Every human makes mistakes: Exploring the sensitivity of deep-learned object detectors to human annotation noise

Laurens Michielsen

Supervisor(s): Dr. Jan van Gemert, Dr. Osman S. Kayhan

[1]**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: Laurens Michielsen
Final project course: CSE3000 Research Project
Thesis committee: Dr. Jan van Gemert, Dr. Osman S. Kayhan, Dr. Petr Kellnhofer

Abstract

The annotation effort associated with object detection is extremely costly. One option to reduce cost is to relax the demands on annotation quality, effectively allowing annotation noise. Current research primarily focuses on noise correction before or during training. However, there remains a gap in the research regarding the impact of specific types of human annotation noise on object-detector performance. This research aimed to determine how sensitive object detectors are to human annotation noise. A systematic methodology was developed to generate and quantify the effects of four noise types: missing annotations, extra annotations, inaccurate bounding boxes, and wrong classification labels. Additionally, evaluations were conducted on YOLOv8 and Faster R-CNN using the PASCAL VOC 2012, VisDrone, and Brain-Tumor datasets. The experiments demonstrated that adding noise to smaller datasets adversely affects the performance of object detectors trained on these datasets more than it does for those trained on larger datasets. Similarly, annotation noise in small objects affects detector performance more than large objects. Furthermore, YOLOv8 is resilient to low levels of missing annotations and inaccurate bounding boxes but is sensitive to all levels of incorrect classification labels. Interestingly, extra annotations seem to have a regularization effect on YOLOv8. In contrast, Faster R-CNN is generally more susceptible to annotation noise compared to YOLOv8, particularly concerning extra annotations, though both models display similar trends regarding inaccurate bounding boxes.

1 Introduction

Object detection involves identifying and locating objects within an image [1]. Recent success in this area can be attributed to the availability of large-scale, crowd-sourced datasets that are meticulously annotated with precise bounding boxes and classification labels [2–6]. Notable examples of these datasets include MS COCO [7], PASCAL VOC 2012 [8], and ImageNet [9]. Although these datasets generally feature high-quality annotations, the associated human annotation effort is extremely time-consuming and expensive [3, 5, 10–12]. Precisely drawing one bounding box on Amazon's Mechanical Turk takes 50.8 seconds on average, while the process of quality and coverage verification takes an additional 37.2 seconds on average [5].Therefore, achieving high-quality labels requires significant time and effort, resulting in substantial costs.

Innovative methods to reduce the annotation effort have been explored. One method is center click annotations [13], where annotators click the center of an imaginary bounding box around the object. These clicks are then incorporated into Multiple Instance Learning techniques for object localization. This method reduces annotation time by 9x to 18x but requires additional annotator training. Another method involves using annotators' eye movements to derive bounding boxes [14]. This method reduces the annotation time by 6.8x, but has limitations in accuracy and requires a complex setup to track eye movements. These methods showed promise in reducing annotation effort but require additional overhead.

Another option to reduce costs is to have less strict demands on the annotation quality. *Li et al.* [2] proposed a learning framework that simultaneously optimizes bounding boxes, classification labels, and hyperparameters by alternating between noise correction and model training. *Wang et al.* [3] proposed a self-correction technique based on a Bayesian filter for prediction ensemble to better leverage noisy bounding box annotations. *Chadwick and Newman* [10] proposed an improvement to co-teaching, a method where two models teach each other by discarding samples with high loss, thereby reducing the impact of annotation noise. While these approaches demonstrate promise in improving performance with noisy data, they still fall short of matching the performance of models trained on noise-free annotations.

Despite these advancements, a gap remains in analyzing the impact of specific types of human annotation noise on object detector performance. Current research [2–4, 10, 15, 16] primarily focuses on noise correction before or during training, aiming to mitigate the effects of noisy annotations. These approaches, while useful, do not provide a detailed understanding of how individual types of annotation noise affect detection accuracy. This lack of granular insights limits the ability to develop targeted strategies for improving model robustness and optimizing the annotation processes.

Various studies categorize human annotation noise differently, leading to inconsistent conclusions and a fragmented understanding of their effects. *Adhikari et al.* [15] describe the following four types of human annotation noise:

1. Missing annotations
2. Inaccurate bounding boxes
3. Extra annotations
4. Wrong classification labels

Tkachenko et al. [4] described similar types, excluding extra annotations. *Chadwick and Newman* [10] proposed an additional type: systematically swapping labels of classes with similar appearances. This study focuses on the four types of noise from *Adhikari et al.* Due to the limited time frame of the study, the fifth type from *Chadwik and Newman* was not included. While this type is significant, it somewhat overlaps with wrong classification labels. The types of human annotation noise considered in this research are visualized in Figure 1.

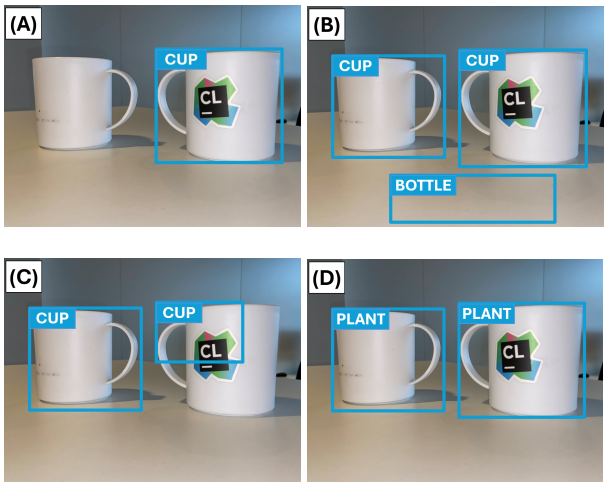

Figure 1: Four types of annotation noise considered in this work. (a) Missing annotation (cup not annotated), (b) Extra annotation (bottom annotation does not correspond to an object) (c) Inaccurately drawn bounding box, (d) Wrong classification label.

This research aims to answer how sensitive deep-learned object detectors are to the four different types of human annotation noise. The performance of the YOLOv8 [17] and Faster R-CNN [18] object detection architectures is evaluated across varying levels of artificially generated noise to understand how these affect model performance. This work has the following contributions:

1. A systematic methodology to generate and quantify the effects of four distinct types of human annotation noise.
2. Extensive evaluation of the performance of YOLOv8 [17] and Faster R-CNN [18] architectures under varying levels of annotation noise using the PASCAL VOC 2012 [8], VisDrone [19] and Brain-Tumor[1] datasets.

2 Related Works

2.1 Reducing the annotation effort

The topic of reducing the annotation effort is extensively investigated in the field of object detection. Innovative strategies such as click center annotations [13] and leveraging eye-tracking data [14] have shown promising results. However, they require additional overhead, including the training time required for annotators and the setup for tracking eye movements. This section explores two additional methodologies to reduce annotation effort: learning with label noise and weakly-supervised learning.

Learning with label noise

One option to reduce the cost is to have less strict demands on the annotation quality, effectively allowing noise in the annotations. To mitigate the effects of noisy data, numerous solutions have been proposed. Meta-Refine-Net (MRNet) adapts to noisy data by assigning lower weights to incorrect labels and generating more accurate bounding box annotations, demonstrating effectiveness in experiments on PASCAL VOC 2012 and MS COCO 2017 [16]. Another learning approach jointly optimizes object labels, bounding box coordinates, and model hyperparameters by alternating noise correction and model training [2]. ObjectLab, an algorithm that detects missing annotations, inaccurate bounding boxes, and incorrect classification labels [4], leverages a trained object detection model to score label quality, prioritizing mislabeled images for review and correction. While these methods aim to mitigate the effects of noisy annotations, they assume that annotation noise negatively impacts performance without explicitly demonstrating it. This research aims to fill this gap by thoroughly exploring and quantifying the specific effects of different types of annotation noise on detector performance.

Weakly-supervised learning

Another option to reduce cost is to train object detectors with only image-level labels, as done in weakly-supervised object detection [20]. Multiple weakly-supervised approaches have been proposed. A min-entropy latent model (MELM) reduces randomness in object locations and detector ambiguity, improving detection performance compared to state-of-the-art weakly-supervised approaches [21]. Another method is a multiple-instance learning approach that iteratively refines object locations, preventing fixation on erroneous localizations [22]. While these methods aim to exploit image-level annotations, which can be considered noisy annotations, this work aims to explore and quantify the effect of noisy annotations on the performance of object detectors.

2.2 Exploring the Effect of Noisy Annotations

Some studies not only propose their own methods but also explore how specific types of annotation noise affect object detector performance. For instance, *Wang et al.* [3] introduced noise into a subset of COCO annotations under relaxed restrictions, simulating real-world conditions. They augmented these annotations with synthesized Gaussian noise scaled to object size in bounding box coordinates. They found that as noise levels increased, there was notable performance degradation when evaluating on popular object detection architectures such as FCOS and Faster R-CNN. Additionally, they suggested a Bayesian filter-based self-correction technique to enhance model robustness against noisy data. While their work aimed to demonstrate performance degradation caused by inaccurate bounding boxes, this work evaluated how sensitive deep-learned object detectors are to all four types of human annotation noise.

Chadwik and Newman [10] investigated the impact of their five types of human annotation noise at 25% and 50% of noisy annotations introduced in the KITTI dataset using the CNN object detector. Their experimental results showed severe degradation in performance, even with limited annotation noise. Finally, they proposed an improvement to co-teaching, a method where two models teach each other by discarding samples with high loss, to mitigate the effects of annotation noise. While their study emphasized showing

[1]The dataset can be found at Ultralytics Brain-Tumor

performance decline, this study focused on exploring the effects of noisy annotations in greater depth by lowering the steps of noise introduction from 25% to 10% and validating the results across multiple architectures and datasets over multiple runs.

Adhikari et al. [15] experimented on training single-stage object detectors with different loss functions and hyperparameter settings at varying levels of inaccurate bounding box noise. Their experiments showed that the focal loss, which down-weights the loss assigned to well-classified examples to focus more on hard, misclassified examples, is impacted more than the cross-entropy loss with high amounts of noise. Finally, it was discovered that larger values of γ, the focusing parameter in focal loss, improve the robustness of the model to label noise such that extreme gamma values make the model indifferent to the noise level. This work aimed to evaluate the sensitivity of single and two-stage object detectors to varying levels of all four types of annotation noise.

