Identifying Labeling Errors Without Access to Ground Truth

Exploring Ensemble Methods for Error Detection and Rectification

Zeryab Alam
Supervisors: Dr. Jan van Gemert, Dr. Osman S. Kayhan
EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Zeryab Alam
Final project course: CSE3000 Research Project
Thesis committee: Dr. Jan van Gemert, Dr. Osman S. Kayhan, Dr. Petr Kellnhofer

Abstract

Object detection heavily relies on accurate annotations, which are costly to obtain but crucial for model performance. Annotation errors can severely impact the reliability of detection models. In response to this challenge, we introduce ***EnsembAudit (EA)****, a novel framework designed to autonomously identify and rectify common labeling errors, thereby reducing annotation efforts. EA leverages ensemble techniques such as* ***Threshold Voting*** *for error identification and* ***Non-Maximum Suppression*** *for error rectification.*

This paper evaluates EA across various noise levels and types of labeling errors to assess its effectiveness. Our experiments demonstrate that EA excels in detecting and rectifying errors in datasets with significant noise, achieving an approximate 20% reduction in errors. However, its efficacy diminishes when applied to datasets with minimal noise. This study highlights EA's potential in enhancing annotation quality and improving the robustness of object detection applications.

1. Introduction

In the domain of computer vision, the accuracy of object detection models heavily relies on meticulously annotated datasets. These annotations, detailing object classes and their bounding box locations within images, are fundamental for training and evaluating object detection algorithms. The annotation process is laborious, time-intensive, and financially demanding, as expressed by *Deng et al.* [10] and many others [7] [42]. Consequently, there's a pressing need to streamline this process while maintaining acceptable accuracy levels.

One approach to reduce the laborious and time-intensive nature of sourcing annotations is to lower the annotation quality requirements as done by *Yixin et al.* in [45]. By relaxing stringent requirements, we can allow for margins of error in labeling, such as class misidentification or imprecise bounding box localization. However, even minor imperfections can affect model accuracy, as noted in [45] and by *Ryoo et al.* in [34].

Well-established methods to lessen the annotation burden in deep learning include Active Learning [24], Transfer Learning [41], and Point Teaching [15], all of which significantly reduce the number of annotations needed for effective model performance. However, most of these methods still require ground truth annotations, or at least a portion of the annotations to be perfectly annotated and human-verified, to ensure their effectiveness.

This raises important questions: How can we ensure that the annotations are truly ground truth? What if there are issues or errors in the annotations? Currently, we often rely on crowd-sourcing to verify annotations for datasets of varying sizes. However, this approach may not be feasible if the images are confidential. How can we verify annotations without human intervention in such cases?

Addressing these questions requires innovative methodologies for assessing and mitigating labeling errors. In this study, we propose a novel approach by combining two established algorithms: Ensemble Algorithms, which aggregate predictions from diverse models, as discussed by *Ángela Casado-García* and *Jónathan Hera* in [5], and Disagreement Algorithms that detect inconsistencies among models' predictions, akin to CrossRectify methods described in [29], which involves training two detectors with different initial parameters and using their disagreements to correct errors, leading to better performance. We refer to this combined approach as **EnsembAudit (EA)** 4. By employing disagreement algorithms to identify problematic annotations and using ensembling techniques to correct them, we aim to advance autonomous error rectification, thereby reducing labeling effort. With EnsembAudit, our goal is to evaluate annotation quality and automatically rectify labeling errors without the need for human intervention.

Through comprehensive evaluation, our goal is to assess the effectiveness of EnsembAudit (EA) across various types and levels of annotation quality. By investigating how different types and levels of annotation errors impact the decision-making process of EA, we aim to offer insights that can inform the development of more resilient and adaptable object detection systems.

2. Background and Related Work

2.1. Identifying Labeling Errors

It is well established that the performance of Deep Neural Networks [36] depend heavily on the amount of data and it's annotation quality [14] [19] [25] [28]. As such, evaluating annotation quality in datasets is crucial for optimizing the performance of object detection models. Labeling errors within datasets pose a significant challenge in machine learning, impacting model performance benchmarks extensively, as detailed in [34]. [2] illustrates how such errors influence various metrics of model disparity, further emphasizing the importance of addressing labeling errors for fair and accurate model evaluations.

Research indicates that commonly used test sets contain approximately 3-6% labeling errors [31], suggesting a potentially higher prevalence throughout entire datasets due to shared distribution characteristics. This issue is particularly critical in object detection tasks, where errors in both classification and localization can occur, significantly affecting the reliability of object detectors.

These findings underscore the prevalence of noise in datasets and emphasize its significant impact on the perfor-

mance of object detectors. Consequently, there has been considerable research interest in developing novel methods to identify and mitigate labeling errors.

One approach to diagnosing labeling errors involves the use of label scores to assess label correctness [38]. This research identifies common labeling errors in datasets and introduces algorithms that detect these errors using label scores. However, these methods often rely on manual intervention to correct images identified with low label quality scores.

Another approach involves using a two-stage object detector and analyzing the loss to identify labeling errors, as described by *Schubert et al.* in [35]. However, loss values can be noisy and influenced by various factors beyond labeling errors [8], such as model architecture, the training process, and the inherent difficulty of the samples. High loss values may arise due to reasons other than labeling errors, such as occlusions, complex backgrounds, or inherently challenging examples [22]. The loss also evolves throughout the training process. Early in training, the model might exhibit high loss values due to a general lack of learning rather than specific labeling errors. Conversely, later in training, the model might show low loss even if some labeling errors are present, as it may have learned to ignore or compensate for them.

