# SYNTHETIC AUGMENTATION METHODS FOR OBJECT DETECTION IN OVERHEAD IMAGERY

By

Nicholas R. Hamilton

A THESIS

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

In Computer Science

MICHIGAN TECHNOLOGICAL UNIVERSITY

2022

This thesis has been approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE in Computer Science.

Department of Computer Science

| | |
|---|---|
| Thesis Advisor: | *Dr. Timothy Havens* |
| Committee Member: | *Dr. Nathir Rawashdeh* |
| Committee Member: | *Dr. Dukka KC* |
| Committee Member: | *Dr. Wenbin Zhang* |
| Department Chair: | *Dr. Zhenlin Wang* |

# Contents

**List of Figures** . . . . . **vii**

**List of Tables** . . . . . **xv**

**Acknowledgments** . . . . . **xvii**

**List of Abbreviations** . . . . . **xix**

**Abstract** . . . . . **xxi**

**1 Introduction** . . . . . **1**

**2 Data Sets** . . . . . **3**

**3 Spin-Set Data Augmentation** . . . . . **7**

3.1 Automated Background Selection . . . . . 9

3.2 Synthetic Data Generation . . . . . 12

3.2.1 Graininess . . . . . 13

3.2.2 Blur . . . . . 14

3.2.3 Discoloration . . . . . 14

3.2.4 Brightness and Contrast . . . . . 15

3.2.5 Occlusion . . . . . 15

3.2.6 Bounding Box Generation . . . . . 16

**4 Experiments** . . . . . **17**

**5 Results** . . . . . **19**

5.1 RGB Imagery . . . . . 19

5.2 LWIR Imagery . . . . . 26

**6 Conclusion** . . . . . **33**

**References** . . . . . **35**

**A Additional Figures** . . . . . **39**

# List of Figures


2.1 Example images from overhead RGB imagery data set. . . . . 4

(a) Route1 . . . . . 4

(b) Location4 . . . . . 4

2.2 Example images from overhead LWIR imagery data set. . . . . 4

(a) Route1 . . . . . 4

(b) Location4 . . . . . 4

3.1 Examples of generated RGB images. . . . . 8

(a) Generated image . . . . . 8

(b) Generated image . . . . . 8

3.2 Examples of generated LWIR images. . . . . 9

(a) Generated image . . . . . 9

(b) Generated image . . . . . 9

3.3 Examples of relevant backgrounds extracted. . . . . 10

(a) Relevant background . . . . . 10

(b) Relevant background . . . . . 10

3.4 Examples of irrelevant backgrounds created. Search terms should be carefully chosen to reduce the incidence of unsuitable backgrounds. 11

(a) Irrelevant background . . . . . 11

(b) Irrelevant background . . . . . 11

5.1 Testing set $F_1$ scores for each RGB subset. Orange bar indicates the median and the green triangle indicates the mean; box and whiskers indicate quartiles and max/min, respectively. More figures are shown later. . . . . 22

(a) Trained on Location1 . . . . . 22

(b) Trained on Location2 . . . . . 22

(c) Trained on Location3 . . . . . 22

(d) Trained on Location4 . . . . . 22

(e) Trained on Route1 . . . . . 22

(f) Trained on Route2 . . . . . 22

5.1 Testing set $F_1$ scores for each RGB subset. Orange bar indicates the median and the green triangle indicates the mean; box and whiskers indicate quartiles and max/min, respectively. Final figures are shown here. . . . . 25

(a) Trained on FlyToward . . . . . 25

(b) Trained on FlyAway . . . . . 25

(c) Trained on synthetic only . . . . . 25

5.2 Results of single subset method trained with Faster R-CNN using LWIR spin-set augmentation. . . . . 27

5.3 Performance increase of single subset method trained with Faster R-CNN relative to results without LWIR spin-set augmentation. . . . 28

5.4 Object recall over different target sizes for Faster R-CNN trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page. . . . . 31

(a) Trained on Location1 . . . . . 31

(b) Trained on Location2 . . . . . 31

(c) Trained on Location3 . . . . . 31

(d) Trained on Location4 . . . . . 31

(e) Trained on Route1 . . . . . 31

(f) Trained on Route2 . . . . . 31

5.4 Object recall over different target sizes for Faster R-CNN trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page. . 32

(a) Trained on FlyToward . . . . . 32

(b) Trained on FlyAway . . . . . 32

(c) Trained on Stage4 . . . . . 32

A.1 Example RGB images from Location1. . . . . 40

A.2 Example RGB images from Location2. . . . . 41

A.3 Example RGB images from Location3. . . . . 42

A.4 Example RGB images from Location4. . . . . 43

A.5 Example RGB images from Route1. . . . . 44

A.6 Example RGB images from Route2. . . . . 45

A.7 Example RGB images from FlyToward. . . . . 46

A.8 Example RGB images from FlyAway. . . . . 47

A.9 Example LWIR images from Location1. . . . . 48

A.10 Example LWIR images from Location2. . . . . 49

A.11 Example LWIR images from Location3. . . . . 50

A.12 Example LWIR images from Location4. . . . . 51

A.13 Example LWIR images from Route1. . . . . 52

A.14 Example LWIR images from Route2. . . . . 53

A.15 Example LWIR images from FlyToward. . . . . 54

A.16 Example LWIR images from FlyAway. . . . . 55

A.17 Example LWIR images from Stage4. . . . . 56

A.18 Results of single subset method trained with SSD using LWIR spin-set augmentation. . . . . 57

A.19 Results of single subset method trained with YOLO using LWIR spin-set augmentation. . . . . 58

A.20 Performance increase of single subset method trained with SSD relative to results without LWIR spin-set augmentation. . . . . 59

A.21 Performance increase of single subset method trained with YOLO relative to results without LWIR spin-set augmentation. . . . . 60

A.22 Results of all-but-one method trained with Faster R-CNN using LWIR spin-set augmentation. . . . . 61

A.23 Results of all-but-one method trained with SSD using LWIR spin-set augmentation. . . . . 62

A.24 Object recall over different target sizes for SSD trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page. . . . . 63

(a) Trained on Location1 . . . . . 63

(b) Trained on Location2 . . . . . 63

(c) Trained on Location3 . . . . . 63

(d) Trained on Location4 . . . . . 63

(e) Trained on Route1 . . . . . 63

(f) Trained on Route2 . . . . . 63

A.24 Object recall over different target sizes for SSD trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page. . . . . 64

(a) Trained on FlyToward . . . . . 64

(b) Trained on FlyAway . . . . . 64

(c) Trained on Stage4 . . . . . 64

A.25 Object recall over different target sizes for YOLO trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page. . . . . 65

(d) Trained on Location1 . . . . . 65

(e) Trained on Location2 . . . . . 65

(f) Trained on Location3 . . . . . 65

(g) Trained on Location4 . . . . . 65

(h) Trained on Route1 . . . . . 65

(i) Trained on Route2 . . . . . 65

A.25 Object recall over different target sizes for YOLO trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page. . . . . 66

(a) Trained on FlyToward . . . . . 66

(b) Trained on FlyAway . . . . . 66

(c) Trained on Stage4 . . . . . 66

A.26 Object recall over different target sizes for Faster R-CNN trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page. . . . . 67

(d) Tested on Location1 . . . . . 67

(e) Tested on Location2 . . . . . 67

(f) Tested on Location3 . . . . . 67

(g) Tested on Location4 . . . . . 67

(h) Tested on Route1 . . . . . 67

(i) Tested on Route2 . . . . . 67

A.26 Object recall over different target sizes for Faster R-CNN trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page. . 68