3 Methodology

This research aims to evaluate how sensitive deep-learned object detectors are to four types of human annotation noise: (i) missing annotations, (ii) extra annotations, (iii) inaccurate bounding boxes, and (iv) wrong classification labels. For each noise type, the experiment assesses the performance of various object-detection architectures on datasets with different levels of artificially generated noise. Initially, the models are trained and evaluated on a noise-free dataset, corresponding to noise level 0. Subsequently, for each noise level ranging from 1 to 5, an additional 10% of the training annotations are intentionally corrupted by introducing the relevant type of annotation noise. For instance, at noise level 3, 30% of the annotations are corrupted, with 10% coming from each previous noise level, namely level 1, 2, and 3. This approach introduces noise incrementally across the different noise levels, ensuring unbiased results. At each noise level, new models are trained on the noisy data and evaluated on a noise-free test set. Finally, the results are aggregated across multiple runs. The pseudo-code outlining the experiment is presented in Algorithm 1.

The datasets used in this study are divided into train, validation, and test sets with proportions of 75:10:15, respectively. Only the train set is intentionally corrupted with annotation noise. The validation set, used during model training, remains free from noise to ensure the validation process accurately reflected the model's performance on clean data. The test set is also kept free noise-free to objectively evaluate detector performance.

The experiment is conducted using various combinations of the YOLOv8 (YOLO) [17] and Faster R-CNN [18] architectures with three distinct datasets: the PASCAL VOC 2012 (PASCAL) dataset [8], the VisDrone dataset [19] and the Brain-Tumor dataset[2]. The combination of YOLO with the PASCAL dataset is run three times to calculate confidence intervals and establish reliable results. It is important to note that the combinations of YOLO with the VisDrone and brain-tumor datasets and Faster R-CNN with the PASCAL dataset are run only once, primarily to validate the findings of the first experiment.

Algorithm 1 Experiment for each noise type

```
for i : 0 ... number_of_runs −1 do
    Train and evaluate the model on noise-free dataset
    for each noise_level in range(5) do
        Generate 10% of noise in the dataset on top of
        existing noise
        Train and evaluate the model on the corrupted
        dataset
    end for
    Aggregate the results over the runs
end for
```

3.1 Detection Architectures

Object detection architectures can be broadly classified into two categories: one-stage detectors and two-stage detectors [1]. One-stage detectors, such as YOLO [23] and SSD [24], perform detection in a single step, predicting both bounding boxes and class probabilities directly from the image in one evaluation. Two-stage detectors, like Faster R-CNN [18] and Mask R-CNN [25], first generate region proposals and then classify these proposals into different object categories.

The choice of YOLO and Faster R-CNN is motivated by their widespread use and representativeness of the one- and two-stage paradigms, respectively. YOLO is known for its speed and efficiency, while Faster R-CNN is known for its high detection accuracy [26]. By evaluating both architectures, this study aims to provide insights into how each paradigm handles different types of annotation noise.

3.2 Datasets

The PASCAL [8], VisDrone [19], and Brain Tumor[3] datasets are used. The PASCAL dataset is a benchmark in object detection, including 20 categories of everyday object in varied and complex ground-level scenes. The dataset consists of 17,125 images and 40,138 annotations and tests the model's ability to handle common detection tasks. In contrast, the VisDrone dataset comprises 7,019 images and 381,964 annotations of aerial scenes captured by drones. The dataset features objects like pedestrians and vehicles in different weather conditions and altitudes. It measures the model's ability to identify smaller and often occluded objects from high vantage points. The Brain Tumor dataset, consisting of 1,116 images and 1,166 annotations for identifying and localizing brain tumors, tests the models' adaptability to specialized domains with highly similar images. The use of three distinct datasets further enhances the robustness of the evaluation by testing the detectors in both ground-level, medical, and aerial image scenarios, which vary

[2]The dataset can be found at Ultralytics Brain-Tumor

[3]The dataset can be found at Ultralytics Brain-Tumor

significantly in terms of object size, occlusion, background complexity, and dataset size.

3.3 Noise generation

Each noise type is generated to simulate human annotation errors. The methods for noise generation are detailed below. The pseudo-code for all the noise scripts can be found in Appendix A.

Missing annotations: An annotation is deleted by randomly selecting an image and uniformly removing an annotation.

Extra annotations: The sensitivity of deep-learned object detectors is evaluated to small and big extra annotations. The small extra annotations are created by uniformly generating an x and y coordinate. The width and height are uniformly generated up to half the allowed dimensions. Big extra annotations are generated by drawing the center from a normal distribution with a mean of 0.5 and standard deviation of 0.185. The width and height are uniformly generated from half to the full allowed dimensions. For both, the centers of the new annotations are placed at a Euclidean distance of at least 0.005 from the original annotations, the minimum distance between centers in the PASCAL dataset. Finally, the classification label is uniformly assigned.

Inaccurate bounding boxes: The centers are randomly shifted by 10%, and the corresponding width and height are uniformly shifted by 10% in each direction.

Wrong classification labels: An image is randomly selected and noise is introduced by randomly selecting a classification label and uniformly replacing it with another.

4 Experimental Setup

The experiments with YOLO [17] use the YOLOv8s[4] model pretrained on MS COCO. For the PASCAL dataset, training is performed with a batch size of 64 over 100 epochs. For the VisDrone dataset, training utilizes a batch size of 32 over 150 epochs. Finally, for the brain-tumor dataset, the model is trained with a batch size of 8 for 75 epochs. All experiments are perfomed using YOLO's default hyperparameters[5].

The experiments with Faster R-CNN [18] employ a ResNet-50-FPN [27] backbone and are conducted exclusively with the PASCAL dataset. This experiment is run for 10 epochs with a batch size of 16. Stochastic Gradient Descent (SGD) is used as the optimizer, with a learning rate of 0.005, momentum of 0.9, and a weight decay of 0.0005. Early stopping is implemented with a patience of 2 epochs; if the validation loss does not improve for 2 consecutive epochs, the best model weights are restored and the learning rate is reduced by a factor of 10.

The datasets are split into training, validation, and test sets with a ratio of 75:10:15, ensuring that each set is representative of the overall data. To ensure reproducibility, the data splits are available in the repository[6].

Seeds are used for all experiments and related methods to ensure reproducibility. The seeds are calculated based on the noise level and number of runs using the formula: seed = $(i * 6) + noise\_level + 1$, where i represents the run specified in the outer loop of Algorithm 1. A noise level of 0 corresponds to training and evaluation of the model on the noise-free dataset.

5 Results

The results are discussed separately for each noise type, with each subsection following a consistent structure. Then, detection performance is measured using the mAP@50-95 metric. mAP@50-95 stands for mean Average Precision (mAP) average over Intersection-over-Union thresholds ranging from 50-95% in steps of 5%. This metric is commonly used in object detection to evaluate overall precision at different IoU overlap thresholds between predicted and ground truth boxes. A higher mAP@50-95 indicates better average precision or recall of objects across different classes and IoU thresholds.

For each noise type, a graph presents the percentage decrease in mAP@50-95 for each noise level compared to the baseline (noise level 0). For YOLO trained on PASCAL, the 95% confidence intervals are shown to account for variance, while other detector-dataset combinations report the direct percentage decreases as they are run only once. Trends in detection performance across noise levels and combinations are then analyzed.

5.1 Missing Annotations

Statistics are collected on the removed annotations for YOLO with PASCAL experiments. They reveal notable variability in the number of annotations removed per class, with the Person class having a substantially higher number of annotations removed compared to other classes. This disparity is due to the dataset containing significantly more annotations for the Person class. Interestingly, despite the higher removal rate, the Person class remains one of the more robust classes, indicating that more training data enhances model robustness against missing annotations. Additionally, the analysis shows that both small and large annotations are removed across the noise levels. The statistics demonstrate that the generated noise effectively simulates missing annotations. A detailed analysis can be found in Appendix B.1.

YOLO with PASCAL: Figure 2 illustrates a modest, linear decline from noise level 0 to 3 with mAP@50-95 decreasing from 0.541 to 0.519, followed by a slight increase at noise level 4 to 0.522 and a significant drop at noise level 5 to 0.462. Notably, the confidence intervals at noise levels 2, 3, and 4 are significantly wider compared to noise levels 0, 1, and 5. This indicates greater variability in performance when a moderate number of annotations are

[4] Available at Ultralytics website

[5] Available at Ultralytics' Train documentation

[6] Available on Github

removed. Furthermore, mAP@50, mAP@75, and mAP@95 are investigated and reveal consistent trends. Their graphs are detailed in Appendix C.1. These findings suggest that YOLO exhibits a degree of robustness to moderate levels of missing annotations in the PASCAL dataset, while highlighting increased variability in performance at these levels.

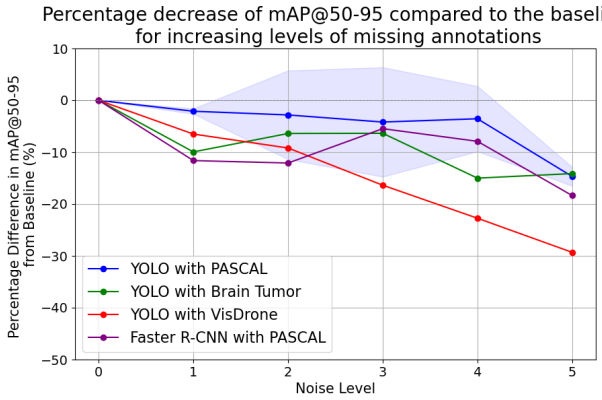

Figure 2: Percentage decrease in mAP@50-95 indicating YOLO's greater robustness to missing annotations compared to Faster R-CNN. YOLO maintains performance with greater variability at lower noise levels. Smaller missing annotations and smaller datasets have a greater negative impact on performance compared to their larger counterparts.

YOLO with VisDrone: A consistent notable linear decrease in performance across all noise levels is observed in Figure 2, with mAP@50-95 dropping 30% from 0.246 to 0.174 at noise level 5. This decline can be attributed to the dataset containing a large number of small objects, which are harder for the model to learn. This suggests that missing small annotations negatively impacts performance more than bigger annotations.

YOLO with Brain-Tumor: Interestingly, Figure 2 shows a significant initial performance drop of 10%, followed by improved and stable performance at noise levels 2 and 3, which align well with the confidence intervals of YOLO with PASCAL. Noise level 4 displays a significant drop of an additional 9%, followed by a slight performance increase. This fluctuation is likely due to the dataset's smaller size, indicating that smaller datasets are more sensitive to missing annotations, as evident from the initial drop at noise level 1. However, the stable performance at noise levels 2 and 3 underscores that YOLO exhibits a partial degree of robustness to moderate levels of missing annotations.

Faster R-CNN with PASCAL: Similar to YOLO with Brain-Tumor, Figure 2 shows an initial performance drop of 12%. Noise level 2 performs similarly to noise level 1. Remarkably, levels 3 and 4 align within YOLO with PASCAL's confidence intervals, while level 5 shows slightly worse performance for Faster R-CNN. These findings suggest Faster R-CNN is more sensitive to lower levels of missing annotations than YOLO. For moderate and higher levels, their sensitivity appears comparable.