In a different approach to tackle the problem, *Jakubik et al.* describes the UQ-LED which combine various different techniques like Confident Learning [32], Entropy Measures and Ensemble Learning [23] to detect labeling errors in [20]. They highlight the limitations of using methods such as Soft-max Probabilities [39], Bayesian Neural Networks [4] and others individually, and suggest that combining them can achieve better results by flagging a label as incorrect only if a certain threshold is met. However, determining the appropriate threshold is challenging: if set too high, many mislabeled samples might go undetected; if set too low, too many correct labels might be falsely flagged, leading to unnecessary revisions. Additionally, different techniques use varying criteria for flagging errors: Confident Learning might flag low-confidence samples, while Entropy Measures might flag high-uncertainty samples. Balancing these methods to avoid contradictions complicates the detection process, making proper calibration and alignment crucial to the ensemble's success.

Identifying labeling errors remains a challenging task in machine learning. Current approaches typically involve algorithms to detect images with labeling errors or utilizing subsets of data with verified clean labels [6] [30] during the training process. Despite these efforts, achieving accurate labeling often requires manual intervention and as such, identifying labeling errors remains a challenging task in machine learning.

2.2. Ensemble Methods

Ensembling methods have been extensively researched for their ability to improve prediction accuracy by combining outputs from multiple models [26] [18]. These methods can be particularly useful in addressing annotation errors, as they help resolve discrepancies among models, leading to more precise and refined predictions [5]. By leveraging ensemble techniques, we aim to develop a robust approach for identifying and rectifying labeling errors, ultimately enhancing the quality of datasets and the performance of machine learning models.

Wang [43] has conducted extensive research on the limitations of ensemble models, detailing individual model accuracies, diversity, and other influencing factors. Many of these findings are reproduced in this paper when showcasing the results of certain models and their ensembles.

Tan et al. [37] thoroughly explains how subtle changes, imperceptible to the human eye, can also deceive classification models, leading them to make incorrect predictions. However, ensemble methods could help alleviate this issue by combining the strengths of multiple models to improve robustness against such perturbations.

3. Methods

3.1. Introducing Noise into the Dataset

We operate under the assumption that the PASCAL [13] dataset is meticulously annotated, serving as the foundation for training and evaluating our models. However, given the objectives of our research, we deliberately introduce noise into the dataset to experimentally measure the effectiveness of our ensemble techniques.

Different studies have been conducted, identifying various types of label noise in object detection. Specifically, *Adhikari et al.* identified the following four types of noise [3]:

1. Incorrect Bounding Box
2. Incorrect Class Label
3. Additional Annotations
4. Missing Annotations

Tkachenko et al. [38], in a separate inquiry, acknowledged all the aforementioned noise types except for additional annotations. We aim to consider all of these types of noise when altering our dataset with corrupted labels.

We initiate this process by introducing classification noise into the dataset, meticulously replacing classes within specific labels with random alternatives. The algorithm governing this procedure is outlined in Algorithm 1.

Subsequently, we introduce localization noise to the clean dataset by adjusting the bounding boxes of annotations within certain labels using a random scale ranging from 0.1 to 0.5 times the size of the existing bounding box.

Algorithm 1 Generating Classification Noise

```
Randomly select a specified percentage of data points (10%, 25%, 50%).
for each selected data point do
    for each annotation in the data point's label do
        Replace with a random, different class
    end for
end for
```

This introduces diversity and variation in the noise added to the labels. The algorithm governing this process is outlined in Algorithm 2.

Algorithm 2 Generating Localization Noise

```
Randomly select a specified percentage of data points (10%, 25%, 50%).
for each selected data point do
    for each annotation in the data point's label do
        Randomly choose noise range
        Move the bounding box
        Ensure new box within image limits
    end for
end for
```

After independently addressing classification and localization noise, we merge both types and apply them anew to the clean dataset. Additionally, we introduce random removals of bounding boxes and the inclusion of ghost or extra bounding boxes. This encompasses all four types of noise previously outlined. The algorithm that manages this integrated noise addition is outlined in 3.

Algorithm 3 Generating Noise

```
Randomly select a specified percentage of data points (10%, 25%, 50%).
for each selected data point do
    for each annotation in the data point's label do
        25% chance to remove this annotation
        if annotation is not removed then
            Apply localization noise similar to 2
            Apply classification noise similar to 1
        end if
        25% chance to add a ghost annotation
    end for
end for
```

By introducing noise into what's presumed to be a clean dataset, our intent is to rigorously assess the utility of ensemble methods in detecting and potentially rectifying errors within the corrupted dataset.

3.2. EnsembAudit: Ensemble-based Framework for Labeling Error Detection and Correction

EnsembAudit 4 is the main meat and bones of this research. Our framework is designed to enhance error detection accuracy and rectify these labeling errors efficiently. It consists of two key components: an ensemble technique and a disagreement monitoring system.

The disagreement monitor evaluates predictions from different models to identify discrepancies in localization, classification, or both. Notably, if substantial deviations emerge among models trained on identical data, it flags a probable mislabeling instance for the corresponding image.