(a) Tested on FlyToward . . . . . 68

(b) Tested on FlyAway . . . . . 68

(c) Tested on Stage4 . . . . . 68

A.27 Object recall over different target sizes for SSD trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page. . . . . 69

(d) Tested on Location1 . . . . . 69

(e) Tested on Location2 . . . . . 69

(f) Tested on Location3 . . . . . 69

(g) Tested on Location4 . . . . . 69

(h) Tested on Route1 . . . . . 69

(i) Tested on Route2 . . . . . 69

A.27 Object recall over different target sizes for SSD trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page. . . . . 70

(a) Tested on FlyToward . . . . . 70

(b) Tested on FlyAway . . . . . 70

(c) Tested on Stage4 . . . . . 70

# List of Tables


2.1 Number of images in each RGB subset. . . . . 5

2.2 Number of images in each LWIR subset. . . . . 5

3.1 Parameters used for image generation. . . . . 13

5.1 $F_1$ score for training set and validation set with 0 synthetic generated images. . . . . 23

5.2 $F_1$ score for training set and validation set with 500 synthetic generated images. . . . . 23

5.3 $F_1$ score for training set and validation set with 1000 synthetic generated images. . . . . 24

5.4 $F_1$ score for training set and validation set with 5000 synthetic generated images. . . . . 24

# Acknowledgments

This project was supported in part by grants from the Office of Naval Research (#N00014-20-1-2793) and NGA (#HM0476-20-C-0020). We would like to thank Redmon and all the GitHub contributors for their work on the Darknet framework [1]. Additionally, some images in this paper were generated from videos on YouTube [2, 3, 4, 5]. Approved for public release, 21-365.

Content in this thesis was re-used from *Augmentation methods for object detection in overhead imagery* [6] with further results acquired since publication:

Nicholas Hamilton, Adam Webb, Zachary DeKraker, Ben Hendrickson, Matt Blanck, Erin Nelson, Wiley Roemer, and Timothy C. Havens "Augmentation methods for object detection in overhead imagery", Proc. SPIE 11729, Automatic Target Recognition XXXI, 117290I (12 April 2021); https://doi.org/10.1117/12.2588502

# List of Abbreviations

| | |
|---|---|
| Faster R-CNN | Faster Region-Based Convolutional Neural Networks |
| GEOINT | Geospatial Intelligence |
| LWIR | Long-Wave Infrared |
| RGB | Red-Green-Blue Color Model |
| SSD | Single Shot MultiBox Detector |
| YOLO | You Only Look Once |
| YOLOv4 | You Only Look Once (Version 4) |

# Abstract

The multidisciplinary area of *geospatial intelligence* (GEOINT) is continually changing and becoming more complex. From efforts to automate portions of GEOINT using machine learning, which augment the analyst and improve exploitation, to optimizing the growing number of sources and variables, there is no denying that the strategies involved in this collection method are rapidly progressing. The unique and inherent complexities involved in imagery analysis from an overhead perspective——e.g., target resolution, imaging band(s), and imaging angle——test the ability of even the most developed and novel machine learning techniques. To support advancement in the application of object detection in overhead imagery, we have developed a spin-set augmentation method that leverages synthetic data generation capabilities to augment the training data sets. We then test this method with the popular object detection networks YOLO, SSD, and Faster R-CNN. This thesis analyzes the synthetic augmentation method in terms of algorithm detection performance, computational complexity, and generalizability.

# Chapter 1

# Introduction

In the efforts to automate *geospatial intelligence* (GEOINT), computer vision techniques such as object detection are being used at an increasing rate [7]. One significant challenge when it comes to training object detection models is acquiring enough training data. Many computer vision models are trained using online, publicly available (and manually labelled) imagery [8]. While this is sufficient for many purposes, it becomes irrelevant as the focus becomes more niche. This means that often new data sets must be created that have considerably fewer trainable images.

To make up for a lack of training data, various studies have been done on using synthetic imagery for training computer vision models [8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. Several methods, of varying complexity, have been proposed to

create synthetic data that has the properties of real-world data. One method that is very computationally expensive is the Monte Carlo ray tracing proposed by Hartwig and Ropinski [10]. On the other hand, Georgakis et al. [9] used a technique where object images were extracted from large data sets and superimposed onto backgrounds of the desired environment. These studies were done in an indoor setting. They are quite different from the outdoor, overhead imaging environment, particularly Hartwig and Ropinski's research which involved reflective objects. However, we would expect the techniques used to be helpful for an outdoor overhead situation as well; our technique is similar to that of Georgakis et al. [9]. Mayer et al. [8] also used a very similar technique where 2D renders of 3D models were superimposed onto assorted backgrounds.

In this thesis we explore the efficacy of augmenting extremely small overhead imagery data sets with rapidly-produced synthetic imagery to improve object detection, and how the number of synthetic images affects model performance. The remainder of this thesis is organized as follows. Chapter 2 provides a short description of the data sets used for training. Chapter 3 explores the details of how our synthetic images are generated. We describe our experiments in Chapter 4. We show our results in Chapter 5 and finally we summarize in Chapter 6.

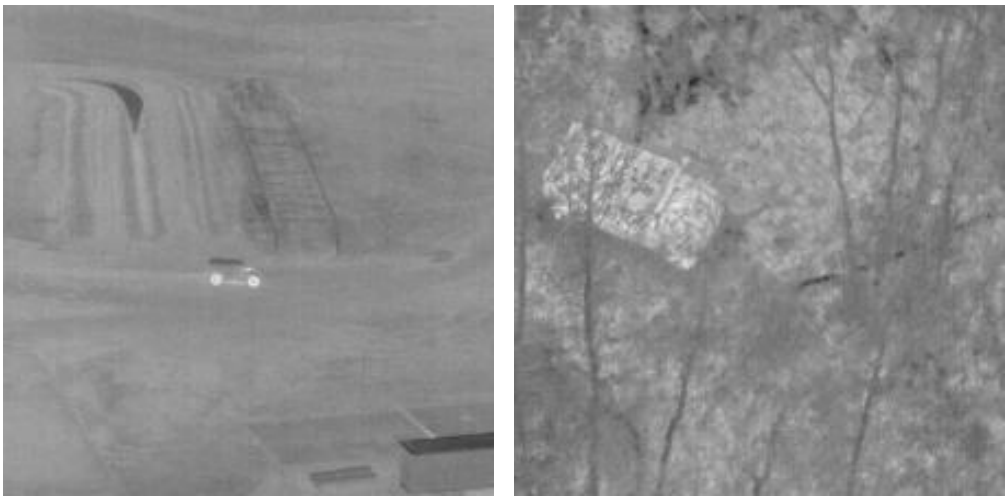
# Chapter 2

# Data Sets

In this work we used a data set of 1,091 color (RGB) images and 1,513 infrared (LWIR) images taken at varying heights and distances to a target vehicle, as seen in Figures 2.1, 2.2, and A.1 to A.17. The data sets are broken up into eight subsets for RGB images and nine for LWIR, each representing a different environment or imaging platform motion. The number of RGB images in a subset can be seen in Table 2.1 and varies from 17 (1.56%) up to 377 (34.56%), while the number of LWIR images can be seen in Table 2.2 and varies from 123 (8.13%) up to 270 (17.85%). The small number of real-world training imagery made these data sets useful for this research, however there are limitations that had to be acknowledged. The first is that there was only one type of vehicle used. Our object detection networks were trained to recognize six vehicle types but only one can be accurately tested. Because of this, we

(a) Route1 (b) Location4

**Figure 2.1:** Example images from overhead RGB imagery data set.



(a) Route1 (b) Location4

**Figure 2.2:** Example images from overhead LWIR imagery data set.

only consider the results of the one vehicle type. The combination of subsets having a widely varying number of images and the fact that some subsets are visually more similar to each other than others makes performance difficult to assess when training.

