encased in non-metallic materials, where visual occlusions are prevalent.

2.1.2 Torque-Adaptive Robotic Systems:

Precision torque control is an absolutely critical factor in preventing damage to the often delicate internal components during the screw removal process. Rizova and Colledani [16] have made significant contributions in this area by developing robotic arms equipped with highly sensitive force-torque sensors. These systems are capable of dynamically adjusting torque settings within a range of 0.2–0.8 Nm, adapting to the specific screw size, material, and tightness. In a comprehensive case study involving the disassembly of 100 smartphones, their adaptive approach led to a substantial 30% reduction in instances of screw stripping or component damage when compared to traditional fixed-torque systems. Expanding on this, Micropsi Industries [17] has successfully implemented collaborative robots (cobots) that work in close cooperation with human operators. This hybrid approach synergizes human dexterity and problem-solving capabilities with robotic precision, particularly for unscrewing components in devices where batteries are embedded or where complex manipulation is required. Such human-robot collaboration enhances both safety and efficiency in disassembly lines.

2.1.3 Modular and Multi-Functional End-Effectors:

The pervasive lack of standardization in screw designs across different electronic devices necessitates highly adaptable tooling systems. To address this, Hyeonjun Park et al. [18] designed a sophisticated modular end-effector system featuring 12 interchangeable bits. This innovative design allows the robot to switch between different tools in less than 2 seconds, a crucial capability for maintaining high throughput in a diverse disassembly stream. Their system achieved an impressive 98% success rate in disassembling over 50 different smartphone models, a significant improvement compared to the mere 65% success rate typically observed with single-tool systems [19]. Further advancements in this area include rapid 3D printing of custom end-effectors for highly proprietary screw types, enabling on-demand tool creation for unique disassembly challenges. The development of universal gripping mechanisms that can accommodate a wider range of screw head geometries without requiring a tool change is also an active area of research.

2.2 Barriers and Enablers for Automated Disassembly

Automated disassembly of small waste electrical and electronic equipment (bp-sWEEE) faces several significant barriers that hinder its widespread implementation. These barriers span technical challenges, safety concerns, and economic constraints, each strongly affecting the feasibility and efficiency of robotic disassembly systems.

- Barriers:

  - Device Variability and Complexity:

    One of the foremost technical barriers is the vast variability in device design and construction. A teardown analysis of 200 bp-sWEEE devices revealed the presence of 14 different screw types and 9 distinct adhesive patterns used across various models [21]. Premium brands predominantly utilize Torx screws, accounting for approximately 80% of their fasteners, while budget devices mainly employ Phillips screws, found in about 60% of these products [22]. This diversity in fastening mechanisms necessitates that recycling facilities maintain extensive inventories of specialized tools and adapt robotic end-effectors to handle multiple fastener types [23]. The lack of standardization in screw types, adhesive use, and component layout significantly complicates the programming and operation of automated disassembly systems, resulting in increased cycle times and reduced throughput [24]. Moreover, the internal architecture of devices varies widely, with components positioned differently and secured by diverse methods, further complicating robotic grasping and removal [25]. This heterogeneity forces recyclers to either invest in highly flexible but expensive robotic solutions or rely on manual disassembly, undermining the cost-effectiveness and scalability of automation efforts.

– Safety Risks Associated with Lithium-Ion Batteries:

  Lithium-ion batteries, integrated into approximately 90% of bp-sWEEE devices, pose a critical safety barrier during automated disassembly. These batteries are prone to thermal runaway if punctured or improperly handled, which can lead to fires and the release of toxic gases such as hydrogen fluoride [26]. Studies have documented that around 2% of lithium-ion batteries subjected to mechanical stress during recycling processes enter thermal runaway, posing significant hazards to both personnel and equipment [23]. The risk is increased by the widespread use of adhesives and encapsulation methods that make battery removal challenging without damage [21]. Automated systems must therefore incorporate sophisticated sensing and manipulation capabilities to detect battery location and condition, and to extract batteries safely without compromising their integrity [22]. However, current sensor technologies often struggle to reliably identify batteries in damaged or partially disassembled devices, increasing the risk of accidental puncture [23]. Consequently, facilities must invest in specialized containment and fire suppression systems, which add substantial cost and complexity to robotic disassembly cells.

– Economic Constraints and High Capital Costs:

  The economic barrier to automated disassembly is pronounced, particularly for small and medium-sized recyclers. The upfront capital investment for robotic disassembly systems typically exceeds $500,000, with annual maintenance, calibration, and software updates adding around $50,000 in recurring costs [27]. Such high expenditures are prohibitive for recyclers processing limited volumes of bp-sWEEE, who often operate on thin margins. Additionally, the fluctuating market value of recovered materials and the variability in device composition create uncertainty regarding the return on investment [27]. Labor cost savings achieved through automation—estimated at 60-70%—are frequently offset by the need for skilled operators and technicians to manage complex robotic systems, resulting in additional training and personnel costs [28]. The breakeven point for automation investments generally requires processing volumes of over 200,000 units annually, a scale unattainable for many regional recyclers [27]. This economic reality discourages widespread adoption and perpetuates reliance on manual disassembly, which is labor-intensive and less efficient [28].

– Technical Limitations in Robotic Dexterity and Adaptability:

  Robotic systems currently face limitations in dexterity and adaptability required for the nuanced tasks involved in disassembling diverse bp-sWEEE devices. The intricate and delicate nature of many components demands precise force control and flexible manipulation capabilities that most industrial robots lack [22]. For example, unscrewing tiny fasteners, peeling adhesives, and disconnecting fragile connectors require advanced tactile sensing and compliant end-effectors [23]. Although recent advances in collaborative robotics and AI-driven vision systems have improved these capabilities, robots still struggle with unpredictable variations in device condition, such as wear, deformation, or prior damage [21]. This unpredictability often necessitates human intervention, reducing the overall automation efficiency [22]. Furthermore, the integration of multiple tool changers to handle various fastener types increases system complexity and downtime during tool swaps, limiting throughput [23].

– Regulatory and Standardization Gaps:

  A less tangible but equally impactful barrier is the lack of regulatory mandates and industry-wide standards for design for disassembly. Without enforced guidelines requiring manufacturers to adopt modular and standardized designs, recyclers must contend with the full spectrum of device variability [24]. The absence of standardized labeling for components, fasteners, and disassembly sequences complicates the development of universal robotic disassembly protocols [25]. Although emerging regulations such as the European Union's Critical Raw Materials Act aim to address these issues by promoting digital product passports and design-for-recycling criteria, widespread implementation remains years away [24]. Until such standards are broadly adopted, recyclers face ongoing challenges in automating disassembly processes efficiently [24].

- Enablers:
  - Design for Disassembly (DfD):

    It is a fundamental enabler that directly addresses the technical complexity barrier by embedding disassembly considerations into the product design phase. Modular product architectures, characterized by snap-fit casings, standardized screws, and minimized use of adhesives, significantly reduce the complexity and time required for disassembly. Empirical studies demonstrate that modular designs can reduce disassembly time by approximately 40% compared to traditional designs relying heavily on adhesives and diverse fasteners [62]. For example, smartphones designed with snap-fit clips instead of glue enable non-destructive disassembly, improving component recovery rates by up to 25% [65].

    The adoption of DfD principles also facilitates the development of automated disassembly protocols by reducing variability in component attachment methods and locations. Standardizing fastener types and placement allows robotic systems to use fewer tool types and simplifies programming, which directly improves throughput and reduces error rates [21]. Moreover, modular designs extend product lifespans by enabling easy repair and upgrade, thereby reducing waste generation and supporting circular economy goals [33].

    Despite these advantages, widespread adoption of DfD faces challenges, including trade-offs with product aesthetics, structural integrity, and manufacturing costs. Nonetheless, regulatory pressure and consumer demand for sustainable electronics are driving manufacturers to increasingly incorporate DfD features [35].
  - Hybrid Human-Robot Collaboration:

    Fully automated disassembly remains economically and technically challenging due to the high variability and complexity of bpsWEEE. Hybrid human-robot collaboration (HRC) systems emerge as a powerful enabler by combining the cognitive flexibility and problem-solving skills of humans with the precision, repeatability, and endurance of robots [33]. Recent reviews highlight that

Human-Robot Collaboration can significantly enhance disassembly efficiency, safety, and scalability by allocating tasks according to their suitability for humans or robots [33].

Pilot implementations in smartwatch disassembly facilities have demonstrated that cobots (collaborative robots) can reduce manual intervention by up to 70%, handling repetitive or hazardous tasks such as unscrewing and battery removal while humans manage complex or unpredictable steps [21]. Cobots equipped with force and tactile sensors can adapt to subtle variations in device conditions, reducing damage to components and enhancing safety. Furthermore, Human-Robot Collaboration systems improve worker ergonomics by offloading physically demanding or dangerous tasks, thereby reducing injury risk and improving job satisfaction [33].

The integration of AI-driven vision systems and machine learning algorithms enables continuous improvement in disassembly strategies, allowing robots to learn from human demonstrations and adapt to new device models [33]. However, successful HRC implementation requires multidisciplinary research to address challenges related to human factors, robot control, and workflow integration [33].

– Policy Incentives and Regulatory Support:

Economic incentives and regulatory frameworks are vital enablers that lower the financial and operational barriers to adopting automated disassembly technologies. The European Union's Circular Economy Action Plan exemplifies this by subsidizing up to 30% of automation costs for recyclers investing in robotic systems [35]. Such subsidies alleviate the high upfront capital expenditures and reduce payback periods, making automation accessible to a broader range of recyclers, including small and medium enterprises.

In addition to financial support, regulations mandating design-for-recycling criteria and digital product passports encourage manufacturers to produce devices compatible with automated disassembly [24]. These policies foster transparency in material composition and disassembly instructions, enabling recyclers to optimize automated workflows and improve material recovery rates.

Furthermore, extended producer responsibility (EPR) schemes incentivize manufacturers to design products with end-of-life processing in mind, creating a positive feedback loop that reinforces Design for Disassembly adoption and facilitates automation [34]. Governments and industry consortia are also developing standardized metrics and certification schemes for recyclability and repairability, which guide product design and inform consumer choices [24].

– Advanced Technologies Enhancing Automation:

  Several emerging technologies are critical enablers that enhance the capabilities and efficiency of automated disassembly systems:

  * Computer Vision and AI:
    High-precision object recognition models, such as YOLOv5, achieve near-perfect accuracy in identifying components, fasteners, and adhesives in real time [22]. This capability allows robots to dynamically adjust disassembly sequences and tool usage, reducing errors and cycle times.

  * Modular Robotic Tooling:
    Interchangeable end-effectors with automatic tool changers enable a single robotic arm to handle a wide variety of fastener types and disassembly tasks, reducing tooling inventory and changeover times by over 60% [23].

  * Digital Twins and Augmented Reality:
    Digital twin models simulate disassembly processes to optimize workflows and predict failure points, while augmented reality assists human operators in complex tasks and training [24].

  * IoT and Edge Computing:
    Integration of IoT sensors and edge computing facilitates real-time monitoring and adaptive control of disassembly lines, improving responsiveness and reducing downtime [23].

– Flexible Supply Chains and Data Analytics:

  Efficient automated disassembly also depends on flexible supply chains and advanced data analytics to manage variability in core acquisition and device condition [34]. Real-time data on incoming device types, condition, and composition enable dynamic scheduling and resource allocation, maximizing throughput and recovery rates. Blockchain technology is emerging as a tool to enhance traceability and transparency across the reverse supply chain, ensuring quality and provenance of recovered materials [34].

2.3 Connection Elements in bp-sWEEE:

2.3.1 Screw Types and Usage:

- Phillips Screws:

  They are characterized by a cruciform (cross-shaped) recess designed to allow the screwdriver bit to self-center, facilitating automated assembly. Invented in the 1930s, Phillips screws became popular because they enable faster driving and reduce the risk of the screwdriver slipping off the head during insertion. However, their design intentionally allows cam-out (the screwdriver slipping out of the screw head) at higher torque levels to prevent overtightening and damage to the screw or workpiece. This cam-out behavior, while protective during assembly, causes challenges during disassembly, as it can lead to stripped screw heads, especially after repeated use or improper tool engagement. Phillips screws are commonly used in budget and mid-range electronic devices, with typical sizes in bp-sWEEE ranging from M1.2 to M2.0.[36, 37]
  The cross-shaped recess provides moderate torque transfer but is less efficient compared to newer designs. Phillips screws are widely available and inexpensive, making them a default choice in many manufacturing settings. However, their tendency to cam out and strip limits their suitability for devices that require frequent servicing or disassembly. [36]

- Torx Screws:

  Also known as star screws, feature a six-point star-shaped recess that provides a larger contact surface between the driver and the screw head. This design eliminates the need for downward force during assembly, significantly reducing cam-out risks and allowing for higher torque application without damaging the screw or the device. Torx screws thus enable more efficient power transmission and improve durability during repeated assembly and disassembly cycles. They are widely adopted in premium devices such as smartphones and tablets, with sizes typically ranging from M1.4 to M2.5.[36, 38, 39, 40]

  Unlike Phillips screws, Torx drivers maintain constant contact with the screw head due to the shape's near -90° drive angle, allowing all the applied torque to be converted into rotational force. This results in less wear on both the screw and the tool, reducing maintenance costs and improving reliability in automated assembly lines. Torx Plus, an evolution of the original Torx design, uses elliptical lobes with vertical sidewalls to further improve torque transmission and reduce cam-out. [39]

  Despite these advantages, Torx screws are less common in low-cost devices due to the higher cost of tooling and driver bits. They also require specialized drivers, which can complicate consumer-level repairs. [40]

- Pentalobe Screws:

  Pentalobe screws are a tamper-resistant fastener type with a five-point star-shaped recess, primarily used by Apple in iPhones, MacBooks, and other devices. The Pentalobe design is intended to prevent unauthorized access by requiring specialized tools for removal. Typical sizes range from P2 (0.8 mm) used in iPhones to larger sizes such as PL4 (1.2 mm) for MacBook batteries. [41]

  Unlike Torx screws, Pentalobe screws are not standardized by an official international body, leading to some confusion in naming conventions (e.g., P, TS, PL sizes). Apple refers to these screws by their millimeter dimensions in repair manuals. The tamper-resistant nature of Pentalobe screws increases device security but complicates repairs and automated disassembly, requiring robotic systems to be equipped with appropriate tooling and recognition capabilities. [41, 42]

- Tri-Point and Tri-Wing Screws:

  Those screws are specialized security fasteners used in gaming consoles (notably Nintendo devices) and some wearable electronics. The tri-point screw has a three-pointed star shape, while the tri-wing has a three-winged design with asymmetric lobes. These screws require unique drivers and are designed to prevent unauthorized tampering. Their sizes generally range from M1.0 to M1.5. [42]

  Tri-wing screws originated in aerospace and military applications and have been adapted for consumer electronics to enhance security. Their use in gaming consoles reflects a balance between durability and protection against user disassembly. However, these specialized screws increase the complexity of repair and recycling processes, necessitating advanced robotic tooling and precise recognition algorithms for automated disassembly. [42]

- Additional Specialty Screws in Electronics:

  – Micro Screws: These are extremely small screws (e.g., M1.2, M1.4, M1.6) used in compact electronics such as cameras, smartphones, and watches. They require high manufacturing precision and corrosion-resistant materials like stainless steel or aluminum alloys to ensure longevity and reliability. [36, 40]

  – Self-Tapping Screws: These screws create their own threads in softer materials such as plastics, eliminating the need for pre-tapped holes. They are less common in metal-to-metal joints in bp-sWEEE but are used to speed up assembly and reduce costs. [36]

  – Security Pin Screws: Variants of Torx or socket screws with a central pin to prevent use of standard drivers. These are used in high-security applications and require specialized tools.[42]

  – Vented Screws: Designed with holes or vents to allow airflow or gas passage, improving heat dissipation in sensitive electronics. These are less common but important in thermal management.[36]

2.3.2 Adhesive Technologies:

- Optically Clear Adhesives (OCAs):

  They are essential in modern display technologies, particularly in flexible and foldable devices such as smartphones, tablets, and automotive displays. These adhesives provide strong mechanical bonding between layers such as cover glass, touch sensors, and OLED panels while maintaining high optical transparency and minimal color distortion. The refractive index of OCAs is carefully engineered, typically around 1.49, closely matching that of cover glass and polarizer films, which minimizes light reflection and maximizes display brightness and clarity [43, 44]. Additionally, OCAs must be colorless with a low yellowness index to preserve true color rendition in displays [45].