When such disparities are detected, the ensemble technique intervenes to aggregate predictions from the various models. This aggregated prediction aims to provide a more reliable estimate, considering inputs from multiple sources.

Subsequently, this fused prediction serves as a foundation for generating a revised label for the image, effectively rectifying any labeling inaccuracies.

Our ensemble framework emphasizes adaptability by supporting various ensemble techniques and offering customizable strategies for monitoring disagreement. In our experimental setup, we predominantly employ the **Fusion Ensemble Method** [5]. This method involves aggregating predictions and employing non-maximum suppression [17] to identify the most promising predictions based on confidence scores. Details of the algorithm can be found in 5.

Furthermore, our approach to monitoring disagreement centers on **Threshold Voting**. This method ensures that predictions are validated only when models above a specified threshold reach a consensus; otherwise, the instance is flagged for correction using the ensembling technique. This agreement process aids in detecting discrepancies in both classification and localization. For specifics, refer to Algorithm 6.

Furthermore, our framework is built to facilitate future enhancements. For example, alternative ensembling techniques could replace Fusion, and different disagreement monitoring methods could be seamlessly integrated in place of Threshold Voting.

Algorithm 4 EnsembAudit Algorithm

```
for each image_prediction in all_image_predictions do
    decision ← Apply Algorithm 6
    if decision then
        Apply Algorithm 5
    else
        Ignore image_prediction
    end if
end for
```

Algorithm 5 Ensemble Method: Fusion

Require: List of model predictions $image\_predictions$

$new\_preds \leftarrow$ Apply Non Max Suppression on $image\_predictions$

return new_preds

Algorithm 6 Disagreement Method: Threshold Voting

Require: List of model predictions $image\_predictions$

$prediction\_groups \leftarrow []$
for each $model\_preds$ in $image\_predictions$ **do**
  for each $pred$ in $model\_preds$ **do**
    $matched \leftarrow$ False
    for each $group$ in $prediction\_groups$ **do**
      if classes are equal and IOU above 0.5 **then**
        add $pred$ to $group$
        $matched \leftarrow$ True
        break
      end if
    end for
    if not $matched$ **then**
      add $[pred]$ to $prediction\_groups$
    end if
  end for
end for
$group\_counts \leftarrow$ [length of $group$ for $group$ in $prediction\_groups$]
$most\_sup \leftarrow \max(group\_counts, default = 0)$
if $most\_sup \geq 4$ **then return** false **else return true**

4. Experimental Setup

4.1. Dataset Preparation

The dataset used in this study includes images and annotations from the PASCAL VOC (Visual Object Classes) dataset, specifically from the 2007 and 2012 versions [13][11][12]. The combined dataset consists of 21,503 data points, with each data point comprising an image and its corresponding label annotations.

For the purpose of testing, 4,000 data points were randomly selected from the 2007 dataset's test set, which contains a total of 4,952 data points [11]. This subset serves as the unseen test set for all models in this study. The selection of 4,000 data points was necessitated by hardware limitations, as testing more than this amount would result in memory issues. The remaining 952 data points were merged with the 16,551 data points from the training sets to create a final training set consisting of 17,503 data points.

To facilitate robust model training and evaluation, a careful partitioning strategy was employed. The training dataset was distributed into five non-overlapping folds, ensuring that each data point appeared exactly once in the validation set across the folds. This approach, known as k-fold cross-validation, minimizes the risk of over-fitting by providing a comprehensive evaluation of the model's performance on different subsets of the data [16]. Training on these five folds not only ensures thorough validation but also introduces a degree of variance among the models, as each model is exposed to a slightly different training set. This variance is beneficial for ensembling, as combining the predictions from multiple diverse models can lead to improved overall performance by reducing the likelihood of simultaneous errors.

For the training-validation split, a ratio of 4:1 was maintained, with 14,002 data-points allocated for training and 3,501 for validation in each split. This allocation provides a sufficient amount of data for model training while allowing for robust validation of model performance on unseen data.

The utilization of such a comprehensive and meticulously prepared dataset, along with the rigorous cross-validation strategy, is crucial for ensuring the robustness and reliability of the trained models' performance across various tasks and scenarios.

4.2. Model Selection & Training

The YOLOv8 model was selected as the primary object detection framework due to its efficiency and effectiveness in real-time detection tasks. Unlike traditional two-stage detectors, YOLOv8 is a one-stage detector, directly predicting bounding boxes and class probabilities from a single pass through the network, leading to faster inference times [33].

The decision to utilize a one-stage detector like YOLOv8 is motivated by its ability to achieve a balance between speed and accuracy, making it suitable for applications where real-time performance is essential.

To ensure model diversity, we employ a 5-fold training strategy, as outlined in 4.1. This approach yields 5 sets of weights, each trained on a separate fold of the dataset, resulting in 5 models with diverse training data.

Training sessions were conducted using two main setups: an ASUS ROG Strix G15 Laptop and the DelftBlue supercomputer [9]. The laptop is equipped with an AMD Ryzen 7 4800H CPU with Radeon Graphics and an NVIDIA RTX 3060 Laptop GPU featuring 6GB of GDDR6 memory. Each model iteration was capped at 75 epochs, taking approximately 5-8 hours per training session on a specific dataset split detailed in 4.1, which remained consistent across all training and evaluation phases.