**Table 2.1**
Number of images in each RGB subset.

| Subset | Images | Percentage | Notes |
|---|---|---|---|
| Location1 | 377 | 34.56% | Target is stationary |
| Location2 | 117 | 10.72% | Target is stationary |
| Location3 | 47 | 4.31% | Target is stationary and often occluded |
| Location4 | 81 | 7.42% | Target is stationary and often occluded |
| Route1 | 17 | 1.56% | Target is moving |
| Route2 | 77 | 7.06% | Target is moving |
| FlyAway | 183 | 16.77% | Target is moving |
| | | | Wide range of target scales |
| | | | Constant camera orientation |
| FlyToward | 192 | 17.60% | Target is stationary |
| | | | Same environment as FlyAway |
| | | | Constant camera orientation |

**Table 2.2**
Number of images in each LWIR subset.

| Subset | Images | Percentage | Notes |
|---|---|---|---|
| Location1 | 270 | 17.85% | Target is stationary |
| Location2 | 132 | 8.72% | Target is stationary |
| Location3 | 140 | 9.25% | Target is stationary and often occluded |
| Location4 | 131 | 8.66% | Target is stationary and often occluded |
| Route1 | 123 | 8.13% | Target is moving |
| Route2 | 178 | 11.76% | Target is moving |
| FlyAway | 166 | 10.97% | Target is moving |
| | | | Wide range of target scales |
| | | | Constant camera orientation |
| FlyToward | 191 | 12.62% | Target is stationary |
| | | | Same environment as FlyAway |
| | | | Constant camera orientation |
| Stage4 | 182 | 12.03% | Target is moving |
| | | | Same environment as Route2 |

# Chapter 3

# Spin-Set Data Augmentation

Spin-set training consists of generating synthetic training data by superimposing rendered images of desired target classes onto frames of videos related to expected locations. In order to generate training data in this way, two resources are needed: target spin-sets and background videos. We first rendered 3D models of target vehicles (both RGB and LWIR) from 72 pitch angles encompassing a 360 degree rotation each with nine different yaw angles (between 45 and 90 degrees) for a total of 648 rendered images per vehicle. The angles were chosen to be representative of targets seen from an overhead perspective. These images were created with transparent backgrounds so that they could easily be added to images, as shown in Figures 3.1 and 3.2. Individual frames from YouTube videos were automatically extracted to be used as backgrounds for the generated imagery, with the backgrounds being converted to

(a) Generated image (b) Generated image

**Figure 3.1:** Examples of generated RGB images.

grayscale for LWIR images. These videos were automatically chosen based on user-provided keywords describing the expected environments, which we describe next. This form of augmentation certainly does not generate the most realistic synthetic imagery; however, it does benefit from being both significantly easier to set up and very efficient to produce compared to other forms of synthetic data such as rendering full 3D scenes. Our technique is also easily generalizable, as one would only have to replace the target images and change what environments of which the backgrounds should consist.

(a) Generated image (b) Generated image

**Figure 3.2:** Examples of generated LWIR images.

## 3.1 Automated Background Selection

Ideally, object detection should be effective regardless of the environment. We designed a tool that automatically uses screenshots from YouTube videos as backgrounds for synthetic training data as a form of domain randomization [11]. The user must enter lists of keywords that they would use to search to find videos of relevant environments. For example, "overhead footage nature," "overhead footage wilderness," and "overhead footage forest" could be useful search terms in the case of overhead object detection in forested regions. The tool collects the first ten (by default) video links for each search term list. When generating an image, a random $N \times N$ crop of a random frame from a randomly chosen video is extracted for the background.

(a) Relevant background (b) Relevant background

**Figure 3.3:** Examples of relevant backgrounds extracted.

This background selection method is an easy and quick way to generate a large amount and variety of different backgrounds. This made generating training data very quick and cheap, and is much less tedious than other methods, such as creating 3D scenes of environments by scratch. Furthermore, this usually provides acceptable backgrounds such as those shown in Figure 3.3. By increasing the number of videos searched for, we can greatly increase the variety of backgrounds, though as this number increases the relevance of videos used can decrease.

This method is not without drawbacks, however. If search terms are not chosen carefully, the tool could easily use highly irrelevant videos, such as those shown in Figure 3.4. These screenshots are from videos found when we used keyword searches such as "drone nature aerial forest," "drone nature overhead trees," "drone footage overhead wilderness," and similar terms. This issue could be mitigated by skimming

(a) Irrelevant background (b) Irrelevant background

**Figure 3.4:** Examples of irrelevant backgrounds created. Search terms should be carefully chosen to reduce the incidence of unsuitable backgrounds.

through videos before they are used in image generation, however this does reduce the level of automation that the method offers. Additionally, even highly relevant videos may have sections that are not useful for image generation. An example of this would be a nature documentary: there are many video sections that could be very useful for simulating outdoor environments; however, there may be scenes of people talking indoors that would not be suitable.

## 3.2 Synthetic Data Generation

When generating training images, many effects are applied to further augment the data and perform domain randomization [11]: discoloration, graininess, blurring, scaling, occlusion, brightness, and contrast. These effects are meant to simulate how a target may be seen from a camera in various conditions. For each effect $e$, we define a minimum strength $e_{min}$ and maximum strength $e_{max}$, as well as a power parameter $e_{pow}$. Using these parameters, we can define the strength $s_e$ of $e$ for a single generated image as

$$s_e = x^{e_{pow}} \cdot (e_{max} - e_{min}) + e_{min}, \tag{3.1}$$

where $x$ is a uniform random real value between 0 and 1. This enables easy adjustment of each effect, increasing or decreasing the range of augmentation from an effect. This is particularly useful for effects such as blurring, where we would like to be able to detect targets from a blurry image but most images are expected to be fairly crisp. By setting $e_{pow}$ to a high value, greater than one, most training images will have blurring closer to the minimum strength. Additionally, we define a probability $e_{prob}$ that an effect will be applied. The parameters for each effect we used in this study are listed in Table 3.1.

**Table 3.1**
Parameters used for image generation.

| Effect | Probability | Min | Max | Power |
|---|---|---|---|---|
| Graininess | 0.8 | 0.00 | 0.12 | 2.00 |
| Blur | 0.5 | 0.00 | 1.00 | 2.00 |
| Discoloration | 0.8 | 0.00 | 0.12 | 2.00 |
| Brightness | 1.0 | -0.30 | 0.30 | 1.0 |
| Contrast | 1.0 | -0.30 | 0.30 | 1.0 |
| #Occlusions | 0.7 | 1 | 10 | 1.0 |
| Occlusion | N/A | 0.10 | 0.50 | 0.5 |

### 3.2.1 Graininess

For each RGB pixel in the generated image, denoted $p \in [0,1]^3$, we adjust the pixel to $p'$, which is determined by linearly interpolating between $p$ and a randomly chosen pixel value using $s_{graininess}$. We use this effect to model noise that could appear in images taken with a camera. We decided to implement this version of noise because it was quick to develop, provided fast image generation, and had visually acceptable results. It is entirely possible that other forms of noise such as salt and pepper noise or perhaps Poisson-Gaussian noise used by Carlson [21] would be more effective and further tests should be done to determine this.