In conclusion, smaller missing annotations have a greater negative impact on performance compared to larger ones. Surprisingly, YOLO shows more robustness to missing annotations at lower noise levels than Faster R-CNN. YOLO is moderately affected by missing annotations, with accuracy declining slightly and variability increasing at moderate levels of missing data, yet exhibiting a pronounced decrease in performance at high levels of missing annotations. Finally, smaller datasets affect performance more than larger datasets.

5.2 Extra Annotations

Small Extra Annotations

For the YOLO and PASCAL experiments, statistics are computed for the small, extra annotations. The statistics show that the extra annotations are evenly distributed across the classes. Additionally, the statistics reveal that the generated extra annotations cover small areas and do not match real objects. In conclusion, the statistics show that the noise generation process effectively simulates small extra annotations. A detailed analysis can be found in Appendix B.2.

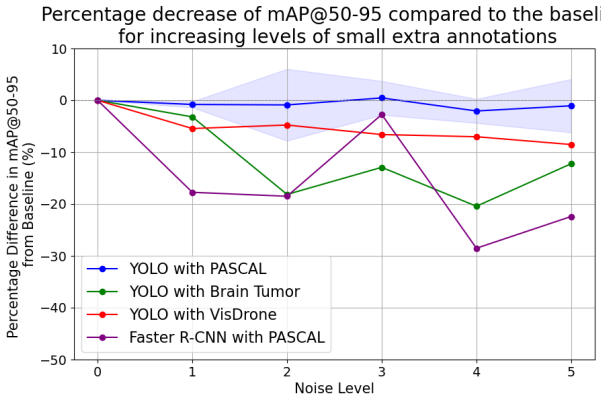

Figure 3: Percentage decrease in mAP@50-95 compared to the baseline indicating that small extra annotations have a regularizing effect for YOLO. Faster R-CNN is sensitive to small extra annotations. Smaller datasets and a higher number of small annotations lead to more pronounced performance declines.

YOLO with PASCAL: Surprisingly, Figure 3 shows consistent performance across the noise levels, with narrower confidence intervals observed at noise levels 0 and 1 compared to the remaining levels. The average mAP@50-95 fluctuates from 0.534 to 0.545. The results suggest that small extra annotations have a regularizing effect. Further analysis of mAP@50, mAP@75, and mAP@95 revealed similar trends in mAP@50 and mAP@75 across noise levels. However, mAP@95 exhibits a significant initial drop, from 0.14 to 0.12, after which performance remains relatively stable across the remaining noise levels. Graphs for mAP@50, mAP@75, and mAP@95 are available in Appendix C.2. In summary, extra annotations did not notably affect mAP overall but did reduce precision at the highest IOU threshold, suggesting regularization with potential cost to localization accuracy.

YOLO with VisDrone: Remarkably, Figure 3 demonstrates an initial performance drop of 5%, with performance remaining relatively stable across subsequent noise levels. The initial drop can be attributed to the dataset containing many small objects, which became harder to learn after the addition of small extra annotations. The subsequent stability further emphasizes that small extra annotations act as regularization terms.

YOLO with Brain-Tumor: Figure 3 shows an unexpected trend compared to YOLO with PASCAL: it displays an initial performance drop followed by a more pronounced decline to 19%. The performance remains relatively stable across the subsequent noise levels. These initial drops can be attributed to the small size of the dataset, which makes it more sensitive to small extra annotations. Additionally, the consistent performance from noise levels 2 to 5 further emphasizes that small extra annotations act as regularization terms.

Faster R-CNN with PASCAL: Notably, Figure 3 reveals a substantial initial drop of 18% followed by inconsistent behavior across the remaining noise levels. These results suggest that YOLO is more robust to small, extra annotations than Faster R-CNN.

In summary, small extra annotations have a regularizing effect on YOLO. Furthermore, smaller datasets show a more pronounced performance decline with small extra annotations. Additionally, their effect is more noticeable when datasets contain a substantial number of smaller annotations. Lastly, Faster R-CNN appears more sensitive to small additional annotations compared to YOLO.

Big Extra Annotations

Statistics are computed for big extra annotations, for the YOLO and PASCAL dataset experiments. They reveal that the generated annotations are evenly distributed across the classes. Additionally, they show that the extra annotations cover large areas and match real objects to some extent, which is expected for big extra annotations. In conclusion, the statistics show that the noise generation process effectively simulates big extra annotations. A detailed analysis can be found in Appendix B.2.

YOLO with PASCAL: Interestingly, Figure 4 shows that the performance across noise levels mirrors the pattern observed with small extra annotations. The only notable difference is a performance drop at noise level 1 with big extra annotations. The mAP@50-95 fluctuates from 0.557 to 0.532. Further analysis attributes this drop to baseline performance disparities: small extra annotations averaged 0.54 mAP@50-95, while big extra annotations achieved 0.55. Analysis of mAP@50, mAP@75, and mAP@95 yield similar trends as observed with small extra annotations. The graphs of mAP@50, mAP@75, and mAP@95 are detailed in Appendix C.2. In summary, big extra annotations seem to have a similar effect as small extra annotations.

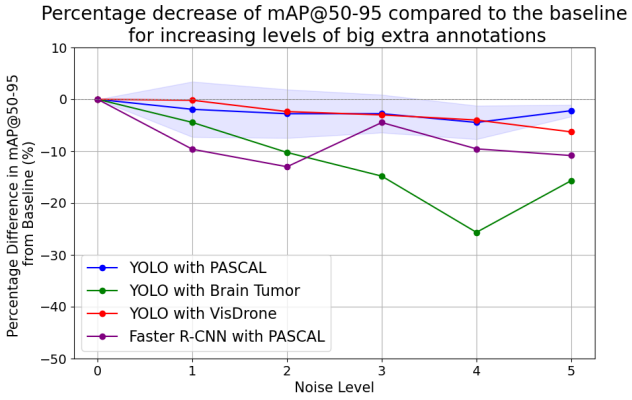

Figure 4: Percentage decrease in mAP@50-95 compared to the baseline indicating that big extra annotations have a regularizing effect for YOLO. Faster R-CNN is sensitive to big extra annotations. Smaller dataset with big extra annotations lead to more pronounced performance declines compared to big datasets.

YOLO with VisDrone: Remarkably, Figure 4 presents a performance evolution nearly identical to YOLO with PASCAL. The only deviation is a performance drop observed for VisDrone at noise level 5, likely caused by VisDrone having significantly more extra annotations than PASCAL. These results suggest that YOLO with VisDrone is more resilient to big extra annotations compared to small ones. These findings emphasize that big extra annotations have a regularization effect and highlight the greater resilience of larger datasets to big extra annotations.

YOLO with Brain Tumor: Figure 4 shows a significant, linear decrease in performance up until noise level 4, where performance dropped a remarkable 25% compared to the baseline, followed by a notable increase of 7% at noise level 5. The consistent initial decline highlights the impact of big extra annotations on a smaller datasets.

Faster R-CNN with PASCAL: Notably, Figure 4 shows a similar, but less pronounced trend as in Figure 3. In Figure 4, the first two noise levels exhibit significantly worse performance compared to YOLO with PASCAL. This is followed by a recovery to nearly identical performance at noise level 3, which then declines again in subsequent noise levels.

In conclusion, extra annotations have a regularizing effect on YOLO. Moreover, smaller datasets seem to be more sensitive to extra annotations. Finally, Faster R-CNN seems to be less robust to extra annotations than YOLO.

5.3 Inaccurate Bounding Boxes

For the YOLO and PASCAL experiments, statistics are computed for inaccurate bounding boxes. The statistics reveal significant overlap with the original bounding boxes while still maintaining substantial difference such that inaccurate bounding boxes are effectively simulated.

Additionally, the average number of inaccurate bounding boxes per class is computed. As expected, the Person class is most affected, which can be attributed to the characteristics of the PASCAL dataset. The statistics reveal that the generated noise effectively simulates inaccurate bounding boxes. A detailed analysis can be found in Appendix B.3.

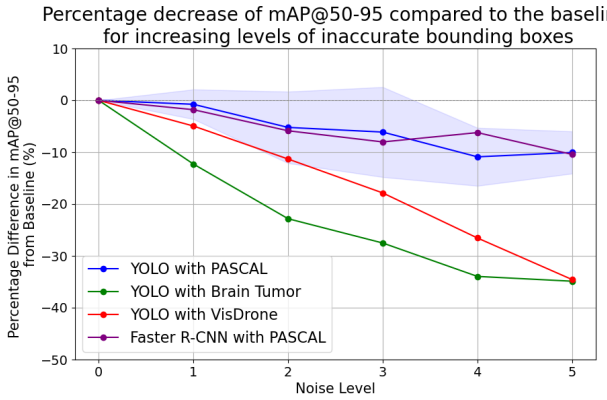

Figure 5: Percentage decrease in mAP@50-95 compared to the baseline showing that YOLO and Faster R-CNN show equal robustness to inaccurate bounding box noise, with moderate sensitivity at low noise levels and severe sensitivity at high noise levels. Inaccurate bounding boxes for small objects and smaller datasets harm performance more significantly.

YOLO with PASCAL: Figure 5 shows a relatively consistent, linear decrease in performance from noise levels 0 to 4, where performance dropped 10% compared to the baseline. The confidence intervals at levels 2 to 5 are significantly larger than at levels 0 and 1, indicating greater variability as more boxes are inaccurate. These findings highlight YOLO's sensitivity to inaccurate bounding boxes. Additional analysis of mAP@50, mAP@75, and mAP@95 revealed similar trends, except for mAP@95. mAP@95 decreased sharply and linearly from 0.14 at noise level 0 to 0.025 at noise level 5, indicating a severe impact on precise detection. Plots of mAP@50, mAP@75, and mAP@95 can be found in Appendix C.3.

YOLO with VisDrone: Notably, Figure 5 shows a significant linear decrease over the noise levels, with a 35% performance loss at noise level 5, and performs much worse than YOLO and Faster R-CNN with PASCAL. These findings suggest that inaccurate bounding boxes of small objects negatively impact performance more than big objects since the VisDrone dataset contains significantly more small objects than PASCAL.

YOLO with Brain-Tumor: Interestingly, Figure 5 shows a significant linear decrease in performance from noise level 0 to 4, with a 35% performance decrease at noise level 4, after which performance stabilizes at noise level 5. The initial steady decline highlights the impact of inaccurate bounding boxes on a smaller dataset, which lacks the robustness seen in larger datasets.