The DelftBlue supercomputer played a crucial role in facilitating the training process alongside the laptop. It provided additional computational power and resources, enhancing the efficiency and scale of training operations.

4.3. Model Prediction Evaluation

We use mean Average Precision (mAP) as a metric for evaluating our algorithms. Mean Average Precision is a widely adopted evaluation metric in object detection tasks. It provides a comprehensive measure of a model's precision across various levels of recall, making it an effective tool for assessing performance.

The mAP metric calculates the average precision for each class and then takes the mean of these values. Precision, in this context, is defined as the ratio of true positive detections to the total number of positive detections (both true and false positives). Recall is defined as the ratio of true positive detections to the total number of actual positives (true positives and false negatives).

To compute the Average Precision (AP) for a specific class, the precision is plotted against recall at different threshold levels, and the area under this precision-recall curve is calculated. The mean of these AP values across all classes gives us the mean Average Precision (mAP). This metric is particularly valuable because it accounts for both the precision and recall of the model, providing a balanced view of its performance.

For further details on the mAP metric and its significance in object detection, we refer to the work by *Lin et al.* [27], which provides a thorough explanation of its application and benefits.

In our evaluations, we specifically use mAP at an Intersection over Union (IoU) threshold of 0.50, commonly referred to as mAP@50. This means that a detection is considered correct if the predicted bounding box overlaps with the ground truth bounding box by at least 50%. Additionally, we incorporate confidence scores when measuring mAP to account for the confidence level of each detection, ensuring that the model's certainty is factored into the evaluation.

By using mAP@50, we ensure that our evaluation comprehensively reflects the accuracy and reliability of our models across different scenarios and object classes.

5. Experiment Results

5.1. Assessing Ensemble Model Performance

The aim of this experiment is to evaluate and compare the effectiveness of individual models versus ensemble models. We categorize the models into two groups: those trained from scratch and those fine-tuned from pretrained models using our dataset. This approach allows us to observe and analyze the performance differences between pretrained and scratch models, providing a baseline for subsequent experiments.

For the YOLOv8 model, we employ two training strategies: training from scratch and fine-tuning a pretrained model. When training from scratch, the YOLOv8 model initializes with random weights [21] and learns directly from our dataset without prior knowledge. This approach allows the model to derive features and representations directly from the data.

In contrast, fine-tuning involves taking a pretrained YOLOv8 model, originally trained on a large-scale dataset such as COCO [27], and adjusting its parameters on our dataset. Fine-tuning leverages the knowledge acquired from a diverse dataset, potentially leading to faster convergence and improved performance, especially beneficial when the target dataset is smaller or more specific.

Each approach is trained independently on the five folds of the dataset, resulting in five sets of weights for each model (i.e., YOLOv8 trained from scratch and YOLOv8 fine-tuned from a pretrained model). This setup enabled a thorough evaluation of model performance across various data subsets.

The evaluation results for these models are presented in Figure 1, illustrating the effectiveness of the ensemble models, particularly evidenced by their superior mAP@50 scores compared to individual models. Additionally, the pretrained models demonstrate notably better performance than models trained from scratch.

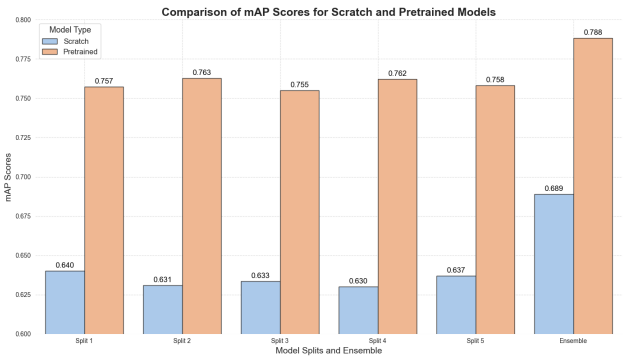

Figure 1. mAP@50 scores showing performance of individual Scratch and Pretrained models, and their ensembled versions. Both ensembled models outperform individual ones, with Pretrained models performing significantly better than Scratch.

After establishing the baseline performance, we utilized EnsembAudit (EA) 4 to evaluate the error estimation of the models. Notably, among 17,503 data-points, even though the dataset contained no actual errors, Scratch models identified 906 false positives, whereas Pretrained models identified 200 false positives. This highlights a significantly lower false positive rate in Pretrained models compared to Scratch models.

To gauge the performance of the newly generated labels against the original labels, we examined key metrics, specifically precision and recall. Both metrics showed moderate performance, ranging between 0.5 and 0.6. This indicates that while Pretrained models demonstrate better accuracy in

error detection, there remains room for improvement in labeling accuracy. Importantly, as these were false positives, the performance of these labels is not critical, since ideally, the models should have agreed on all instances without any disagreements.

For the upcoming experiments, we will continue to fine-tune pretrained models on our modified datasets. While models trained from scratch would be suitable for experiments where EnsembAudit (EA) benefits from disagreements among models, in practical applications, fine-tuning a pretrained model on the dataset tends to yield faster convergence and higher metric accuracies. Therefore, we will assess the effectiveness of using EnsembAudit on pretrained models in the next experiments rather than on models trained from scratch. It's important to note that each experiment involves resetting and using models pretrained on COCO, which means the pretrained model is effectively 'fresh' for each experiment and has not been trained on our specific dataset beforehand.