### 3.2.2 Blur

A Gaussian filter is applied over the entire image. The standard deviation of this filter is set to $s_{blur}$, used for the blur strength. The filter is applied to account for two phenomena. The first is to simulate an unfocused lens. The second is to simulate motion blur. There is motion blur in a select few of our real-world images and it is likely to occur in similar applications. Though this model is not an accurate representation of directional motion blur, it is an easy way to augment the training data.

### 3.2.3 Discoloration

This effect is applied to target images before being added to the background image. A random vector is chosen in $[-1, 1]^3$, normalized, and then scaled such that the magnitude is equal to $s_{discoloration}$. This vector is then added to every pixel in the target image and then the pixels are clamped between 0 and 1. This effect was added to account for changes in lighting, as our object detection models should be performant regardless of color temperature.

### 3.2.4 Brightness and Contrast

Our real-world images contain a wide variety of brightness levels, and this is likely to be the case for similar outdoor applications. There are a few reasons for this such as camera quality, exposure time, and sky brightness. Every pixel in the generated image $p$ is altered such that $p' = p \cdot (2s_{contrast} + 1) - s_{contrast} + s_{brightness}$. Alterations were added to these properties to account for differences in environmental lighting as well as camera exposure.

### 3.2.5 Occlusion

Occlusions are common, particularly in a natural environment where trees or other objects could block parts of an object. To counter this, we implemented rudimentary occlusions in the form of blocked out regions of the target. This has the additional benefit of discouraging models from focusing on a specific part of a vehicle to recognize it. Occlusions were added to generated images by erasing randomly shaped rectangular sections of target images before adding them to the background. We first determine the number of occlusions to apply, which is equal to $\lfloor s_{numOcclusion} \rfloor$. The width and height of the rectangular region are determined with $s_{occlusion_1} \cdot width$ and $s_{occlusion_2} \cdot height$, respectively. Finally, the center of the region is chosen in a uniform

random fashion, within the target bounding box.

### 3.2.6 Bounding Box Generation

Automatically determining a bounding box for a target using this technique is quite easy. After all effects have been applied to a target image, we determine the top-, bottom-, left-, and right-most pixels that are not fully transparent. When adding the image to the video frame background, we keep track of where those four pixels end up on the final image. These pixels end up determining the top-left and bottom-right corner. From there it is easy enough to transform this information into whatever format is needed for training (such as YOLO format).

# Chapter 4

# Experiments

Experimentation involved combining our real imagery with varying numbers of spin-set images into one training data set. For each subset of measured images, spin-set images were added. The spin-set images were added in quantities of 0, 500, 1,000, and 5,000 for RGB images. Our original publication involved only RGB images. Later when LWIR images were considered this was changed to percentages relative to the total number of measured images in each subset ranging from 5% to 300%.

These augmented data sets were then used to train our various models. The model performances were compared to that of training with real-world images only. We used two methods of training: single subset and all-but-one subset, although only the single subset method was used for RGB images. In the single subset method,

object detection models are trained on one subset (and synthetic augmentations), validated with another subset, and tested on the remaining seven subsets. This was done seven times for RGB images and eight times for LWIR images for each training set such that every remaining subset was used for validation once. One benefit to this method was that we could train one model for each subset and then retroactively decide when to stop training and what the best detection threshold was based on each validation set, allowing us to save significant amounts of time in exchange for frequently saving network weights to storage to be accessed later. The all-but-one subset method involved training on 90% of eight subsets (as only LWIR images were used), validating the network on the remaining 10% of those subsets, and testing on the one subset left out. This was then repeated such that every subset had a turn as the testing set. This allowed us to repeat the experiment up to 10 times, although due to time constraints only 5 folds were used.

# Chapter 5

# Results

Our original publication included results for RGB images using YOLO (specifically YOLOv4) with the single subset method. Later work added results for LWIR images using YOLO, SSD, and Faster R-CNN with both the single subset method as well as the all-but-one method.

## 5.1 RGB Imagery

We trained YOLO [22, 23] models using one subset of our data set for training, one for validation, and the remaining subsets for testing. We then repeated this experiment for each subset as the training data. All models were trained for up to 40,000 batches

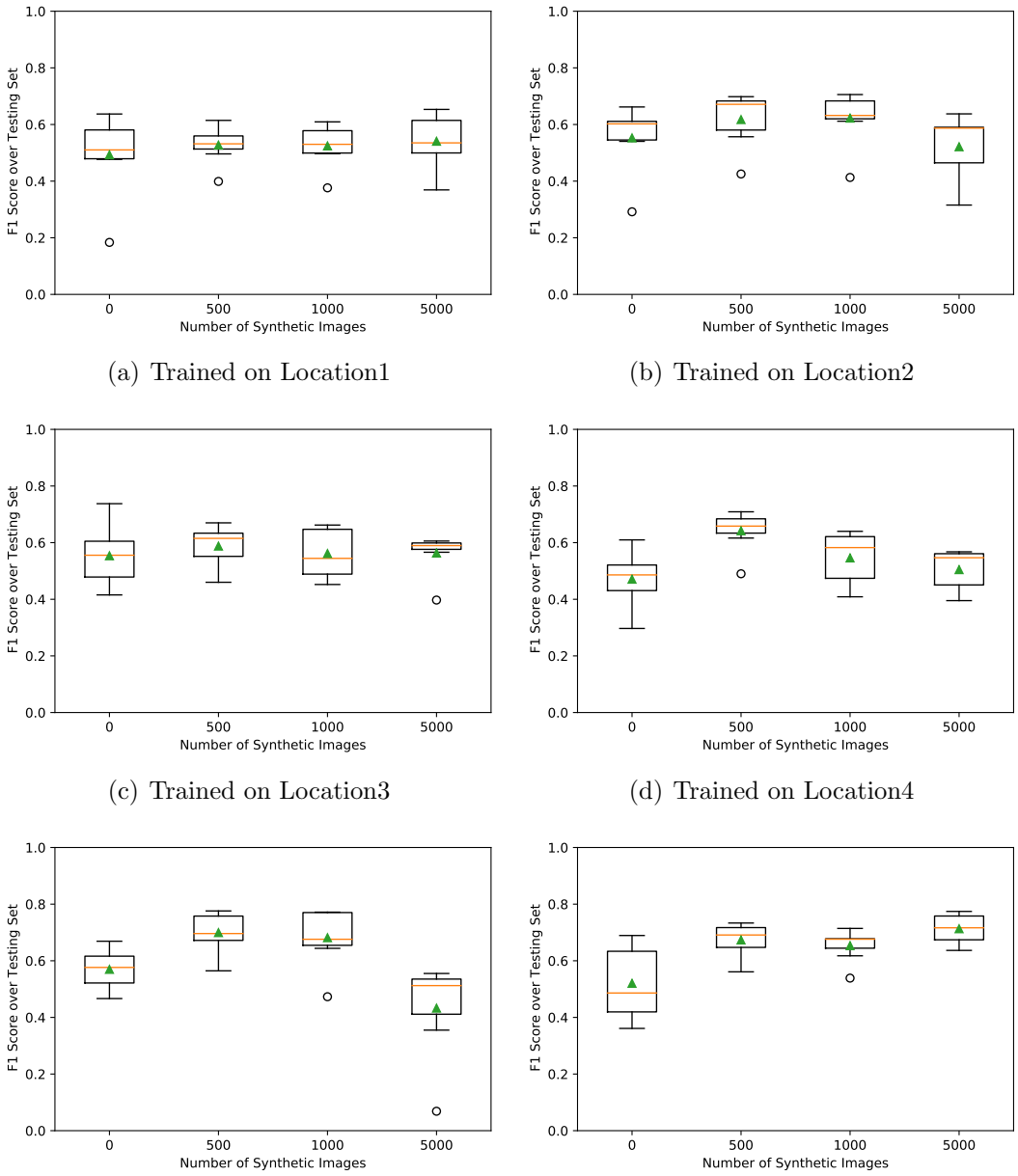
of 16 images each. It should be noted that in addition to our data generation, the minimal built-in data augmentation of Darknet [1] was used when training YOLO [22, 23] models. $F_1$ scores for the respective testing subsets are shown in Tables 5.1, 5.2, 5.3, and 5.4. Additionally, we trained models with synthetic data only, whose results are included in the "None" column of each table.

