applied sciences

MDPI

*Article*

# Machine Learning-Based Label Quality Assurance for Object Detection Projects in Requirements Engineering

**Neven Pičuljan * and Željka Car**

Faculty of Electrical Engineering and Computing, University of Zagreb, 10000 Zagreb, Croatia; zeljka.car@fer.hr
* Correspondence: neven.piculjan@fer.hr

**Featured Application: Our machine learning-based label quality assurance demo showcases the potential of our approach to improve object detection projects within the data requirement stage of requirements engineering. The accompanying API enables easy integration with existing platforms. This approach reduces the resources needed for label quality assurance.**

**Abstract:** In recent years, the field of artificial intelligence has experienced significant growth, which has been primarily attributed to advancements in hardware and the efficient training of deep neural networks on graphics processing units. The development of high-quality artificial intelligence solutions necessitates a strong emphasis on data-centric approaches that involve the collection, labeling and quality-assurance of data and labels. These processes, however, are labor-intensive and often demand extensive human effort. Simultaneously, there exists an abundance of untapped data that could potentially be utilized to train models capable of addressing complex problems. These raw data, nevertheless, require refinement to become suitable for machine learning training. This study concentrates on the computer vision subdomain within artificial intelligence and explores data requirements within the context of requirements engineering. Among the various data requirement activities, label quality assurance is crucial. To address this problem, we propose a machine learning-based method for automatic label quality assurance, especially in the context of object detection use cases. Our approach aims to support both annotators and computer vision project stakeholders while reducing the time and resources needed to conduct label quality assurance activities. In our experiments, we trained a neural network on a small set of labeled data and achieved an accuracy of 82% in differentiating good and bad labels on a large set of labeled data. This demonstrates the potential of our approach in automating label quality assurance.

**Keywords:** artificial intelligence; computer vision; data requirements; data-centric artificial intelligence; deep learning; label quality assurance; machine learning; object detection; requirements engineering



**Citation:** Pičuljan, N.; Car, Ž. Machine Learning-Based Label Quality Assurance for Object Detection Projects in Requirements Engineering. *Appl. Sci.* **2023**, *13*, 6234. https://doi.org/10.3390/app13106234

Academic Editors: Wenjie Zhang and Zhengyi Yang

Received: 29 April 2023
Revised: 15 May 2023
Accepted: 17 May 2023
Published: 19 May 2023



## 1. Introduction

Modern artificial intelligence has come to rely heavily on artificial neural networks, which are predominantly trained through supervised learning with labeled data. A variety of neural network architectures have emerged to tackle diverse problems, and their performance is often dependent on the quality of the underlying labeled data. Large language models (LLMs), such as GPT-3 [1], have gained widespread popularity due to their impressive capabilities. The labeling tasks required for training models such as GPT-3 are relatively simple, facilitating easy scaling. These autoregressive models predict future words based on past words, and labels can be programmatically generated from vast text corpora. However, to harness more specific outputs from these models, fine-tuning of specially labeled data is necessary [2]. In contrast, generating inexpensive, high-quality labels for images is a more challenging process, even for pretraining large computer vision models. Labels, in the context of supervised learning, are the desired outputs or target values associated with the input data. They are used to guide the learning process of the

neural network by providing a reference to the correct output. In object detection tasks, labels are the annotations that provide information about the location and category of objects within an image. For object detection, labels typically consist of two main components: bounding boxes and class labels. Bounding boxes are rectangular boxes drawn around each object of interest in the image. A bounding box is usually represented by the coordinates of its top–left and bottom–right corners, which define the position and size of the box. Class labels are the categories or classes assigned to each object within the bounding boxes. For example, if an object is a cat, its class label would be "cat". Another example of a task is to generate photo-realistic images from textual inputs, as demonstrated by models such as DALL·E 2 [3] and Stable Diffusion [4]. These models require datasets that contain both images and their corresponding textual descriptions. Labels play a crucial role in ensuring the quality of the generated images. Errors or inaccuracies in the textual descriptions can lead to the generation of images that do not align with the intended content.

This research paper concentrates on the computer vision subfield within the broader context of modern artificial intelligence. Computer vision enables machines to extract meaning or semantics from images or videos, thereby allowing them to understand the contents. There are three popular tasks in computer vision: detection, recognition and segmentation. Three sample images from the Microsoft Common Objects in Context (MS COCO) dataset [5] are showcased, each demonstrating one of these tasks with a cat as the target object. Detection involves generating a bounding box around the object [6] (Figure 1), recognition involves assigning a tag to describe the object in the image [7] (Figure 2), and segmentation involves classifying pixels at the individual level [8] (Figure 3). Segmentation is the most comprehensive task, as detection and recognition can usually be derived from it.

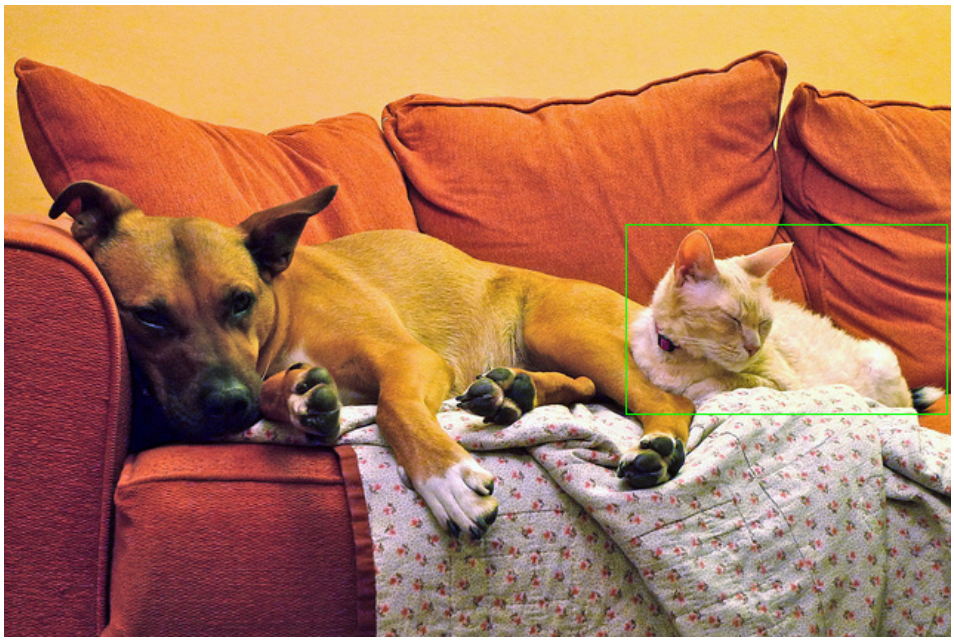


**Figure 1.** Detection: cat enclosed by a bounding box.

**Figure 2.** Recognition: cat identified and tag is assigned to the image.

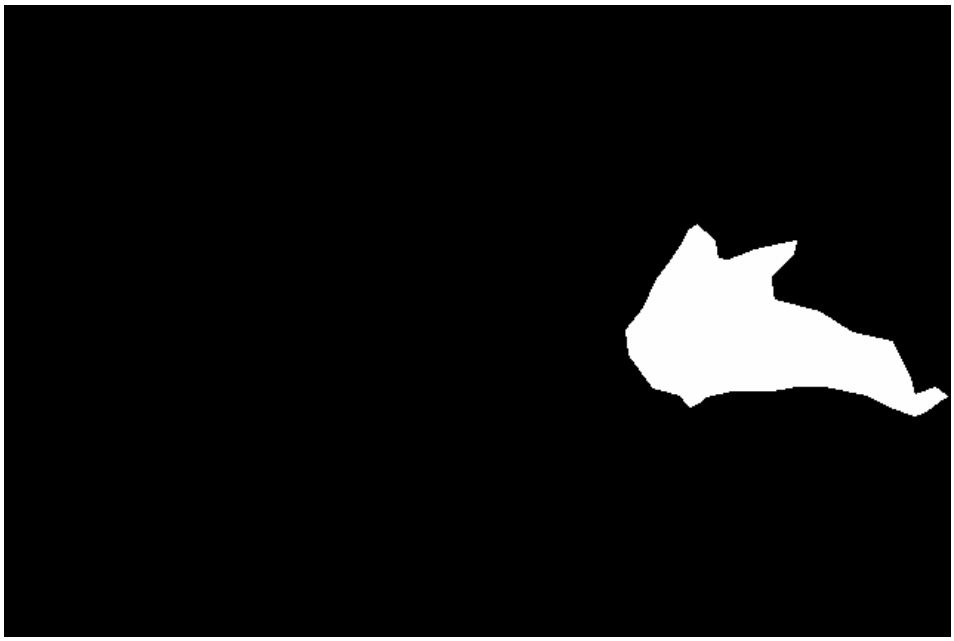


**Figure 3.** Segmentation: cat and background pixels distinguished.

In our research, we address problems related to data requirements for computer vision, with a particular focus on quality assurance of data labels. We propose a comprehensive framework for data collection, labeling and quality assurance of data and labels for machine learning projects. Furthermore, we provide a detailed explanation of our machine learning-based label quality assurance method to assess whether the ground truth assigned for a specific task and image is indeed accurate. This method allows us to determine if the labeled data accurately represent the ground truth, ensuring the reliability of the training data for machine learning models. Our method aims to reduce the time and resources needed for label quality assurance activity by offering immediate estimates of label quality to annotators and computer vision project stakeholders. While we focus on object detection as an example, our method can be extended to other tasks and artificial intelligence subfields.

In order to provide a clear overview of the paper's structure, we present an article map outlining the content of each section.

- **Related Work**: This section explores the crucial role of data requirements in artificial intelligence projects, distinguishes them from traditional software engineering needs and surveys various methodologies for label quality assurance, such as manual review, inter-annotator agreement, deep active learning and algorithmic approaches.
- **Proposed Process for Data Requirements within Requirements Engineering**: This section outlines a proposed process for incorporating data requirements within requirements engineering for artificial intelligence projects, placing the "Data Requirements" phase between "Requirements Elicitation" and "Requirements Analysis" activities.
- **Machine Learning-Based Label Quality Assurance**: This section discusses a machine learning-based label quality assurance method that involves a series of steps to build a quality assurance model that classifies object detection labels as good or bad, helping annotators focus on samples with potentially incorrect labels.
- **Results and Discussion**: This section details the performance of a machine learning-based label quality assurance model utilizing ResNet-18 components and a fully connected neural network that attains 82% accuracy in discerning good and bad labels for object detection tasks, even when trained on a small dataset.
- **Conclusions**: This section underscores the significance of data requirements for artificial intelligence development and underlines a pioneering machine learning methodology for automating label quality assurance, which, while demonstrated for object detection, offers flexibility for adaptation across a variety of computer vision tasks and other machine learning subfields.

## 2. Related Work

In this section, we discuss the related work in two areas: data requirements in the context of requirements engineering and approaches to label quality assurance. We explore how requirements engineering has been adapted to accommodate the unique needs of artificial intelligence projects. Furthermore, we examine various methods for ensuring label quality, ranging from manual review to deep active learning. This review of related work provides a foundation for understanding the significance of data requirements and label quality assurance in the development of artificial intelligence solutions.

### *2.1. Data Requirements in the Context of Requirements Engineering*

Software requirements are the needs and constraints placed on a software product that contribute to finding a solution for a specific real-world problem. The systematic handling of requirements is referred to as "requirements engineering" [9]. Common phases in requirements engineering include requirements elicitation, requirements analysis, requirements specification and requirements validation.

The field of artificial intelligence has expanded considerably in recent years. Projects in this domain differ significantly from traditional software engineering projects due to their heavy reliance on data. There have been attempts to adapt requirements engineering for artificial intelligence projects to accommodate these unique characteristics. In computer vision projects, the goal is to automatically process images to extract hidden knowledge, creating both value and experience for users. Outputs of the project are a trained instance of a neural network (or model), extracted knowledge, generated images, etc., and their quality depends on the data.

A few papers discuss the need to refine requirements engineering to support the needs of artificial intelligence projects. One paper suggests that training data are an integral part of any machine learning system. The authors argue that data requirements may play a larger role in specifying machine learning systems than in conventional systems, potentially introducing a new class of requirements called data requirements [10]. Another paper posits that to find a suitable technical solution, data requirements must be clarified. Business owners should provide data sources and examine potential data ethics issues. Domain

experts should confirm the appropriate use of data. Data scientists should closely examine data completeness, sample distribution and the assumption that the data are independent and identically distributed, among other factors. It is mentioned that about 80% of the time in machine learning application development is spent preparing data. As coding is not as challenging, data requirements should likely dominate the cost [11]. The authors of a related paper emphasize the importance of collaboration between software engineers and data scientists to harness big data analytics during the software development process [12]. They propose a new requirements engineering model that enables both parties to work together in discovering hidden business values through data mining and analytics. This approach ensures that software systems are developed to fully exploit existing and newly generated data, ultimately leading to more effective and evidence-based decision making.