5.2. Detection of Labeling Errors with Classification Noise

This experiment aims to identify labeling errors in datasets modified with classification noise. Using Algorithm 1 to generate classification noise into the dataset, we train pretrained models on the modified dataset using the same splits as in Experiment 5.1. This enables us to accurately assess whether EA aids in classification error detection.

Testing the various noise levels against the test set reveals differences in model performance as per Figure 2. Introducing 10% and 20% noise did not significantly impair the models, despite noticeable performance drops, indicating their robustness. However, 50% noise resulted in a 10% decrease in mAP@50 compared to the ensembled version of the pretrained models. These results suggest that even when models are fine-tuned on datasets with substantial noise, they maintain robustness and achieve performance scores nearly comparable to those fine-tuned on noise-free datasets.

The results of applying EA can be observed in Table 1. The error detection rates are comparable for 10% and 25% noise levels, but the detection accuracy increases to 16% with a 50% classification noise level. Moreover, the precision and recall of the identified errors are moderately high, suggesting effective label correction. For instance, Figure 3 illustrates instances where incorrect labels have been corrected.

5.3. Detection of Labeling Errors with Localization Noise

In this experiment, we shift our focus from classification to localization, aiming to identify labeling errors in datasets affected by localization noise. We utilize Algorithm 2 to introduce localization noise into the dataset. Subsequently, we train pretrained models on this modified dataset, using the same splits as in the previous two experiments (5.1, 5.2). This approach allows us to specifically analyze the impact of localization noise and evaluate the effectiveness of EA.

Testing the models against the test set yields promising results. Despite the introduction of noise by Algorithm 2, the models exhibit robust performance, as evidenced by the minimal drops observed in mAP@50. Figure 4 illustrates that the models are resilient against minor labeling errors introduced by the algorithm, which have negligible impact on their performance.

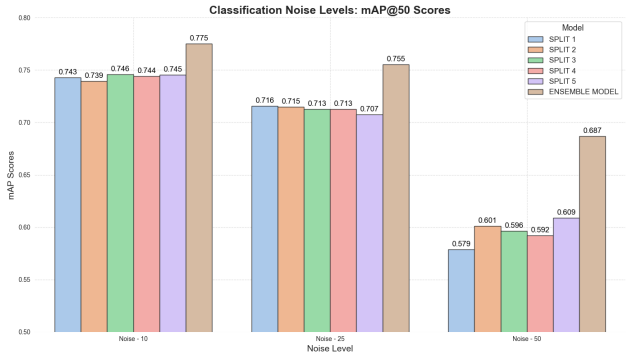

Figure 2. mAP@50 scores showing performance of individual models, and their ensembled versions against varying levels of classification noise.

Model	Actual Errors	Detected Errors	Verified Errors	Performance		
				Precision	Recall	mAP@50
10% Noise	1750	371	87	0.6346	0.7734	0.3611
25% Noise	4375	690	288	0.6753	0.7585	0.2948
50% Noise	8751	2130	1397	0.8099	0.6638	0.2818

Table 1. Classification Noise: Performance of EA across different thresholds of classification noise. EA accurately identifies 4%, 6%, and 16% of labeling errors with moderately good metric scores, suggesting effective error rectification compared to the original labels.

Figure 3. Classification Noise Rectification: Instances of images where the incorrect classes were rectified by EA. Observations can be made on EA's rectification on incorrect class assignments.

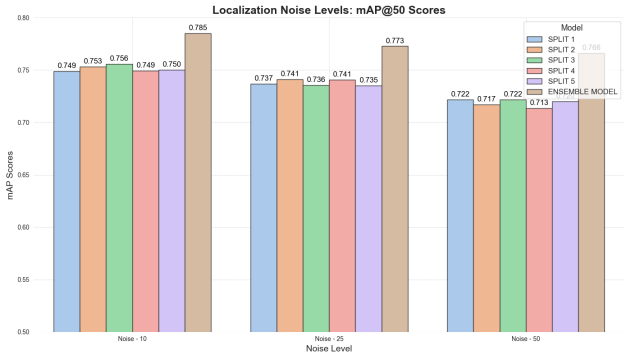

Figure 4. The mAP@50 scores depict the performance of individual models and their ensembled versions across varying levels of localization noise. Remarkably, the mAP@50 scores show no significant drop across different noise levels, indicating the models' robustness against minor localization noise.

Despite our model demonstrating robustness, as previously discussed, disagreements among the models will also be minimal, resulting in limited error detection. Nevertheless, we proceed with EA to observe its outcomes.

Table 2 summarizes the results of applying EA. As anticipated, the error detection rate remains exceedingly low, even at 50% localization noise. It seems that merely shifting bounding boxes slightly in random directions with random scales may not sufficiently challenge the model's ability to learn effectively. Nevertheless, Figure 5 showcases notable instances where such errors were detected and corrected."