Our experiments indicated that including a small number of spin-set images provided a benefit, but increasing the number of spin-set images too much decreased detection performance. Our conjecture is that generated data were dissimilar enough from the measured data that models would overfit on synthetic imagery when a large enough proportion of the training set was synthetic. We can see in Figure 5.1 that including 500 synthetic images consistently improves performance regardless of which subset is used for training. However, the amount of increase varies noticeably depending on the training subset, and the trend is not consistent once 1,000 images are used, and including 5,000 synthetic images tends to be detrimental to performance. Spin-set augmentation most notably improved results when training on Location4, Route1, Route2, and FlyToward; whereas Location1, Location2, Location3, and FlyAway had minimal improvement. Our experiments also showed that synthetic imagery alone was not sufficient to obtain reasonable performance.

It is difficult to know for sure why certain subsets were improved by the augmentation more than others, though there are some subsets where we can reasonably speculate

the reason. Our most surprising result was how little augmentation improved FlyAway, when the images were so similar to those of FlyToward. Both subsets were from the same environment and the camera's orientation was relatively constant in both as it moved backwards or forwards, respectively. There was, however, one key difference: the target moved in circles in FlyAway whereas the target was stationary in FlyToward. This meant that FlyAway had images of the target from many more angles compared to FlyToward. We suspect that because of this, augmentation was much more helpful when training on FlyToward since it provided angles that would not have otherwise been trained on. This cannot be the only reason, however, since the target was visible from many angles in the other subsets and yet some improved significantly while others did not. This is the only noticeable difference between the two subsets, though, so it is likely to be a factor in determining performance increase.

Overall, these results can be summarized by the statement that including a relatively small number of synthetic *spin-set* images in the training set improves detection performance. Care should be taken that the algorithm does not overtrain on the synthetic imagery, which could be accomplished by ensuring that the proportion of real-world and synthetic imagery is kept roughly equal.




(a) Trained on Location1

(b) Trained on Location2

(c) Trained on Location3

(d) Trained on Location4

(e) Trained on Route1

(f) Trained on Route2

**Figure 5.1:** Testing set $F_1$ scores for each RGB subset. Orange bar indicates the median and the green triangle indicates the mean; box and whiskers indicate quartiles and max/min, respectively. More figures are shown later.

**Table 5.1**
$F_1$ score for training set and validation set with 0 synthetic generated images.

| **Valid.** | **Training Set** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Loc1 | Loc2 | Loc3 | Loc4 | Route1 | Route2 | Toward | Away | None |
| Loc1 | - | 0.292 | 0.415 | 0.297 | 0.496 | 0.460 | 0.483 | 0.657 | - |
| Loc2 | 0.184 | - | 0.466 | 0.525 | 0.467 | 0.361 | 0.638 | 0.524 | - |
| Loc3 | 0.481 | 0.603 | - | 0.486 | 0.610 | 0.628 | 0.676 | 0.702 | - |
| Loc4 | 0.477 | 0.550 | 0.491 | - | 0.623 | 0.379 | 0.731 | 0.632 | - |
| Route1 | 0.534 | 0.619 | 0.555 | 0.465 | - | 0.486 | 0.518 | 0.629 | - |
| Route2 | 0.510 | 0.540 | 0.569 | 0.517 | 0.547 | - | 0.762 | 0.692 | - |
| Toward | 0.627 | 0.602 | 0.738 | 0.610 | 0.669 | 0.689 | - | 0.661 | - |
| Away | 0.637 | 0.662 | 0.641 | 0.396 | 0.577 | 0.639 | 0.762 | - | - |
| Mean | 0.493 | 0.553 | 0.554 | 0.471 | 0.570 | 0.520 | 0.653 | 0.643 | - |
| StdDev | 0.140 | 0.113 | 0.102 | 0.093 | 0.066 | 0.122 | 0.105 | 0.055 | - |

**Table 5.2**
$F_1$ score for training set and validation set with 500 synthetic generated images.

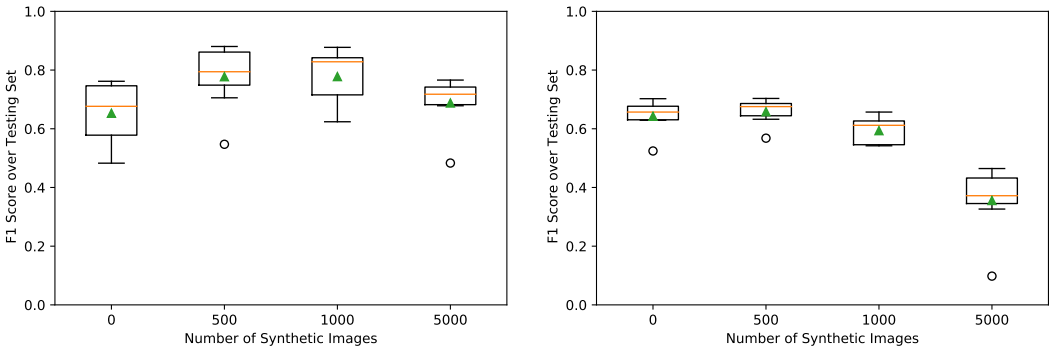
| **Valid.** | **Training Set** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Loc1 | Loc2 | Loc3 | Loc4 | Route1 | Route2 | Toward | Away | None |
| Loc1 | - | 0.425 | 0.460 | 0.490 | 0.651 | 0.561 | 0.705 | 0.632 | 0.361 |
| Loc2 | 0.399 | - | 0.617 | 0.651 | 0.693 | 0.645 | 0.792 | 0.675 | 0.171 |
| Loc3 | 0.532 | 0.698 | - | 0.616 | 0.758 | 0.705 | 0.854 | 0.686 | 0.411 |
| Loc4 | 0.530 | 0.674 | 0.650 | - | 0.776 | 0.730 | 0.880 | 0.656 | 0.444 |
| Route1 | 0.496 | 0.671 | 0.615 | 0.686 | - | 0.649 | 0.547 | 0.703 | 0.400 |
| Route2 | 0.532 | 0.693 | 0.570 | 0.658 | 0.696 | - | 0.868 | 0.686 | 0.165 |
| Toward | 0.587 | 0.604 | 0.670 | 0.683 | 0.565 | 0.733 | - | 0.568 | 0.428 |
| Away | 0.615 | 0.557 | 0.533 | 0.709 | 0.757 | 0.691 | 0.794 | - | 0.432 |
| Mean | 0.527 | 0.617 | 0.588 | 0.642 | 0.699 | 0.673 | 0.777 | 0.658 | 0.352 |
| StdDev | 0.064 | 0.092 | 0.067 | 0.068 | 0.069 | 0.056 | 0.109 | 0.043 | 0.108 |