Faster R-CNN with PASCAL: Unexpectedly, Figure 5 shows almost identical performance to YOLO with PASCAL. This suggests that both YOLO and Faster R-CNN are somewhat robust to inaccurate bounding boxes, especially at lower noise levels.

In conclusion, YOLO and Faster R-CNN seem to be equally robust to inaccurate bounding box noise. They demonstrate moderate sensitivity to inaccurate bounding boxes at low noise levels but demonstrate severe sensitivity at higher noise levels, highlighted by the wider confidence intervals of YOLO. Furthermore, inaccurate bounding boxes for small objects seem to harm performance more than big objects. Finally, smaller datasets are more sensitive to inaccurate bounding boxes than larger datasets.

5.4 Random Wrong Classification Labels

For the YOLO and PASCAL experiments, statistics are computed. They show that all classes are affected. As for missing annotations and inaccurate bounding boxes, the Person class is affected most while the other classes are affected in a relatively similar manner. In summary, the statistics show that the noise effectively simulates wrong classification labels. A detailed analysis can be found in Appendix B.4.

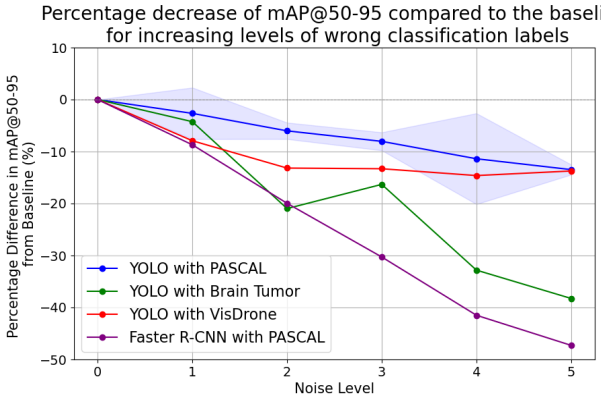

Figure 6: Percentage decrease in mAP@50-95 indicating that YOLO is severly sensitive to wrong classification labels at all noise levels. Faster R-CNN is even significantly more sensitive. Smaller datasets and smaller objects with wrong classification labels affect detector performance more, particularly at lower noise levels.

YOLO with PASCAL: Figure 6 shows a notable, consistent linear decrease from noise level 0 to noise level 5, with a 12% performance loss at noise level 5. The confidence intervals are very tight except at noise levels 1 and 4. These results indicate that YOLO is sensitive to wrong classification labels at all noise levels. Additionally, the mAP@50. mAP@75, and mAP@95 are investigated and detailed in Appendix C.4. The mAP@50 and mAP@75 followed similar trends. However, the mAP@95 showed a significant initial drop from noise level 0 to noise level 1 as the mAP@95 dropped from 0.15 to 0.095. This highlights that even 10% of wrong classification labels severely impacts the ability of the model to accurately detect objects.

YOLO with VisDrone: Interestingly, Figure 6 exhibits a linear decrease from noise level 0 to noise level 2, with a performance decrease of 12% at noise level 2, after which the performance stabilizes. These results suggest that wrong classification labels of small objects negatively impact performance more than big objects, especially at lower noise levels.

YOLO with Brain-Tumor: Remarkably, Figure 6 shows two initial drops,resulting in a 21% performance loss at noise level 2, followed by a slight improvement at noise level 3 to 17%. Finally, the performance drops significantly to 33% and to 38% for the remaining noise levels, respectively. The significant drops and instability can be explained by the smaller size of the dataset.

Faster R-CNN with PASCAL: Figure 6 shows a consistent linear decrease over the noise levels much more severe than YOLO with PASCAL, with a staggering performance loss of 48% at noise level 5. These results suggest that Faster R-CNN is less robust to wrong classification labels than YOLO.

In summary, YOLO demonstrates severe sensitivity to wrong classification labels, highlighted by the wide confidence interval at noise level 1. Faster R-CNN seems more sensitive to wrong classification labels than YOLO. Moreover, smaller datasets seem more sensitive to wrong classification labels. Finally, wrong classification labels of smaller objects impact performance more severely than large objects, especially at lower noise levels.

6 Discussion

This research aimed to evaluate the sensitivity of deep-learned object detectors to four types of human annotation noise: (i) missing annotations, (ii) extra annotations, (iii) inaccurate bounding boxes, and (iv) wrong classification labels. To address this, the performance of YOLOv8 and Faster R-CNN was assessed for each noise type on different datasets with incrementally added noise levels. The experiments revealed the following:

1. Annotation noise in smaller datasets impacts object detector performance more negatively than annotation noise in larger datasets.
2. Annotation noise in smaller objects impacts detector performance more negatively compared to bigger objects.
3. YOLOv8 shows resilience to low levels of missing annotations and inaccurate bounding boxes but is sensitive to all levels of wrong classification labels.
4. Extra annotations have a regularizing effect on YOLOv8.
5. Faster R-CNN is more sensitive to all noise types compared to YOLOv8, except for inaccurate bounding boxes where performance is impacted similarly.

The conclusions were drawn from an extensive analysis of the results based on multiple metrics. However, some conclusions were based on single runs for each noise type. Specifically, conclusions 1, 2, and 5 need to be interpreted with caution due to being derived from single runs. The remaining conclusions are based on three runs and are more reliable.

The performance decrease caused inaccurate bounding box noise for the YOLO and Faster R-CNN architecture follows a similar trend as for FCOS and Faster R-CNN as shown in [3]. Furthermore, *Chadwick and Newman* [10]showed severe performance degradation for the CNN object detector at low levels of all types of noise. The results show that YOLO shows more robustness to annotation noise across all levels. To obtain the best results reliably, data needs to be meticulously annotated. However, when limited resources are available, having less strict demands on missing annotations, extra annotations, and inaccurate bounding boxes will not severely impact YOLO performance. Additionally, building on top of existing knowledge in the machine learning domain that larger datasets enhance model robustness, smaller datasets indeed harm resilience of the detectors against annotation noise. Similarly, the greater impact on smaller objects is logical, given that the performance of state-of-the-art object detectors on larger objects significantly outperforms smaller objects, indicating that smaller objects are harder to learn. Additionally, the relative size of noise-induced errors is larger for smaller objects, leading to greater performance degradation.

The regularizing effect of extra annotations on YOLOv8 might be due to the model's ability to learn from additional, noisy data, enhancing its generalization capabilities. YOLOv8's sensitivity to incorrect classification labels at all noise levels could stem from the architecture's reliance on accurate class predictions for effective object detection. The similar impact of inaccurate bounding boxes on YOLO and Faster R-CNN suggests a shared vulnerability in localizing objects precisely under such conditions.

Future research should perform hyperparameter optimization in a similar experiment to see if noise can be mitigated in a simple manner. Additionally, more runs should be conducted on Faster R-CNN compared to YOLOv8 to validate the observed trends. Moreover, the performance of multiple single and two-stage detectors should be compared. Finally, resources should be allocated to investigate the effects of the fifth noise type not considered in this study: systematically swapping classes with similar appearances.

7 Responsible Research

The results presented in this study are derived directly from the experiments without any manipulation. Adhering to principles of responsible and transparent research, all data, trained models, and code used in this study are openly available in a dedicated repository[7]. This repository includes

[7] Available on Github

tools to replicate the experiments and verify results, ensuring the reliability of the findings reported here.

In line with the FAIR principles, the models and datasets used in this research are easily discoverable via dedicated web pages, ensuring they are findable. Accessibility is ensured as all models and datasets are freely available without requiring any credentials for download. This open access guarantees that these resources are available to anyone interested in this research. The datasets include annotations in standard formats (XML), and detector-specific formats are well-documented and accessible online, ensuring interoperability. Extensive documentation of the datasets and models is available, making them reusable for further research.

Moreover, this research is designed to be fully reproducible. All relevant code is available in the GitHub repository, with clear instructions on how to set up and run the experiments. The repository includes detailed information on the seeds, specific models, hyperparameters, and other essential configurations used in the experimental setup. This transparency ensures that others can replicate the experiments and verify the results, contributing to the robustness and reliability of the research findings.

References

[1] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[2] J. Li, C. Xiong, R. Socher, and S. Hoi, "Towards noise-resistant object detection with noisy annotations," *arXiv preprint arXiv:2003.01285*, 2020.

[3] S. Wang, J. Gao, B. Li, and W. Hu, "Narrowing the gap: Improved detector training with noisy location annotations," *IEEE Transactions on Image Processing*, vol. 31, pp. 6369–6380, 2022.

[4] U. Tkachenko, A. Thyagarajan, and J. Mueller, "Objectlab: Automated diagnosis of mislabeled images in object detection data," *arXiv preprint arXiv:2309.00832*, 2023.

[5] H. Su, J. Deng, and L. Fei-Fei, "Crowdsourcing annotations for visual object detection," in *Workshops at the twenty-sixth AAAI conference on artificial intelligence*, 2012.

[6] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from noisy large-scale datasets with minimal supervision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 839–847, 2017.

[7] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755, Springer, 2014.

[8] 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.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[10] S. Chadwick and P. Newman, "Training object detectors with noisy data," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1319–1325, IEEE, 2019.

[11] X.-Y. Zhang, C.-L. Liu, and C. Y. Suen, "Towards robust pattern recognition: A review," *Proceedings of the IEEE*, vol. 108, no. 6, pp. 894–922, 2020.

[12] B. Frenay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2014.

[13] D. P. Papadopoulos, J. R. Uijlings, F. Keller, and V. Ferrari, "Training object class detectors with click supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6374–6383, 2017.

[14] D. Papadopoulos, A. Clarke, F. Keller, and V. Ferrari, "Training object class detectors from eye tracking data," in *Computer Vision – ECCV 2014*, Lecture Notes in Computer Science, pp. 361–376, Springer International Publishing, Sept. 2014. European Conference on Computer Vision 2014, ECCV 2014 ; Conference date: 05-09-2014 Through 12-09-2014.

[15] B. Adhikari, J. Peltomäki, S. Bakhshi Germi, E. Rahtu, and H. Huttunen, "Effect of label noise on robustness of deep neural network object detectors," in *Computer Safety, Reliability, and Security. SAFECOMP 2021 Workshops* (I. Habli, M. Sujan, S. Gerasimou, E. Schoitsch, and F. Bitsch, eds.), Lecture Notes in Computer Science, pp. 239–250, Springer, 2021. JUFOID=62555; International Conference on Computer Safety, Reliability, and Security ; Conference date: 07-09-2021 Through 10-09-2021.

[16] Y. Xu, L. Zhu, Y. Yang, and F. Wu, "Training robust object detectors from noisy category labels and imprecise bounding boxes," *IEEE Transactions on Image Processing*, vol. 30, pp. 5782–5792, 2021.