Model	Actual Errors	Detected Errors	Verified Errors	Performance		
				Precision	Recall	mAP@50
10% Noise	1750	248	40	0.5536	0.6596	0.2867
25% Noise	4375	286	104	0.4889	0.5906	0.2809
50% Noise	8751	423	262	0.5316	0.6359	0.4128

Table 2. Localization Noise: Performance of EA across different thresholds of classification noise. EA accurately identifies around 2% of labeling errors across all noise levels with moderate metric scores, suggesting poor error detection and moderate error rectification

5.4. Identifying Mistakes in Dataset Modified With Both Classification and Localization Noise

In our final experiment, we integrate both classification and localization errors, using Algorithm 3, to provide a holistic evaluation of our approach in detecting and rectifying labeling errors. Again, we train pretrained models on this altered dataset using the same splits as the previous experiment (5.1, 5.2, 5.3). This comprehensive assessment allows us to examine the performance of EA in scenarios where errors occur simultaneously in both class labels and bounding box positions.

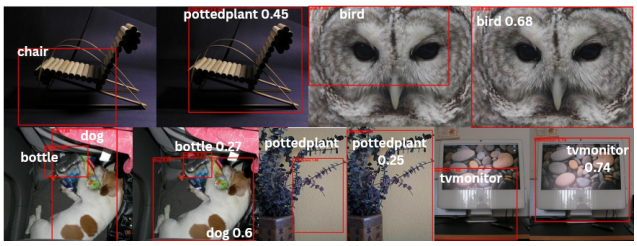

Figure 5. Localization Noise Rectification: Examples of images where EA corrected bounding box positions for various objects improving bounding box accuracy and precision compared to noisy labels.

Testing these models against the test set (Figure 6), reveals a significant decrease in performance as the noise level increases. The performance degradation is particularly pronounced at the 50% noise level, with the mAP@50 score for the ensembled model dropping to 0.67 compared to 0.77 at the 10% noise level.

Applying EA, the results are summarized in Table 3. Detection accuracy remains consistent at around 7% for noise levels 10% and 25%, but increases significantly to 20% for a noise level of 50%. This indicates that EnsembAudit (EA) performs notably better with heavily noisy datasets. Additionally, we observe a high precision score and moderately good recall score, suggesting effective error rectification in localization aspects, despite some challenges in classification aspects. Figure 7 illustrates various instances where EA successfully corrected incorrect labels, showcasing its strengths in handling datasets with substantial noise.

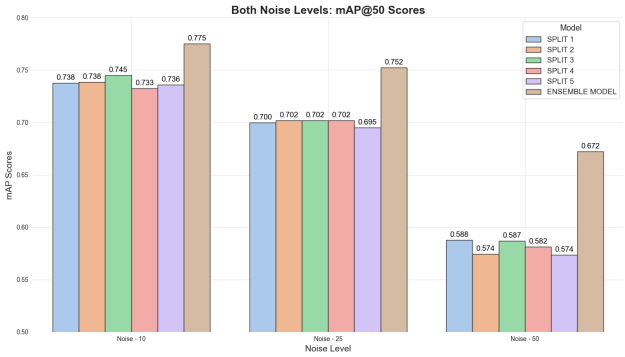

Figure 6. mAP@50 scores showing performance of individual models, and their ensembled versions against varying levels of both types of noise. Even with a variety of noise types, the models still perform on par with the no-noise pretrained models. However, a huge decrease in performance can be observed at noise level 50%.

Model	Actual Errors	Detected Errors	Verified Errors	Performance		
				Precision	Recall	mAP@50
10% Noise	1750	391	124	0.6289	0.7394	0.3015
25% Noise	4375	778	393	0.6391	0.6868	0.2422
50% Noise	8751	2600	1833	0.7731	0.5844	0.2699

Table 3. Both Noise: Performance of EA across different thresholds of classification noise. EA accurately identifies 7%, 8%, and 20% of labeling errors with moderately good metric scores, suggesting good error detection and rectification.

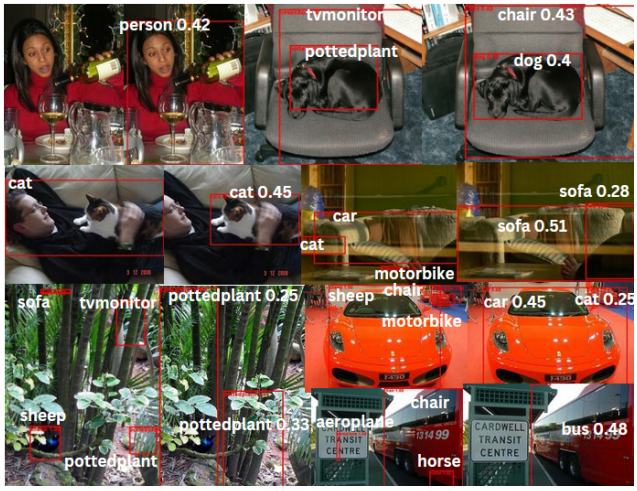

Figure 7. Noisy annotations and their rectifications: Some instances of EA correctly detecting noise and rectifying it's label with a new properly annotated one

6. Discussion and Conclusion

This thesis proposed **EnsembAudit (EA)**, a novel approach for autonomously detecting and rectifying labeling errors in weakly supervised object detection. EA leverages ensemble methods and disagreement monitoring tools to identify erroneous labels and correct them automatically. The framework of EA is versatile, capable of incorporating various ensembling techniques and disagreement monitoring tools to enhance its performance.