A study on modern deep learning systems addresses the challenges of determining the appropriate amount and type of data needed for optimal performance [13]. The researchers propose a novel paradigm for modeling the data collection workflow as a formal optimal data collection problem, enabling designers to specify performance targets, collection costs, a time horizon and penalties for not meeting the targets. This approach reduces the risks of failing to meet performance targets in various tasks, such as classification, segmentation and detection, while keeping total collection costs low. A different paper proposes a new approach to estimate the number of samples needed for a model to achieve the targeted performance, overcoming the limitations of the power law when extrapolating from small datasets [14]. The authors utilize a random forest regressor trained via meta-learning, which generalizes across various tasks and architectures. This method significantly improves performance estimation across classification and detection datasets while also reducing over-estimation of data. Another paper addresses the critical question of determining how much additional data are needed to reach a target performance given a small training dataset and a learning algorithm, particularly in applications where data collection is expensive and time-consuming [15]. The authors systematically investigate a family of functions that generalize the power law function to better estimate data requirements across various computer vision tasks. By incorporating a tuned correction factor and collecting data over multiple rounds, their approach significantly improves data estimators' performance, ultimately saving development time and data acquisition costs.

One of the examined papers highlights that the current focus of the artificial intelligence industry is on machine learning as a data-driven approach due to the information technology (IT) infrastructural developments available today, such as fast processing power and inexpensive data storage [16]. The authors state that healthcare is one of the areas most attractive for artificial intelligence applications, yet it faces notable obstacles such as the lack of mandatory standards or continuous data exchange. By properly understanding data requirements activities, these problems can be resolved, and engineers can proceed with building artificial intelligence solutions once requirements specifications, with an emphasis on data requirements, are finalized. This specific healthcare case can be addressed by understanding HIPAA [17] and GDPR [18] regulations and FHIR [19] and DICOM [20] standards used for handling sensitive patient medical data.

### 2.2. Approaches to Label Quality Assurance

The most effective method for ensuring label quality is to manually review samples and flag those with bad-quality labels for reannotation. However, this approach can be time-consuming and tedious for human reviewers, and it may become less accurate if they lose concentration while working on a monotonous task.

An alternative approach is to assign multiple annotators to the same image and calculate inter-annotator agreement [21,22]. Low inter-annotator agreement indicates that the image likely needs to be reannotated. This approach has some challenges, such as defining a suitable inter-annotator agreement measure for different machine learning tasks. Various authors have proposed methods for assessing the quality of suggested dataset labels in the context of human labeling [23–25]. Another issue with inter-annotator agreement

is handling specific cases wherein not all annotators annotated all samples [26]. In a relevant study, the authors explore the inter-annotator agreement among multiple expert annotators when segmenting lesions and abnormalities on medical images, which is crucial for the performance of artificial intelligence-based medical computer vision algorithms [27]. They propose the use of three metrics for qualitative and quantitative assessment. The experiments demonstrate the consistency of the inter-annotator reliability assessment and highlight the importance of combining different metrics to avoid biased evaluations. In a study focusing on lung ultrasounds for detecting COVID-19 manifestations, the authors investigated the agreement among physicians when identifying signs associated with the disease [28]. In a separate study that concentrates on manual segmentation of gliomas using magnetic resonance imaging (MRI), the authors scrutinized the level of agreement between novices and experts at different stages of the disease. MRI is a non-invasive medical imaging technique that employs strong magnets, radio waves and a computer to generate detailed images of the body's internal structures with the intention of diagnosis [29]. The authors found that the inter-rater agreement varied depending on the stage of the disease, with higher agreement generally observed between experts than between novices, particularly for non-glioblastoma cases. A different study presents a quality control protocol using an automated tool for assessing functional MRI data quality and assesses the inter-rater reliability of four independent raters [30]. The authors suggest several approaches to increase rater agreement and reduce disagreement for uncertain cases, ultimately aiming to improve classification consistency in data quality assessments.

Label quality assurance can also be achieved using a form of deep active learning [31–33]. Deep active learning focuses on a continuous loop between the model being trained and the human annotator. This approach requires a neural network architecture that can already produce reasonably accurate predictions for the target task and a large amount of high-quality labels. For example, to implement deep active learning for an object detection task, the entire pipeline for building the object detection model must be set up with a model capable of providing predictions of a certain quality. Samples requiring attention are identified by the model and subsequently directed for reannotation.

Platforms such as Amazon Mechanical Turk enable crowdsourcing of data and labels [34]. Annotators from various parts of the world and with different backgrounds participate, making it challenging to ensure the quality of large-scale labeling tasks. It is important to note that not only human reviewers or annotators can perform label quality assurance. Various algorithms can flag suspicious samples based on specific heuristics. Machine learning approaches such as deep active learning can be employed to flag low-quality labels. Filters based on label dimensions, shape or color of the annotated region can be applied to address certain issues. Images that should not be labeled at all can be filtered using heuristics or machine learning approaches as part of data quality assurance activity. A blur detector [35] can be used to remove overly blurry images. Width and height information about an image can be employed to eliminate images that are too small or too large and thus immediately unsuitable for further labeling tasks.

## 3. Proposed Process for Data Requirements within Requirements Engineering

We propose a process for incorporating data requirements within requirements engineering for artificial intelligence projects. The typical requirements engineering activities include "Requirements Elicitation", "Requirements Analysis", "Requirements Specification" and "Requirements Validation". Our proposed "Data Requirements" phase is inserted between the "Requirements Elicitation" and "Requirements Analysis" activities and consists of "Legal and Regulatory Data Requirements", "Production-like Image Data Acquisition", "Data Quality Assurance", "Data Labeling" and "Label Quality Assurance" activities. Figure 4 illustrates the general requirements engineering activities with the addition of the data requirements activities.

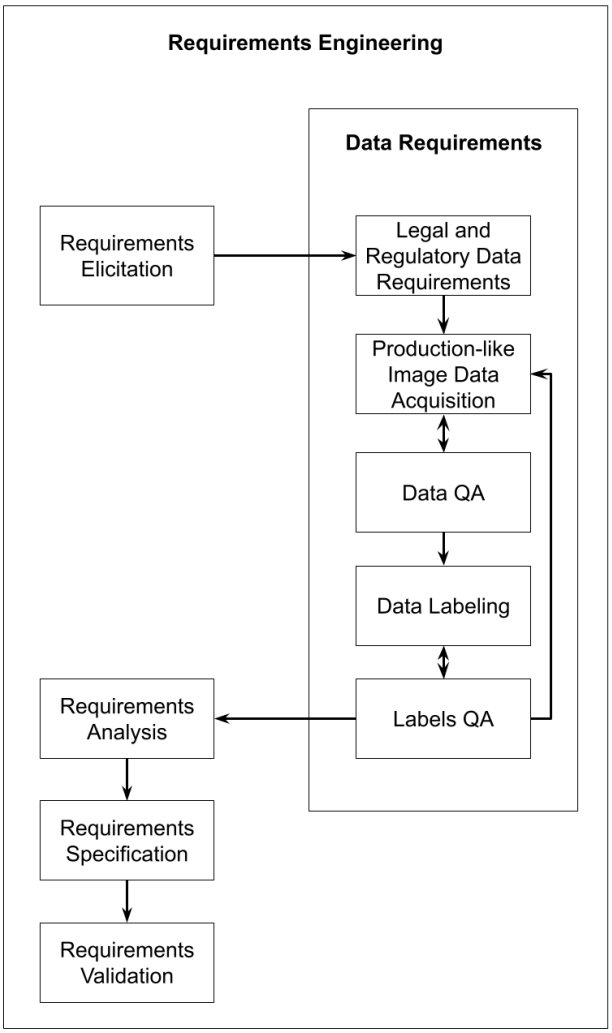



**Figure 4.** Requirements engineering process with data requirements activities for machine learning projects. "QA" stands for "Quality Assurance".

During the requirements elicitation activity, which is the initial phase of understanding the problem that the software must address, it is essential to determine whether a continuous stream of similar data is available and whether these data will be accessible once the project is completed. This consideration is specifically tied to artificial intelligence projects because the performance and effectiveness of machine learning models heavily rely on the availability and quality of data. Ensuring that there is a consistent flow of relevant data for training, validation and testing purposes allows the model to adapt and improve over time, leading to better overall performance and long-term viability of the artificial intelligence solution. The effort required to extract knowledge from the data, whether human or programmatic, must also be assessed. Additionally, it is crucial to identify any issues related to the speed, accuracy or cost of this effort. If a significant investment is being made, the next question to answer is whether the task can be framed as a segmentation, recognition or detection task with a fixed number of predefined classes. Examples of such tasks include detecting abnormalities in MRI scans, classifying traffic lights, detecting cats or creating a visual substitution system for blind people.

Following the requirements elicitation activity, the data requirements phase begins. During the legal and regulatory data requirements activity, any relevant regulations are identified and steps are taken to ensure compliance. For example, with medical images, HIPAA and GDPR regulations require patient consent or data de-identification [36]. Once

regulatory compliance is understood, production-like image data acquisition can commence. Data should be from the same distribution as the data in production. This means that the data used during the project should closely resemble the real-world data the model will ultimately process when deployed. Further, maintaining a connection to the data source throughout the project's life cycle is crucial. Data sources can include raw camera photos or MRI scans from specific MRI scanners. It is also important to remove any irrelevant data and to ensure a balanced dataset. Data ready for machine learning processes can be stored in commercial storage solutions such as Amazon S3 on Amazon Web Services [37], Azure Blob Storage on Microsoft Azure [38] or Cloud Storage on Google Cloud [39].

Subsequently, data quality assurance is performed by checking structural attributes such as image width, height and pixel histograms and visually inspecting the data's semantics. If any issues with the data are found (e.g., images being too small, too large or containing excessive noise), the production-like image data acquisition activity should be repeated. The frequency of repeating the production-like image data acquisition activity depends on data quality and project requirements. A representative subset of the collected data should be assessed, and if significant issues are found, the process may be repeated for all or affected images. Decisions should consider the project's needs, resources and timeline.

The next step is to label the data according to the defined classes. Labels should correspond to the expected output of the trained model, and in some cases, expert knowledge may be necessary. The amount of labeled data needed for training a machine learning model on image data depends on the variance in factors such as content, lighting, image quality, backgrounds and camera angles. High-variance image datasets require more labeled data to help the model generalize better, while low-variance datasets with simpler patterns may need less labeled data to achieve good performance.

The labels quality assurance activity follows, during which the quality and distribution of labels (proportions of different objects or classes within the labeled dataset, as well as the spatial distribution of these objects in the images) are checked. If the quality is bad, the data labeling activity should be repeated. This work further elaborates on what constitutes good or bad label quality in Section 4.3. If there are many outliers in the labels and the reason is related to the data content, it may be necessary to return to the production-like image data acquisition activity. The final product is computer vision software that generates predictions, such as a cat detector system, face recognition system or other similar applications.

There are certain assumptions about the project. Although this work focuses on detection, recognition and segmentation tasks in computer vision and presents a machine learning-based label quality assurance method for object detection tasks, the approach can be extended to other computer vision tasks and other machine learning domains, ranging from natural language processing to tabular data analysis. The software can be cloud-based. Security, privacy, types of cloud resources, cost and other aspects are not discussed in this work, as it is assumed that industry standards are followed. The response time of the system is also not discussed, but it is assumed to be reasonable—not in real-time, but not taking several minutes either. A response time in the range of seconds is expected. Some limitations are present in the proposed data requirements process. Over-regulation can be an issue, as highly regulated data for solving specific tasks can potentially halt the project. Moreover, if there is high variance in a large dataset or a multitude of different labels, creating a model with the current state of technology may be impossible (e.g., a general object detection model).

In the following section, a novel machine learning-based label quality assurance method is presented. It uses a small amount of carefully annotated data to build a model that can solve the binary classification task of estimating whether a label for a given image is of good or bad quality. By automating the process of determining label quality, the need for manual inspection and validation of labels is reduced, saving time and resources.

## 4. Machine Learning-Based Label Quality Assurance

*Demo is accessible at https://label-qa.piculjantechnologies.ai/. The accompanying API can be found at https://label-qa.piculjantechnologies.ai/?view=api (accessed on 16 May 2023).*

In this section, we describe the proposed machine learning-based label quality assurance method. To contextualize this within the proposed process for data requirements within requirements engineering, let us assume that during the "Requirements Elicitation" activity, the objective is to build an object detection computer vision software product capable of detecting objects in images belonging to 80 classes (e.g., "apple", "banana", "person", "umbrella" and so on). It is assumed that the "Legal and Regulatory Data Requirements" activity is completed, as well as the "Production-like Image Data Acquisition" activity, which a dataset that accurately represents the problem is collected. For the "Data Quality Assurance" activity, we assume that all images are of acceptable sizes and qualities.