**Table 5.3**
$F_1$ score for training set and validation set with 1000 synthetic generated images.

| **Valid.** | **Training Set** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Loc1 | Loc2 | Loc3 | Loc4 | Route1 | Route2 | Toward | Away | None |
| Loc1 | - | 0.413 | 0.452 | 0.409 | 0.644 | 0.539 | 0.764 | 0.612 | 0.298 |
| Loc2 | 0.376 | - | 0.633 | 0.612 | 0.665 | 0.671 | 0.830 | 0.615 | 0.193 |
| Loc3 | 0.497 | 0.688 | - | 0.631 | 0.771 | 0.714 | 0.854 | 0.542 | 0.348 |
| Loc4 | 0.530 | 0.706 | 0.661 | - | 0.772 | 0.618 | 0.877 | 0.542 | 0.338 |
| Route1 | 0.502 | 0.629 | 0.513 | 0.462 | - | 0.676 | 0.667 | 0.549 | 0.332 |
| Route2 | 0.565 | 0.632 | 0.544 | 0.640 | 0.769 | - | 0.624 | 0.657 | 0.324 |
| Toward | 0.609 | 0.679 | 0.662 | 0.582 | 0.473 | 0.678 | - | 0.638 | 0.284 |
| Away | 0.591 | 0.612 | 0.464 | 0.486 | 0.676 | 0.678 | 0.828 | - | 0.301 |
| Mean | 0.524 | 0.622 | 0.562 | 0.546 | 0.682 | 0.653 | 0.778 | 0.594 | 0.302 |
| StdDev | 0.072 | 0.091 | 0.084 | 0.085 | 0.099 | 0.054 | 0.090 | 0.045 | 0.046 |

**Table 5.4**
$F_1$ score for training set and validation set with 5000 synthetic generated images.

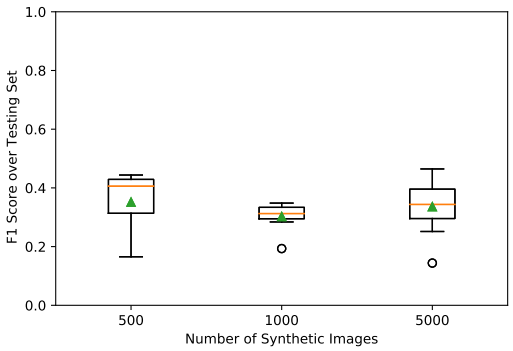
| **Valid.** | **Training Set** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Loc1 | Loc2 | Loc3 | Loc4 | Route1 | Route2 | Toward | Away | None |
| Loc1 | - | 0.316 | 0.397 | 0.395 | 0.355 | 0.662 | 0.686 | 0.434 | 0.377 |
| Loc2 | 0.369 | - | 0.566 | 0.441 | 0.467 | 0.686 | 0.718 | 0.098 | 0.144 |
| Loc3 | 0.535 | 0.589 | - | 0.547 | 0.541 | 0.716 | 0.759 | 0.364 | 0.340 |
| Loc4 | 0.589 | 0.587 | 0.601 | - | 0.555 | 0.747 | 0.766 | 0.464 | 0.347 |
| Route1 | 0.497 | 0.408 | 0.590 | 0.461 | - | 0.637 | 0.678 | 0.326 | 0.310 |
| Route2 | 0.502 | 0.521 | 0.606 | 0.568 | 0.530 | - | 0.725 | 0.372 | 0.251 |
| Toward | 0.640 | 0.591 | 0.587 | 0.555 | 0.069 | 0.774 | - | 0.430 | 0.465 |
| Away | 0.653 | 0.637 | 0.597 | 0.567 | 0.512 | 0.770 | 0.483 | - | 0.454 |
| Mean | 0.541 | 0.521 | 0.563 | 0.505 | 0.433 | 0.713 | 0.688 | 0.356 | 0.336 |
| StdDev | 0.091 | 0.109 | 0.069 | 0.066 | 0.161 | 0.050 | 0.089 | 0.114 | 0.098 |




(a) Trained on FlyToward




(b) Trained on FlyAway




(c) Trained on synthetic only

**Figure 5.1:** Testing set $F_1$ scores for each RGB subset. Orange bar indicates the median and the green triangle indicates the mean; box and whiskers indicate quartiles and max/min, respectively. Final figures are shown here.

## 5.2 LWIR Imagery

We trained Faster R-CNN, SSD, and YOLO with the single subset method. For Faster R-CNN and SSD (available through TensorFlow 2 Detection Model Zoo), weights pretrained on the COCO 2017 data set and built-in data augmentation of TensorFlow's object detection API were used in addition to our synthetic augmentations [24, 25]. It should be noted that these weights were trained using RGB images and were not originally intended for LWIR use. For YOLO, the built-in data augmentations of Darknet were used and the models were trained from scratch. Due to time constraints, only Faster R-CNN and SSD were used when training with the all-but-one method, which used the same pretrained weights and data augmentation as in the single subset method. Figures for the results for the single subset method trained with Faster R-CNN are shown, and the remaining figures are available in the appendix.

Our experiments indicated that similarly to RGB spin-set augmentation, LWIR spin-sets were beneficial to data sets with a small number of images. We found that while the most effective number of spin-set images to include varies depending on the architecture and base training set images, it was beneficial for all architectures over all subsets in the single subset tests. Including spin-set images for Faster R-CNN in the range of 10% to 50% of the training set size was the most beneficial; including any more than that began to reduce the effectiveness in the majority of subsets. Still,

**Figure 5.2:** Results of single subset method trained with Faster R-CNN using LWIR spin-set augmentation.

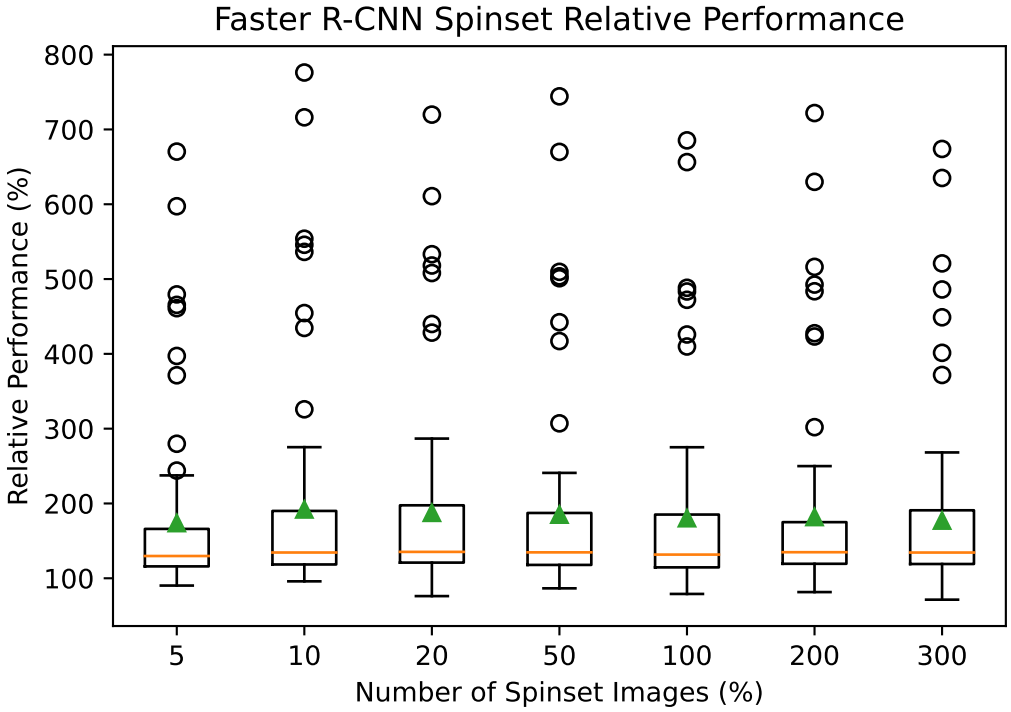



**Figure 5.3:** Performance increase of single subset method trained with Faster R-CNN relative to results without LWIR spin-set augmentation.