The experiments conducted in this study aimed to evaluate EA's effectiveness and resilience against datasets contaminated with labeling errors. The results highlighted several strengths and weaknesses of EA. One of its significant strengths is its robustness in handling datasets with a high volume and variety of label noise. EA autonomously detected and rectified approximately 20% of labeling errors in such noisy datasets, demonstrating its potential for real-world applications in improving object detection accuracy.

However, a notable limitation observed in this study is EA's reduced efficacy when dealing with datasets containing only minimal amounts of noise, which is more typical in practical scenarios. In these cases, EA exhibited limited ability to detect and flag erroneous labels due to the inherent resilience of pretrained models used in the experiments. Additionally, EA also displays a significant False Positives rate. When tuning ensembling techniques, a balance needs to be struck among the thresholds to mitigate this issue. False positives can lead to unnecessary label changes, potentially confusing the labeling process. However, upon manual review, it was observed that EA primarily flags images with inherently challenging characteristics, and errors often involve mis-classifications rather than incorrect bounding box placements.

Looking ahead, future research could focus on enhancing the robustness and efficacy of ensembling techniques and disagreement systems used in EA. There is ample opportunity to develop more sophisticated methods that can reliably detect and correct labeling errors across a broader spectrum of datasets, including those with varying quantities and qualities of noise. Additionally, comparing EA's performance against models trained from scratch would provide insights into its comparative advantages and areas for improvement.

Furthermore, validating these methods against diverse datasets and conducting additional experiments will be crucial for understanding their generalizability and effectiveness across different contexts. By addressing these challenges and exploring these avenues for improvement, EA and similar frameworks hold promise for significantly advancing the reliability and efficiency of weakly supervised object detection systems in practical applications.

7. Responsible Research

7.1. Reproducibility

This research paper adheres to the four principles outlined in the FAIR principles for scientific data [44]: Findable, Accessible, Interoperable, and Reusable.

Findable: All programming scripts, aspects, and code used in this paper are available on Github[1]. Every concept, term, library, and dataset mentioned in this paper is thoroughly established and well-referenced. All training logs, weights, scripts and results are publicly available on the GitHub repository.

Accessible: All data and materials used in this research are easily accessible to the public. The GitHub repository hosting the scripts and datasets ensures that anyone can access the resources without barriers. Detailed instructions for accessing and using the data are provided in the repository's README file to facilitate ease of use.

Interoperable: The data and code are structured to ensure compatibility with various systems and tools. The datasets were obtained through Ultralytics and follow the

[1]https://github.com/zeri27/EnsembAudit

YOLO labeling format [2]. The code is written in Python 3.12 [3], a widely-used programming language. Additionally, the main framework used is PyTorch [4] and Ultralytics [40], both of which are widely used in the machine learning industry, ensuring ease of integration with other research.

Reusable: The provided resources are designed for versatile reuse. Clear documentation, including comprehensive code comments and detailed explanations in the repository, ensures that other researchers can understand and repurpose the materials for their studies. Scripts involving aspects dependent on randomness, such as K-Fold splitting and Test Set Selection, are shared along with the random seed to ensure consistency. Additionally, any noise added to the files has been meticulously logged and tracked.

7.2. Integrity

This research paper adheres to the five principles of research integrity as defined in the Netherlands Code of Conduct for Research Integrity [1]: honesty, scrupulousness, transparency, independence, and responsibility. We have presented all aspects of this work truthfully and referenced sources properly, ensuring full transparency by discussing both the benefits and challenges of our methods. Any data not included in the paper has been logged and made publicly accessible for further analysis or interpretation.

No external influences have affected this research, and no ethical concerns have arisen in its conduct. All pre-existing code utilized in this study is available to the public.

References

[1] *Netherlands Code of Conduct for Research Integrity*. Association of Universities in the Netherlands (VSNU), 2018. Translated version of the "Nederlandse Gedragscode Wetenschappelijke Integriteit (2018)". Licensed under CC-BY 4.0. 9

[2] Julius Adebayo, Melissa Hall, Bowen Yu, and Bobbie Chern. Quantifying and mitigating the impact of label errors on model disparity metrics, 2023. 1

[3] Bishwo Adhikari, Jukka Peltomäki, Saeed Bakhshi Germi, Esa Rahtu, and Heikki Huttunen. Effect of label noise on robustness of deep neural network object detectors. In Ibrahim Habli, Mark Sujan, Simos Gerasimou, Erwin Schoitsch, and Friedemann Bitsch, editors, *Computer Safety, Reliability, and Security. SAFECOMP 2021 Workshops*, Lecture Notes in Computer Science, pages 239–250. Springer, 2021. JUFOID=62555; International Conference on Computer Safety, Reliability, and Security ; Conference date: 07-09-2021 Through 10-09-2021. 2

[4] Julyan Arbel, Konstantinos Pitas, Mariia Vladimirova, and Vincent Fortuin. A primer on bayesian neural networks: Review and debates, 2023. 2

[5] A. Casado-García and J. Heras. Ensemble methods for object detection, 2019. https://github.com/ancasag/ensembleObjectDetection. 1, 2, 3

[6] Pengfei Chen, Benben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels, 2019. 2

[7] Per H. Christensen and Wojciech Jarosz. The path to path-traced movies. *Foundations and Trends® in Computer Graphics and Vision*, 10(2):103–175, 2016. 1

[8] Lorenzo Ciampiconi, Adam Elwood, Marco Leonardi, Ashraf Mohamed, and Alessandro Rozza. A survey and taxonomy of loss functions in machine learning, 2023. 2