The method starts with labeling a small random subset (approximately 5% of the full collected dataset) carefully to ensure that the labels are of very high quality. Let us call this the *small set*, and the remaining approximately 95% of the full collected dataset is the *large set*. For the purpose of this work and for building and evaluating the machine learning-based label quality assurance method, we rely on a publicly available labeled object detection dataset.

Using the ground truth label information about a carefully annotated *small set*, large amounts of good and bad labels for every sample can be created to train a machine learning-based label quality assurance model. With this model, there is no need to assign a reviewer to examine all remaining samples from the *large set* and find the ones with bad labels or to assign multiple annotators to all samples from the *large set* to perform inter-annotator agreement. Instead, the model can determine whether a label is good or bad by being trained on a large number of good and bad labels. Once it has learned the patterns and features that distinguish good labels from bad ones, it can then evaluate the quality of new labels. Consequently, a single annotator can annotate one sample from the *large set*, and the trained model can assess the label quality, streamlining the annotation process and ensuring higher-quality labeled data. There is also no need to set up a full machine learning pipeline with a larger amount of labeled data of good quality to perform a deep active learning approach.

Each image from *large set* receives labels from a single annotator, which may be a machine learning model that generates labels, a program or a human. The label quality assurance neural network can immediately determine if a label is good or not and notify the annotator to pay more attention to the sample if it is classified as a bad sample. If the label quality assurance neural network produces a false negative prediction, the annotator can confirm that they have inspected the sample carefully and their confirmation serves as a strong signal that the sample is outside the *small set* distribution used for training the model. The creation of a machine learning-based label quality assurance neural network within the proposed data requirements process is illustrated in Figure 5.

Subsequently in the development process, the *small set* is utilized by machine learning engineers and researchers as a validation set, while the *large set* serves as a training set for the primary task being addressed: in this case, the creation of an object detection model. The percentages for set cutoffs are determined based on the typical data splits used in machine learning projects. Further details are illustrated in Figure 6.

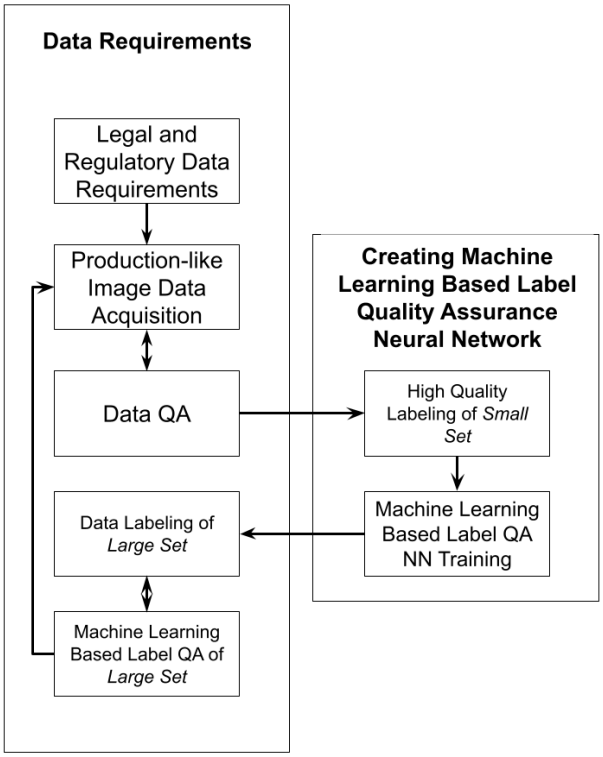



**Figure 5.** Developing a machine learning-based label quality assurance neural network. "QA" represents "Quality Assurance", and "NN" denotes "Neural Network".

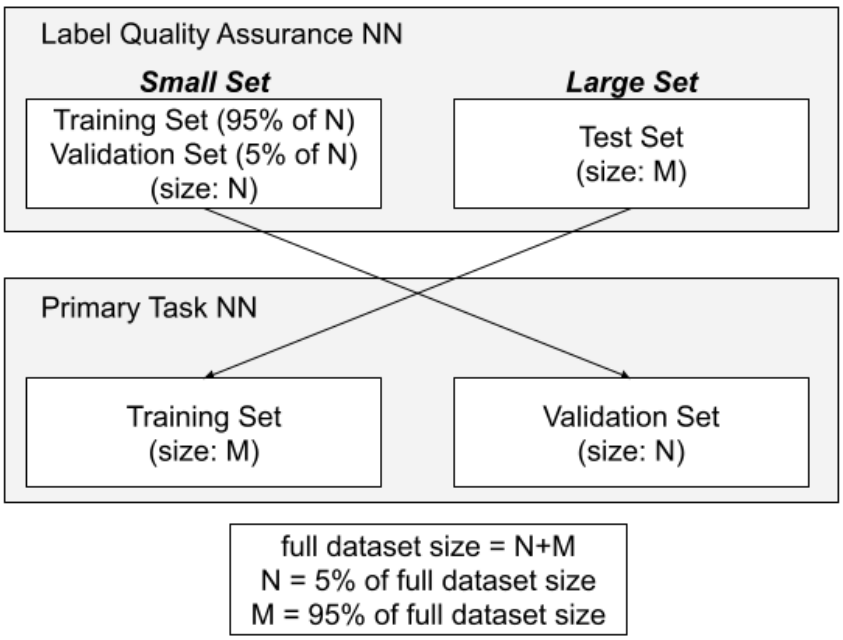



**Figure 6.** The training set used to create the label quality assurance neural network validation set is used to solve the primary task (object detection), and the test set used to create the label quality assurance neural network is the training set used to create the neural network for the primary task. "NN" stands for "Neural Network".

### 4.1. Machine Learning-Based Label Quality Assurance Method

In this section, we outline the algorithmic steps for creating a machine learning-based label quality assurance model that can be utilized to streamline the annotation process and improve label quality for various computer vision tasks.

1. Collect a dataset that accurately represents the problem to be solved. This dataset should be large enough to train a machine learning model for the primary task (e.g., object detection).
2. Split the dataset into two parts: a *small set* (approximately 5% of the full dataset) and a *large set* (approximately 95% of the full dataset).
3. Carefully annotate the *small set* with high-quality labels. This can be done by assigning multiple annotators to each sample and checking inter-annotator agreement or by assigning one or more annotators per sample, followed by reviewers checking the quality of all labels.
4. Generate large amounts of good and bad labels for every sample in the *small set* based on the ground truth label information.
5. Train a machine learning-based label quality assurance model using the generated good and bad labels from the *small set*.
6. Use the trained label quality assurance model to classify the quality of labels from the *large set*, which are created by a machine learning model that generates labels, a program or a human. Notify annotators to pay more attention to samples classified as having bad labels.
7. If there are false-negative or false-positive predictions from the model, which occur when the model incorrectly predicts the presence or absence of certain labels, have annotators confirm that they have carefully inspected the sample. Their confirmation serves as a strong signal that the sample is outside the *small set* distribution.
8. Use the *small set* as a validation set for the primary task and the *large set* as a training set.

These steps outline the process for creating a machine learning-based label quality assurance model that can be adapted and extended to various computer vision tasks and other machine learning subfields with appropriate label representation and uncertainty region ranges. By implementing this approach, researchers can enhance the efficiency and quality of the data labeling process, ultimately improving the performance of their primary task models.

### 4.2. Dataset for Machine Learning-Based Label Quality Assurance Method Verification

The dataset employed for the machine learning-based label quality assurance method experiments is Microsoft Common Objects in Context (MS COCO) [40]. MS COCO is a widely used dataset containing a diverse collection of images for various tasks, including object detection, segmentation, key-point detection and captioning. For the purpose of these experiments, we focus on the object detection task, where the goal is to identify bounding boxes surrounding objects in the images. The MS COCO dataset comprises a training set of 118,287 images and a validation set of 5000 images, each with labeled bounding boxes. Some labels carry an "iscrowd = 1" flag, indicating that large groups of objects are annotated using a single bounding box. We exclude these samples, resulting in a filtered training set of 109,172 samples and a filtered validation set of 4589 samples. The MS COCO dataset contains 80 predefined classes, with each annotated bounding box for the object detection task belonging to one of these classes based on the object it surrounds. The classes include: "airplane", "apple", "backpack", "banana", "baseball bat", "baseball glove", "bear", "bed", "bench", "bicycle", "bird", "boat", "book", "bottle", "bowl", "broccoli", "bus", "cake", "car", "carrot", "cat", "cell phone", "chair", "clock", "couch", "cow", "cup", "dining table", "dog", "donut", "elephant", "fire hydrant", "fork", "frisbee", "giraffe", "hair drier", "handbag", "horse", "hot dog", "keyboard", "kite", "knife", "laptop", "microwave", "motorcycle", "mouse", "orange", "oven", "parking meter", "person", "pizza", "potted

plant", "refrigerator", "remote", "sandwich", "scissors", "sheep", "sink", "skateboard", "skis", "snowboard", "spoon", "sports ball", "stop sign", "suitcase", "surfboard", "teddy bear", "tennis racket", "tie", "toaster", "toilet", "toothbrush", "traffic light", "train", "truck", "TV", "umbrella", "vase", "wine glass" and "zebra".

In this study, we consider the MS COCO validation set as a *small set*, which is used as the training and validation set for constructing the label quality assurance model. The MS COCO training set is treated as a test set for evaluating the label quality assurance model. Figure 7 presents a labeled sample from the label quality assurance training set (a subset of the MS COCO validation set):


**Figure 7.** A labeled sample from MS COCO belonging to the training set used for building the label quality assurance model.

### *4.3. Uncertainty Regions for Determining Good and Bad Labels*

During the training process, the machine learning-based label quality assurance neural network is exposed to both positive and negative samples. Positive samples comprise a pair of the original image and a good label, while negative samples consist of the original image and a bad label. To comprehend the distinction between good and bad labels, we first introduce the concept of uncertainty regions. These regions represent the area surrounding ground truth labels, delineating the space where all acceptable labels for a given object can reside. Figure 8 illustrates the ground truth label for a given image.

Given that $x1$ and $y1$ denote the top–left coordinates of the ground truth bounding box label and $x2$ and $y2$ represent the bottom–right coordinates, the uncertainty region for coordinate $x1$ is defined as follows:

$$[x1 - width \times 0.2, x1 + width \times 0.0], \tag{1}$$

for $y1$:

$$[y1 - height \times 0.2, y1 + height \times 0.0], \tag{2}$$

for $x2$:

$$[x2 + width \times 0.2, x2 - width \times 0.0], \tag{3}$$

and for $y2$:

$$[y2 + height \times 0.2, y2 - height \times 0.0], \tag{4}$$

where $width$ and $height$ are the width and height of the input image, and $x1$, $y1$, $x2$ and $y2$ are absolute coordinates; $x1$ and $x2$ are in the $[0, width]$ range, and $y1$ and $y2$ in the $[0, height]$ range.

Throughout the training process, positive and negative samples are randomly generated from a single pair of raw image and the ground truth label. A good label is fully enclosed within the uncertainty region, i.e., inside the outer (red) bounding box and outside the inner (green) bounding box, as illustrated in Figure 9. All bounding box coordinates must be located within their corresponding uncertainty region ranges.