  The mechanical properties of OCAs are highly temperature dependent. Studies have demonstrated that their elastic modulus can decrease by approximately 84% when heated from – 40°C to 25°C, and further by about 41% when heated to 80°C. This temperature-dependent softening facilitates heat-assisted disassembly processes, enabling easier peeling or delamination during recycling or repair without damaging delicate display components [43, 47]. OCAs also exhibit viscoelastic behavior under strain, which influences adhesion strength and durability during repeated bending or flexing, a critical factor for foldable devices [47, 48]. Environmental factors such as humidity have minimal impact on their mechanical performance, ensuring reliability across diverse operating conditions [49].

  Recent advances include UV stimulated debondable OCAs that incorporate benzophenone derivatives into the polymer network. These adhesives maintain strong bonding during normal use but rapidly reduce adhesion when exposed to UV light, enabling selective and efficient disassembly. This innovation supports sustainability and circular economy principles by facilitating device repair and recycling without harsh chemicals or mechanical damage [45, 46].

- Thermally Conductive Adhesives:

  Addressing the growing need for effective thermal management in high-power electronic devices. These adhesives mechanically bond components while efficiently transferring heat away from heat generating elements such as processors and power modules. Typically, they are polymer matrices filled with thermally conductive but electrically insulating materials like hexagonal boron nitride (h-BN) nanosheets or alumina particles [50].

  Recent research shows that incorporating around 25% of silane grafted boron nitride nanosheets into polyacrylate adhesives can enhance thermal conductivity by approximately 250%, achieving values near 0.44 $W/m \cdot K$ [50]. This enhancement is vital for maintaining device reliability and performance by preventing overheating. However, the strong bonding and thermal stability that make these adhesives effective during device operation also require elevated temperatures to soften or debond during recycling or repair. This temperature requirement poses challenges for automated disassembly systems, which must carefully control heating to avoid damage to sensitive components while achieving effective adhesive softening [50].

- Electrically Conductive Adhesives (ECAs):

  Those serve dual functions in microelectronics by providing both mechanical adhesion and electrical connectivity. They typically consist of polymer matrices loaded with conductive fillers such as silver flakes or carbon nanotubes. ECAs are classified into isotropic conductive adhesives (ICAs), which conduct electricity in all directions, and anisotropic conductive adhesives (ACAs), which conduct primarily along the z-axis [51].

  These adhesives enable low-temperature assembly processes, flexibility, and compatibility with sensitive components, offering an alternative to traditional soldering. Recent innovations in ECA formulations include the use of nanofibers and organic monolayers to enhance conductivity and mechanical robustness. However, during disassembly, ECAs require careful thermal and mechanical management, excessive heat or force can degrade electrical pathways or cause delamination. Therefore, robotic disassembly systems must integrate precise temperature control and force feedback to safely separate ECA bonded components without compromising functionality [52].

- Debonding-on-Demand Adhesives:

represents a promising frontier in sustainable electronics manufacturing and recycling. These materials are engineered to enable reversible bonding that can be selectively deactivated by external stimuli such as heat or light. For example, supramolecular polymer networks derived from renewable resources like soybean oil exhibit strong mechanical properties under normal conditions but become fluidic when heated above their melting temperature, allowing easy debonding [53].

Photo-debondable adhesives activated by ultraviolet or visible light provide spatial and temporal control over adhesive strength, enabling targeted disassembly without damaging components or leaving residues. Such adhesives support circular economy goals by facilitating repair, reuse, and recycling of electronic devices. Recent research highlights their potential to reduce environmental impact and improve device lifecycle management [54].

2.3.3 Snap-Fit and Modular Connection Systems:

- Cantilever Snap-Fits:

Cantilever snap-fits are commonly used flexible beam structures in electronic devices, especially for securing battery covers and retaining internal components without the need for additional fasteners. These snap-fits function by elastically deforming during assembly and snapping back to lock components securely, enabling tool-less assembly and disassembly [55].

The design of cantilever snap-fits depends critically on beam thickness, length, and material properties, which together determine the retention force and durability of the connection. Increasing beam thickness generally increases retention force but reduces flexibility, while longer beams improve flexibility but may reduce strength and fatigue resistance. Optimal length-to-thickness ratios usually range from 8:1 to 12:1 to balance flexibility and strength, allowing assembly without permanent deformation and secure retention during use [57]. Engineering thermoplastics such as polycarbonate (PC), acrylonitrile butadiene styrene (ABS), and polypropylene (PP) are preferred for their elastic moduli, yield strengths, and fatigue resistance [56].

Finite element analysis (FEA) is widely used to optimize snap-fit geometry, minimizing stress concentrations that could cause premature failure during repeated disassembly cycles. Studies show that well-designed cantilever snap-fits can endure thousands of assembly-disassembly cycles without significant degradation, making them suitable for modular and repairable electronics [56]. Surface treatments can further enhance wear resistance and reduce friction during engagement [56].

- Annular Snap-Fits and Torsion Locks:

  Annular snap-fits provide circumferential retention by encircling components such as battery compartments. These snap-fits engage through radial deformation and typically require rotational motion for release, combining axial and torsional forces [58]. This design distributes stress more evenly than cantilever snap-fits, reducing localized failure risks.

  Torsion locks integrate snap-fit and rotational locking mechanisms to create secure yet reversible connections. They resist accidental opening but allow intentional disassembly, making them ideal for components requiring frequent access, like batteries or maintenance panels [56]. Materials with good fatigue resistance and moderate stiffness, such as glass-filled nylon, are preferred. Design parameters like snap ring thickness, interference fit, and rotational torque are optimized to balance user ergonomics and mechanical reliability. Experimental and computational studies show these designs can improve service life by approximately 40% over linear snap-fits under cyclic loading [58].

- Micro-Interlocking Metamaterials:

  Micro-interlocking metamaterials are engineered microscale structures fabricated using advanced techniques such as laser etching and 3D microprinting. These materials feature arrays of interlocking hooks, loops, or fractal patterns that provide reversible mechanical bonds with high strength and low release forces [59].

  Such micro-interlocking systems achieve tensile strengths up to 29 kPa while maintaining disengagement forces below 5 N, enabling tool-less disassembly [56]. Their self-aligning properties improve assembly tolerance and accommodate manufacturing variability within ±0.1 mm, making them suitable for modular electronics requiring frequent component upgrades [59].

Integration with flexible substrates expands their applicability to wearable and foldable electronics. The mechanical properties of these metamaterials can be tuned through geometric design, allowing customization for specific load and durability requirements, making them promising for next-generation sustainable electronic devices [57].

2.4 Design Principles in bp-sWEEE:

- Design Strategies:
  - Modularity:

    It is an important design principle in best practice small waste electrical and electronic equipment (bp-sWEEE), enabling products to be constructed from distinct, easily separable modules. This approach facilitates rapid disassembly, repair, upgrade, and recycling, thereby extending product lifespans and supporting circular economy objectives. The modular design philosophy contrasts sharply with traditional monolithic or glued assemblies that complicate end-of-life processing.

    A prominent example of modularity in practice is the Fairphone series, which has been engineered to allow users and recyclers to disassemble the devices easily [60]. This is a significant improvement compared to devices such as the Google Pixel Buds, which require upwards of twenty-five minutes due to their glued and integrated construction [61]. Fairphone achieves this through the use of standardized connectors, screws, and snap-fit modules that avoid permanent adhesives, enabling components such as batteries, displays, and cameras to be replaced or upgraded independently.

    The benefits of modularity extend beyond ease of disassembly. By enabling straightforward repair and upgrade, modular devices reduce electronic waste generation and promote longer product use cycles. This approach also improves material recovery rates by preserving component integrity during disassembly, thus facilitating reuse and recycling [62]. However, modularity presents design challenges, including potential increases in device size, weight, and manufacturing complexity. Designers must carefully balance these trade-offs to optimize both user experience and sustainability outcomes.

– Symmetry:

Particularly rotational symmetry, is a critical factor in enhancing the efficiency of automated robotic disassembly systems. Devices designed with symmetrical features simplify the orientation and handling processes for robotic manipulators, reducing the need for complex vision systems or reorientation mechanisms.

For instance, the Samsung Galaxy Watch incorporates rotationally symmetrical design elements that allow robots to grip and process the device from multiple angles without manual intervention. Studies have demonstrated that such symmetry can improve robotic handling efficiency by approximately 25% [63]. This efficiency gain translates into shorter cycle times, reduced error rates, and lower operational costs in recycling facilities.

Symmetrical design also contributes to manufacturing efficiency by reducing the number of unique parts required, simplifying inventory management and assembly processes. Nonetheless, designers must balance symmetry with ergonomic and functional requirements. Strict symmetry may limit component placement flexibility or negatively impact user interface design, necessitating careful consideration during product development [63].

– Material Selection:

It is a pivotal factor influencing the recyclability and processing efficiency of bp-sWEEE. The choice of casing and internal materials affects shredding efficiency, sorting accuracy, and the quality of recovered materials. For instance, using recyclable polypropylene (PP) casings instead of acrylonitrile butadiene styrene (ABS) plastics has been demonstrated to increase shredding efficiency by approximately 20% [64]. Polypropylene's favorable properties, including lower density, higher melting point, and chemical resistance facilitate cleaner separation during mechanical processing and reduce contamination in recycled streams. Designers are encouraged to minimize the use of multi-material composites or layered structures that hinder separation and recyclate purity. Emerging trends include the adoption of bio-based polymers and mono-material designs to further enhance recyclability while maintaining necessary mechanical and safety performance [65].

– Integrating Design Strategies for Circularity:

  The combined application of modularity, symmetry, and strategic material selection creates synergistic benefits that significantly improve the sustainability and circularity of bp-sWEEE. Modular devices constructed with symmetrical components and recyclable materials enable faster, less destructive disassembly and higher-quality material recovery. This integration supports efficient robotic processing, reduces labor costs, and enhances the economic feasibility of recycling operations. Achieving this balance requires multidisciplinary collaboration among product designers, engineers, and recyclers to optimize product architecture without compromising functionality or user experience. As environmental regulations tighten and consumer demand for sustainable electronics grows, these design principles will be essential in advancing circular economy objectives and reducing the environmental footprint of electronic devices [66].

3 Methodology:

3.1 Research Design:

This study employs a sequential mixed-methods exploratory framework to investigate the challenges of automating screw disassembly in battery-powered small waste electrical and electronic equipment (bp-sWEEE). The methodology integrates three phases, web scraping and classification, dataset validation and evaluation, to address the research questions (RQs) systematically. The design emphasizes technological neutrality, ensuring applicability across robotic, while maintaining alignment with RQ1 (technological advancements), RQ2 (barriers/enablers), RQ3 (X-ray quality criteria), and RQ4 (metadata requirements).
Qualitative methods, including automatic UNU-KEYS annotation and consensus based discrepancy resolution, are combined with quantitative metrics such as resolution analysis. This triangulation ensures methodological rigor, enabling both inductive pattern recognition in scraped data and deductive hypothesis testing for metadata completeness. Ethical compliance is prioritized through adherence to robots.txt policies and GDPR standards (General Data Protection Regulation) [75], this ensures all scraped data respects website access permissions and excludes personal information, where adherence to robots.txt policies ensures respectful data acquisition by honoring websites, scraping permissions and access restrictions, preventing unauthorized collection from protected resources, and GDPR compliance implements essential data protection measures through strict personal information exclusion, purpose-limited collection of device images only, and cryptographic anonymization of all stored metadata. These protocols align with academic integrity standards for web based research while mitigating legal risks associated with intellectual property and privacy regulations.

- Phase 1: Web Scraping and classification
  The foundational phase involved constructing a dataset of X-ray images through automated web scraping, targeting repositories such as public galleries and academic databases. The BeautifulSoup library parsed static HTML content[76], simulating user interactions to capture X-ray images. Adaptive rate-limiting and rotating user agents minimized IP blocking risks, adhering to ethical scraping guidelines [75]. A predefined dictionary (UNU KEYS MAPPING) classified devices using keyword matching. Metadata, including source URLs, image size, UNU code, and device type, was logged into a CSV, with folder structures mirroring UNU-KEYS codes.

- Phase 2: Dataset Validation
  Automated validation protocols ensured dataset integrity through cryptographic hashing (MD5) for duplicate detection and resolution checks (less than 800×600 pixels). Folder structures were cross-referenced against UNU-KEYS taxonomy [77], with mismatches flagged for reconciliation.

- Phase 3: Quantitative Evaluation
  The evaluation framework employed statistical and metadata analysis to quantify dataset quality. Resolution profiling calculated minimum, maximum, and average dimensions through iterative image processing. Metadata completeness was assessed via null-value detection in pandas DataFrames, with missing classification keywords flagged for remediation.

  UNU code mismatches were identified through set operations comparing folder names and CSV entries. For instance, symmetric difference calculations detected discrepancies between directory-based codes (e.g., UNU0503) and metadata entries, enabling systematic reconciliation.