[17] R. Varghese and S. M., "Yolov8: A novel object detection algorithm with enhanced performance and robustness," in *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, pp. 1–6, 2024.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[19] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, "Detection and tracking meet drones challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7380–7399, 2021.

[20] H. Bilen and A. Vedaldi, "Weakly supervised deep detection networks," *CoRR*, vol. abs/1511.02853, 2015.

[21] F. Wan, P. Wei, Z. Han, J. Jiao, and Q. Ye, "Min-entropy latent model for weakly supervised object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2395–2409, 2019.

[22] R. G. Cinbis, J. Verbeek, and C. Schmid, "Weakly supervised object localization with multi-fold multiple instance learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 1, pp. 189–203, 2016.

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

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21–37, Springer, 2016.

[25] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[26] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. N. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *CoRR*, vol. abs/2104.11892, 2021.

[27] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

A Noise Generation Pseudo-code

The following algorithms present the pseudocode of the noise generation scripts.

A.1 Missing annotations

Algorithm 2 Generate Missing Annotations

1: **num_annotations_to_remove** = $0.1*$total_annotations
2: Initialize random seed with seed
3: **for** i: 0 ... num_annotations_to_remove **do**
4: Select a random annotation file
5: Remove a random annotation from the file
6: **end for**

A.2 Extra Annotations

Algorithm 3 Generate Small Extra Annotation

x_center = uniformly generate between 0 and 1
y_center = uniformly generate between 0 and 1
while $\min(\text{Euclidean distance}(\text{new\_center}, \text{centers})) < \epsilon$ **do**
x_center = uniformly generate between 0 and 1
y_center = uniformly generate between 0 and 1
end while
width_range = $\min(1-\text{x\_center}, \text{x\_center})$
height_range = $\min(1-\text{y\_center}, \text{y\_center})$
width = uniformly generate between 0 and width_range
height = uniformly generate between 0 and height_range

Algorithm 4 Generate Big Extra Annotation

x_center = $\mathcal{N}(0.5, 0.185^2)$
y_center = $\mathcal{N}(0.5, 0.185^2)$
while $\min(\text{Euclidean distance}(\text{new\_center}, \text{centers})) < \epsilon$ **do**
x_center = $\mathcal{N}(0.5, 0.185^2)$
y_center = $\mathcal{N}(0.5, 0.185^2)$
end while
width_range = $\min(1 - \text{x\_center}, \text{x\_center})$
height_range = $\min(1 - \text{y\_center}, \textbf{y\_center})$
width = uniformly generate between $0.5 *$ width_range and width_range
height = uniformly generate between $0.5 *$ height_range and height_range

Algorithm 5 Generate Extra Annotations

num_annotations_to_remove = $0.1*$total_annotations
Initialize random seed with seed
for i: 0 ... in num_annotations_to_remove **do**
Select a random label file from labels_folder
Generate extra annotation by calling Extra Annotation
class_label = uniformly generate
Add the extra annotation
end for

A.3 Inaccurate Bounding Boxes

Algorithm 6 corrupt_bbox

Input: class_id, x_center, y_center, bbox_width, bbox_height
Output: Corrupted bounding box annotation
x_center_new = x_center + random value between max(-0.1, -x_center) and min(0.1, 1 - x_center)
y_center_new = y_center + random value between max(-0.1, -y_center) and min(0.1, 1 - y_center)
max_width = min(1 - x_center_new, x_center_new) * 2
max_height = min(1 - y_center_new, y_center_new) * 2
max_width = min(1.1 * bbox_width, max_width)
max_height = min(1.1 * bbox_height, max_height)
bbox_width_new = random value between 0.9 * min(bbox_width, max_width) and max_width
bbox_height_new = random value between 0.9 * min(bbox_height, max_height) and max_height

Algorithm 7 generate_noise_type3

Input: labels_folder, target_labels_folder, seed, total_annotations
Output: Labels and images with corrupted annotations
num_annotations_to_corrupt = $0.1*$total_annotations
Initialize random seed with seed
for $i = 0$ to num_annotations_to_corrupt -1 **do**
Select a random label file from labels_folder
Collect the annotations from the label file
Corrupt the annotation using corrupt_bbox
Replace the annotation with the corrupted one
end for

A.4 Wrong classification labels

Algorithm 8 Generate Wrong Classification Label

Input: seed, total_annotations
Output: Modified labels and images with missing annotations
num_annotations_to_remove = $0.1*$total_annotations
Initialize random seed with seed
for annotation in num_annotations_to_remove **do**
Select a random label file
Select a random annotation from the label file
Randomly Generate a different classification label
end for

B Statistics for the experiments

B.1 Missing Annotations

Statistics are collected on the removed annotations for experiments involving YOLO and PASCAL. A notable finding is the substantial difference in the number of removed annotations per class. On average, around 6178.0 annotations for the Person class are removed at noise level 5, while other classes have between 192.33 and 812.0 annotations removed. This variation can be attributed to the PASCAL dataset containing significantly more annotations for the Person class compared to other classes. The average number of removed annotations per class across noise levels is reported in Table 1. Additionally, the mean, standard deviation, and quantiles of the removed annotation areas are calculated for each noise level. These statistics are shown in Table 2. The results demonstrate that annotations of both small and large boxes were removed. Removing annotations of varying object sizes is important for evaluating how sensitive object detectors are to missing annotations, as it simulates realistic human annotation errors and tests detectors' robustness across different object scales.

Table 1: Average number of annotations removed per class over different noise levels showing that the Person class was affected most.

Classes	Noise Level 1	Noise Level 2	Noise Level 3	Noise Level 4	Noise Level 5
aeroplane	110.33	211.67	311.33	391.33	471.67
bicycle	65.33	127.67	194.67	261.67	328.67
bird	134.33	252.67	372.33	470.33	572.67
boat	74.33	145.67	213.67	287.33	358.67
bottle	62.67	140.67	231.00	331.67	432.00
bus	51.67	98.00	146.67	202.33	256.67
car	150.33	308.33	452.00	612.00	792.33
cat	160.67	308.67	453.33	566.00	678.00
chair	126.33	260.00	417.67	586.33	778.00
cow	47.33	112.00	170.67	232.67	290.67
diningtable	33.00	72.33	104.00	149.00	192.33
dog	174.33	352.33	514.67	669.67	812.00
horse	64.33	130.00	196.67	269.00	331.33
motorbike	67.00	129.67	189.33	255.00	322.00
person	1344.33	2693.33	4040.00	5387.00	6718.00
pottedplant	62.00	115.00	178.00	247.67	320.33
sheep	62.00	122.00	190.00	255.67	321.67
sofa	60.00	126.00	189.00	249.00	323.00
train	84.67	164.00	234.33	303.67	356.33
tvmonitor	61.00	122.00	188.67	256.67	323.67

Noise levels	1	2	3	4	5
Mean	0.2986	0.2913	0.2830	0.2759	0.2683
Std Dev	0.2752	0.2722	0.2699	0.2686	0.2661
25th %	0.0615	0.0591	0.0552	0.0509	0.0472
50th %	0.2135	0.2056	0.1941	0.1844	0.1752
75th %	0.4793	0.4662	0.4541	0.4414	0.4268

Table 2: Statistics of the Areas of the Removed Annotations at Different Noise Levels showing that small, medium and large objects were removed.

In addition, for the YOLO with PASCAL experiments, the average percentage decrease in AP@50-95 relative to the baseline for each class is computed across different noise levels. Figure 7 shows that the Person class is one of the more robust classes to missing annotations. This robustness can be attributed to the PASCAL dataset having significantly more annotations for the Person class. This suggests that more training data makes the model more robust to missing annotations. Additionally, Figure 7 shows that the Sofa class is affected most. The Sofa class contains the least amount of annotations in the training data, further underscoring that more training data makes the model more robust to missing annotations.

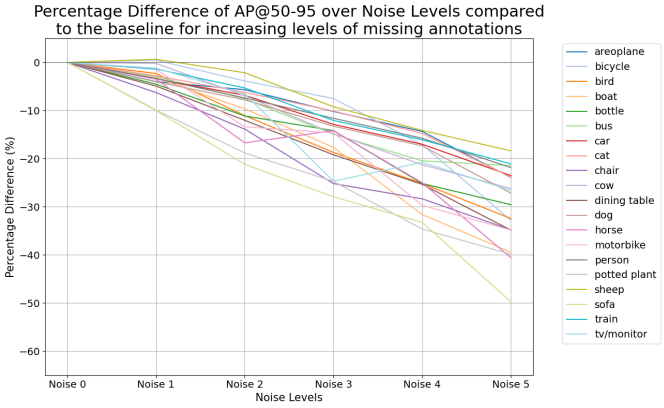

Figure 7: Average percentage decrease in AP@50-95 for each class in the PASCAL dataset trained with YOLO. The graph shows that the Person class remains one of the more robust classes against missing annotations.

B.2 Extra annotations

Small extra annotations

For the YOLO and PASCAL experiments, statistics are collected for the generated small, extra annotations. In contrast to removed annotations, the extra annotations are distributed evenly across classes. The average number of extra annotations per class ranges from 717.0 to 769.66 at noise level 5. For other noise levels, annotations are also evenly distributed. The average extra annotations per class can be found in Table 3. Additionally, the mean, standard deviation, and quantiles are calculated for the areas of the extra annotations. These statistics are reported in Table 4. As shown, the areas are very small compared to those of the removed annotations. The average at the noise levels for missing annotations ranged from 0.26 to 0.29, while for the small extra annotations this range from 0.01 to 0.02. Finally, the mean, standard deviation, and quantiles of the highest IoU of the new annotations with original annotations in images were calculated. These statistics are in Table 5. The IoU was used to measure the overlap with real objects. They demonstrate that the generated annotations did not correspond to any real objects. From these statistics, we can conclude that extra annotations were evenly distributed per class and corresponded to small additional annotations that did not match real objects. Therefore, the noise generation process was able to simulate small extra annotations and is suitable to study the sensitivity of object detectors to small extra annotations.