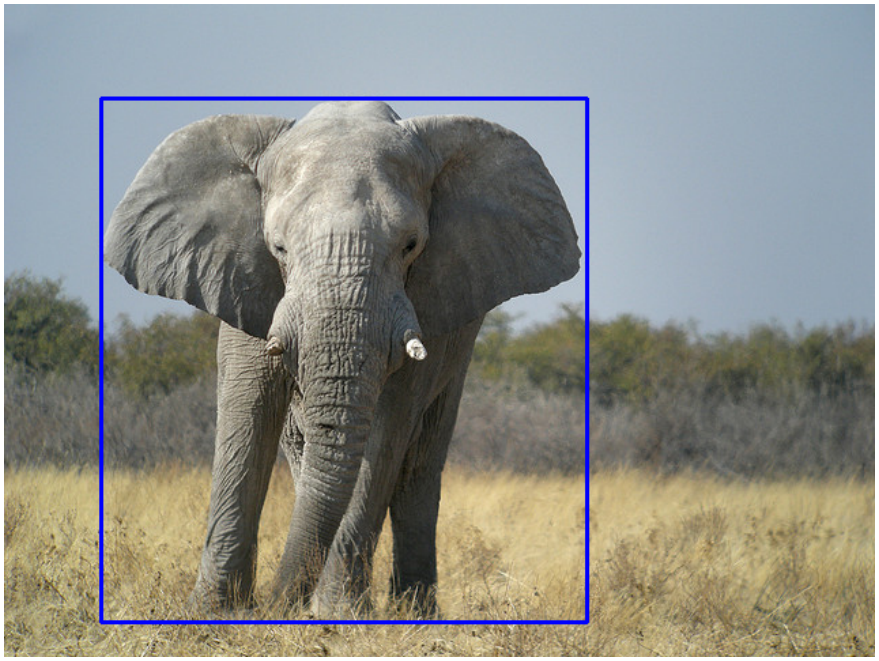


**Figure 8.** Ground truth for the object "elephant" in the image.

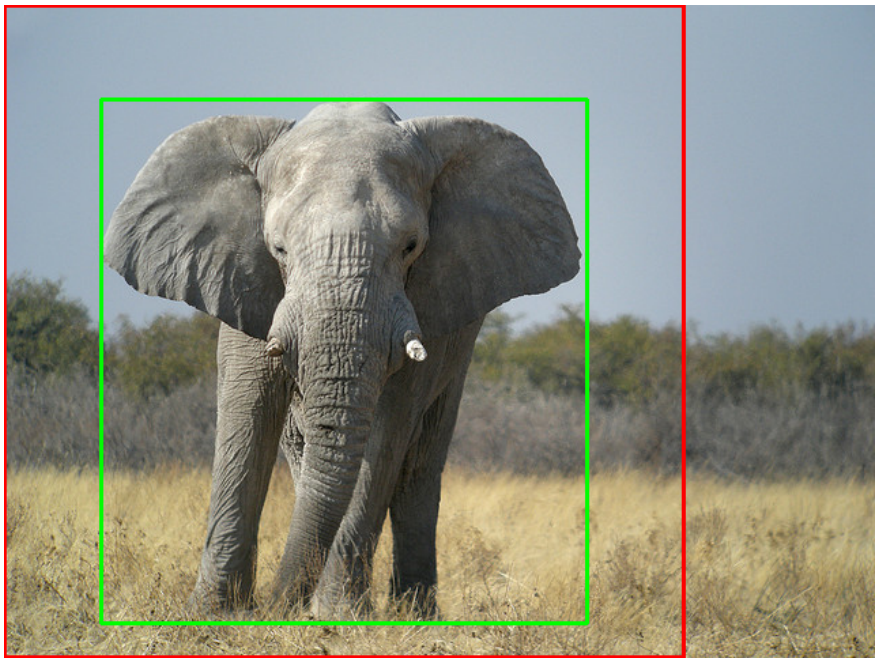


**Figure 9.** Uncertainty region for the object "elephant" in the image.

Figure 10 displays an example of a good label situated within the outer bounding box and outside the inner bounding box, as indicated in blue.

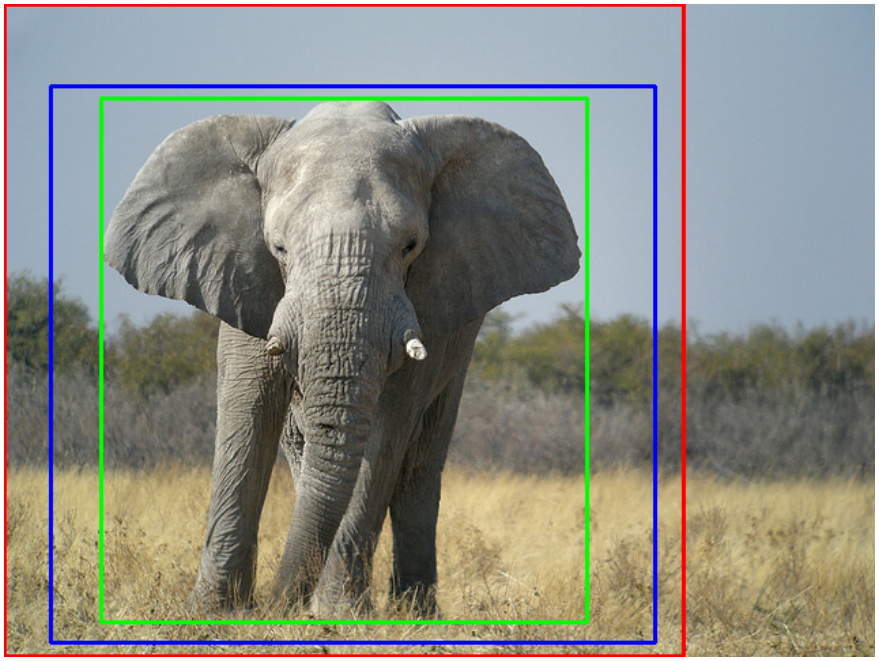


**Figure 10.** Example of a good bounding box label for object "elephant" inside the uncertainty region.

In contrast, a bad label is any label that lies partially or entirely outside the uncertainty region: either outside the red bounding box or inside the green bounding box, as shown in Figure 11. At least one of the bounding box coordinates must fall outside its corresponding uncertainty region range.

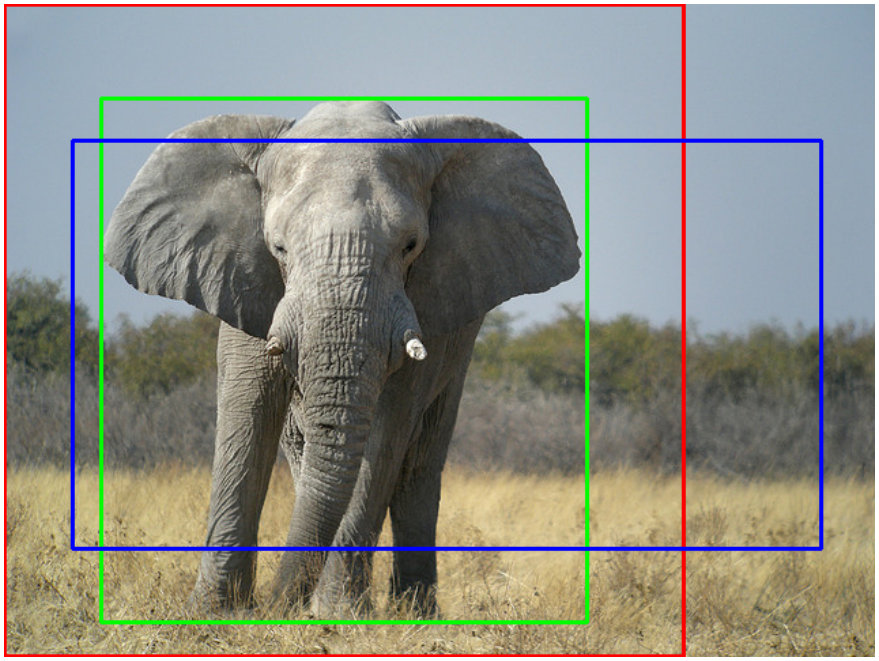


**Figure 11.** Example of a bad bounding box label for object "elephant" outside the uncertainty region.

It is crucial to note that one uncertainty region corresponds to a single bounding box label. If multiple bounding boxes are labeled and are intended to be fully enclosed within a single uncertainty region, that sample is considered a negative sample containing a bad label. Furthermore, if a sample contains multiple labeled bounding boxes and at least one of them is a bad bounding box, it is deemed a negative sample. The MS COCO dataset provides annotated bounding boxes, with each bounding box belonging to one of 80 classes.

Each uncertainty region is associated with a specific class. The space for bad bounding box labels is extensive. To manage this vastness, error types are introduced to systematically cover the most common errors annotators may make. Error types are divided into two major categories: Type A, which includes labels placed inside uncertainty regions that are damaged or manipulated; and Type B, which involves the creation of new labels outside of uncertainty regions. Table 1 displays the subtypes A1, A2, A3, B1 and B2 of the two major types: Type A (damage to labels inside uncertainty regions) and Type B (creation of new labels outside of uncertainty regions). Figure 12 presents examples of Type A errors, and Figure 13 displays examples of Type B errors. Images on the left side show ground truth labels, while the ones on the right side show labels with errors.


**Figure 12.** Error types A.

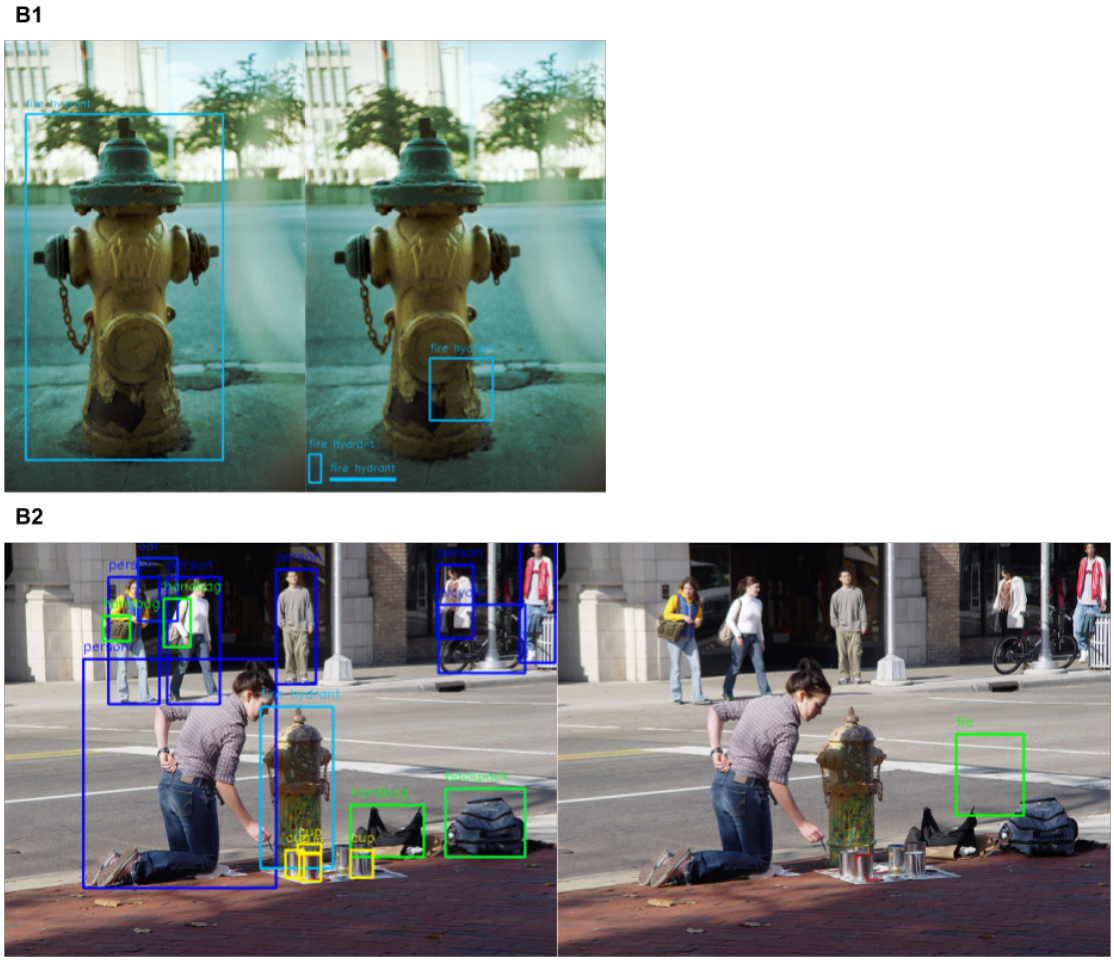



**Figure 13.** Error types B.

**Table 1.** Error subtypes with descriptions.

| | |
|---|---|
| **A1** | erase one or multiple label(s) from uncertainty regions |
| **A2** | distort label that is inside uncertainty region to become outside of uncertainty region |
| **A3** | swap classes of labels from uncertainty regions |
| **B1** | add new labels to ground truth classes, outside of any uncertainty region |
| **B2** | add new labels to classes that do not belong to any ground truth classes, outside of any uncertainty region |

### *4.4. Neural Network Architecture for Machine Learning-Based Label Quality Assurance*

The neural network is designed to accept both the image and its corresponding label, performing classification to determine if the label is good (outputting 1) or bad (outputting 0). All images are resized to 640 × 640, maintaining their aspect ratio, as the maximum width and height in the training set are both 640. Black padding is added if the width and/or height are smaller. Labels are represented as 640 × 640 images with 80 channels, where pixels representing the border of the labeled bounding box are set to 1 and all other pixels are set to 0. For the padded image in Figure 14, the label representation is shown in Figure 15 (displaying only the image planes corresponding to the classes present in the image: class "sink" and class "bottle"; the remaining 78 image planes are black).