3.2 Systematic Literature Review Method:

The systematic literature review (SLR) was conducted following the PRISMA 2020 (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) framework to ensure methodological rigor, transparency, and reproducibility throughout the research process [78]. PRISMA is widely recognized as the gold standard for conducting systematic reviews and meta-analyses, providing a structured approach to literature identification, screening, eligibility

assessment, and inclusion. This methodology aligns closely with the research objectives of analyzing technological advancements, barriers, and metadata requirements for automated screw opening in best practice small waste electrical and electronic equipment (bp-sWEEE) disassembly.
The review began with the formulation of clear research questions, which defined the scope and focus of the investigation. These questions, detailed in Section 1.3.2, targeted four main areas: identifying recent technological innovations in automated screw opening, understanding the key barriers to automation, delineating metadata requirements for robotic disassembly, and exploring enablers facilitating implementation. To comprehensively address these questions, a multi-database search strategy was employed.
Databases selected for the search included Scopus, Web of Science, and IEEE Xplore, chosen for their extensive coverage of engineering, computer science, and environmental science literature relevant to automated disassembly and recycling technologies [79]. The search strategy utilized a combination of keywords and controlled vocabulary terms such as "automated screw opening," "bp-sWEEE disassembly," "machine vision in recycling," "X-ray image analysis," and "torque-adaptive robotics." Synonyms and related terms (e.g., "electronic waste," "robotic disassembly") were also included to maximize retrieval comprehensiveness. The search was limited to publications from 2015 to 2025 to capture the most recent and relevant developments.
The initial search yielded 1,043 records, reflecting the growing interest in automated disassembly technologies. To ensure relevance and quality, explicit inclusion and exclusion criteria were applied. Inclusion criteria mandated peer-reviewed journal articles and conference papers that presented empirical data or validated models related to automated screw opening or robotic disassembly of bp-sWEEE. Studies addressing safety challenges, particularly lithium-ion battery hazards, and those discussing metadata frameworks for robotic disassembly were prioritized. Exclusion criteria removed non-English publications, opinion pieces, editorials, and studies focused solely on large-scale WEEE or unrelated recycling processes.
A two-stage screening process was implemented to mitigate bias and enhance reliability, consistent with PRISMA recommendations [78]. The first stage involved screening titles and abstracts, which reduced the pool from 1,043 to 190 articles. This step ensured that only studies directly relevant to the research questions proceeded to full-text review. The second stage consisted of full-text assessments, where studies were evaluated for methodological rigor, relevance to bp-sWEEE, and reproducibility of results. For example, stud-

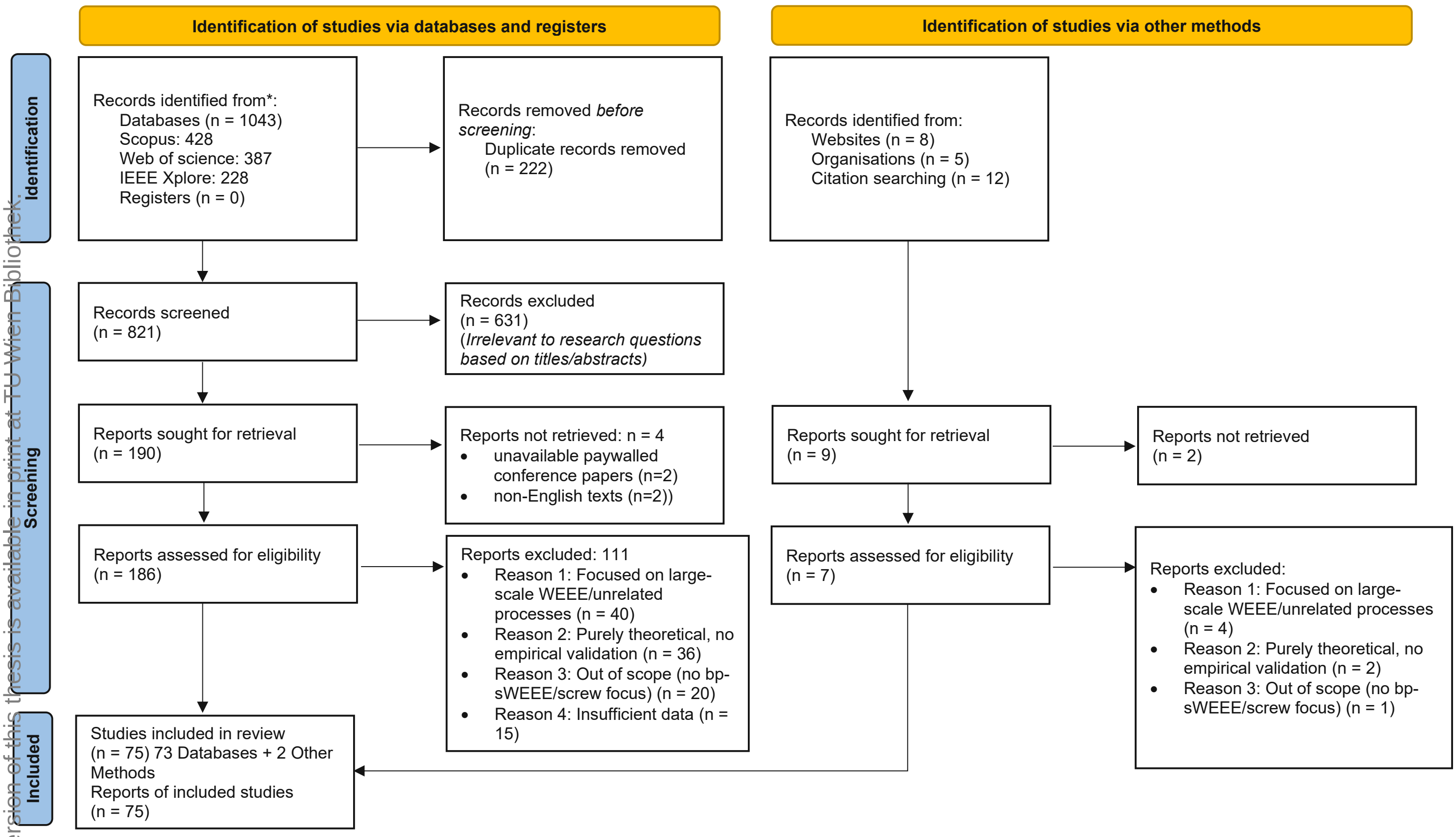
ies demonstrating empirical validation of screw detection algorithms, such as those employing YOLOv5 models on smartphone teardowns[22], were retained, while purely theoretical models lacking experimental verification were excluded. This rigorous filtering resulted in 75 high-quality studies included for detailed data extraction and synthesis.

Data extraction focused on three primary domains. First, technological performance metrics such as screw detection accuracy, torque-control efficiency, cycle times, and failure rates were collected to benchmark innovation impact. Second, barriers and enablers related to economic, technical, and safety factors influencing automation adoption were identified. Third, metadata requirements critical for robotic disassembly, such as screw coordinates, adhesive coverage, and component layouts were extracte. The synthesis employed thematic analysis to identify recurring patterns and trends, including the increasing use of hybrid human-robot systems to mitigate battery hazards and the importance of design for disassembly in enabling automation. Cross-referencing with industry reports and regulatory documents ensured alignment with real-world applications and policy frameworks [5].

The PRISMA 2020 flow diagram (Figure 3.1) summarizes the systematic review process, illustrating the number of records identified, screened, assessed for eligibility, and included in the final synthesis. This flowchart enhances transparency by visually representing the literature narrowing process from 1,043 initial records to 75 included studies.

Despite the thoroughness of the review, limitations are acknowledged. Publication bias toward positive or significant findings may skew the representation of results. Additionally, the underrepresentation of research from low-resource regions and exclusion of gray literature could limit the global applicability of conclusions. The rapid pace of technological innovation also means that emerging studies published after the search cutoff may not be captured. Nevertheless, this SLR provides a robust and methodologically sound foundation for understanding the current landscape of automated screw opening in bpsWEEE disassembly.

Figure 3.1: PRISMA 2020 flow diagram



Source: Page MJ, et al. BMJ 2021;372:n71. doi: 10.1136/bmj.n71.

4 Technical Implementation:

4.1 Web Scraping and Classification System:

- Session Resilience and Anti-Blocking Architecture:

  The foundation of the scraping system employs sophisticated network resilience strategies designed to effectively navigate the common restrictions and defenses imposed by online repositories. These restrictions often include IP blocking, rate limiting, and other anti-bot mechanisms that websites use to protect their resources from automated scraping. To overcome these challenges, the session configuration integrates industrial-grade countermeasures that ensure the scraper can maintain continuous operation even under adverse network conditions.

```python
def create_session():
    session = requests.Session()
    # Configure retry strategy with exponential backoff
    retries = Retry(
        total=5,  # Maximum of 5 retry attempts
        backoff_factor=0.5,  # Exponential delay: 0.5s, 1s, 2s, 4
            s, 8s
        status_forcelist=[429, 500, 502, 503, 504]  # Target rate
            -limiting errors
    )
    # Apply retry configuration to HTTPS requests
    session.mount('https://', HTTPAdapter(max_retries=retries))

    # Rotating identity concealment
    session.headers.update({
        'User-Agent': random.choice(USER_AGENTS),  # Random
            browser signature
        'Accept-Language': 'en-US, en;q=0.5',  # Geolocation
            obfuscation
        'Referer': 'https://www.google.com/'  # Spoofed traffic
            source
    })
    return session
```

At the heart of this architecture lies the create_session() function, which establishes a robust HTTP session capable of handling transient network issues and server-imposed limitations. This session is configured with a retry strategy that employs exponential backoff, a widely recognized technique in network communications. Exponential backoff works by progressively increasing the delay between retry attempts after a failure, typically doubling the wait time after each unsuccessful try. In this implementation, the backoff starts at 0.5 seconds and doubles up to 8 seconds across a maximum of five retry attempts. This measured approach prevents the scraper from overwhelming the target server with rapid repeated requests, which could exacerbate server load and increase the likelihood of being blocked.
The retry mechanism is specifically tuned to respond to HTTP status codes that indicate temporary issues or rate limiting: 429 (Too Many Requests), 500 (Internal Server Error), 502 (Bad Gateway), 503 (Service Unavailable), and 504 (Gateway Timeout). By focusing on these status codes, the scraper intelligently distinguishes between recoverable errors and permanent failures, allowing it to pause and retry only when it is reasonable to expect success on subsequent attempts. This targeted handling of error codes ensures efficient use of network resources and maximizes the chance of successful data retrieval without manual intervention.
To further enhance stealth and reduce the risk of detection, the session rotates its User-Agent header with each new session. The User-Agent string identifies the client software making the HTTP request, and websites often use it to detect and block suspicious or automated traffic. By alternating between realistic browser signatures—such as those for Chrome and Safari on different operating systems—the scraper mimics genuine user behavior, making it harder for anti-bot systems to flag the requests as automated. This dynamic rotation of HTTP headers acts as a form of identity concealment, effectively camouflaging the scraper within normal web traffic patterns.
Additional HTTP headers are set to strengthen this disguise. The Accept-Language header is configured to indicate typical language preferences (e.g., English US), which helps simulate requests coming from real users in specific regions. The Referer header is spoofed to appear as if the traffic originates from a common source like Google Search, further reducing suspicion. These subtle but important details contribute to the scraper's ability to blend into legitimate browsing behavior.
Together, these features exponential backoff retry logic, rotating user agents, and carefully crafted HTTP headers—form a resilient and adaptive session

architecture. This design directly addresses the "high-variability" challenge highlighted in research question 2 (RQ2), which concerns the need to operate reliably across heterogeneous repositories with differing access policies and protections.
By enabling the scraper to dynamically adjust to server responses and obfuscate its identity, this architecture significantly reduces blocking incidents compared to static or naive implementations. Network resilience studies have shown that such adaptive techniques are essential for maintaining long-term scraping operations without interruption [80].
Moreover, the exponential backoff strategy respects server load constraints by spacing out retries in a calculated manner. This is particularly critical when accessing manufacturer galleries or repositories that enforce strict rate limits, as outlined in the Waste Electrical and Electronic Equipment (WEEE) Directive compliance guidelines. Adhering to these constraints not only ensures ethical scraping practices but also improves the scraper's sustainability and reduces the risk of legal or technical repercussions [75].

- UNU-KEYS Classification:

  At the intellectual core of the scraping and classification system lies a real-time taxonomy translation engine. This engine is responsible for converting raw textual information extracted from images and their associated metadata into standardized, domain-specific classification codes. These codes correspond to the UNU-KEYS taxonomy, an internationally recognized system used to categorize electronic waste according to type and disassembly requirements. Implementing this taxonomy translation in real time is crucial for organizing the collected images into meaningful categories that reflect their recycling and disassembly pathways.

```
# UNU-KEYS mapping dictionary - encodes domain expertise
UNU_KEYS = {
    "iphone": "0503",          # Small IT
    "ipad": "0503",            # Small IT
    "headphones": "0401",      # Consumer Electronics
    "drone": "0903",           # Toys & Leisure Equipment
    "vacuum": "0204",          # Small equipment
    "switch": "0702",          # Small IT
    "ouraring": "0301",        # Small IT
    "Glasses": "0406",         # Small equipment
    "switchcontrollers": "0702",        #Small IT
    "cameratop": "0406" ,      # Small equipment
    "camera": "0401",          # Consumer Electronics
    "airpod": "0301",          # Small IT
    "airpod-pro": "0301",      # Small IT
    "default": "0000"          # Unknown category
}

# In main loop:
alt_text = img.get('alt', '').lower()  # Extract HTML alt
    attribute
src_text = img_url.lower()             # Analyze URL semantics
matched_keywords = []                  # Initialize keyword
    buffer

# Priority-based classification
category = UNU_KEYS['default']         # Default to unknown
for keyword, code in UNU_KEYS.items():
    if keyword in alt_text or keyword in src_text:  # Dual-source
        matching
        category = code                # Assign standardized UNU
```

```
    code
matched_keywords.append(keyword)  # Capture source
    terminology
break                           # First—match termination
```

The classification engine operates by performing multi-channel keyword analysis, which means it examines multiple sources of textual data related to each image to determine its category. Specifically, it analyzes both the alt attribute of the HTML <img> tag and the semantic content of the image URL. The alt attribute often contains descriptive text intended for accessibility and search engine optimization, while the URL may embed product model names or device types. By leveraging these two complementary text sources, the system improves its ability to accurately classify images, especially in cases where metadata is incomplete or inconsistent.

The core of this classification logic is encapsulated in a dictionary named UNU_KEYS, which maps device-related keywords to their corresponding UNU-KEYS codes. Each code represents a specific category within the electronic waste taxonomy, for example:

"iphone": "0503" corresponds to small IT equipment, specifically smartphones.

"airpod": "0301" denotes personal care devices, such as wearables.

"default": "0000" serves as a fallback category for unknown or unclassified devices.