Table 3: Average number of small extra annotations per class over the noise levels showing that the small extra annotations were evenly distributed over the classes

Classes	Noise Level 1	Noise Level 2	Noise Level 3	Noise Level 4	Noise Level 5
aeroplane	159.00	305.33	446.00	586.00	730.33
bicycle	135.33	290.67	440.33	598.00	750.33
bird	146.00	296.67	448.33	600.33	745.67
boat	148.67	309.67	463.67	613.00	766.33
bottle	148.67	292.00	437.00	592.33	734.00
bus	142.33	300.33	447.67	594.33	748.67
car	156.33	297.67	452.33	614.00	769.00
cat	156.67	308.67	456.33	601.33	758.67
chair	146.33	295.67	436.33	578.67	729.33
cow	152.33	296.33	445.67	594.33	742.00
diningtable	159.00	303.00	469.67	629.67	777.00
dog	156.00	310.00	467.33	612.33	760.33
horse	158.33	311.67	465.33	622.00	767.67
motorbike	139.00	275.00	436.00	583.33	727.33
person	143.33	297.00	437.33	587.33	744.33
pottedplant	150.00	297.33	431.67	571.67	717.00
sheep	153.33	304.33	454.00	601.33	754.33
sofa	147.00	292.67	448.67	596.33	744.67
train	143.67	296.33	443.67	588.00	743.33
tvmonitor	154.67	296.33	460.67	619.67	769.67

Table 4: Statistics of the areas of the small extra annotations showing that the generated annotations indeed covered small areas.

Noise levels	1	2	3	4	5
Mean	0.0158	0.0157	0.0156	0.0156	0.0156
Std Dev	0.0232	0.0230	0.0230	0.0231	0.0230
25th %	0.0015	0.0015	0.0015	0.0015	0.0015
50th %	0.0065	0.0065	0.0065	0.0064	0.0064
75th %	0.0201	0.0199	0.0199	0.0198	0.0198

Table 5: Statistics of the highest IoU of the added annotations with the original annotations at different noise levels demonstrating that the small extra annotations do not correspond to real objects.

Noise levels	1	2	3	4	5
Mean	0.0289	0.0286	0.0287	0.0284	0.0284
Std Dev	0.0640	0.0632	0.0640	0.0640	0.0641
25th %	0.0000	0.0000	0.0000	0.0000	0.0000
50th %	0.0006	0.0006	0.0006	0.0005	0.0005
75th %	0.0256	0.0253	0.0254	0.0245	0.0244

Furthermore, for the YOLO and PASCAL experiments, the average percentage decrease in AP@50-95 for each class was calculated across the various noise levels. Figure 8 demonstrates that all classes experience similar fluctuations in performance within narrow intervals across the noise levels, further highlighting that small extra annotations act as regularization terms.

Big Extra Annotations

For the PASCAL datasets experiments, statistics are collected for the generated big, extra annotations. Similarly to the small extra annotations, the big extra annotations are distributed evenly per class. The average number of big extra annotations per class ranges from 719.0 to 767.67 at noise level 5. The extra annotations are also distributed evenly over the remaining noise levels. The average extra annotations per class can be found in Table 6. Additionally, the mean, standard deviation, and quantiles are calculated for the areas of the extra annotations. These statistics are shown in Table 8. The areas are big in comparison to those

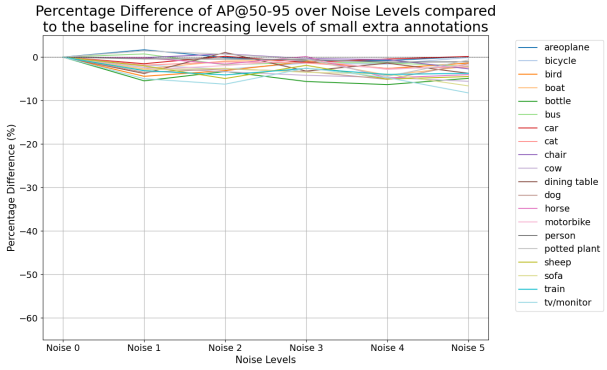

Figure 8: Average percentage decrease in AP@50-95 for each class in the PASCAL dataset trained with YOLO. The graph shows that all the classes have a similar performance evolution with small extra annotations added.

of the small extra annotations, with the means of the big extra annotations ranging from 0.28 to 0.29 in comparison to 0.01 to 0.02. Additionally, the mean of the areas of the big extra annotations is very similar to that of the removed annotations. Furthermore, the standard deviation is smaller for the big extra annotations showing that the experiment indeed generates big extra annotations. Finally, the mean, standard deviation, and quantiles of the highest IoU of the new annotations with original annotations in images are calculated. These statistics are reported in Table 7. They show that the generated annotations do have some overlap with existing annotations, in contrast to the small extra annotations. This difference was expected as big extra annotations take up significant space of the image and are bound to overlap to some extent with actual annotations. From these statistics we can conclude that the big extra annotations were evenly distributed per class and correspond to realistic big extra annotations.

Table 6: Average number of big extra annotations per class over the noise levels showing that the annotations are distributed evenly over the classes.

Classes	Noise Level 1	Noise Level 2	Noise Level 3	Noise Level 4	Noise Level 5
aeroplane	147.67	307.67	452.67	604.33	745.33
bicycle	146.00	295.33	458.67	615.00	755.33
bird	159.67	310.33	453.67	610.00	760.00
boat	145.33	297.33	434.67	594.33	746.33
bottle	153.33	301.00	457.67	605.00	752.00
bus	152.33	303.67	452.33	604.00	756.00
car	152.67	304.00	452.33	604.00	765.67
cat	152.00	297.00	434.67	581.33	743.33
chair	151.00	298.00	449.00	602.33	755.67
cow	150.67	291.33	440.00	586.67	741.33
diningtable	149.67	303.67	467.67	611.00	767.67
dog	142.33	307.33	462.33	613.67	757.00
horse	161.00	313.33	462.33	604.33	750.33
motorbike	160.00	310.67	466.67	602.33	751.67
person	146.67	299.00	448.33	593.00	739.67
pottedplant	146.33	284.33	424.00	574.00	719.00
sheep	135.67	288.00	446.33	599.33	749.00
sofa	147.00	293.00	443.67	592.33	740.67
train	146.33	298.33	443.33	597.00	743.00
tvmonitor	150.33	288.67	437.67	590.00	741.00

Noise levels	1	2	3	4	5
Mean	0.2806	0.2826	0.2830	0.2829	0.2832
Std Dev	0.1480	0.1488	0.1493	0.1495	0.1498
25th %	0.1712	0.1740	0.1743	0.1741	0.1736
50th %	0.2649	0.2651	0.2650	0.2648	0.2651
75th %	0.3698	0.3729	0.3738	0.3738	0.3746

Table 7: Statistics of the areas of the big extra annotations at different noise levels showing that the they cover large areas.

Noise levels	1	2	3	4	5
Mean	0.2490	0.2485	0.2490	0.2493	0.2489
Std Dev	0.1886	0.1871	0.1871	0.1871	0.1869
25th %	0.0872	0.0888	0.0890	0.0890	0.0894
50th %	0.2320	0.2323	0.2333	0.2338	0.2330
75th %	0.3829	0.3791	0.3803	0.3805	0.3793

Table 8: Statistics of the highest IoU of the big extra annotations with the original annotations at different noise levels demonstrating that the big extra annotations cover some parts of real objects.

Additionally, for the YOLO and PASCAL runs, the averagepercentage decrease in AP@50-95 for each class is computed across the various noise levels. Figure 9 shows that all classes experience similar fluctuations in performance, similar to the experiment with small extra annotations. However, the intervals for bigger annotations are bigger. These findings further highlight that extra annotations act as regularization terms.

B.3 Inaccurate Bounding Boxes

For the PASCAL dataset experiments, the IoU between the corrupted bounding box and original bounding box is calculated. The values for each noise level can be found in Table 9. The table shows there is significant overlap between the corrupted and original bounding boxes, effectively simulating human annotation noise for bounding boxes. Additionally, the average number of corrupted bounding boxes per class is computed. As with missing annotations, there was a large discrepancy between the Person class and other classes. At noise level 5, on average 6,678 bounding boxes were corrupted for the Person class, while for other classes the average corruptions ranged from 304.0 to 811.66. This disparity can be attributed to the PASCAL dataset having significantly more annotations for the Person class than other classes. The average number of corrupted bounding boxes per class can be found in Table 10. In summary, the generated noise corrupted bounding boxes while maintaining significant overlap with original annotations, effectively simulating noisy human labels.

Additionally, the percentage decrease in AP@50-95 for each class in the PASCAL dataset for YOLO is computed across the various noise levels. Figure 10 shows that despite having the most corrupted bounding boxes, the Person class remains relatively robust across the noise levels. This further highlights that more data enhances model robustness inaccurate bounding boxes.

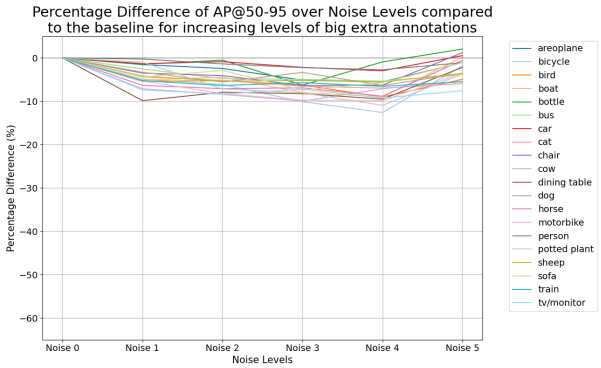

Figure 9: Average percentage decrease in AP@50-95 for each class in the PASCAL dataset trained with YOLO. The graph shows that all the classes have a similar performance evolution with big extra annotations added and follows a similar pattern as the small extra annotations but with slightly bigger intervals.

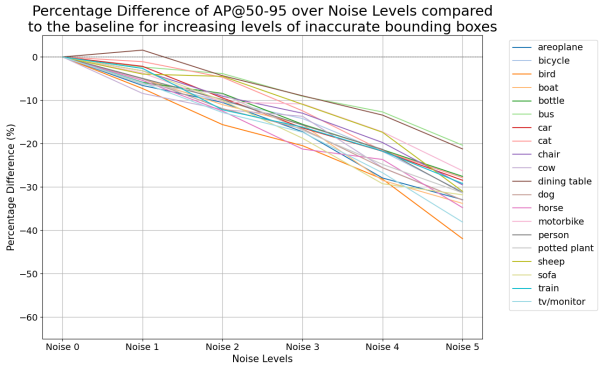

Figure 10: Average Percentage decrease in AP@50-95 for each class in the PASCAL dataset trained with YOLO. The graph shows that the Person class, despite having the most corrupted bounding boxes, remains relatively robust.

Noise levels:	1	2	3	4	5
Mean	0.5543	0.5487	0.5449	0.5381	0.5305
Std Dev	0.2480	0.2491	0.2513	0.2541	0.2571
25th %	0.4128	0.4030	0.3969	0.3818	0.3655
50th %	0.6273	0.6215	0.6177	0.6115	0.6041
75th %	0.7417	0.7383	0.7368	0.7342	0.7304

Table 9: Statistics of IoU of the corrupted bounding boxes with the original bounding boxes at different noise levels showing overlap that simulates human annotation noise.