[9] Delft High Performance Computing Centre (DHPC). DelftBlue Supercomputer (Phase 2). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2, 2024. 4

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html. 4

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 4

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010. 2, 4

[14] Di Feng, Christian Haase-Schutz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multimodal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, March 2021. 1

[15] Yongtao Ge, Qiang Zhou, Xinlong Wang, Zhibin Wang, Hao Li, and Chunhua Shen. Point-teaching:

[2] https://docs.ultralytics.com/datasets/detect/voc/
[3] https://www.python.org/downloads/release/python-3120/
[4] https://pytorch.org/

Weakly semi-supervised object detection with point annotations, 2022. 1

[16] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, chapter 7, pages 241–249. Springer, 2 edition. 4

[17] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression, 2017. 3

[18] Hao Huang, Yongtao Wang, Zhaoyu Chen, Zhi Tang, Wenqiang Zhang, and Kai-Kuang Ma. Rpattack: Refined patch attack on general object detectors, 2021. 2

[19] Paul F. Jaeger, Simon A. A. Kohl, Sebastian Bickelhaupt, Fabian Isensee, Tristan Anselm Kuder, Heinz-Peter Schlemmer, and Klaus H. Maier-Hein. Retina u-net: Embarrassingly simple exploitation of segmentation supervision for medical object detection, 2018. 1

[20] Johannes Jakubik, Michael Vössing, Manil Maskey, Christopher Wölfle, and Gerhard Satzger. Improving label error detection and elimination with uncertainty quantification, 2024. 2

[21] Glenn Jocher. Comment on initialization and weight control in yolov8 models. https://github.com/ultralytics/ultralytics/issues/7193#issuecomment-1870558981, 2023. Accessed: 2024-06-23. 5

[22] Fehmi Kahraman, Kemal Oksuz, Sinan Kalkan, and Emre Akbas. Correlation loss: Enforcing correlation between classification and localization, 2023. 2

[23] Azal Ahmad Khan, Omkar Chaudhari, and Rohitash Chandra. A review of ensemble learning and data augmentation models for class imbalanced problems: combination, implementation and evaluation, 2023. 2

[24] Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Introducing geometry in active learning for image segmentation, 2015. 1

[25] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control, 2019. 1

[26] Siyuan Liang, Longkang Li, Yanbo Fan, Xiaojun Jia, Jingzhi Li, Baoyuan Wu, and Xiaochun Cao. A large-scale multiple-objective method for black-box attack against object detection, 2022. 2

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015. 5

[28] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, December 2017. 1

[29] Chengcheng Ma, Xingjia Pan, Qixiang Ye, Fan Tang, Weiming Dong, and Changsheng Xu. Crossrectify: Leveraging disagreement for semi-supervised object detection. *Pattern Recognition*, 137:109280, 2023. 1

[30] Kento Nishi, Yi Ding, Alex Rich, and Tobias Höllerer. Augmentation strategies for learning with noisy labels, 2021. 2

[31] Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks, 2021. 1

[32] Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. Confident learning: Estimating uncertainty in dataset labels, 2022. 2

[33] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 4

[34] Kwangrok Ryoo, Yeonsik Jo, Seungjun Lee, Mira Kim, Ahra Jo, Seung Hwan Kim, Seungryong Kim, and Soonyoung Lee. Universal noise annotation: Unveiling the impact of noisy annotation on object detection, 2023. 1

[35] Marius Schubert, Tobias Riedlinger, Karsten Kahl, Daniel Kröll, Sebastian Schoenen, Siniša Šegvić, and Matthias Rottmann. Identifying label errors in object detection datasets by loss inspection, 2023. 2

[36] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era, 2017. 1

[37] Wenyi Tan, Yang Li, Chenxing Zhao, Zhunga Liu, and Quan Pan. Doepatch: Dynamically optimized ensemble model for adversarial patches generation, 2023. 2

[38] Ulyana Tkachenko, Aditya Thyagarajan, and Jonas Mueller. Objectlab: Automated diagnosis of mislabeled images in object detection data, 2023. 2

[39] Weijie Tu, Weijian Deng, Liang Zheng, and Tom Gedeon. What does softmax probability tell us about classifiers ranking across diverse test conditions?, 2024. 2

[40] Ultralytics. *YOLOv8 Documentation*. Accessed: 2024-04-19. 9

[41] Boxiang Wang, Yunan Wu, and Chenglong Ye. The art of transfer learning: An adaptive and robust pipeline, 2023. 1

[42] Shaoru Wang, Jin Gao, Bing Li, and Weiming Hu. Narrowing the gap: Improved detector training with noisy location annotations. *IEEE Transactions on Image Processing*, 31:6369–6380, 2022. 1

[43] Wenjia Wang. Some fundamental issues in ensemble methods. In *2008 IEEE International Joint Confer-*

ence on Neural Networks (IEEE World Congress on Computational Intelligence), pages 2243–2250, 2008. 2

[44] Mark D. Wilkinson et al. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3:160018–, March 2016. 8

[45] Yixin Zhang, Shen Zhao, Hanxue Gu, and Maciej A. Mazurowski. How to efficiently annotate images for best-performing deep learning based segmentation models: An empirical study with weak and noisy annotations and segment anything model, 2023. 1