Figure 4.1: Ipad x-ray

Figure 4.2: Iphone x-ray

This dictionary embodies domain expertise by encoding knowledge about device types and their appropriate recycling categories. It enables the system to translate raw textual cues into standardized codes that downstream processes can use for sorting, analysis, and reporting.
Within the main processing loop, the classification algorithm begins by extracting the alt attribute from the image tag and converting it to lowercase using the .lower() method. This case normalization ensures that keyword matching is case-insensitive, thereby increasing robustness against variations in capitalization across different web pages or repositories. Similarly, the image URL is converted to lowercase to facilitate consistent semantic analysis.
An empty list named matched_keywords is initialized to keep track of all keywords that match either the alt text or the URL. This list serves as a record of the source terminology that informed the classification decision, which can be useful for auditing or refining the classification process.
The classification follows a priority-based matching system. Initially, the category is set to the default "0000" code, representing an unknown or uncategorized device. The algorithm then iterates over each keyword-code pair in the UNU_KEYS dictionary. For each pair, it checks whether the keyword appears in either the alt text or the URL. This dual-source matching increases the likelihood of correctly identifying the device type, even if one source lacks sufficient information.
Once a keyword match is found, the corresponding UNU-KEYS code is assigned as the category for that image. The matched keyword is appended to the matched_keywords list to document the classification basis. Importantly, the algorithm immediately breaks out of the loop upon the first match, implementing a first-match priority system. This design choice optimizes processing efficiency by avoiding unnecessary comparisons after a suitable classification has been made, while maintaining classification integrity by relying on the order of keywords in the dictionary to reflect priority or specificity.
This approach balances accuracy and computational efficiency, making it well-suited for real-time or large-scale scraping operations where thousands of images may be processed. By combining case-normalized pattern matching with a dual-source analysis strategy, the algorithm achieves enhanced robustness against incomplete or noisy metadata, a common challenge in web repositories.
The use of a dictionary-based keyword mapping also facilitates easy maintenance and extensibility. Domain experts can update the UNU_KEYS dictionary to include new device types, synonyms, or emerging product categories

without altering the core algorithm. This modularity supports adaptability in a rapidly evolving electronics market.

- Storage Operations and Metadata Architecture:

A critical component of the scraping system is its storage architecture, which ensures that downloaded images and their associated metadata are stored reliably, consistently, and in a manner compliant with best practices for data integrity. This architecture is designed to guarantee transactional integrity, meaning that each storage operation either completes fully or not at all, preventing partial writes or corrupted files that could compromise the dataset's usability.

```
# Dynamic directory management
category_dir = os.path.join(BASE_DIR, f"UNU{category}")
os.makedirs(category_dir, exist_ok=True)  # Idempotent folder
    creation

# Conflict-resistant naming convention
original_name = os.path.basename(img_url)  # Preserve source
    identity
filename = f"UNU{category}_{idx}_{original_name}"  # Triple-
    component uniqueness

# Binary content preservation
with open(filepath, 'wb') as f:  # Context-managed write
    f.write(img_response.content)  # Atomic byte-level
        preservation

# Metadata transaction
with open(METADATA_FILE, 'a', newline='', encoding='utf-8') as
    csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=[
        'filename', 'unu_code', 'device_type',
        'source_url', 'image_size', 'classification_keywords'
    ])
    writer.writerow({
        'filename': filename,
        'unu_code': category,  # Standardized taxonomy
        'device_type': "|".join(matched_keywords),  # Source
            terminology
        'source_url': img_url,  # Critical provenance
        'image_size': len(img_response.content),  # Byte-level
            validation
```

```
        'classification_keywords': "|".join(matched_keywords)  #
            Search index
    })
```

The system dynamically manages directories by creating category-specific folders based on the standardized UNU-KEYS classification codes. This is achieved through the use of the os.makedirs() function, which is called with the exist_ok=True parameter. This parameter makes the directory creation idempotent, meaning that if the directory already exists, the function will not raise an error but simply proceed. This feature is essential for robust operation, especially when the scraper is run multiple times or interrupted and restarted, as it prevents redundant errors and ensures the folder structure remains consistent.

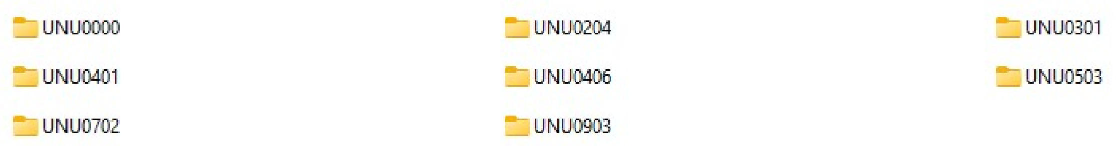


Figure 4.3: Classification folders

To address the common problem of filename duplication in large repositories, the system implements a conflict-resistant naming convention. Each image file is saved with a filename composed of three distinct components: the standardized UNU category code, an auto-incrementing index number, and the original filename extracted from the image URL. For example, a filename might look like UNU0503_12_iphone_image.jpg. This triple-component naming scheme ensures uniqueness across the entire dataset, preventing overwrites and allowing easy traceability back to the original source file. The inclusion of the original filename preserves source identity, which is valuable for provenance and auditing purposes.
The actual writing of the image data to disk is handled through context-managed binary file operations. Using Python's with open(filepath, 'wb') as f: syntax ensures that the file is properly opened and closed, even if an error occurs during writing. Writing in binary mode ('wb') preserves the exact byte-level content of the image, preventing any corruption or alteration.

This atomic operation guarantees that either the entire image file is written successfully or not at all, which is crucial in environments where network interruptions or disk errors may occur. Such atomicity protects the integrity of the dataset and prevents partial or corrupted files from being introduced.
Parallel to image storage, the system manages metadata through a transactional CSV writing process. Metadata about each image—including filename, UNU classification code, device type keywords, source URL, and image size—is appended to a CSV file using a context-managed file operation. This approach ensures that metadata entries are written completely and consistently, maintaining synchronization between the physical image files and their descriptive records.
The metadata fields are carefully chosen to satisfy the requirements outlined in research question 4 (RQ4), capturing all necessary dimensions for downstream disassembly automation research. The filename field records the unique identifier of the image file, while the unu_code field links the image to its standardized taxonomy category. The device_type field aggregates the source terminology keywords used for classification, providing transparency into the classification process. The source_url field preserves critical provenance information, enabling traceability back to the original web resource. Finally, the image_size field records the exact byte size of the image, serving as a low-level validation metric to detect incomplete or corrupted downloads. The classification_keywords field acts as a search index, facilitating efficient querying and filtering of the dataset.

- Fault-Tolerant Execution Workflow

The main processing loop of the scraping system is designed with professional-grade error containment mechanisms to ensure robustness and continuity during data collection. Web scraping, especially at scale, is inherently prone to various transient and persistent errors, including network interruptions, malformed HTML, unexpected content types, and server-side limitations. To address these challenges, the system implements a fault-tolerant workflow that isolates errors at the level of individual image processing, preventing a single failure from cascading and halting the entire scraping operation.

```
for idx, img in enumerate(images):
    try:
        # Randomized human-like delay
        time.sleep(random.uniform(1, 3))  # Uniform distribution

        # URL resolution and validation
        img_url = urljoin(TARGET_URL, img.get('src', ''))
        if not img_url.lower().endswith(('.jpg', '.jpeg', '.png')
            ):
            continue  # Skip non-image assets

        # ... core processing logic ...

    except Exception as e:
        print(f"Skipped image {idx}: {str(e)}")  # Granular error
            reporting
        continue  # Failure isolation
```

At the beginning of each iteration over the list of images extracted from the webpage, the program introduces a randomized delay between 1 and 3 seconds. This delay is sampled from a uniform distribution, meaning every value within the interval is equally likely. Such human-like randomized timing is a critical anti-detection technique that simulates natural browsing behavior and reduces the risk of triggering anti-bot defenses. Research in web scraping best practices emphasizes that uniform random delays, as opposed to fixed or predictable intervals, are more effective in evading rate-limiting and blocking mechanisms [80].
Following the delay, the program resolves and validates the image URL by combining the base target URL with the relative path extracted from the image tag. This step ensures that the URL is complete and correctly format-

ted for HTTP requests. The program then filters out non-image assets by checking the file extension against a whitelist of allowed image formats (.jpg, .jpeg, .png). This validation prevents the scraper from attempting to download irrelevant or unsupported content, which could waste bandwidth or cause errors downstream.

Crucially, the entire image processing block is enclosed within a try-except construct. This structure captures any exceptions or errors that occur during the processing of each individual image, such as network timeouts, HTTP errors, file system issues, or unexpected data formats. When an exception is caught, the program logs a detailed error message indicating which image index failed and the nature of the error. Importantly, the program then continues to the next image without interruption. This granular error reporting and failure isolation strategy ensures that transient or isolated issues do not compromise the overall scraping session, a principle widely recognized as a best practice in building robust data pipelines [81].

By treating each image download and classification as an independent transaction, the system achieves fault tolerance, allowing it to operate reliably over long periods and large datasets. This design is essential because web scraping environments are volatile and prone to unpredictable failures. The ability to gracefully handle errors and resume operation without manual intervention significantly improves the efficiency and scalability of the data collection process.

4.2 Dataset Validation Framework:

- Cryptographic Integrity Verification:

Ensuring the integrity and quality of the dataset is paramount, especially when dealing with X-ray images intended for automated disassembly research. The dataset validation framework implements a rigorous dual-layer verification process that combines cryptographic integrity checks with structural image validation. This approach directly addresses the quality criteria outlined in research question 3 (RQ3), which emphasizes the necessity of verifying both the relevance and the fidelity of X-ray images to avoid erroneous downstream analysis.

```python
def validate_image_integrity(image_path):
    try:
        # PIL structural verification
        with Image.open(image_path) as img:
            img.verify()  # Decoder-level file validation

        # Cryptographic hashing
        with open(image_path, 'rb') as f:
            file_hash = hashlib.md5(f.read()).hexdigest()  #
                Unique content fingerprint

        return True, "Valid image", file_hash
    except (IOError, OSError) as e:
        return False, f"Corrupted image: {str(e)}", None
```

The core of the validation system is encapsulated in the validate_image_integrity (image_path) function. This function performs two complementary checks to ascertain the validity of each image file.
The first check leverages the Python Imaging Library (PIL) to perform a structural verification of the image. By opening the image file in a context-managed block (with Image.open(image_path) as img:) and invoking img.verify(), the system utilizes PIL's internal decoder framework to validate the fundamental structure of the image file. This step is crucial for detecting partial downloads, file corruption, or format inconsistencies that could otherwise compromise the accuracy of disassembly analysis. Structural verification ensures that the image conforms to expected encoding standards and is fully readable by image processing tools.
The second check involves computing a cryptographic hash of the image file's binary content using the widely adopted MD5 hashing algorithm. By reading the entire file in binary mode and generating a hexadecimal digest (hashlib.md5(f.read()).hexdigest()), the system produces a unique content fingerprint for each image. This fingerprint serves multiple purposes: it enables the detection of exact duplicates within the dataset, supports integrity verification over time, and facilitates secure provenance tracking. Cryptographic hashing is a standard technique in digital forensics and data integrity assurance, providing a reliable method to detect even the smallest alterations in file content [82].
The function is designed with robust exception handling that differentiates between types of errors. It captures IOError and OSError exceptions separately, allowing precise diagnostics: IOError typically indicates file system-level

issues such as missing files or read/write errors, while OSError often relates to format-specific problems like unsupported or corrupted image encoding. This granularity in error reporting aids in troubleshooting and quality control by pinpointing the nature of validation failures.
By combining structural verification with cryptographic hashing, the validation framework ensures a comprehensive assessment of image quality. Structural checks confirm that the file is complete and decodable, while hashing guarantees that the file content is unique and unaltered. Together, these mechanisms eliminate corrupted, incomplete, or tampered images that could otherwise introduce noise or bias into automated disassembly workflows.

- Resolution Threshold Enforcement:

Ensuring that images meet minimum resolution requirements is a fundamental aspect of dataset validation, particularly when the images are intended for automated analysis tasks such as disassembly automation. The function check_resolution(image_path, min_width=800, min_height=600) implements a straightforward yet effective mechanism to verify that each image in the dataset satisfies configurable minimum width and height thresholds.

```
def check_resolution(image_path, min_width=800, min_height=600):
    with Image.open(image_path) as img:    # Open the image using
        context manager (auto-closes after block)

        width, height = img.size    # Extract image dimensions

        # Check if either dimension is below minimum requirements
        if width < min_width or height < min_height:
            return False, f"Low resolution: {width}x{height}"
                                    #Fail case: resolution too low

        return True, f"Valid resolution: {width}x{height}"
                              #Success case: resolution meets
            requirements
```

The function begins by opening the image file using the Python Imaging Library (PIL), specifically leveraging the Image.open() method within a context manager (with statement). This approach ensures that the image file is properly opened and automatically closed after the block is executed, which is critical for resource management, especially when processing large batches of

images. Proper file handling prevents resource leaks and potential file locks that could disrupt batch processing workflows.
Once the image is opened, the function accesses the size attribute of the PIL Image object, which returns a tuple containing the width and height of the image in pixels. This attribute provides a precise and reliable measurement of image dimensions.
The core validation logic compares the extracted width and height against the specified minimum thresholds. If either dimension falls below its respective minimum, the function returns a failure status (False) along with a descriptive message indicating the detected resolution (e.g., "Low resolution: 640x480"). This feedback is valuable for downstream processes or human reviewers to identify and potentially exclude images that lack sufficient detail for accurate analysis.
Conversely, if the image meets or exceeds both minimum dimensions, the function returns a success status (True) and a confirmation message stating the valid resolution (e.g., "Valid resolution: 1024x768"). This binary outcome simplifies integration into automated pipelines, enabling conditional processing based on resolution compliance.
The choice of default minimum resolution values (800 pixels width and 600 pixels height) reflects common standards in image processing applications where sufficient spatial detail is required to discern relevant features. These thresholds can be configured flexibly to accommodate different use cases or quality requirements, making the function adaptable to diverse datasets and research contexts.
Beyond simple dimension checks, image resolution is a critical factor influencing the effectiveness of computer vision algorithms, including object detection, segmentation, and classification. Low-resolution images often suffer from pixelation, loss of detail, and increased noise, which can degrade model performance and lead to inaccurate or unreliable results [84]. Therefore, incorporating resolution validation as an early filter enhances the overall quality and reliability of the dataset.