even including as little as 5% provided dramatic boosts to $F_1$ scores. The results for SSD and YOLO, however, differ quite dramatically from Faster R-CNN. Both architectures perform much worse without spin-sets, even SSD with its pretrained weights. We see a more gradual increase in performance with SSD and YOLO as more spin-set images are used for training. The models continued to improve with as much as 300% spin-sets, where the experiments were cut off and it is likely that including more than 300% spin-sets may produce even better results. This shows that spin-set images used for augmentation are beneficial for small data sets, but precisely how many images are necessary for optimal performance is highly dependent on the

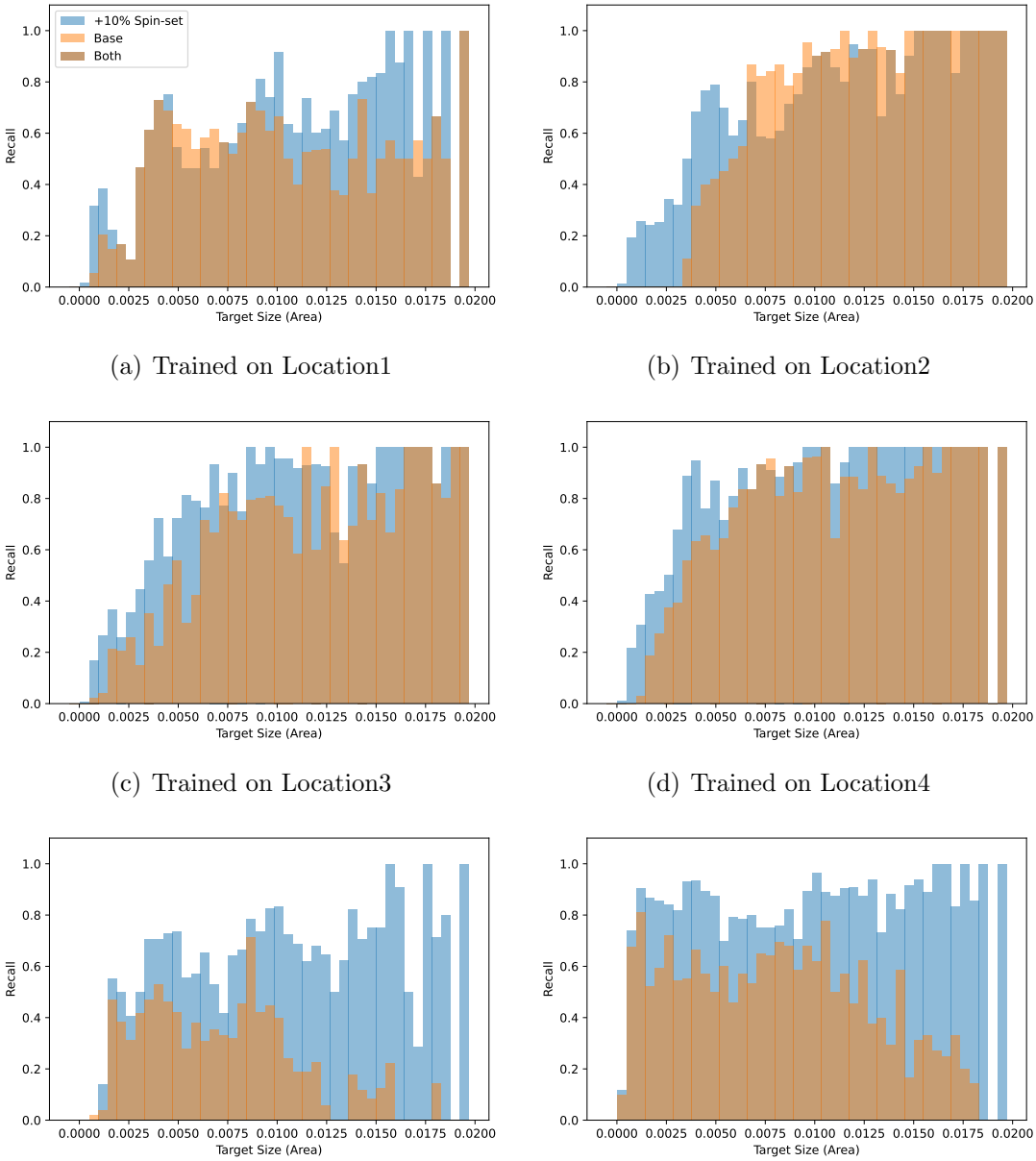
model architecture.

For LWIR imagery, we also performed the all-but-one method, which involved training with eight out of nine subsets and testing on the single remaining subset. For each test, we used Faster R-CNN and SSD each to train two models: one with no augmentation and one with a large but fixed number (500) of spin-set images. We found that spin-set augmentation had much more inconclusive results for these experiments. There were subsets where spin-set augmentation significantly improved performance, significantly decreased performance, and some where it remained consistent (see Figures A.22 and A.23). It should be noted that the charts generated for the all-but-one method cannot be directly compared to those created from the single-subset method results as the $F_1$ scores are calculated from different sets of images; the single-subset models were tested on eight subsets (all subsets other than the one they were trained and validated on) while the all-but-one models were tested on one subset. Nevertheless, the detection rates are certainly nowhere near as dramatic overall.

To gather a greater understanding of how spin-set augmentation affected various object detection networks, we further compared the detection rate of differing target sizes. For each architecture in the single subset method, we compared the base model to the model trained with the highest performing number of spin-sets (10% for Faster R-CNN, 200% for SSD, and 300% for YOLO) over all images not in the training subset. We found that over all subsets nearly every target size sees an improved

detection rate, with a few exceptions such as mid-size targets when Faster R-CNN is trained on Location 2 and small targets when YOLO is trained with Route 1 and 2 (Figures 5.4, A.24, and A.25). This shows that, in general, synthetic spin-set augmentation is largely beneficial over all scales for object detection when data sets are extremely limited and improve the efficacy when testing over a range of different testing environments.

Interestingly, all-but-one subset target size results diverge from these previous results. Both Faster R-CNN and SSD have similar trends. In two testing environments (Location 1 and 3) spin-set augmentation improves detection rate while one environment (Location 4) has a significant decline, with smaller changes in the other subsets (Figures A.26 and A.27).