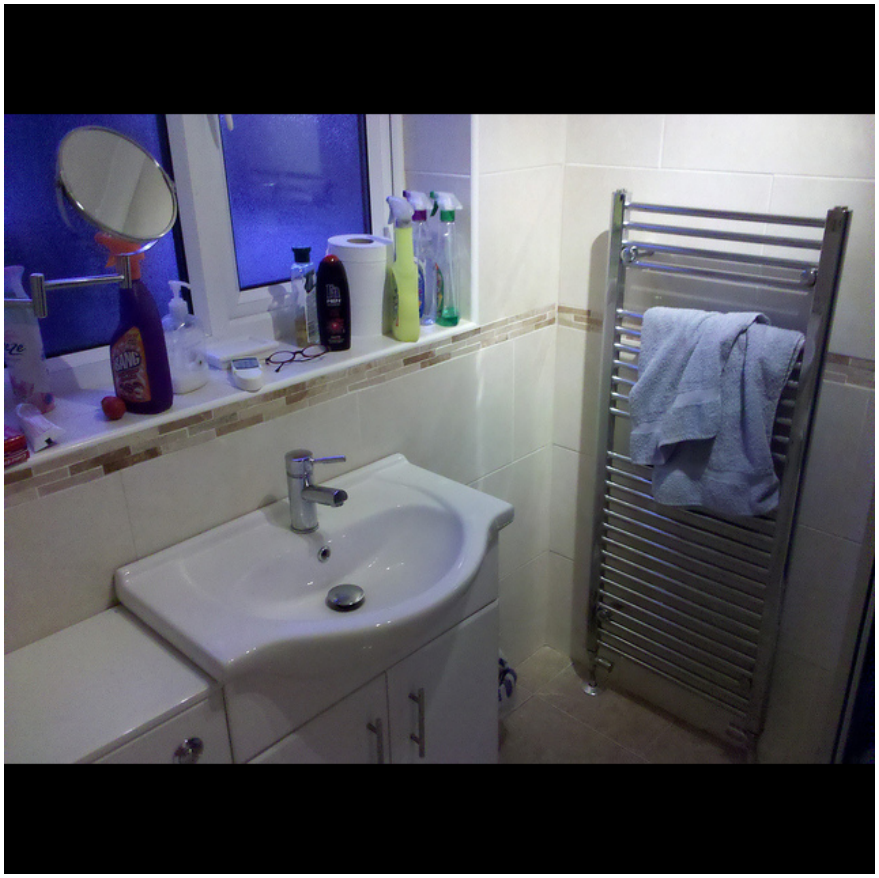


**Figure 14.** Padded image that goes to the label quality assurance neural network.

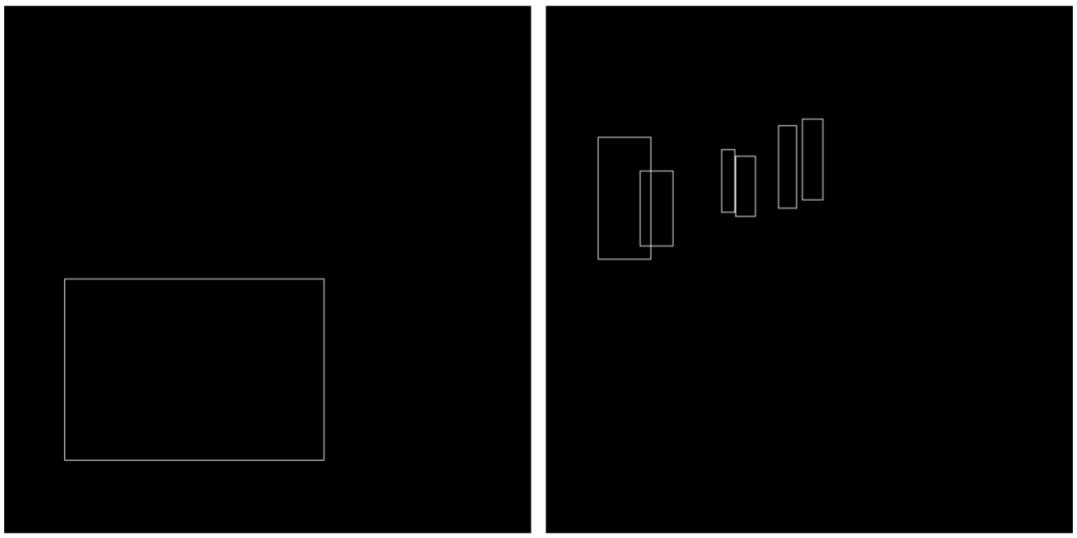


**Figure 15.** Representation of the labels for objects "sink" and "bottle" that goes to the label quality assurance neural network. Other planes are all black because there are no other MS COCO classes annotated for this particular image.

Figure 16 illustrates the neural network architecture being utilized.

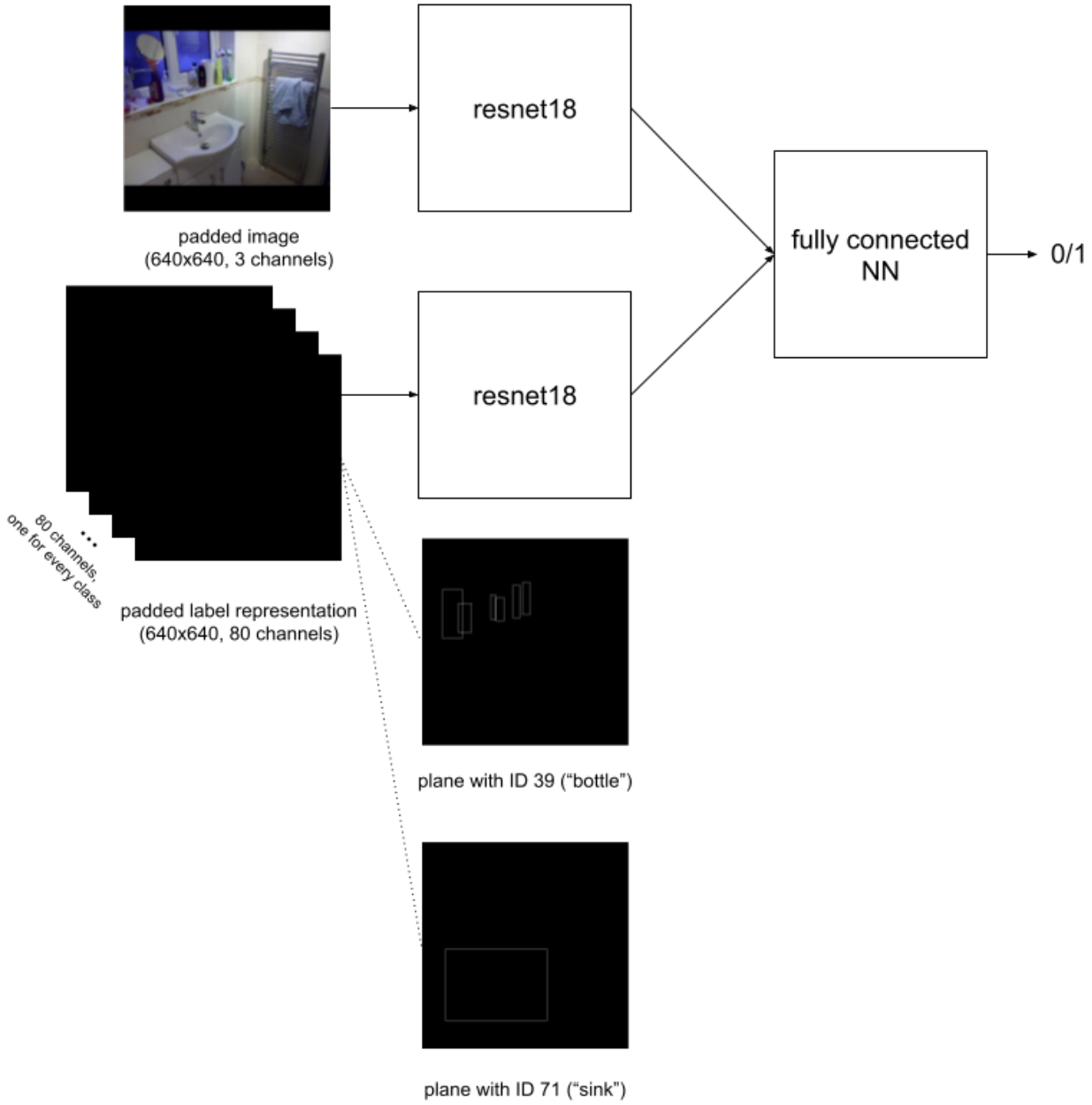



**Figure 16.** Label quality assurance neural network architecture. The ellipsis displayed on the image signifies the existence of 80 channels, which are not visually presented. "NN" stands for "Neural Network''.

In this study, a neural network is employed as the feature extractor. The images are input into the neural network in their original format, while the labels are provided using the aforementioned label representation method. It is worth noting that some studies suggest that incorporating wavelet transform-based feature extraction techniques could potentially enhance the accuracy of neural networks [41]. This is an area that warrants further exploration in future research. Additionally, there are a few studies that discuss the integration of wavelet transforms and neural networks, which presents promising avenues for improving the effectiveness of neural networks in machine learning-based label quality assurance tasks [42,43]. Furthermore, alternative traditional machine learning algorithms, such as support vector machines (SVMs), random forests and others could also be employed for this task. Feature extraction techniques, such as gray-level co-occurrence matrix (GLCM), can be utilized in conjunction with these algorithms to further improve classification performance.

## 5. Results and Discussion

We employed ResNet-18 [44] components and a fully connected neural network [45] with random initialization for our classification task. Optimization was carried out using the cross-entropy loss function and the Adam optimizer [46], featuring a learning rate of $1 \times 10^{-3}$, betas set to (0.9, 0.999), eps of $1 \times 10^{-8}$ and weight decay of 0. Training was executed on a single NVIDIA A100 GPU with 40GB of GPU memory [47] using a batch size of 64 to maximize GPU memory utilization. The model was trained for 250 epochs, with each epoch involving a full pass through all training images. Samples were dynamically determined as positive or negative with equal probabilities. The ground truth label from the MS COCO dataset was utilized to generate good or bad labels. To create a negative sample, a single error type and subtype were introduced to the ground truth label. There was a 50% probability of choosing either major Type A or major Type B for generating a negative sample. If major Type A was selected, there was a 33.3% chance that A1, A2 or A3 would be performed. If major Type B was chosen, there was a 50% chance that either B1 or B2 would be performed. These probabilities were designed to ensure that both major Types A and B as well as their subtypes had equal chances of being selected.

Table 2 displays the performance metrics for the trained model [48] on the test set. The MS COCO training set serves as the test set for the machine learning-based label quality assurance model, with good and bad labels generated for testing and each having an equal probability of occurrence. The classification report on the test set indicates that the trained model can effectively differentiate between good-labeled and bad-labeled samples. A baseline random classifier, which provides random good or bad predictions for labeled samples, would have an accuracy of approximately 50%. In comparison, our classifier achieves an accuracy of 82%, correctly identifying the quality of labels for 82% of the samples. The support metric reveals that Class 0 (bad label) contains 54,473 samples, while Class 1 (good label) consists of 54,699 samples. Since there is an equal chance of a sample being assigned a good or bad label, these numbers are closely matched. The total number of test samples is 109,172. The classification report displays class-wise precision, recall, F1-score, macro and weighted averages, along with overall accuracy, precision and recall. Metrics demonstrate the predictive power of the machine learning-based quality assurance model. Figure 17 presents the training loss, validation loss and validation accuracy plots, with the epoch number on the $x$-axis and loss or accuracy on the $y$-axis. The decreasing trend in both training and validation losses, combined with the increasing validation accuracy, indicates that the neural network is effectively learning from the data and is able to distinguish between good and bad labels.

**Table 2.** Test classification report.

| Class | Precision | Recall | F-Score | Support |
|---|---|---|---|---|
| 0 | 0.86 | 0.75 | 0.80 | 54,473 |
| 1 | 0.78 | 0.88 | 0.83 | 54,699 |
| accuracy | | | **0.82** | 109,172 |
| macro avg | 0.82 | 0.82 | 0.82 | 109,172 |
| weighted avg | 0.82 | 0.82 | 0.82 | 109,172 |
| accuracy | 0.8166 | | | |
| precision | 0.8222 | | | |
| recall | 0.8166 | | | |