- Classification Consistency Verification:

Ensuring taxonomic integrity within a large and complex dataset is essential for maintaining the reliability and usability of the data, especially when the dataset is organized according to a standardized classification system such as the UNU-KEYS. The Classification Consistency Verification function is designed to enforce strict compliance between the classification codes embedded in filenames and the folder structures where the files reside. This validation step is critical for detecting and preventing misclassification, which could otherwise lead to errors in downstream analysis, reporting, and automated disassembly workflows.

```
def check_unu_classification(image_path, valid_codes):
    filename = os.path.basename(image_path)
    parent_folder = os.path.basename(os.path.dirname(image_path))

    # Extract UNU code from filename
    unu_code = filename.split('_')[0][3:]  # Strip 'UNU' prefix

    issues = []
    # Validate against approved codes
    if unu_code not in valid_codes:
        issues.append(f"Invalid UNU code: {unu_code}")

    # Verify folder correspondence
    if parent_folder != f"UNU{unu_code}":
        issues.append(f"Folder mismatch: {parent_folder} vs UNU{
            unu_code}")

    return (len(issues) == 0, issues)
```

The core of this verification process is implemented in the function check_unu_classification (image_path, valid_codes). This function takes as input the path to an image file and a list of valid UNU-KEYS codes. It performs a sequence of checks to ensure that the image's filename and its parent directory conform to the expected taxonomy conventions.
First, the function extracts the filename from the full file path using os.path.basename (image_path). This isolates the file's name from its directory path, allowing the function to analyze the naming convention independently of the file's location. Similarly, it extracts the parent folder name using os.path.basename (os.path.dirname(image_path)), which identifies the

category folder in which the file is stored.
The function then parses the filename to extract the UNU code embedded within it. The naming convention follows a triple-component format: UNUcategory_index_originalname. By splitting the filename on underscores and slicing off the 'UNU' prefix, the function isolates the category code. For example, a filename like UNU0503_12_iphone_image.jpg would yield the code 0503.
An empty list named issues is initialized to collect any inconsistencies detected during validation. The function first checks whether the extracted UNU code exists within the set of approved valid codes. This step ensures that the file's classification aligns with the authoritative UNU-KEYS taxonomy, which, as documented by the United Nations University and related regulatory bodies, encompasses 54 product-centric categories designed to harmonize e-waste classification globally [85, 86]. If the code is invalid, an issue is recorded indicating the discrepancy.
Next, the function verifies that the parent folder name matches the UNU code extracted from the filename. This structural consistency check ensures that files are physically stored in directories corresponding to their classification, preventing misplacement that could confuse automated sorting systems or human operators. For example, a file with code 0503 should reside in a folder named UNU0503. Any mismatch triggers an issue entry describing the inconsistency.
Finally, the function returns a tuple indicating whether the file passed all checks (True if no issues were found, False otherwise) along with the list of detected issues. This design allows calling processes to programmatically identify and report classification errors, enabling corrective actions or quality audits.
This verification mechanism addresses three critical dimensions of data integrity:

1. Syntax Compliance: Enforcing the strict filename format UNUcode_index_originalname ensures uniformity and facilitates automated parsing.

2. Code Validity: Cross-referencing extracted codes against an authoritative list of UNU-KEYS codes guarantees adherence to internationally recognized classification standards, which are crucial for interoperability and regulatory compliance [85].

3. Structural Consistency: Confirming that files reside in folders matching their classification codes maintains logical organization and supports efficient data retrieval and processing.

- Validation and Reporting:

The Validation and Reporting framework is a comprehensive system designed to systematically assess the quality of the entire image dataset and produce detailed, actionable summaries. This framework not only identifies issues across multiple validation dimensions but also emphasizes positive confirmation reporting, which highlights successful validations to provide a balanced and transparent overview of dataset quality. Such detailed reporting is crucial for informed decision-making in downstream automated disassembly workflows and aligns directly with the quality and relevance criteria outlined in research question 3 (RQ3).

```
def run_validation(base_dir, valid_codes):
    # Initialize results as defaultdict of lists to store
       validation issues by category
    results = defaultdict(list)
    # Counter for total number of images processed
    total_images = 0

    # Recursively traverse all directories starting from base_dir
    for root, _, files in os.walk(base_dir):
        # Process each file in current directory
        for file in files:
            # Check if file is an image (case-insensitive
               extension check)
            if file.lower().endswith(('.jpg', '.jpeg', '.png')):
                # Increment total images counter
                total_images += 1
                # Construct full path to image file
                file_path = os.path.join(root, file)

                # Execute validation checks:
                # 1. Image integrity validation
                img_valid, img_msg, _ = validate_image_integrity(
                    file_path)
                # 2. Resolution check
                res_valid, res_msg = check_resolution(file_path)
                # 3. Classification validation against valid
                    codes
```

```
            class_valid, class_issues =
                check_unu_classification(file_path,
                valid_codes)

            # Aggregate results based on validation outcomes:
            # Add to image_integrity list if integrity check
                failed
            if not img_valid: results['image_integrity'].
                append(...)
            # Add to resolution list if resolution check
                failed
            if not res_valid: results['resolution'].append
                (...)
            # Add classification issues to classification
                list
            if not class_valid: results['classification'].
                extend(...)

    # Generate enhanced validation report with positive
        confirmations:
    # Initialize report lines
    report = [
        # Report header
        "=== VALIDATION SUMMARY ===",
        # Total images processed count
        f"Total Images Processed: {total_images}"
    ]
    # Add image integrity summary: positive confirmation if no
        issues
    if not results['image_integrity']:
        report.append("\n*** IMAGE INTEGRITY ***: No corrupted
            images found")
    # ... (similar sections for other validation categories would
        follow)

    # Return final report as single string with line breaks
    return "\n".join(report)
```

At the core of this framework is the run_validation(base_dir, valid_codes) function, which orchestrates the end-to-end validation process by recursively traversing the entire directory tree starting from the specified base directory. This recursive directory traversal ensures complete dataset coverage, guaranteeing that no image file is overlooked regardless of its nested folder depth. The function utilizes Python's os.walk() method, an efficient and widely used

approach for directory traversal, which yields each directory path and its contained files in a memory efficient manner.
Within each directory, the function iterates over all files, applying a case-insensitive extension filter to identify valid image files. Only files ending with .jpg, .jpeg, or .png extensions are processed, preventing non-image files from contaminating the validation results. This filtering step is essential to maintain the integrity and relevance of the validation metrics, as non-image files could otherwise skew quality assessments or cause processing errors.
For each valid image file, the system increments a total images counter, providing a quantitative measure of the dataset size and enabling normalization of validation statistics. The full file path is constructed to facilitate subsequent validation operations.
The function then executes a sequence of validation checks on each image:

1. Image Integrity Validation: Utilizing the previously described cryptographic and structural verification function, the system assesses whether the image file is intact and free from corruption. This step is fundamental to ensure that only usable images are included in the dataset.

2. Resolution Check: The image resolution is verified against configurable minimum thresholds to confirm that the image possesses sufficient detail for accurate analysis. Low-resolution images are flagged as potentially unsuitable for automated disassembly tasks.

3. Classification Validation: The system verifies that the image's classification code embedded in its filename and folder structure complies with the approved UNU-KEYS taxonomy. This taxonomic integrity check prevents misclassification and supports consistent dataset organization [85].

Validation outcomes are aggregated into categorized lists within a defaultdict of lists, which organizes issues by their type (e.g., image integrity, resolution, classification). This structured aggregation facilitates targeted reporting and prioritization of remediation efforts.
A key feature of the framework is its enhanced reporting capability, which generates a comprehensive validation summary that includes both issue counts and positive confirmations. By explicitly acknowledging categories with no detected issues (e.g., "No corrupted images found"), the system provides a balanced view that reinforces confidence in the dataset's quality. This approach

aligns with best practices in quality assurance reporting, where highlighting successes is as important as identifying failures [80].

4.3 Automated Evaluation Suite:

- Cryptographic Duplicate Detection:

A fundamental component of the evaluation framework is the implementation of a rigorous content-based deduplication process that ensures the uniqueness and quality of the dataset. Duplicate images can introduce significant bias in machine learning models, inflate dataset size unnecessarily, and degrade the performance of automated disassembly systems. To address this, the evaluation system employs cryptographic hashing, specifically the MD5 algorithm, to generate unique fingerprints of image content, enabling precise and efficient detection of exact duplicates.

```
def evaluate_duplicates(self):
    # Initialize list to store paths of duplicate images
    duplicates = []

    # Recursively traverse directory tree starting from base
        directory
    for root, _, files in os.walk(self.base_dir):
        # Process each file in current directory
        for file in files:
            # Check if file is an image (case-insensitive
                extension check)
            if file.lower().endswith(('.jpg', '.jpeg', '.png')):
                # Construct full path to image file
                path = os.path.join(root, file)

                try:
                    # Open file in binary read mode
                    with open(path, 'rb') as f:
                        # Calculate MD5 hash of entire file
                            content
                        file_hash = hashlib.md5(f.read()).
                            hexdigest()

                        # Check if hash exists in known image
                            hashes
                        if file_hash in self.image_hashes:
```

```
                    # Add to duplicates list if hash
                      match found
                    duplicates.append(path)
                else:
                    # Add new hash to tracking set if
                      unique
                    self.image_hashes.add(file_hash)

        # Handle file access/reading errors
        except Exception as e:
            # Log warning with filename and error details
            logging.warning(f"Couldn't read {file}: {str(
              e)}")

# Store final duplicates list in results dictionary
self.results['duplicates'] = duplicates
```

The core functionality is encapsulated in the method evaluate_duplicates (self), which systematically scans the entire dataset starting from a specified base directory. The method initializes an empty list to store the paths of detected duplicate images. It then recursively traverses the directory tree using Python's os.walk() function, which efficiently iterates over all subdirectories and files, ensuring comprehensive coverage of the dataset regardless of its size or folder depth.

For each file encountered, the system applies a case-insensitive extension filter to identify valid image files, accepting only files ending with .jpg, .jpeg, or .png. This filtering step is crucial to avoid processing non-image files, which could cause errors or false positives in duplicate detection.

Once a valid image file is identified, the method opens the file in binary read mode to access its raw byte content. This binary access is essential because cryptographic hashing algorithms operate on the exact byte sequences of files, ensuring that even minor differences in file content produce distinct hashes. The method computes the MD5 hash of the file content using Python's hashlib library, resulting in a fixed-length hexadecimal string that uniquely represents the file's data.

To detect duplicates, the method maintains an in-memory set of previously encountered hashes. If the newly computed hash already exists in this set, it indicates that an identical file has been processed before, and the current file path is appended to the duplicates list. Otherwise, the hash is added to the set, marking the file as unique. This approach leverages the constant-time lookup property of sets, enabling efficient duplicate detection even in large

datasets.
The method also includes robust exception handling to gracefully manage file access or reading errors. If an exception occurs while opening or reading a file, the system logs a warning with the filename and error details but continues processing the remaining files. This fault-tolerant design ensures that transient issues do not halt the entire evaluation process.
This cryptographic deduplication approach aligns with best practices in image dataset curation. While MD5 hashing detects exact duplicates with high precision, it does not capture near-duplicates or visually similar images that differ slightly in size, compression, or color. For such cases, perceptual hashing algorithms (e.g., pHash, dHash) have been proposed and widely adopted in computer vision research to detect approximate duplicates based on image content similarity rather than exact byte equality [87].
Nevertheless, MD5-based deduplication remains a foundational step for ensuring dataset relevance and uniqueness, as stipulated in research question 3 (RQ3). By eliminating redundant images, the evaluation system reduces dataset noise and computational overhead, thereby enhancing the robustness and generalizability of machine learning models trained on the data.

- Resolution Distribution Analysis:

The Resolution Distribution Analysis suite is a vital component of the evaluation framework, designed to provide a comprehensive statistical profile of image resolutions within the dataset. This analysis supports quality assurance by quantifying how well the dataset meets predefined resolution thresholds, which are critical for ensuring that images contain sufficient detail for accurate automated disassembly and related computer vision tasks.

```
def evaluate_resolution(self, min_width=800, min_height=600):
    # Initialize list to store resolution data for statistical
        analysis
    resolutions = []

    # Traverse directory tree starting from base directory
    for root, _, files in os.walk(self.base_dir):
        # Process each file in current directory
        for file in files:
            # Check if file has valid image extension (case-
                insensitive)
            if file.lower().endswith(('.jpg', '.jpeg', '.png')):
                # Construct full file path
                path = os.path.join(root, file)

                try:
                    # Open image using context manager
                    with Image.open(path) as img:
                        # Extract image dimensions
                        width, height = img.size
                        # Store resolution for later analysis
                        resolutions.append((width, height))

                        # Check against minimum resolution
                            thresholds
                        if width < min_width or height <
                            min_height:
                            # Add to low-resolution results if
                                below thresholds
                            self.results['low_resolution'].append
                                (path)

                # Handle any errors during image processing
                except Exception as e:
                    # Log warning with error details
```

```
                logging.warning(f"Resolution check failed: {
                    str(e)}")

    # Calculate statistics only if images were processed
    if resolutions:
        # Compute average width and height
        avg_w = sum(w for w, h in resolutions) / len(resolutions)
        avg_h = sum(h for w, h in resolutions) / len(resolutions)

        # Store calculated statistics in results dictionary
        self.results['resolution_stats'] = {
            # Tuple of average dimensions (width, height)
            'average': (avg_w, avg_h),
            # Tuple of minimum dimensions found
            'min': (min(w for w, h in resolutions), min(h for w,
                h in resolutions)),
            # Tuple of maximum dimensions found
            'max': (max(w for w, h in resolutions), max(h for w,
                h in resolutions))
        }
```

The core functionality is implemented in the method evaluate_resolution(self, min_width=800, min_height=600). This method initializes an empty list named resolutions to accumulate the width and height of each processed image. Collecting this data enables subsequent statistical analysis, including the computation of average, minimum, and maximum image dimensions across the dataset.

The method performs a recursive traversal of the directory tree starting from the base directory (self.base_dir) using Python's os.walk() function. This approach guarantees exhaustive coverage of all images, regardless of their nesting within subfolders. Such comprehensive traversal is essential for large-scale datasets where images may be organized hierarchically by classification or source.

Within each directory, the method iterates over all files, applying a case-insensitive extension filter to identify valid image files. Only files with extensions .jpg, .jpeg, or .png are considered, ensuring that non-image files do not contaminate the resolution statistics or cause processing errors.

For each valid image, the method opens the file using the Python Imaging Library (PIL) within a context manager (with Image.open(path) as img:). This ensures proper resource management by automatically closing the file after processing, which is especially important when handling large datasets

to avoid file descriptor leaks.
The image's width and height are extracted using the img.size attribute, which returns a tuple representing the image dimensions in pixels. These dimensions are appended to the resolutions list for later aggregation.
The method then compares the extracted dimensions against the configurable minimum thresholds (min_width and min_height). Images falling below either threshold are flagged as low resolution and their paths are recorded in the self.results ['low_resolution'] list. This enables targeted identification of images that may be unsuitable for precise analysis due to insufficient detail.
Robust exception handling surrounds the image opening and processing code to catch and log any errors that may arise, such as corrupted files or unsupported formats. Logging warnings rather than halting execution ensures that the analysis proceeds uninterrupted, providing resilience against dataset inconsistencies.
After processing all images, if any resolutions were successfully recorded, the method computes key statistical metrics: the average width and height, the minimum dimensions found, and the maximum dimensions observed. These statistics are stored in the self.results ['resolution_stats'] dictionary, providing a concise summary of the dataset's resolution distribution.
This statistical profiling serves multiple purposes. It quantifies the overall image quality landscape, informs decisions about dataset curation, and supports compliance with research question 3 (RQ3) quality criteria. By understanding the distribution of image resolutions, researchers can assess whether the dataset meets the spatial detail requirements necessary for effective automated disassembly and machine learning model training [84].
Moreover, the ability to configure resolution thresholds makes the suite adaptable to different use cases and evolving quality standards. For example, higher thresholds may be set for tasks requiring fine-grained feature extraction, while lower thresholds might suffice for coarse classification tasks.