Table 10: Average number of annotations removed per class over the noise levels showing that the Person Class was impacted most.

Classes	Noise Level 1	Noise Level 2	Noise Level 3	Noise Level 4	Noise Level 5
aeroplane	111.67	209.33	295.67	390.00	471.67
bicycle	54.33	116.67	184.33	249.00	316.33
bird	121.00	239.67	351.67	461.33	559.33
boat	72.67	156.33	237.33	312.00	389.67
bottle	71.67	139.33	222.33	307.00	400.67
bus	48.00	99.33	155.00	207.33	270.33
car	135.33	283.33	449.00	630.33	811.67
cat	169.00	334.00	470.00	591.33	702.33
chair	118.00	259.00	418.67	590.00	776.33
cow	48.67	104.67	159.67	207.33	271.00
diningtable	30.67	70.33	111.00	156.33	200.33
dog	176.00	342.33	509.33	663.67	804.67
horse	70.00	136.67	202.00	278.67	343.00
motorbike	56.67	117.00	182.33	256.33	325.33
person	1399.67	2745.67	4058.00	5373.67	6678.00
pottedplant	53.00	109.67	177.00	248.33	325.67
sheep	57.67	112.33	174.67	244.00	314.00
sofa	61.67	130.00	198.67	256.67	330.33
train	80.33	157.33	229.33	293.00	360.33
tvmonitor	60.00	129.00	202.00	267.67	329.00

B.4 Wrong Classification Labels

For the experiments involving the PASCAL dataset, the confusion matrix for the wrong and original classification labels was computed at each noise level. At noise level 5, all the classes were changed at least 4.3 times with each other. The confusion matrices revealed that all the classes were affected. As for missing annotations and inaccurate bounding boxes, the Person class was affected most. At noise level 5, the Person class was switched at least 238.7 with another class on average. The confusion matrices for each noise level are available in the repository[8]. In summary, the confusion matrices show that the noise script effectively simulated wrong classification labels.

Moreover, for the YOLO and PASCAL runs, the average percentage decrease in AP@50-95 compared to the baseline is computed for each class across the various noise levels. Figure 11 shows that despite having the most classification labels corrupted, the Person class remains the most robust against wrong classification noise. This further highlights that more training data improves model robustness against wrong classification labels.

[8] Available on Github

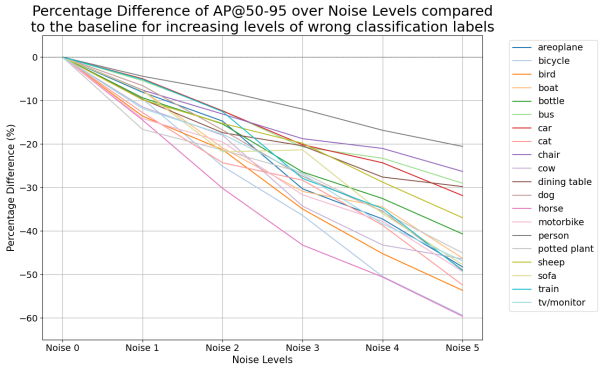

Figure 11: Average percentage decrease in AP@50-95 for each class in the PASCAL dataset trained with YOLO. The graph shows that the Person class, despite having the most labels corrupted, remains the most robust.

C Further Results for the experiments

C.1 Missing Annotations

Figures 12, 13, 14, and 15 present the 95% confidence intervals of the mAP@50-95, mAP@50, mAP@75, and mAP@95 for the YOLO object detector trained on the PASCAL dataset with increasing levels of missing annotations. The figures show that YOLO is somewhat robust to lower levels of missing annotations, but its performance drops significantly at higher noise levels.

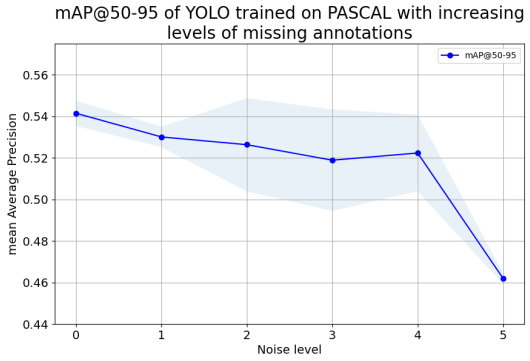

Figure 12: mAP@50-95 95% confidence interval of YOLO object detector for different levels of missing annotations in the PASCAL dataset. YOLO is somewhat resilient to lower levels of missing annotations, but performance drops significantly at higher levels.

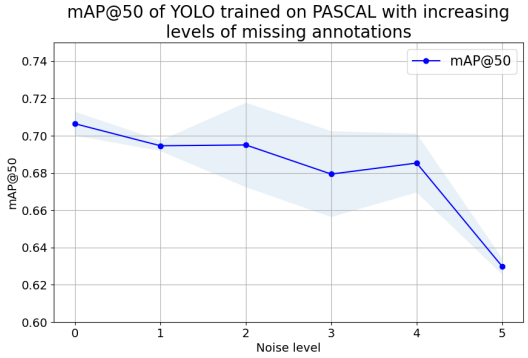

Figure 13: mAP@50 95% confidence interval of YOLO object detector for different levels of missing annotations in the PASCAL dataset. YOLO is somewhat resilient to lower levels of missing annotations, but performance drops significantly at higher levels.

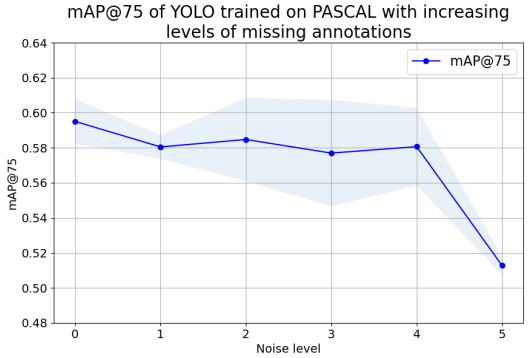

Figure 14: mAP@75 95% confidence interval of YOLO object detector for different levels of missing annotations in the PASCAL dataset. YOLO is somewhat resilient to lower levels of missing annotations, but performance drops significantly at higher levels.

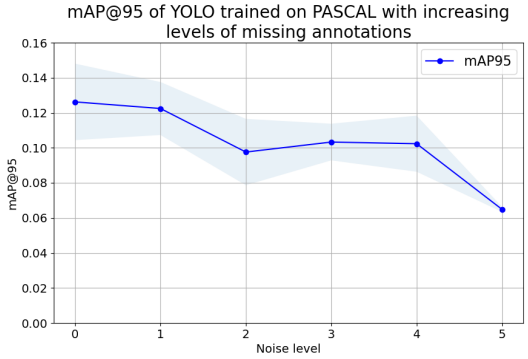

Figure 15: mAP@95 95% confidence interval of YOLO object detector for different levels of missing annotations in the PASCAL dataset. YOLO is somewhat resilient to lower levels of missing annotations, but performance drops significantly at higher levels.

C.2 Extra annotations

Small Extra Annotations

Figures 16, 17, 18, and 19 present the 95% confidence intervals of the mAP@50-95, mAP@50, mAP@75, and mAP@95 for the YOLO object detector trained on the PASCAL dataset with increasing levels of small extra annotations. The figures show that the extra annotations have a regularizing effect. However, Figure 19 shows that mAP@95 drops significantly even at noise level 1. In conclusion, the small extra annotations seem to have a regularizing effect but with cost to localization accuracy at the highest level.

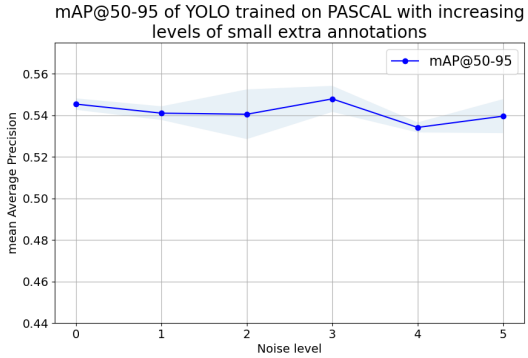

Figure 16: mAP@50-95 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The extra annotations seem to have a regularizing effect.

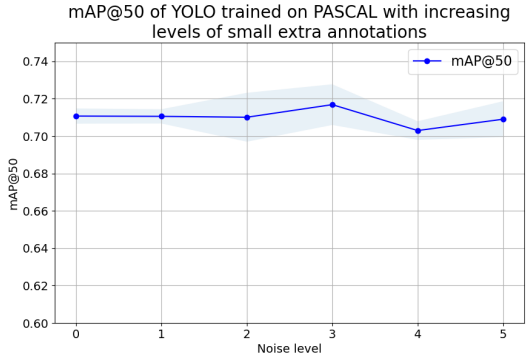

Figure 17: mAP@50 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The extra annotations seem to have a regularizing effect.

Big Extra Annotations

Figures 20, 21, 22, and 23 present the 95% confidence intervals of the mAP@50-95, mAP@50, mAP@75, and mAP@95 for the YOLO object detector trained on the PASCAL dataset with increasing levels of big extra annotations. The figures show that the extra annotations have a regularizing effect. However, Figure 23 shows that mAP@95 drops significantly even at noise level 1. In conclusion, the big extra annotations seem to have a regularizing effect but with cost to localization accuracy at the highest level.

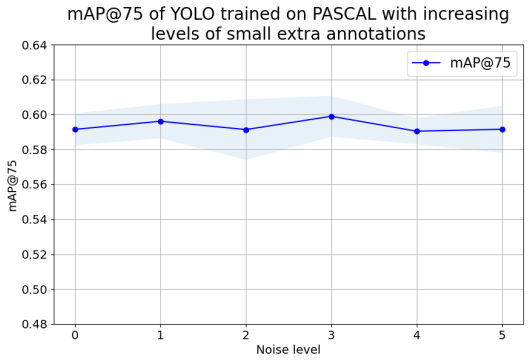

Figure 18: mAP@75 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The extra annotations seem to have a regularizing effect.

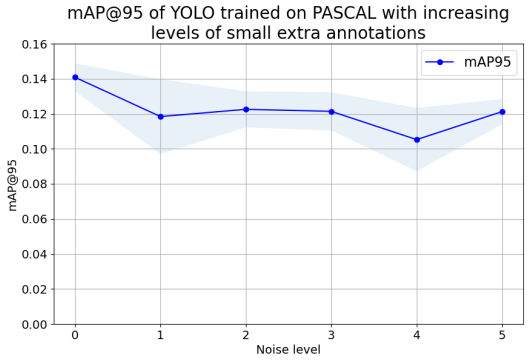

Figure 19: mAP@95 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The initial drop at noise level 1 indicates that the model's ability to precisely detect objects is severely harmed by small extra annotations.