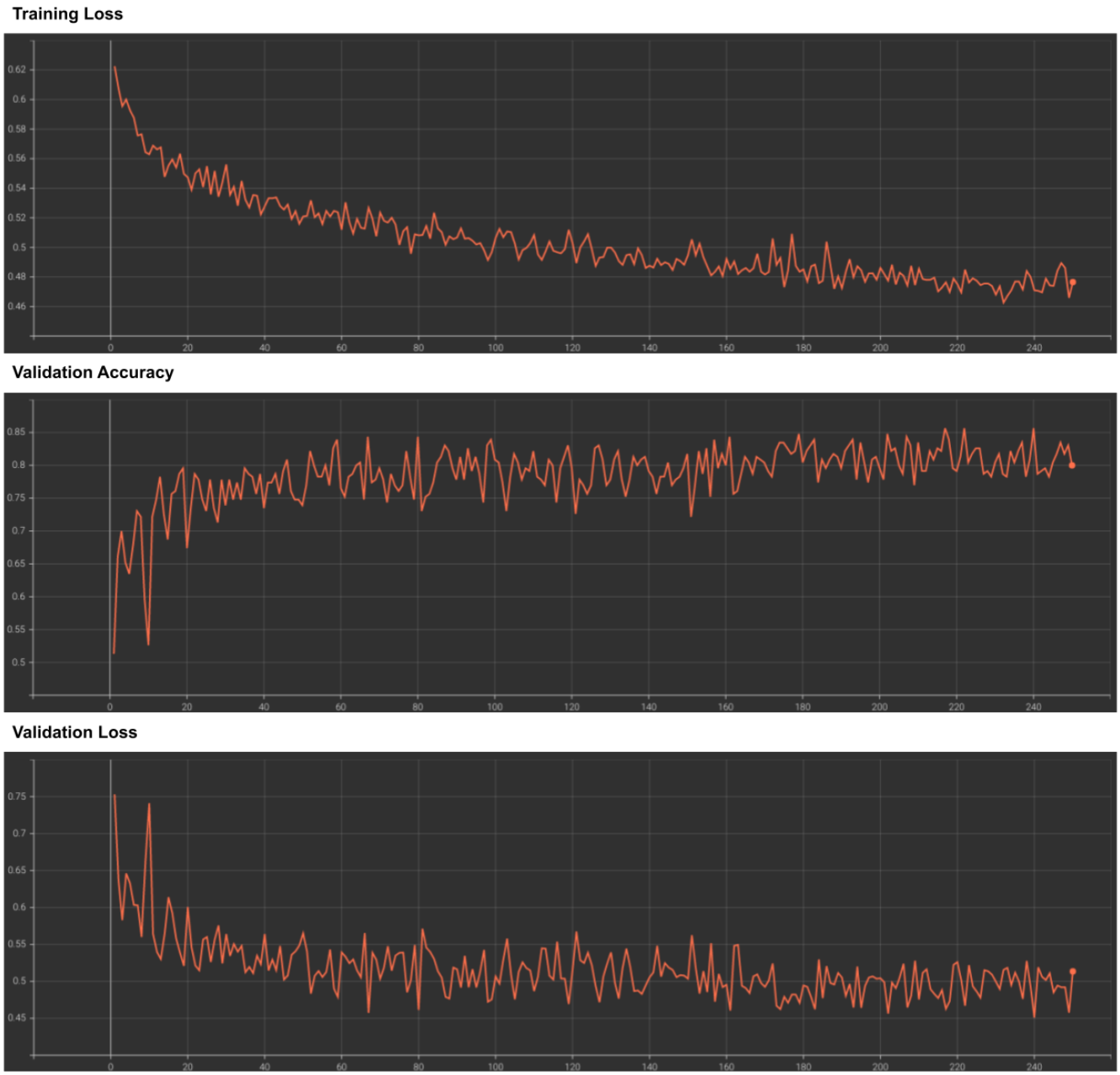



**Figure 17.** Training loss, validation accuracy and validation loss plots.

We also examined the impact of neural network size on the classification accuracy by training smaller neural networks. This was accomplished by reducing the number of layers in the fully connected part of the neural network and removing the last layers within the ResNet-18 components. The analysis of our findings is presented below in Table 3. Our results indicate that increasing the size of the neural network may lead to further improvements in classification accuracy. This suggests that there is potential for enhancing model performance by exploring larger and deeper architectures in future studies. Figure 18 presents the architecture of the large neural network employed in our study, while Figure 19 depicts the architectures of the medium and small neural networks that we also utilized. These architectural diagrams were generated using Netron [49].

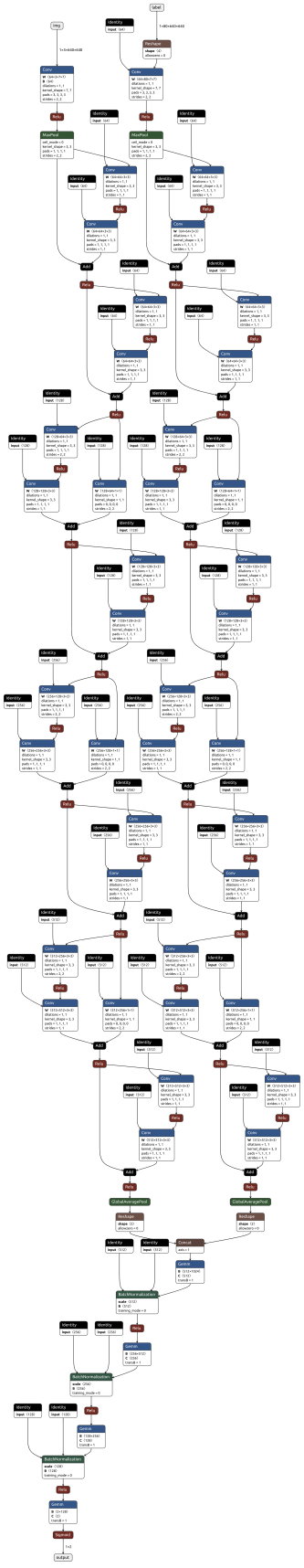
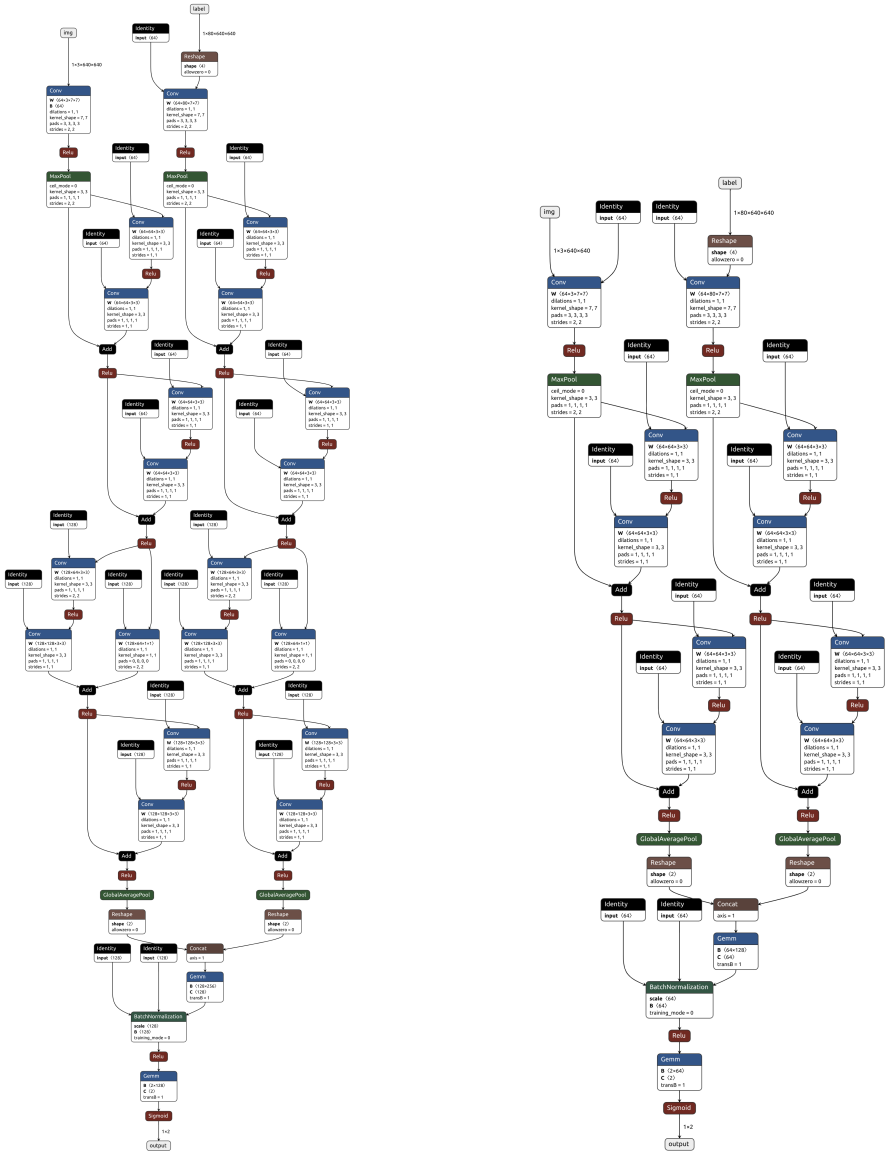


**Figure 18.** Large neural network architecture for machine learning-based label quality assurance.

**Table 3.** The relationship between the number of learnable parameters and the accuracy of a machine learning-based label quality assurance neural network.

| Size | Number of Learnable Parameters | Accuracy |
|---|---|---|
| small | 564,994 | 0.79 |
| medium | 1,641,026 | 0.80 |
| large | 23,285,570 | **0.82** |



(**a**) (**b**)

**Figure 19.** Smaller neural network architectures for machine learning-based label quality assurance. (**a**) Medium neural network architecture. (**b**) Small neural network architecture.

Figure 20 displays several examples for which the model accurately predicts that the labels are good and the corresponding labels are, in fact, of high quality.

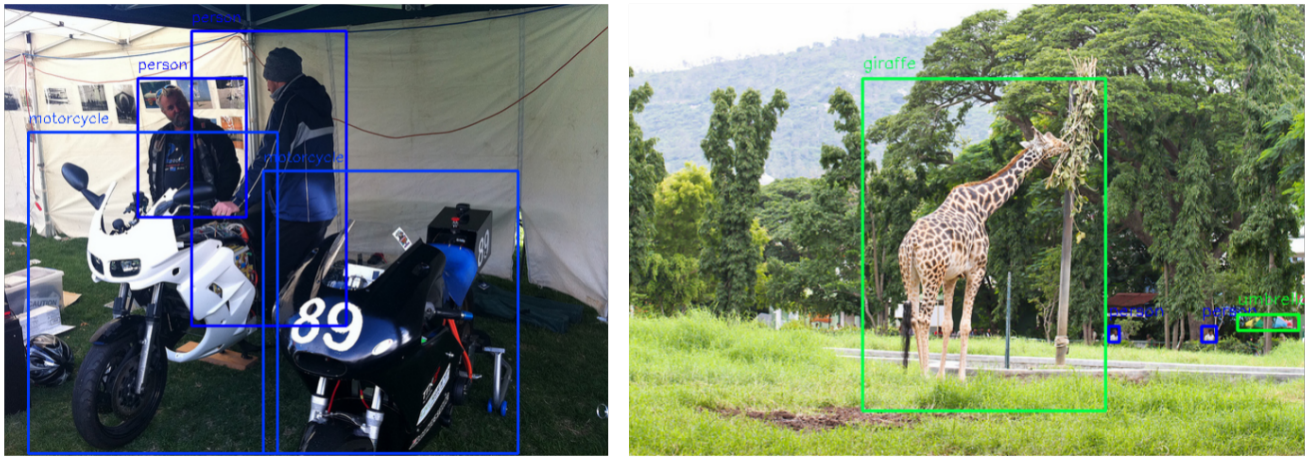



**Figure 20.** Examples of true positives.

Figure 21 presents a selection of examples for which the model correctly identifies the labels as bad and the corresponding labels are indeed of bad quality.

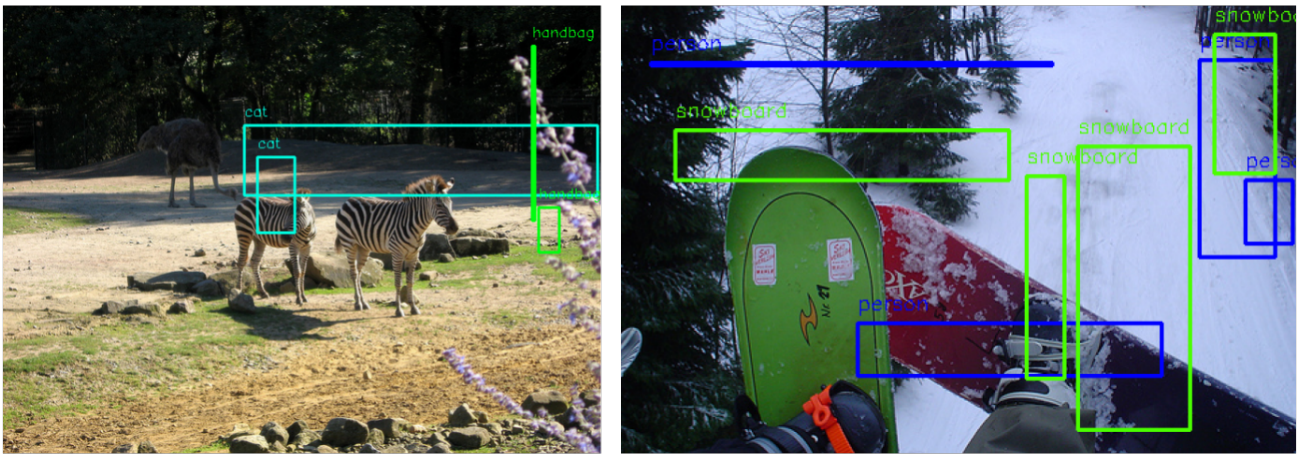



**Figure 21.** Examples of true negatives.

Figure 22 showcases two instances where the model incorrectly predicts the labels as bad when they are actually good. These samples are likely outside the distribution of the *small set* used for training the label quality assurance neural network.