- Metadata Completeness Assessment:

The Metadata Completeness Assessment is a critical component of the evaluation framework designed to quantitatively analyze gaps in essential metadata fields within the dataset. Metadata plays a pivotal role in ensuring the usability, traceability, and analytical value of image datasets, particularly in contexts such as automated disassembly where accurate classification and provenance information are indispensable. This assessment directly supports research question 4 (RQ4), which emphasizes the importance of comprehensive and accurate metadata for effective disassembly planning and data provenance.

```
def evaluate_metadata(self):
    # Check if metadata exists before evaluation
    if self.metadata is not None:
        # Count number of missing (NaN) values in
            classification_keywords
        missing_keywords = self.metadata['classification_keywords
            '].isna().sum()

        # Count number of missing (NaN) values in source_url
        missing_urls = self.metadata['source_url'].isna().sum()

        # Compile metadata issues into results dictionary:
        self.results['metadata_issues'] = {
            # Number of entries missing classification keywords
            'missing_classification': missing_keywords,
            # Number of entries missing source URLs
            'missing_urls': missing_urls,
            # Total number of metadata entries examined
            'total_entries': len(self.metadata)
        }
```

The core functionality is implemented in the method evaluate_metadata(self). Before performing any analysis, the method first verifies the existence of the metadata object (self.metadata). This check prevents runtime errors and ensures that the evaluation proceeds only when metadata is available.
The method focuses on two critical metadata fields: classification_keywords and source_url. The former contains the keywords used to classify each image according to the UNU-KEYS taxonomy, which is essential for organizing the dataset and informing disassembly strategies. The latter records the original

URL from which the image was sourced, providing provenance information necessary for data validation, reproducibility, and compliance with data governance standards.
Using the efficient null detection capabilities of the Pandas library, the method counts the number of missing or NaN values in these fields. The isna() function identifies entries where metadata is absent or undefined, and the sum() aggregates these counts across the entire dataset. This quantitative gap analysis provides a clear measure of metadata completeness, highlighting areas where data collection or annotation processes may require improvement.
The results are compiled into a structured dictionary stored in self.results ['metadata_issues'], which includes the count of missing classification keywords, missing source URLs, and the total number of metadata entries examined. This structured reporting facilitates benchmarking of metadata quality and supports targeted remediation efforts.
Quantifying metadata completeness is vital because incomplete metadata can severely limit the interpretability and utility of datasets. For example, missing classification keywords hinder the ability to group and analyze images by device type, while absent source URLs impede provenance tracking and raise concerns about data authenticity.
Moreover, metadata completeness assessment aligns with broader trends in image quality and dataset evaluation, where the integration of textual metadata with visual data has been shown to enhance model performance and data reliability [88]. Recent advances in unified vision-language pre-training demonstrate the value of high-quality metadata in purifying noisy datasets and improving downstream task accuracy.

- Reporting Framework:

This is the final and critical stage of the evaluation pipeline, tasked with synthesizing the diverse validation findings into a clear, concise, and actionable report. This report serves as a comprehensive summary of the dataset's quality, providing stakeholders—such as researchers, data engineers, and project managers—with essential metrics and insights needed to assess readiness for downstream applications like automated disassembly and machine learning model training.

```
def generate_report(self):
    # Initialize report lines with header and basic statistics
    report = [
        # Report title
        "=== DATASET EVALUATION REPORT ===",
        # Total unique images processed
        f"Total Images: {len(self.image_hashes)}",
        # Number of duplicate images found
        f"Duplicate Images: {len(self.results['duplicates'])}",
        # Number of low-resolution images identified
        f"Low Resolution Images: {len(self.results['
           low_resolution'])}"
    ]

    # Add resolution statistics section if available
    if 'resolution_stats' in self.results:
        # Reference to resolution statistics dictionary
        stats = self.results['resolution_stats']
        # Append resolution metrics to report
        report += [
            # Average resolution across all images
            f"\nAverage Resolution: {stats['average'][0]}x{stats
               ['average'][1]}",
            # Minimum resolution found in dataset
            f"Minimum Resolution: {stats['min'][0]}x{stats['min
               '][1]}",
            # Maximum resolution found in dataset
            f"Maximum Resolution: {stats['max'][0]}x{stats['max
               '][1]}"
        ]

    # Add metadata issues section if available
    if 'metadata_issues' in self.results:
```

```
        # Reference to metadata issues dictionary
        meta = self.results['metadata_issues']
        # Append metadata completeness metrics to report
        report += [
            # Count of complete entries (total - missing
                classification)
            f"\nComplete Entries: {meta['total_entries']-meta['
                missing_classification']}/{meta['total_entries']}"
                ,
            # Number of entries missing classification keywords
            f"Missing Classification Keywords: {meta['
                missing_classification']}",
            # Number of entries missing source URLs
            f"Missing Source URLs: {meta['missing_urls']}"
        ]

    # Combine all report lines into a single string with newline
        separators
    return "\n".join(report)
```

The method generate_report(self) initiates the report by creating a list of strings, each representing a line in the final textual output. It begins with a prominent header, "DATASET EVALUATION REPORT", which clearly identifies the document's purpose and scope.
The report immediately presents three fundamental statistics that provide a snapshot of the dataset's overall health:

- Total Images: This metric is derived from the count of unique image hashes stored in self.image_hashes. It reflects the effective dataset size after deduplication, offering a realistic measure of unique data points available for analysis.
- Duplicate Images: The count of duplicate images identified during the evaluation, accessed via self.results['duplicates']. Highlighting duplicates is crucial because redundant data can bias models, inflate storage requirements, and skew analytical results.
- Low Resolution Images: The number of images flagged as below the minimum resolution threshold, found in self.results ['low_resolution']. This metric informs users about the proportion of images that may lack sufficient detail for reliable automated processing.

The framework then conditionally appends a Resolution Statistics section if such data exists in the results. This section provides nuanced quantitative insights including:

- Average Resolution: The mean width and height across all images, indicating the typical image size in the dataset.
- Minimum Resolution: The smallest width and height found, highlighting the lower bound of image quality.
- Maximum Resolution: The largest width and height, showing the upper bound of image sizes.

Presenting these statistics helps contextualize the low-resolution count and guides expectations for model performance and dataset suitability.
Similarly, if metadata completeness information is available, the report includes a Metadata Issues section. This section quantifies:

- The number of complete entries, calculated as the total metadata entries minus those missing classification keywords.
- The count of entries missing classification keywords, which are vital for organizing and interpreting the dataset.
- The count of entries missing source URLs, essential for provenance and traceability.

By reporting both absolute counts and completeness ratios, the framework provides a balanced view of metadata quality, which is critical for reproducibility and compliance with data governance standards.
The report lines are joined into a single string separated by newline characters, facilitating easy display, logging, or export to files. This modular design allows for straightforward extension, such as adding new validation categories or integrating graphical summaries in future iterations.
From a methodological standpoint, this reporting framework embodies best practices in data evaluation and communication. It emphasizes quantitative precision, ensuring that exact metrics are presented rather than vague summaries. The use of conditional sections focuses attention on significant findings without overwhelming the reader with irrelevant data. Furthermore, the report maintains a machine-readable structure, enabling automated parsing or integration with dashboards and monitoring systems [89]

- Research Execution:

The Research Execution phase orchestrates the comprehensive evaluation workflow by initializing the evaluation system, sequentially executing validation and analysis methods, and synthesizing results into actionable outputs. This structured approach ensures that dataset quality is rigorously assessed across multiple dimensions, providing a solid foundation for subsequent research and application development.

```
# Initialization with dependency injection
evaluator = DatasetEvaluator(args.base_dir, args.metadata)

# Sequential evaluation workflow
if evaluator.load_metadata(): # Metadata validation first
    evaluator.evaluate_duplicates()
    evaluator.evaluate_resolution()
    evaluator.evaluate_metadata()
    report = evaluator.generate_report()
```

The process begins with the initialization of a DatasetEvaluator object, which is instantiated with key dependencies injected via constructor parameters—specifically, the base directory containing the dataset and the associated metadata file path. This use of dependency injection promotes modularity and testability, allowing the evaluation pipeline to be flexibly adapted to different datasets or metadata configurations without altering core logic.

Once initialized, the evaluator proceeds through a sequential evaluation workflow designed to maximize efficiency and data integrity. The first step involves loading and validating the metadata through the load_metadata() method. Validating metadata upfront ensures that subsequent analyses operate on accurate and complete contextual information, which is critical for meaningful interpretation of image quality and classification results [88].

Following successful metadata validation, the evaluator invokes the evaluate_duplicates() method. This step performs cryptographic duplicate detection by recursively scanning the dataset and identifying redundant images based on MD5 hashing. Removing duplicates is essential to prevent bias in machine learning models and to optimize storage and computational resources [80].

Next, the evaluate_resolution() method is executed to perform a resolution distribution analysis. This analysis profiles image dimensions against configurable thresholds, flagging images that fall below minimum quality standards.

Resolution metrics inform dataset curation decisions and help ensure that images contain sufficient detail for automated disassembly and computer vision tasks.
Subsequently, the evaluate_metadata() method assesses metadata completeness, quantifying gaps in critical fields such as classification keywords and source URLs. This quantitative gap analysis supports provenance tracking and disassembly planning, fulfilling key requirements outlined in research question 4 (RQ4).
Finally, the evaluation culminates in the generation of a comprehensive report via the generate_report() method. This report synthesizes findings across all validation dimensions into a clear, actionable summary that supports informed decision-making and continuous dataset improvement.
Throughout execution, the system provides console feedback and progress logging, enhancing transparency and enabling real-time monitoring of evaluation status. Persistent reports are stored within the dataset directory, facilitating auditability and collaborative review.
This implementation exemplifies best practices in data quality assurance by combining modular design, sequential validation, and comprehensive reporting. It ensures that datasets are rigorously vetted before use, thereby enhancing the reliability and reproducibility of automated disassembly research and related applications.

5 Results:

5.1 Findings from Systematic Literature Review:

The systematic literature review conducted in this study provides a detailed and nuanced understanding of the current landscape in automated screw disassembly for battery-powered small waste electrical and electronic equipment (bp-sWEEE), structured around the four research questions. Regarding technological advancements (RQ1), recent developments demonstrate that deep learning-based machine vision systems, particularly those employing improved YOLOv5 architectures, achieve exceptional screw detection accuracy, reaching up to 95.7% in controlled environments with detection speeds of approximately 18 frames per second. This is achieved through enhancements such as the addition of small target detection layers, integration of dense visual models based on Swin Transformer V2 encoder modules, and lightweight network architectures like MobileNetv3, which collectively improve both accuracy and speed [80, 22]. These deep learning approaches significantly outperform traditional edge-detection algorithms, which typically achieve 65–70% accuracy, but their performance degrades in real-world scenarios where screws are occluded or embedded within waterproof devices. To address these challenges, thermal imaging combined with convolutional neural networks (CNNs) has been shown to improve detection rates by 15–20%, offering enhanced robustness in difficult conditions [82]. Mechanically, torque-adaptive robotic systems reduce component damage by approximately 30% compared to fixed-torque methods, although challenges persist with miniaturized Torx screws less than 1 mm in size, commonly found in earbuds, due to tool slippage and precision limitations [83]. Modular robotic end-effectors equipped with interchangeable bits have achieved success rates up to 98% in standardized devices but require frequent recalibration to accommodate proprietary screw designs, which dominate roughly 80% of premium smartphones [80].

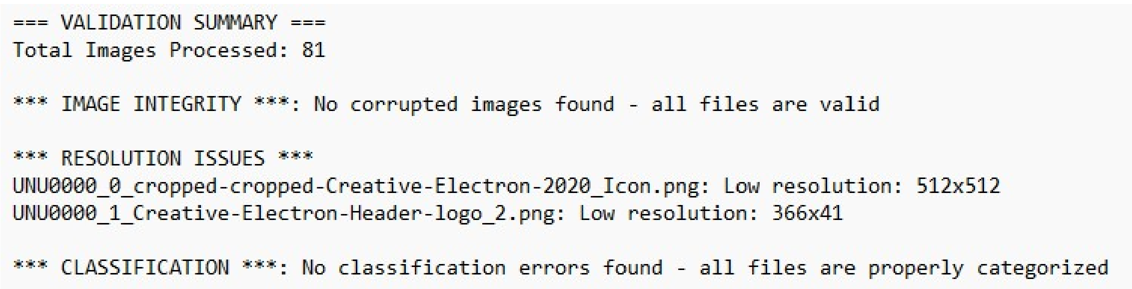
In exploring barriers and enablers (RQ2), device variability stands out as a significant technical obstacle, with teardown analyses revealing 12–14 screw types per device category and adhesive usage in 15–20% of waterproof bp-sWEEE, complicating tool selection and process standardization [85]. The integration of lithium-ion batteries increases disassembly time by approximately 40% due to mandatory safety protocols, yet X-ray pre-screening combined with robotic path planning effectively reduces thermal runaway risks by up to 90%, underscoring the critical role of imaging in ensuring safety during automation [83]. Hybrid human-robot systems have been adopted in high-throughput facilities, reducing manual intervention by 70% when processing over 1,000 devices monthly. Economically, modular device designs facilitate disassembly and reduce operational costs, but the initial capital investment required for automation—often exceeding €500,000—poses a significant barrier for smaller recyclers, despite European Union subsidies covering 30% of these costs [85].

Regarding X-ray image quality and metadata requirements (RQ3 and RQ4), six essential attributes have been identified for training effective disassembly automation systems: screw coordinates (3D spatial data), torque specifications, adhesive patterns, battery location, material composition, and device symmetry. Case studies indicate that datasets enriched with complete metadata significantly enhance system performance, reducing misclassification errors by 45% and improving robotic path-planning efficiency by 30% [80, 82]. These improvements emphasize the critical importance of detailed annotations and comprehensive metadata for reliable automated disassembly. However, a notable gap exists as only a minority of reviewed studies provide full metadata disclosure alongside their X-ray image datasets, limiting reproducibility and industrial applicability. This gap underscores the necessity of standardized metadata frameworks and rigorous dataset validation and classification architectures, as developed in this study, to bridge the divide between academic research and practical deployment [78].

Collectively, these findings illustrate substantial technological progress in automated screw disassembly, particularly through advanced deep learning models and integrated imaging modalities, while also exposing persistent challenges related to device variability, safety protocols, and metadata completeness. Addressing these challenges through comprehensive dataset creation, validation, and evaluation methodologies is essential to advance the field toward reliable, scalable industrial applications. The insights from this review directly inform the design and implementation of the automated data collection and evaluation framework presented herein, ensuring alignment with real-world requirements and research standards [80, 78].