C.3 Inaccurate Bounding Boxes

Figures 24, 25, 26, and 27 present the 95% confidence intervals of the mAP@50-95, mAP@50, mAP@75, and mAP@95 for the YOLO object detector trained on the PASCAL dataset with increasing levels of inaccurate bounding boxes. The figures show that YOLO exhibits some robustness at lower noise levels, but performance drops significantly at higher noise levels. Moreover, Figure 27 shows a linear decrease of mAP@95 over the noise levels, indicating that even minimal inaccurate bounding boxes severely impact the ability of YOLO to accurately detect objects at the highest precision.

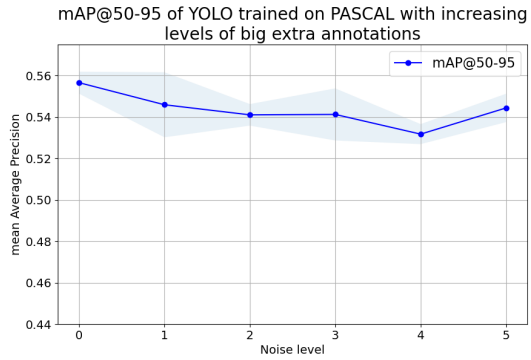

Figure 20: mAP@50-95 95% confidence interval of YOLO object detector for different levels of big extra annotations in the PASCAL dataset. The extra annotations seem to have a regularizing effect.

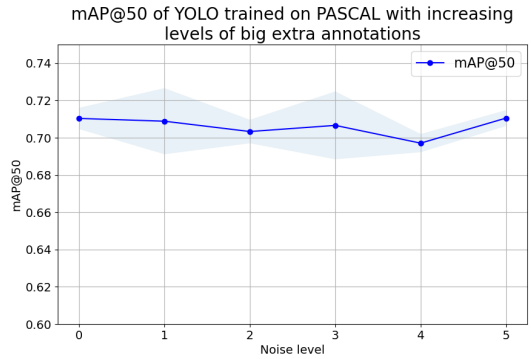

Figure 21: mAP@50 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The extra annotations seem to have a regularizing effect.

C.4 Wrong Classification Labels

Figures 28, 29, 30, and 31 present the 95% confidence interval of the mAP@50-95, mAP@50, mAP@75, and mAP@95 for YOLO trained on PASCAL with increasing levels of wrong classification labels. The figures show that YOLO is sensitive to all levels of wrong classification labels.

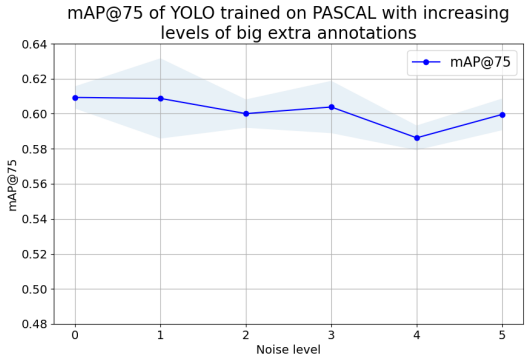

Figure 22: mAP@75 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The extra annotations seem to have a regularizing effect.

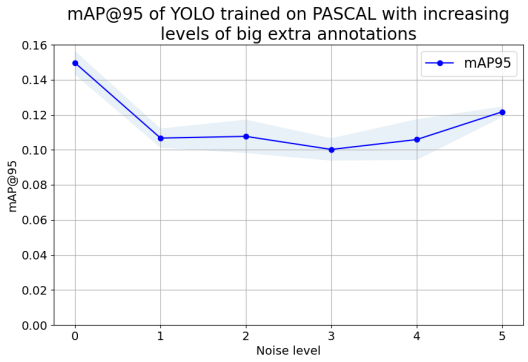

Figure 23: mAP@95 95% confidence interval of YOLO object detector for different levels of small extra annotations in the PASCAL dataset. The initial drop at noise level 1 indicates that the model's ability to precisely detect objects is severely harmed by small extra annotations.

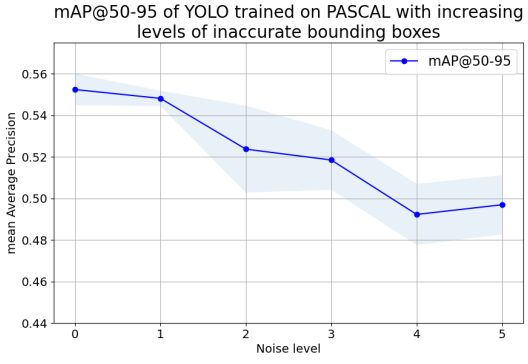

Figure 24: mAP@50-95 95% confidence interval of YOLO for different levels of inaccurate bounding boxes in the PASCAL dataset. YOLO seems somewhat robust to inaccurate bounding boxes at low noise levels but performance drops significantly at higher noise levels.

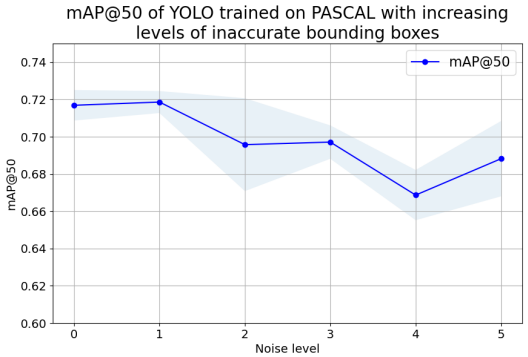

Figure 25: mAP@50 95% confidence interval of YOLO for different levels of inaccurate bounding boxes in the PASCAL dataset. YOLO seems somewhat robust to inaccurate bounding boxes at low noise levels but performance drops significantly at higher noise levels.

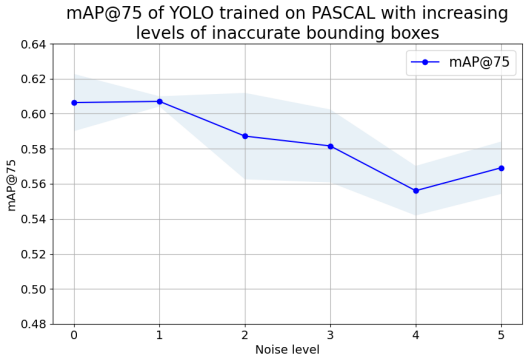

Figure 26: mAP@75 95% confidence interval of YOLO for different levels of inaccurate bounding boxes in the PASCAL dataset. YOLO seems somewhat robust to inaccurate bounding boxes at low noise levels but performance drops significantly at higher noise levels.

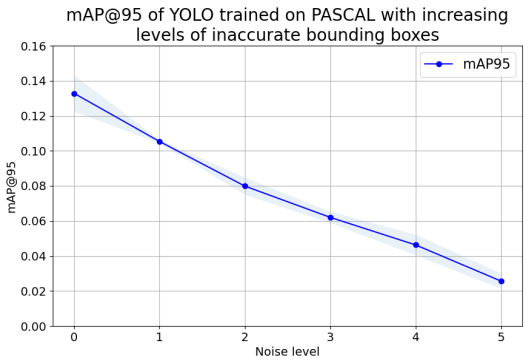

Figure 27: mAP@95 95% confidence interval of YOLO for different levels of inaccurate bounding boxes in the PASCAL dataset. Inaccurate bounding boxes severely impact YOLO's ability to accurately detect objects at the highest precision.

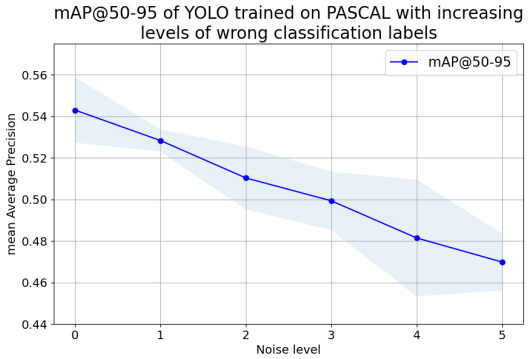

Figure 28: mAP@50-95 95% confidence interval of YOLO for different levels of wrong classification labels in PASCAL. YOLO is sensitive to all levels of wrong classification labels.

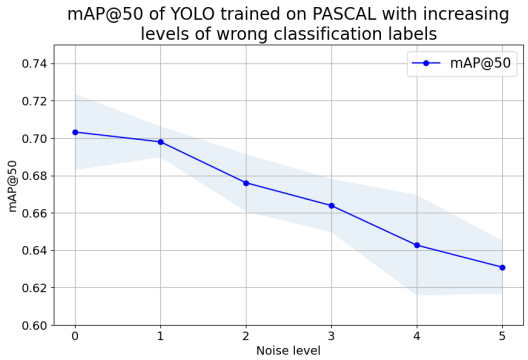

Figure 29: mAP@50 95% confidence interval of YOLO for different levels of wrong classification labels in PASCAL. YOLO is sensitive to all levels of wrong classification labels.

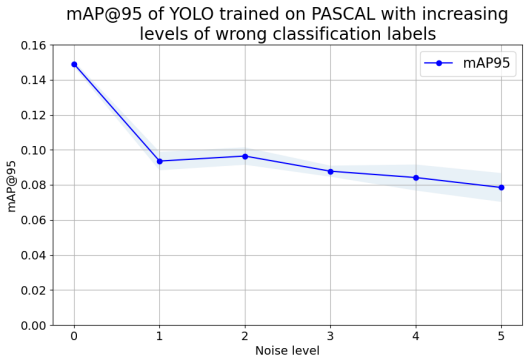

Figure 31: mAP@95 95% confidence interval of YOLO for different levels of wrong classification labels in PASCAL. YOLO is sensitive to all levels of wrong classification labels.

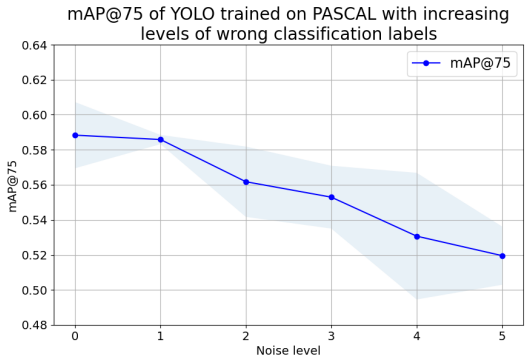

Figure 30: mAP@75 95% confidence interval of YOLO for different levels of wrong classification labels in PASCAL. YOLO is sensitive to all levels of wrong classification labels.