Figure 23 presents two examples for which the model erroneously predicts the labels as good when they are actually bad. Similar to the false negatives mentioned earlier, false-positive predictions are likely to be outside the distribution of the *small set* used for training the label quality assurance neural network.

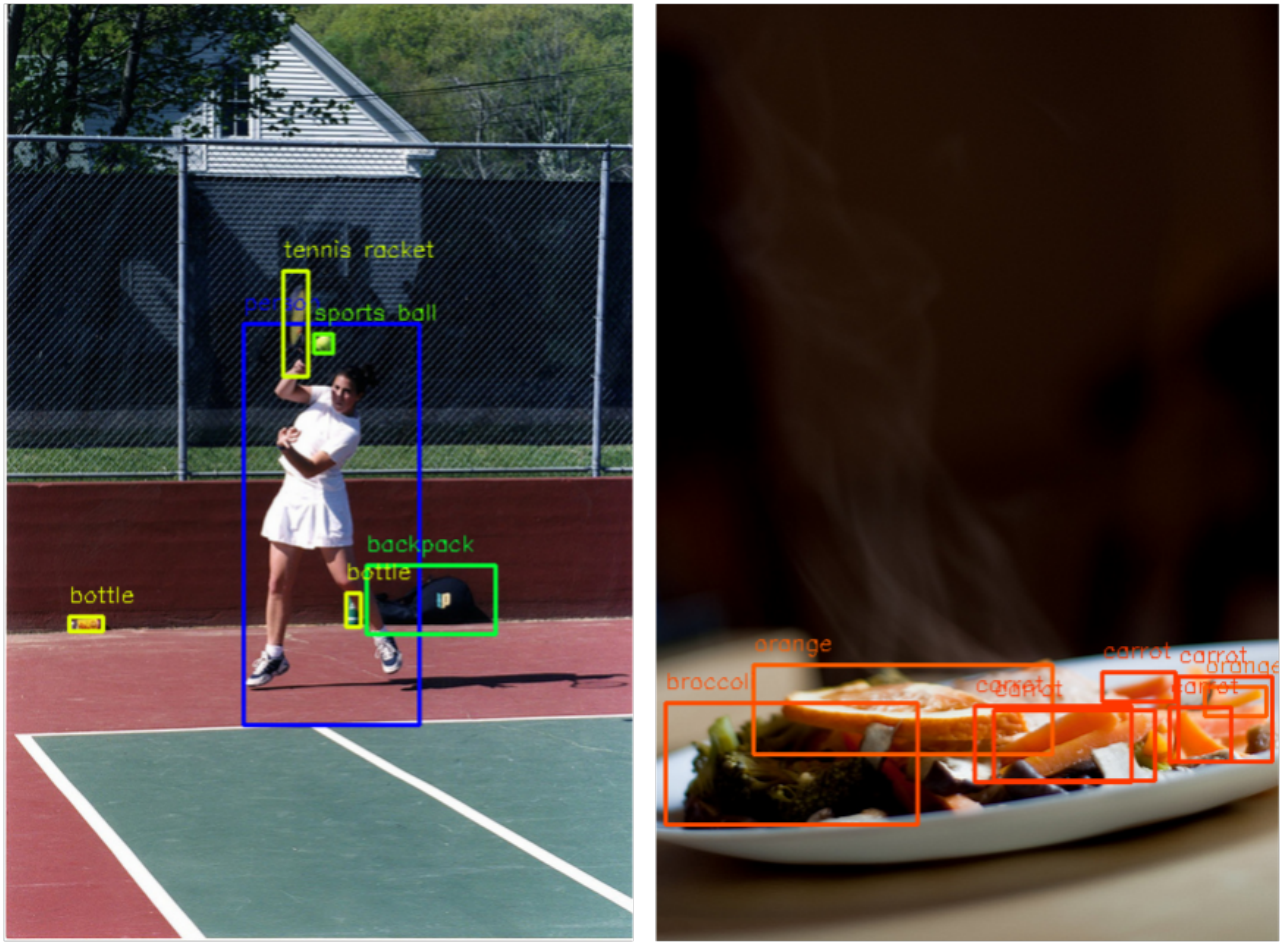



**Figure 22.** Examples of false negatives.

During training, only one of the error subtypes (A1, A2, A3, B1, or B2) is applied to the label if the sample is chosen to be negative. Further analysis involving different error types and subtypes and testing on specific combinations is conducted and presented in Table 4. The order of operations is important, and meaningful orders are shown. For instance, A1A2 means that A1 is first performed on the ground truth label, followed by A2 on the already modified ground truth label. There is an equal chance for a sample to be classified as positive or negative. If a sample is positive, labels are created within uncertainty regions, and if it is negative, a combination of error subtypes is performed on ground truth labels. The evaluation is performed using 1000 random samples from the test set 10 times for every combination. The mean and standard deviation for 10 runs per combination are shown. The label quality assurance neural network performs the worst on the A2 error subtype (distorting labels inside uncertainty regions to become outside of them), which is reasonable, as it can be challenging even for a human to determine if a bounding box with a good class is entirely inside the uncertainty region or not. The neural network best detects the B2 and B1B2 combinations (B1: adding new labels to ground truth classes outside any uncertainty region; B2: adding new labels to classes that do not belong to any ground truth classes, outside any uncertainty region). This outcome makes sense because these errors involve adding entirely new bounding boxes that do not belong to any uncertainty region.

**Table 4.** Mean ± standard deviation for accuracy per error combination.

| Error Combination | Mean ± Std |
|---|---|
| A1 | 0.71 ± 0.01 |
| A2 | 0.65 ± 0.02 |
| A3 | 0.71 ± 0.02 |
| A1A2 | 0.77 ± 0.01 |
| A1A3 | 0.78 ± 0.01 |
| A2A3 | 0.78 ± 0.01 |
| A1A2A3 | 0.83 ± 0.01 |
| B1 | 0.91 ± 0.01 |
| B2 | 0.92 ± 0.01 |
| B1B2 | 0.92 ± 0.01 |
| A1B1 | 0.77 ± 0.01 |
| A2B1 | 0.74 ± 0.01 |
| A3B1 | 0.77 ± 0.01 |
| A1A2B1 | 0.81 ± 0.01 |
| A1A3B1 | 0.83 ± 0.01 |
| A2A3B1 | 0.81 ± 0.01 |
| A1A2A3B1 | 0.85 ± 0.01 |
| A1B2 | 0.77 ± 0.02 |
| A2B2 | 0.74 ± 0.01 |
| A3B2 | 0.79 ± 0.01 |
| A1A2B2 | 0.83 ± 0.01 |
| A1A3B2 | 0.83 ± 0.01 |
| A2A3B2 | 0.83 ± 0.01 |
| A1A2A3B2 | 0.87 ± 0.01 |
| A1B1B2 | 0.82 ± 0.01 |
| A2B1B2 | 0.8 ± 0.01 |
| A3B1B2 | 0.83 ± 0.01 |
| A1A2B1B2 | 0.86 ± 0.01 |
| A1A3B1B2 | 0.86 ± 0.01 |
| A2A3B1B2 | 0.87 ± 0.01 |
| A1A2A3B1B2 | 0.88 ± 0.01 |

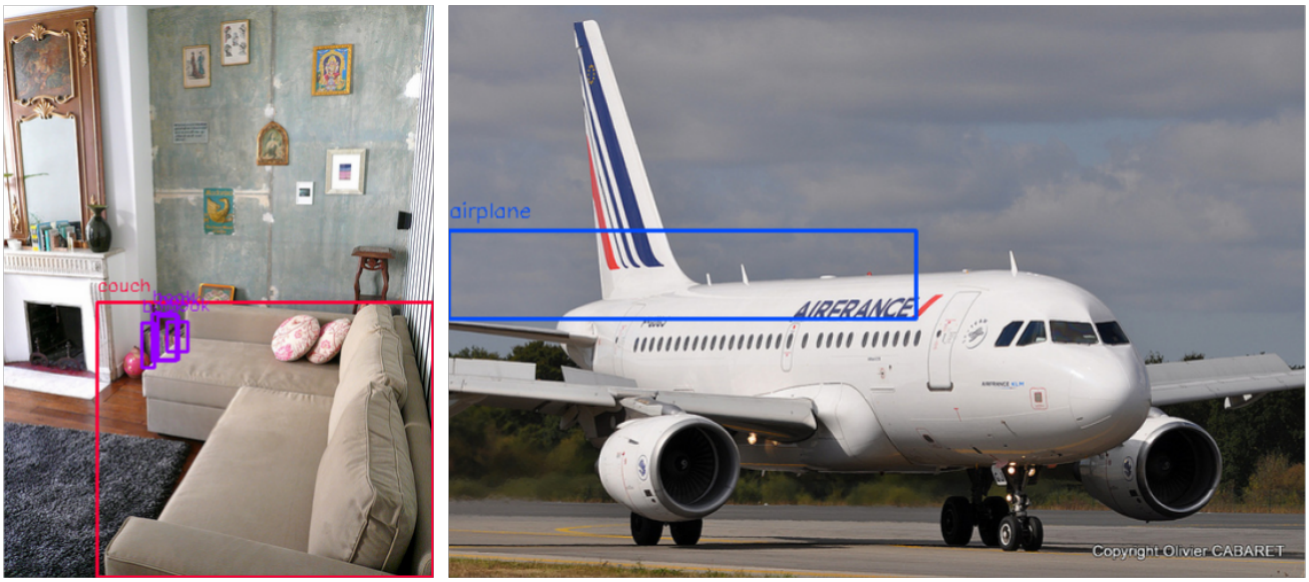



**Figure 23.** Examples of false positives.

The results indicate that training a reasonably effective neural network for label quality evaluation is achievable using a *small set*. Once trained, this neural network can be applied to a *large set*, automatically assessing whether the labels are of good quality or not. To recapitulate label quality assurance from a requirements engineering perspective, let us review Figure 4 for the "Production-like Image Data Acquisition" activity. In this activity, relevant data are collected. The subsequent activity, "Data Quality Assurance", involves manual visual inspection of a random subset for a sanity check, along with statistics about widths, heights, pixel histograms and semantic similarity analysis. The "Data Labeling" activity follows, which includes careful and high-quality labeling of a *small set* of collected data by the internal team. A rule of thumb for the size of the subset is approximately 5% of the full dataset. This work uses 4.2% of all relevant data for training the label quality assurance model. Once the label quality assurance model is complete, the remaining unlabeled data from the *large set* can be distributed among multiple annotators, and the model will immediately alert annotators if a label is suspicious.

The quantitative parameters that demonstrate the efficacy of this approach include an accuracy of 82%, precision of 82% and recall of 82%, as shown in the test classification report (Table 2). For qualitative parameters, the paper presents various examples of true positives, true negatives, false positives and false negatives (Figures 20–23). These examples help illustrate the model's ability to correctly identify good and bad labels as well as highlight cases where the model makes incorrect predictions, providing a visual assessment of the label quality assurance performance.

In summary, our findings demonstrate the effectiveness of the machine learning-based label quality assurance model by differentiating between good and bad labels with an accuracy of 82%. The model performs particularly well in detecting B2 and B1B2 error combinations, while it faces challenges in detecting the A2 error subtype. The approach we proposed enables efficient and accurate label quality assurance in object detection tasks, allowing for rapid evaluation of labels in large datasets. This method can greatly reduce the time and effort spent on manual review, ultimately improving the overall quality and reliability of the labeled data used for training computer vision models.

The proposed approach has several advantages. First, it automates the label quality assurance process, saving significant time and effort that would have been spent on manual review. Second, it allows for the rapid assessment of label quality in large datasets, which can be crucial for developing high-performance computer vision models. Finally, the approach is adaptable and can be extended to other tasks beyond detection, such as seg-

mentation, with appropriate modifications to the representation of the label and uncertainty region ranges. However, there are some limitations to our study. One limitation is that the label quality assurance model may struggle to detect certain error subtypes, such as the A2 error subtype. This can be addressed by exploring different neural network architectures or training strategies or incorporating additional features to improve the model's ability to distinguish between subtle differences in label quality. Another limitation is that our approach relies on a *small set* for training the label quality assurance model. The performance of the model may be sensitive to the quality and diversity of this *small set*, which should be carefully curated to ensure its representativeness of the entire dataset. Future research could focus on the investigation of more advanced techniques to improve the detection of challenging error subtypes, such as incorporating additional context information or improving the neural network architecture. Additionally, the approach could be extended and adapted to a wider range of computer vision tasks. By exploring future directions, our approach can pave the way for more efficient, accurate and reliable labeling processes in artificial intelligence projects, ultimately leading to the development of higher-quality and more robust machine learning models.