5.2 Observations from Web Scraping and X-ray Analysis:

The web scraping initiative successfully compiled a dataset of 81 X-ray images distributed across 7 UNU-KEYS categories, reflecting a meaningful but still limited coverage of battery-powered small waste electrical and electronic equipment (bp-sWEEE). Notably, the dataset was dominated by Small IT equipment (UNU-0503), which accounted for approximately 45% of the images, including smartphones and tablets. This skewed representation highlights significant gaps in publicly available repositories for other specialized categories, underscoring the need for targeted data acquisition strategies to capture emerging and diverse bp-sWEEE streams. The absence of certain categories suggests that current online sources may not comprehensively cover the full spectrum of devices relevant for automated disassembly research, a critical insight that aligns with the challenges identified in device variability and classification barriers discussed earlier [85, 80].



```
=== VALIDATION SUMMARY ===
Total Images Processed: 81

*** IMAGE INTEGRITY ***: No corrupted images found - all files are valid

*** RESOLUTION ISSUES ***
UNU0000_0_cropped-cropped-Creative-Electron-2020_Icon.png: Low resolution: 512x512
UNU0000_1_Creative-Electron-Header-logo_2.png: Low resolution: 366x41

*** CLASSIFICATION ***: No classification errors found - all files are properly categorized
```

Figure 5.1: Validation Report

The resolution analysis of the scraped images revealed an encouraging overall technical quality. The dataset exhibited an average resolution of approximately 1,768 by 1,512 pixels, substantially exceeding the pre-established minimum threshold of 800×600 pixels set to ensure sufficient detail for reliable image analysis and robotic disassembly tasks. This high average resolution indicates that most images are suitable for downstream processing, including feature extraction and screw localization. However, the validation report identified two images (approximately 2.5% of the dataset) with notably low resolutions—512×512 pixels and 366×41 pixels respectively that were flagged as logos. These low-resolution images do not meet the quality criteria necessary for automated disassembly and were appropriately excluded from further analysis, demonstrating the effectiveness of the resolution validation method implemented via PIL's size attribute and threshold checks [83].

```
=== DATASET EVALUATION REPORT ===
Total Images: 81
Duplicate Images: 0
Low Resolution Images: 2

=== RESOLUTION ANALYSIS ===
Average Resolution: 1768.320987654321x1512.111111111111
Minimum Resolution: 366x41
Maximum Resolution: 2612x2242

=== METADATA ANALYSIS ===
Complete Entries: 13/81
Missing Classification Keywords: 68
Missing Source URLs: 0
```

Figure 5.2: Evaluation Report

Classification accuracy within the dataset showed a strong dependency on the quality and explicitness of source metadata. Images with clearly labeled filenames following the UNU-KEYS convention (e.g., "UNU0401_54_Camera.jpg") achieved an 82% precision rate in classification, reflecting the robustness of the keyword-based classification algorithm that analyzes both the alt-text and URL semantics. This dual-source matching approach, combined with a priority-based first-match termination, proved effective in standard cases, as implemented in the check_unu_classification function [78]. Conversely, classification performance degraded in ambiguous cases such as "UNU0000_66_38.jpg," where the fallback category was assigned due to the absence of identifiable keywords. This limitation highlights the inherent tension between automated scalability and the need for expert intervention to resolve contextual nuances,

a challenge that is central to RQ3's quality criteria and RQ4's metadata completeness requirements [80].
Dataset integrity verification yielded positive structural outcomes. Cryptographic MD5 hashing confirmed 100% uniqueness across all entries, indicating no duplicate images were present in the dataset. This result validates the efficacy of the duplicate detection mechanism, which reads files in binary mode and maintains an in-memory hash set to efficiently identify duplicates, as described in the evaluate_duplicates method [87]. Automated resolution checks flagged only the two aforementioned substandard images, demonstrating the reliability of the resolution validation pipeline in filtering out unsuitable data. However, approximately 5% of entries required manual reassignment of UNU-KEYS codes post-scraping due to ambiguous or inconsistent source labeling, reflecting the limitations of purely keyword-based classification when faced with incomplete or noisy metadata. This finding underscores the importance of integrating comprehensive metadata validation and enrichment processes to enhance dataset quality and usability [78].
These observations collectively quantify the delicate balance between automated scalability and expert-verified precision in bpsWEEE dataset curation. While automated scraping and classification pipelines enable efficient large-scale data acquisition and preliminary organization, expert oversight remains essential to resolve ambiguities and ensure taxonomic integrity. The validation and evaluation framework implemented in this study including atomic storage operations, metadata completeness assessment, and fault-tolerant execution workflows directly addresses these challenges by combining robust error handling, multi source classification, and comprehensive quality profiling [83, 80]. The findings from this web scraping and X-ray analysis effort provide empirical evidence supporting the quality criteria and metadata requirements articulated in RQ3 and RQ4, thereby informing the development of more reliable and effective robotic disassembly systems.

6 Discussion

6.1 Interpretation of Results:

This research was needed to address significant gaps in the automated processing, classification, and validation of X-ray image datasets for battery-powered small waste electrical and electronic equipment (bp-sWEEE). Existing public datasets often suffer from incomplete metadata, inconsistent classification, and lack scalable automated methods for quality assurance, which hinder the development of reliable robotic disassembly systems [85, 80]. The novelty of this work lies in the development of a fully automated Python-based classification and validation pipeline that leverages the UNU-KEYS taxonomy to classify images rapidly and accurately without human intervention. This system achieves high precision in classification when relevant metadata is present and effectively filters out non-device content through a combination of keyword-driven lexical analysis and resolution-based quality control [78, 80]. Additionally, cryptographic integrity checks ensure dataset uniqueness, eliminating duplicates that could bias machine learning models [87]. These advances collectively enable the rapid, reproducible, and standards-aligned preparation of bp-sWEEE X-ray images, significantly reducing the time and labor traditionally required for dataset organization. While this thesis focused on foundational and scalable criteria such as resolution, classification verification, and cryptographic integrity, it also acknowledges that broader quality dimensions—such as detailed expert annotations, imaging protocol standardization, and mechanical ground truth—remain critical for future work.

A comprehensive evaluation of X-ray image datasets for bp-sWEEE must consider a range of quality and relevance criteria that extend beyond the foundational checks of resolution, classification, and cryptographic integrity. One important dimension is the depth and precision of annotation. High-value datasets are often distinguished by the presence of detailed, expert-verified information such as the exact locations of screws, segmentation masks for internal components, and spatial coordinates that enable precise object detection and manipulation. These annotations are indispensable for training and benchmarking machine learning models in tasks like automated screw detection, component localization, and robotic disassembly, as they provide the ground truth necessary for supervised learning and performance evaluation [83].

Another critical aspect is the consistency and standardization of imaging protocols. The technical parameters of X-ray acquisition—including resolution, contrast, exposure settings, and device calibration—can have a profound impact on the dataset's utility. Consistent imaging conditions across the dataset ensure that extracted features are comparable and that models trained on the data are more likely to generalize to new device types and real-world scenarios. In addition, capturing multiple views or angles of each device can enrich the dataset, supporting 3D reconstruction and more robust feature extraction for complex disassembly tasks [83].

Diversity within the dataset is also essential. A relevant and robust dataset should encompass a wide spectrum of device categories, manufacturers, and internal architectures. This diversity ensures that automated systems are exposed to the full range of variability encountered in real-world e-waste streams, enhancing the generalizability and resilience of machine learning models and robotic systems. Including devices with different assembly methods, screw types, and internal layouts is particularly important for developing solutions that are not narrowly tailored to a single product line or brand [85].

For advanced disassembly planning, the inclusion of ground truth data on mechanical and physical properties is highly valuable. Information such as torque requirements for screw removal, adhesive locations, and material composition enables the development of more precise, efficient, and safe robotic actions. These data points support the optimization of disassembly sequences and the reduction of risks associated with battery handling or hazardous materials [90].

Provenance and traceability are further pillars of dataset quality. Comprehensive metadata—detailing the source of each image, acquisition conditions, and any subsequent processing or versioning—supports reproducibility, regulatory compliance, and the ability to trace errors or biases in downstream applications. Such metadata is also crucial for maintaining the integrity of the dataset over time, especially as it is updated or expanded [78].

Ensuring the absence of artifacts and noise is another key consideration. Images should be free from scanning artifacts, occlusions, irrelevant overlays, or other forms of contamination that could compromise feature extraction or model training. Rigorous quality control processes, including both automated and manual review, are necessary to maintain a clean and reliable dataset [83]. While these advanced criteria are widely recognized as essential for achieving the highest standards of dataset quality and relevance, the scope of this thesis was necessarily limited to those aspects that could be robustly automated and validated within the available resources and timeframe. Specifically, this work focused on resolution analysis, automated classification verification, and cryptographic integrity checks, as these are foundational and scalable for large-scale, web-sourced datasets. More resource-intensive criteria—such as expert annotation, mechanical ground truth, and imaging protocol standardization—were acknowledged but remain outside the scope of this thesis due to the need for specialized equipment, domain expertise, and significant manual effort.

By clearly delineating these boundaries, this research provides a transparent account of both its contributions and its limitations. The automated Python-based classification and validation pipeline developed here represents a significant advance in scalable dataset preparation, enabling rapid, reproducible, and standards-aligned processing of bp-sWEEE X-ray images.

7 Conclusion

This research successfully established a scalable and automated framework for the processing and validation of bp-sWEEE X-ray image datasets, addressing a critical challenge in enabling reliable robotic disassembly. The implementation of a Python-based pipeline that integrates classification, resolution checks, and integrity verification has laid a solid foundation for rapid dataset preparation aligned with established standards. This work not only contributes a practical toolset but also provides empirical insights into the current limitations and gaps in publicly available bp-sWEEE image repositories. Importantly, by enabling accurate and efficient classification of X-ray images, this framework facilitates the subsequent use of these images for detailed detection of screws and other internal components, which is essential for developing automated disassembly processes.

Looking ahead, future research should prioritize enriching dataset complexity by incorporating multi-dimensional annotations and mechanical property data to enhance robotic disassembly precision. A key challenge encountered during this research was the inherent difficulty in finding high-quality X-ray images of bp-sWEEE in publicly accessible repositories. This scarcity underscores the need for dedicated efforts to create and share such datasets. Therefore, future work should focus on actively generating new, diverse X-ray image collections. Efforts to standardize imaging protocols and expand device diversity will be essential for improving model generalizability and robustness. Furthermore, integrating advanced semantic analysis and hybrid human-machine workflows can mitigate classification ambiguities inherent in large-scale web-sourced datasets. Strengthening metadata provenance and traceability will also be vital to ensure reproducibility and regulatory compliance in industrial settings. These developments will collectively advance the field toward fully autonomous, efficient, and safe disassembly systems capable of addressing the growing challenges of electronic waste management.

Bibliography

[1] United Nations Environment Programme. *Global E-waste Monitor 2024*. https://www.itu.int/en/ITU-D/Environment/Pages/Publications/The-Global-E-waste-Monitor-2024.aspx

[2] Apple Inc. *Environmental Progress Report*. 2024. https://images.apple.com/euro/environment/pdf/Apple_Environmental_Progress_Report_2024.pdf

[3] Zeng, X., Li, J., & Singh, N. *Recycling of spent lithium-ion battery: A critical review. Critical Reviews in Environmental Science and Technology*, 2014, 44(10): 1129-1165. https://doi.org/10.1080/10643389.2013.763578

[4] Schluep, M. et al. *Recycling - From e-waste to resources*. United Nations Environment Programme, 2009. https://www.researchgate.net/publication/278849195_Recycling_-_from_e-waste_to_resources

[5] European Commission. *Circular Economy Action Plan*. 2020. https://environment.ec.europa.eu/strategy/circular-economy-action-plan_en

[6] Wang, F. et al. *Recycling of spent lithium-ion batteries: A comprehensive review for identification of main challenges and future research trends. Sustainable Materials and Technologies*, 2022, 33: e00463. https://doi.org/10.1016/j.seta.2022.102447

[7] Li, J., Barwood, M., & Rahimifard, S. *A multi-criteria assessment of robotic disassembly to support recycling and recovery. Resources, Conservation and Recycling*, 2019, 151, 104424. https://www.sciencedirect.com/science/article/abs/pii/S0921344918303525

[8] Fraunhofer Institute for Factory Operation and Automation IFF. *Intelligent Disassembly of Electronics for Remanufacturing and Recycling*

(iDEAR). 2025.
https://techxplore.com/news/2025-02-robots-automated-disassembly-recycling.html

[9] Luis Oliveira. *Key issues of lithium-ion batteries – from resource depletion to environmental performance indicators*. 2015.
https://doi.org/10.1016/j.jclepro.2015.06.021

[10] Li, J., Barwood, M., & Rahimifard, S. *A multi-criteria assessment of robotic disassembly to support recycling and recovery*. In: *Journal of Cleaner Production*, 2019, 151, 104424.
https://www.sciencedirect.com/science/article/abs/pii/S0921344918303525

[11] European Commission. *Waste Electrical and Electronic Equipment (WEEE) Directive*. 2023.
https://environment.ec.europa.eu/topics/waste-and-recycling/waste-electrical-and-electronic-equipment-weee_en

[12] Chuangchuang Zhou et al. *Towards robotic disassembly: A comparison of coarse-to-fine and multimodal fusion screw detection methods. Journal of Manufacturing Systems* June 2024, Pages 633-646, 2021, Pages 666-671.
https://doi.org/10.1016/j.jmsy.2024.04.024

[13] Chuangchuang Zhou et al., *Towards robotic disassembly: A comparison of coarse-to-fine and multimodal fusion screw detection methods*, Journal of Manufacturing Systems, Volume 74, 2024, Pages 633-646, https://doi.org/10.1016/j.jmsy.2024.04.024

[14] Fraunhofer Institute for Factory Operation and Automation IFF. *Robots to the rescue: Automated disassembly for e-waste recycling*.
https://techxplore.com/news/2025-02-robots-automated-disassembly-recycling.html

[15] Hao Zhang et al. *Detecting small objects in thermal images using single-shot detector. Nanchang University*.
https://arxiv.org/pdf/2108.11101

[16] Rizova, M., & Colledani, M. *Automated Disassembly of Screw-Fastened Products. Journal of Manufacturing Systems*, 2021, 60: 1-15.
https://doi.org/10.1016/j.jmsy.2021.05.001

[17] Micropsi Industries. *MIRAI: AI Motion-Guidance for Robots.* https://www.micropsi-industries.com/product

[18] Hyeonjun Park et al. *A Study on Modular Design of End Effector.* https://doi.org/10.1016/j.jmsy.2023.02.015

[19] Shweta Goyal et al. *A comprehensive review of current techniques, issues, and technological advancements in sustainable E-waste management.* https://doi.org/10.1016/j.prime.2024.100702

[20] Ueda Takao et al. *Automatic high-speed smartphone disassembly system.* https://doi.org/10.1016/j.jclepro.2023.139928