In this paper, we emphasize the importance of data requirements for object detection projects within the field of requirements engineering because it heavily relies on image data with labeled object bounding boxes. We introduce a process for integrating data requirements into the requirements engineering process, with a particular emphasis on label quality assurance, a critical component of the overall data requirements process for creating modern artificial intelligence solutions. Our study presents a unique machine learning-based approach to perform label quality assurance automatically, providing immediate feedback to annotators and project stakeholders about the quality of the labeling work and significantly improving the efficiency of the process. Our approach can be extended to various computer vision tasks and other machine learning subfields by adjusting label representation and setting suitable uncertainty region ranges. To the best of our knowledge, this is the first machine learning-based label quality assurance method for object detection projects, setting our work apart from previous research that relied on manual review, inter-annotator agreement and deep active learning approaches.

## 6. Conclusions

In this study, we emphasized the importance of data requirements for building effective object detection solutions within the context of requirements engineering. We proposed a process for integrating data requirements into the requirements engineering process, specifically focusing on the challenges of label quality assurance. This aspect is often sensitive, expensive and time-consuming, making it a critical component of the overall data requirements process for creating modern artificial intelligence solutions. To address these challenges, we demonstrated how machine learning can be employed to create a model that can perform label quality assurance automatically. This model provides immediate feedback to annotators and project stakeholders about the quality of the labeling work, significantly improving the efficiency of the process. Using the object detection task as an example, we showcased the effectiveness of our approach, achieving an accuracy of 82% in differentiating between good and bad labels. The proposed approach is not limited to object detection tasks and can be extended to various computer vision tasks as well as other machine learning subfields. By adjusting the label representation and setting suitable uncertainty region ranges, our method can be adapted to a wide range of applications. For instance, in the case of segmentation tasks, polygons could be used as an alternative to bounding boxes. The problem solver is responsible for determining the appropriate label representation and uncertainty region ranges based on the specific task at hand. In conclusion, our work contributes to the development of more efficient, accurate and reliable labeling processes in object detection projects, which is crucial for the creation of high-quality and robust machine learning models. By integrating our approach into the requirements engineering process, we can significantly enhance the overall performance

of artificial intelligence solutions and empower practitioners, engineers and scientists to tackle a wide array of real-world challenges. Future research should focus on addressing the limitations of our study, such as improving the detection of challenging error subtypes and ensuring the representativeness of the *small set* used as a training set. Additionally, researchers could explore the potential of our approach in a broader range of computer vision tasks and machine learning subfields, as well as investigate the synergy between automated label quality assurance and other quality control techniques. By building on the foundation established in this work, we can continue to drive the field of artificial intelligence forward.

**Author Contributions:** Conceptualization, N.P.; Methodology, N.P.; Software, N.P.; Validation, N.P.; Investigation, N.P.; Resources, N.P.; Data curation, N.P.; Writing—original draft, N.P.; Writing—review & editing, Ž.C.; Visualization, N.P.; Supervision, Ž.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Publicly available dataset was analyzed in this study. This data can be found here: https://cocodataset.org/#home.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
2. Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. Training language models to follow instructions with human feedback. *arXiv* **2022**, arXiv:2203.02155.
3. Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; Sutskever, I. Zero-shot text-to-image generation. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 8821–8831.
4. Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; Ommer, B. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 10684–10695.
5. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part V 13 ; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755.
6. Zaidi, S.S.A.; Ansari, M.S.; Aslam, A.; Kanwal, N.; Asghar, M.; Lee, B. A survey of modern deep learning based object detection models. *Digit. Signal Process.* **2022**, *126*, 103514. [CrossRef]
7. Zhu, Z.; Xie, L.; Yuille, A.L. Object recognition with and without objects. *arXiv* **2016**, arXiv:1611.06596.
8. Hafiz, A.M.; Bhat, G.M. A survey on instance segmentation: State of the art. *Int. J. Multimed. Inf. Retr.* **2020**, *9*, 171–189. [CrossRef]
9. Borque, P.; Fairley, R. *Guide to the Software Engineering Body of Knowledge Version 3.0*; IEEE Computer Society: Washington, DC, USA, 2014.
10. Vogelsang, A.; Borg, M. Requirements engineering for machine learning: Perspectives from data scientists. In Proceedings of the 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW), Jeju Island, Republic of Korea, 23–27 September 2019; pp. 245–251.
11. Pei, Z.; Liu, L.; Wang, C.; Wang, J. Requirements Engineering for Machine Learning: A Review and Reflection. In Proceedings of the 2022 IEEE 30th International Requirements Engineering Conference Workshops (REW), Virtual, 15–19 August 2022; pp. 166–175.
12. Altarturi, H.H.; Ng, K.Y.; Ninggal, M.I.H.; Nazri, A.S.A.; Ghani, A.A.A. A requirement engineering model for big data software. In Proceedings of the 2017 IEEE Conference on Big Data and Analytics (ICBDA), Kuching, Malaysia, 16–17 November 2017; pp. 111–9117.
13. Mahmood, R.; Lucas, J.; Alvarez, J.M.; Fidler, S.; Law, M. Optimizing data collection for machine learning. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 29915–29928.
14. Jain, A.; Swaminathan, G.; Favaro, P.; Yang, H.; Ravichandran, A.; Harutyunyan, H.; Achille, A.; Dabeer, O.; Schiele, B.; Swaminathan, A.; et al. A Meta-Learning Approach to Predicting Performance and Data Requirements. *arXiv* **2023**, arXiv:2303.01598.

15. Mahmood, R.; Lucas, J.; Acuna, D.; Li, D.; Philion, J.; Alvarez, J.M.; Yu, Z.; Fidler, S.; Law, M.T. How Much More Data Do I Need? Estimating Requirements for Downstream Tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 275–284.
16. Belani, H.; Vukovic, M.; Car, Ž. Requirements engineering challenges in building AI-based complex systems. In Proceedings of the 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW), Jeju Island, Republic of Korea, 23–27 September 2019; pp. 252–255.
17. Asghar, M.R.; Lee, T.; Baig, M.M.; Ullah, E.; Russello, G.; Dobbie, G. A review of privacy and consent management in healthcare: A focus on emerging data sources. In Proceedings of the 2017 IEEE 13th International Conference on e-Science (e-Science), Auckland, New Zealand, 24–27 October 2017; pp. 518–522.
18. Linden, T.; Khandelwal, R.; Harkous, H.; Fawaz, K. The privacy policy landscape after the GDPR. *Proc. Priv. Enhanc. Technol.* **2020**, *2020*, 47–64. [CrossRef]
19. Ayaz, M.; Pasha, M.F.; Alzahrani, M.Y.; Budiarto, R.; Stiawan, D. The Fast Health Interoperability Resources (FHIR) standard: Systematic literature review of implementations, applications, challenges and opportunities. *JMIR Med. Inform.* **2021**, *9*, e21929.
20. DICOM. Available online: https://www.dicomstandard.org/ (accessed on 15 May 2023).
21. Ribeiro, V.; Avila, S.; Valle, E. Handling inter-annotator agreement for automated skin lesion segmentation. *arXiv* **2019**, arXiv:1906.02415.
22. Lampert, T.A.; Stumpf, A.; Gançarski, P. An empirical study into annotator agreement, ground truth estimation, and algorithm evaluation. *IEEE Trans. Image Process.* **2016**, *25*, 2557–2572. [CrossRef] [PubMed]
23. Braylan, A.; Alonso, O.; Lease, M. Measuring Annotator Agreement Generally across Complex Structured, Multi-object, and Free-text Annotation Tasks. In Proceedings of the ACM Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 1720–1730.
24. DiPietro, D.M.; Hazari, V. DiPietro-Hazari Kappa: A Novel Metric for Assessing Labeling Quality via Annotation. *arXiv* **2022**, arXiv:2209.08243.
25. Nassar, J.; Pavon-Harr, V.; Bosch, M.; McCulloh, I. Assessing data quality of annotations with Krippendorff alpha for applications in computer vision. *arXiv* **2019**, arXiv:1912.10107.
26. Nørregaard, J.; Derczynski, L. Sparse Probability of Agreement. *arXiv* **2022**, arXiv:2208.06161.
27. Yang, F.; Zamzmi, G.; Angara, S.; Rajaraman, S.; Aquilina, A.; Xue, Z.; Jaeger, S.; Papagiannakis, E.; Antani, S.K. Assessing Inter-Annotator Agreement for Medical Image Segmentation. *IEEE Access* **2023**, *11*, 21300–21312. [CrossRef]
28. Herraiz, J.L.; Freijo, C.; Camacho, J.; Muñoz, M.; González, R.; Alonso-Roca, R.; Álvarez-Troncoso, J.; Beltrán-Romero, L.M.; Bernabeu-Wittel, M.; Blancas, R.; et al. Inter-Rater Variability in the Evaluation of Lung Ultrasound in Videos Acquired from COVID-19 Patients. *Appl. Sci.* **2023**, *13*, 1321. [CrossRef]
29. Visser, M.; Müller, D.M.J.; van Duijn, R.J.M.; Smits, M.; Verburg, N.; Hendriks, E.J.; Nabuurs, R.J.A.; Bot, J.C.J.; Eijgelaar, R.S.; Witte, M.; et al. Inter-rater agreement in glioma segmentations on longitudinal MRI. *NeuroImage Clin.* **2019**, *22*, 101727. [CrossRef]
30. Williams, B.; Hedger, N.; McNabb, C.B.; Rossetti, G.M.; Christakou, A. Inter-rater reliability of functional MRI data quality control assessments: A standardised protocol and practical guide using pyfMRIqc. *Front. Neurosci.* **2023**, *17*, 1070413. [CrossRef]
31. Takezoe, R.; Liu, X.; Mao, S.; Chen, M.T.; Feng, Z.; Zhang, S.; Wang, X. Deep Active Learning for Computer Vision: Past and Future. *arXiv* **2022**, arXiv:2211.14819.
32. Ren, P.; Xiao, Y.; Chang, X.; Huang, P.Y.; Li, Z.; Gupta, B.B.; Chen, X.; Wang, X. A survey of deep active learning. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–40. [CrossRef]
33. Wu, M.; Li, C.; Yao, Z. Deep Active Learning for Computer Vision Tasks: Methodologies, Applications, and Challenges. *Appl. Sci.* **2022**, *12*, 8103. [CrossRef]
34. Kovashka, A.; Russakovsky, O.; Li, F.-F.; Grauman, K. Crowdsourcing in computer vision. *Found. Trends® Comput. Graph. Vis.* **2016**, *10*, 177–243. [CrossRef]
35. Xiao, X.; Yang, F.; Sadovnik, A. Msdu-net: A multi-scale dilated u-net for blur detection. *Sensors* **2021**, *21*, 1873. [CrossRef] [PubMed]
36. Diaz, O.; Kushibar, K.; Osuala, R.; Linardos, A.; Garrucho, L.; Igual, L.; Radeva, P.; Prior, F.; Gkontra, P.; Lekadir, K. Data preparation for artificial intelligence in medical imaging: A comprehensive guide to open-access platforms and tools. *Phys. Med.* **2021**, *83*, 25–37. [CrossRef] [PubMed]
37. Cloud Object Storage—Amazon S3—Amazon Web Services. Available online: https://aws.amazon.com/s3/ (accessed on 15 May 2023).
38. Azure Blob Storage | Microsoft Azure. Available online: https://azure.microsoft.com/en-us/products/storage/blobs/ (accessed on 15 May 2023).
39. Cloud Storage | Google Cloud. Available online: https://cloud.google.com/storage (accessed on 15 May 2023).
40. COCO Dataset | Papers With Code. Available online: https://paperswithcode.com/dataset/coco (accessed on 15 May 2023).
41. Fortuna-Cervantes, J.M.; Ramírez-Torres, M.T.; Martínez-Carranza, J.; Murguía-Ibarra, J.S.; Mejía-Carlos, M. Object detection in aerial navigation using wavelet transform and convolutional neural networks: A first approach. *Program. Comput. Softw.* **2020**, *46*, 536–547. [CrossRef]
42. Vakharia, V.; Kiran, M.B.; Dave, N.J.; Kagathara, U. Feature extraction and classification of machined component texture images using wavelet and artificial intelligence techniques. In Proceedings of the 2017 8th International Conference on Mechanical and Aerospace Engineering (ICMAE), Prague, Czech Republic, 22–25 July 2017; pp. 140–144.

43. Alaba, S.Y.; Ball, J.E. Wcnn3d: Wavelet convolutional neural network-based 3d object detection for autonomous driving. *Sensors* **2022**, *22*, 7010. [CrossRef] [PubMed]
44. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
45. Scabini, L.F.; Bruno, O.M. Structure and performance of fully connected neural networks: Emerging complex network properties. *Phys. A Stat. Mech. Its Appl.* **2023**, *615*, 128585. [CrossRef]
46. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
47. NVIDIA A100 | NVIDIA. Available online: https://www.nvidia.com/en-us/data-center/a100/ (accessed on 15 May 2023).
48. Gösgens, M.; Zhiyanov, A.; Tikhonov, A.; Prokhorenkova, L. Good classification measures and how to find them. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 17136–17147.
49. Roeder, L. Netron. GitHub Repository. 2017. Available online: https://github.com/lutzroeder/netron (accessed on 15 May 2023).