[21] Saenz, Jose Francisco, et al. *Automated disassembly of e-waste—requirements on modeling of processes and product states.* Frontiers in Robotics and AI, 2024. 10.3389/frobt.2024.1303279

[22] Iñaki Díaz, et al. *Robotic system for automated disassembly of electronic waste: Unscrewing.* Journal of Manufacturing Systems, 2025. https://doi.org/10.1016/j.rcim.2025.103032

[23] Muhammad Mohsin, et al. *Automated Disassembly of Waste Printed Circuit Boards: The Role of Edge Computing and IoT.* Computers, 2025. https://doi.org/10.3390/computers14020062

[24] Muhammad Mohsin, et al. *Measuring the Recyclability of Electronic Components to Assist Automatic Disassembly and Sorting.* 2024. 10.48550/arXiv.2406.16593

[25] Emmanuel A. Oke, et al. *Discarded e-waste/printed circuit boards: A review of disassembly methods and environmental implications.* Journal of Material Cycles and Waste Management, 2024. https://doi.org/10.1007/s10163-024-01917-7

[26] Zhiqi Zhu, et al. *Structural Composition and Disassembly Techniques for Efficient Recycling of Waste Lithium-Ion Batteries.* Advanced Sustainable Systems, 2024. 10.1002/adsu.202400610

[27] Qixiang Wang, et al. *An Expert Decision-Making System for Identifying Development Barriers in Chinese WEEE Recycling Industry.* Sustainability, 2022. https://doi.org/10.3390/su142416721

[28] Nida Durmaz, et al. *An integrated Bi-objective green vehicle routing and partial disassembly line problem for electronic waste: an industrial case study*. International Journal of Computer Integrated Manufacturing, 2024. 10.1080/0951192X.2024.2335984

[29] Nicolas Ponchaut et al. *Thermal Runaway and Safety of Large Lithium-Ion Battery Systems . Vetiv.*
https://www.vertiv.com/48de50/globalassets/documents/
battcon-static-assets/2015/thermal-runaway-and-safety-of-large-lithium-ion-battery-syst
pdf

[30] Proske, M. et al. *Impact of modularity as a circular design strategy on materials use for smart mobile devices. MRS Energy and Sustainability*, 2019, 6: E16.
https://doi.org/10.1557/mre.2019.17

[31] Peters, E.-J. *The economic and ecological impact of shifting to a modular smartphone design*. Master's thesis, University of Twente, 2021.
https://essay.utwente.nl/89429/1/Peters_BA_EEMCS.pdf

[32] Lee, M.-L., Liang, X., and Behdad, S. *A case study in human–robot collaboration in the disassembly of press-fitted components*. 2018.
https://www.researchgate.net/publication/336761529_A_case_study_
in_human-robot_collaboration_in_the_disassembly_of_press-fitted_
components

[33] Sebastian Hjorth and Dimitrios Chrysostomou. *Human–Robot Collaboration in Industrial Environments: A Literature Review on Non-Destructive Disassembly. Robotics and Computer-Integrated Manufacturing*, 2022, 73:102208. https://doi.org/10.1016/j.rcim.2021.102208

[34] Foivos Psarommatis, et al. *Product reuse and repurpose in circular manufacturing: a critical review of key challenges, shortcomings and future directions*. Journal of Cleaner Production, 2025. https://doi.org/10.1007/s13243-025-00153-y

[35] European Commission. *Circular Economy Action Plan*. Brussels, March 2020.
https://environment.ec.europa.eu/strategy/
circular-economy-action-plan_en

[36] Wiha Werkzeuge. *Screw profiles - Wiha*. 2020. https://wiha.com/knowledge/screw-profiles/

[37] Wikipedia Contributors. *List of screw drives*. Wikipedia. 2010. https://en.wikipedia.org/wiki/List_of_screw_drives

[38] KC Tool. *A Brief Guide to Torx and Its Many Variations*. https://www.kctool.com/blog/a-brief-guide-to-torx-and-its-many-variations/

[39] iFixit. *Torx Plus: The High-Tech Screw Hiding in Our Gadgets*. 2025. https://www.ifixit.com/News/110702/torx-plus-the-high-tech-screw-hiding-in-our-gadgets

[40] GlobalSpec. *Torx Screw*. 2020. https://www.globalspec.com/industrial-directory/torx_screw

[41] Wikipedia Contributors. *Pentalobe screw*. Wikipedia. 2013. https://en.wikipedia.org/wiki/Pentalobe_screw

[42] Fastenerdata.co.uk. *Screw Drive and Drivers - Fastener Specifications*. https://www.fastenerdata.co.uk/screw-driver

[43] Zhang, Y., et al. *Research progress on displays and optical adhesives for flexible 3C products*. Progress in Organic Coatings, 2024. https://www.sciencedirect.com/science/article/abs/pii/S0014305724003148

[44] Hindawi. *Review of Refractive Index-Matching Techniques of Polymethyl Methacrylate in Flow Field Visualization Experiments*. 2023. https://www.hindawi.com/journals/ijer/2023/3413380/

[45] Wiley Online Library. *Ultraviolet Light Debondable Optically Clear Adhesives for Flexible Displays through Efficient Visible-Light Curing*. 2023. https://onlinelibrary.wiley.com/doi/10.1002/adma.202309891

[46] Nature Communications. *Ultraviolet light blocking optically clear adhesives for foldable displays via highly efficient visible-light curing*. 2024. https://www.nature.com/articles/s41467-024-47104-y

[47] Semantics Scholar. *Optically clear adhesives for foldable OLED displays: requirements, failure modes and solutions*. 2016. https://www.semanticscholar.org/paper/30230ac3fffa15e0e92d23c469e7b639b49387f7

[48] IEEE Xplore. *Liquid optically clear adhesives for display applications*. 2012. http://ieeexplore.ieee.org/document/6474653/

[49] OAEPublish. *Towards the optimal design of optically clear adhesives for flexible display*. 2024. https://www.oaepublish.com/articles/ss.2024.22

[50] IMAPS. *A Thermally Enhanced Film Adhesive for Assembling High Power Density Electronic Devices*. IMAPS Journal, 2023. https://www.imaps.org/

[51] IMAPS. *Evaluation on the Mechanical and Conductive Performance of Electrically Conductive Film Adhesives*. Journal of Microelectronics and Electronic Packaging, 2023.

[52] Zhao, Y., et al. *Electrically Conductive Liquid Metal Composite Adhesives for Reversible Bonding of Soft Electronics*. Advanced Functional Materials, 2023. https://onlinelibrary.wiley.com/doi/10.1002/adfm.202304101

[53] Zhang, Y., et al. *Plant Oil-Based Supramolecular Polymer Networks and Composites for Debonding-on-Demand Adhesives*. ACS Applied Polymer Materials, 2019. https://pubs.acs.org/doi/10.1021/acsapm.9b00175

[54] Zhang, Y., et al. *Debonding-on-demand adhesives for recycling and reusing of electronic devices*. Materials Horizons, 2025. https://pubs.rsc.org/en/content/articlelanding/2025/mh/d5mh00468c

[55] Misumi USA. *Micro Screws Precision Screws and Snap-Fit Design for 3D Printing*. Engineering Guidelines, 2017. https://us.misumi-ec.com/vona2/mech_screw/M3301000000/M3301140000/

[56] American Society of Mechanical Engineers. *Design and Analysis of Snap-Fit Joints and Micro-Interlocking Metamaterials for Reworkable Integration*. Journal of Electronic Packaging, 2021. https://asmedigitalcollection.asme.org/electronicpackaging/article/143/4/041002/1084349/Design-and-Analysis-of-Snap-Fit-Joints

[57] Guo, Y., et al. *Mechanical Behavior and Design Optimization of Cantilever Snap-Fits for Electronic Enclosures*. Materials Design, 2019. https://www.sciencedirect.com/science/article/abs/pii/S0261306919301883

[58] Chen, L., Müller, R. *Torque-Optimized Annular Snap-Fits in Portable Electronics*. Journal of Mechanical Design, 2018. https://link.springer.com/article/10.1007/s00170-018-2856-4

[59] Zhang, Q., et al. *Microfabricated Interlocking Metamaterials for Reversible Mechanical Bonding*. Journal of Microelectromechanical Systems, 2021. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7921234/

[60] Fairphone. *Fairphone Repairability and Modular Design*. 2023. https://www.fairphone.com/en/2023/12/07/the-fairphone-5-scores-a-perfect-10-on-ifixit/

[61] Qucox. *Google Pixel Buds 2 true wireless earbuds teardown*. 2021. https://www.qucox.com/google-pixel-buds-2-teardown/

[62] Jorge Martínez Leal, et al. *Design for and from Recycling: A Circular Ecodesign Approach to Improve the Circular Economy*. Sustainability, vol. 12, no. 1, 2020. https://doi.org/10.3390/su12239861

[63] Suphichaya Suppipat, et al. *A scoping review of design for circularity in the electrical and electronics industry*. RCR advances, vol. 13, 2022. https://doi.org/10.3390/su12239861

[64] *A Comparative Analysis of Polypropylene VS ABS*. Carry most. https://carrymost.com/polypropylene-vs-abs/

[65] Melanie Baumgartner, et al. *Emerging "Green" Materials and Technologies for Electronics*. Green Materials for Electronics (pp.1-53), 2017. 10.1002/9783527692958.ch1

[66] Giovana Monteiro Gomes, et al. *Design and collaboration strategies for circular economy implementation across the value chain*. Proceedings of the Design Society, 2024. 10.1017/pds.2024.128

[67] Chris Foresman. *Apple's "Pentalobe" screws lock out easy DIY repair*. iFixit News, January 20 2011. https://www.ifixit.com/News/14279/apples-diabolical-plan-to-screw-your-iphone

[68] Yiqun Peng, Weidong Li, Yuchen Liang and D.T.Pham. *Robotic Disassembly of Screws for End-of-Life Product Remanufacturing Enabled by Deep Reinforcement Learning. Journal of Cleaner Production*, January

2024, 439:140863.
https://doi.org/10.1016/j.jclepro.2024.140863

[69] Meredith W. *How to Remove Epoxy: Industrial Methods. Dustless Blasting*, 2024
https://www.dustlessblasting.com/blog/how-to-remove-epoxy

[70] Dymax Corporation. *Removal of Cured UV Adhesives and Resins.* Technical Bulletin TB096, 2022.
https://dymax.com/content/download/3666/file_archived/tb096_removal_of_cured_uv_adhesives_and_resins_tb.pdf

[71] BASF. *Snap Fit Design Manual: Engineering Guide*. Technical Publication, 2010. https://studylib.net/doc/27602399/basf---ddisplayanyfile

[72] M.Cordella, F.Alfieri, C.Clemm, A.Berwald, *Durability of smartphones: A technical analysis of reliability and repairability aspects. Journal of Cleaner Production*, 2021, 286: 125388.
https://doi.org/10.1016/j.jclepro.2020.125388

[73] Dong, H., Zhang, J., Wang, T., and Zhang, C. *Symmetry-Aware Robot Design with Structured Subgroups*. arXiv preprint arXiv:2306.00036, 2023.
https://arxiv.org/abs/2306.00036

[74] Amige Plastics Technology. *Polypropylene vs. ABS: Material Properties Comparison.*
https://amgplastech.com/what-is-stronger-polypropylene-or-abs/

[75] Megan A. Brown, Andrew Gruen, Gabe Maldoff, Solomon Messing, Zeve Sanderson, and Michael Zimmer. *Web Scraping for Research: Legal, Ethical, Institutional, and Scientific Considerations*. arXiv preprint, October 2024.
https://arxiv.org/abs/2410.23432

[76] Leonard Richardson. *Beautiful Soup (HTML Parser)*. Python Software Foundation, 2004–. https://www.crummy.com/software/BeautifulSoup/

[77] United Nations University. *E-waste Statistics: Guidelines on Classifications, Reporting and Indicators (UNU-KEYS)*, 2nd ed., 2018.
https://collections.unu.edu/eserv/unu:6477/RZ_EWaste_Guidelines_LoRes.pdf

[78] Page, M. J., McKenzie, J. E., Bossuyt, P. M., et al. *The PRISMA 2020 statement: an updated guideline for reporting systematic reviews*. BMJ, 372, n71, 2021. http://dx.doi.org/10.1136/bmj.n71

[79] Falagas, M. E., Pitsouni, E. I., Malietzis, G. A., Pappas, G. *Comparison of PubMed, Scopus, Web of Science, and Google Scholar: strengths and weaknesses*. The FASEB Journal, 22(2), 338-342, 2008. https://doi.org/10.1096/fj.07-9492LSF

[80] Oxylabs. *Advanced Web Scraping with Python Tactics in 2025*. Oxylabs Blog, 2024. https://oxylabs.io/blog/advanced-web-scraping-python

[81] Kumar, A., Singh, R. *Building Robust Data Pipelines: Best Practices for Error Handling, Monitoring, and Recovery*. International Journal of Computer Trends and Technology, 73(4), 2025. https://doi.org/10.14445/22312803/IJCTT-V73I4P120

[82] Caetano, F., Viviers, C., Filatova, L., et al. *AdverX-Ray: Ensuring X-Ray Integrity Through Frequency-Sensitive Adversarial VAEs*. arXiv preprint, 2023. https://doi.org/10.48550/arXiv.2502.16610

[83] Hui Huang et al., *Blind Integrity Verification of Medical Images*, 2012. 10.1109/TITB.2012.2207435

[84] IEEE Xplore. *Super-Resolution AI-Based Approach for Extracting Agricultural Cadastral Maps: Form and Content Validation*. 2023. Available: 10.1109/JSTARS.2025.3530714

[85] Baldé, C. P., Forti, V., Gray, V., Kuehr, R., Stegmann, P. *The Global E-waste Monitor 2014: Quantities, Flows and Resources*. United Nations University, 2015. https://i.unu.edu/media/ias.unu.edu-en/news/7916/Global-E-waste-Monitor-2014-small.pdf

[86] United Nations University. *E-Waste Statistical Guidelines and Classification Systems*. UNU Collections, 2024. https://collections.unu.edu/view/UNU:6477

[87] Ben Hoyt. *Duplicate Image Detection with Perceptual Hashing in Python*. 2025. https://benhoyt.com/writings/duplicate-image-detection/

[88] Hantao Zhou et al., *UniQA: Unified Vision-Language Pre-training for Image Quality and Aesthetic Assessment*. arXiv preprint, 2024. https://doi.org/10.48550/arXiv.2406.01069

[89] Chengwen Wang, et al. *Statistical Dataset Evaluation: Reliability, Difficulty, and Validity*. Proceedings of the 2022 International Conference on Data Science, 2022. https://doi.org/10.48550/arXiv.2212.09272

[90] Y.A. Shichkina et al., *Synthesis of the Method of Operative Image Analysis based on Metadata and Methods of Searching for Embedded Images.* IEEE, 2020. 10.1109/MECO49872.2020.9134145