(a) Trained on Location1

(b) Trained on Location2

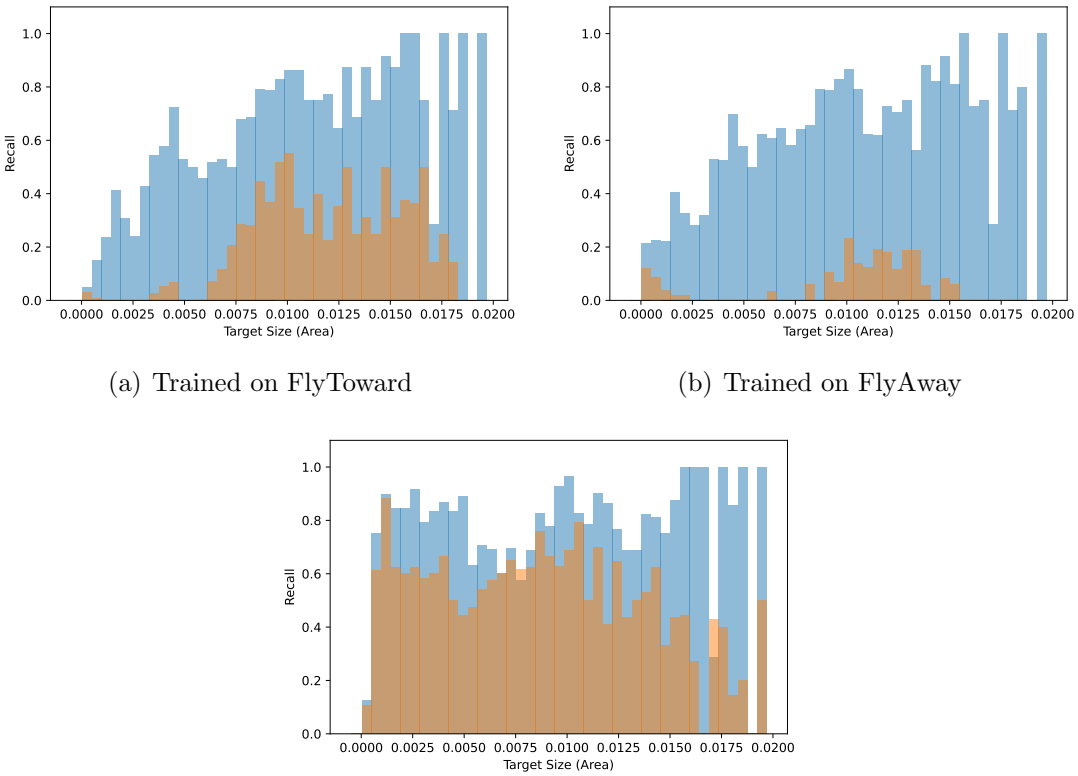
(c) Trained on Location3

(d) Trained on Location4

(e) Trained on Route1

(f) Trained on Route2

**Figure 5.4:** Object recall over different target sizes for Faster R-CNN trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page.




(a) Trained on FlyToward

(b) Trained on FlyAway

(c) Trained on Stage4

**Figure 5.4:** Object recall over different target sizes for Faster R-CNN trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page.

# Chapter 6

# Conclusion

Simulating everything in real-world environments is very difficult. Spin-set augmentation is easy to implement and quick to generate. However, it can be difficult to create imagery that is close enough to true photos. We performed experiments with the YOLO [23], SSD [26], and Faster R-CNN [27] object detection algorithms for overhead detection of a target in varied environments and situations. We introduced spin-set data augmentation, which creates images by choosing relevant background images and superimposing rendered targets onto the image. This has the benefit of being extremely computationally cheap to produce. The results indicate that there is a performance boost from including spin-set augmentation imagery in training. However, the proportion between the number of real-world training images and the synthetic images must be taken into account.

There are many paths toward future work. The one most in need of attention would be determining the relative utility each of the image augmentation effects. It was beyond the scope of this thesis to test each effect individually. It is possible that certain combinations are more useful than using all effects, and some effects may even be harmful.

# References


[1] Darknet: Open source neural networks in c. Redmon, J. http://pjreddie.com/darknet/, **2013–2016**.

[2] 4K drone footage - fall colors of Mountain Home Road, Leavenworth, WA state - 3 hour drone film. https://www.youtube.com/watch?v=Bu-8-gn0PEw.

[3] RISE - Oregon aerial nature | drone video | 4K UHD. https://www.youtube.com/watch?v=xviqK0uFx90.

[4] 4K drone footage - bird's eye view of Croatia, Europe - 3 hour ambient drone film. https://www.youtube.com/watch?v=HSsqzzuGTPo.

[5] Watch a breathtaking monarch butterfly swarm. https://www.youtube.com/watch?v=lWOySU_hAz0.

[6] Hamilton, N.; Webb, A.; DeKraker, Z.; Hendrickson, B.; Blanck, M.; Nelson, E.; Roemer, W.; Havens, T. C. In *Automatic Target Recognition XXXI*, Vol. 11729, pages 140 – 149. International Society for Optics and Photonics, 2021.

[7] Dold, J.; Groopman, J. *Geo-spatial information science* **2017**, *20*(2), 151–162.

[8] Mayer, N.; Ilg, E.; Fischer, P.; Hazirbas, C.; Cremers, D.; Dosovitskiy, A.; Brox, T. *International Journal of Computer Vision* **2018**, *126*(9), 942–960.

[9] Georgakis, G.; Mousavian, A.; Berg, A.; Kosecka, J. *ArXiv* **2017**, *abs/1702.07836*.

[10] Hartwig, S.; Ropinski, T. *ArXiv* **2019**, *abs/1904.00824*.

[11] Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; Abbeel, P. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.

[12] Schraml, D. In Rosenberger, M., Dittrich, P.-G., Zagar, B., Eds., *Photonics and Education in Measurement Science 2019*, Vol. 11144, pages 108 – 120. International Society for Optics and Photonics, SPIE, 2019.

[13] Wang, K.; Shi, F.; Wang, W.; Nan, Y.; Lian, S. *ArXiv* **2019**, *abs/1904.12294*.

[14] Barbosa, I. B.; Cristani, M.; Caputo, B.; Rognhaugen, A.; Theoharis, T. *Computer Vision and Image Understanding* **2018**, *167*, 50–62.

[15] Nikolenko, S. I. *ArXiv* **2019**, *1909.11512*.

[16] Struckmeier, O. *ArXiv* **2019**, *1905.13546*.

[17] Nowruzi, F. E.; Kapoor, P.; Kolhatkar, D.; Hassanat, F. A.; Laganière, R.; Rebut, J. *ArXiv* **2019**, *abs/1907.07061*.

[18] Xu, G.; Zhang, Y.; Zhang, Q.; Lin, G.; Wang, J. *ArXiv* **2017**, *abs/1709.08142*.

[19] Tremblay, J.; Prakash, A.; Acuna, D.; Brophy, M.; Jampani, V.; Anil, C.; To, T.; Cameracci, E.; Boochoon, S.; Birchfield, S. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* **2018**, pages 1082–10828.

[20] Prakash, A.; Boochoon, S.; Brophy, M.; Acuna, D.; Cameracci, E.; State, G.; Shapira, O.; Birchfield, S. *2019 International Conference on Robotics and Automation (ICRA)* **2019**, pages 7249–7255.

[21] Carlson, A.; Skinner, K. A.; Vasudevan, R.; Johnson-Roberson, M. In *ECCV Workshops*, 2018.

[22] Redmon, J.; Divvala, S.; Girshick, R. B.; Farhadi, A. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2016**, pages 779–788.

[23] Bochkovskiy, A.; Wang, C.-Y.; Liao, H. *ArXiv* **2020**, *abs/2004.10934*.

[24] Huang, J.; Rathod, V.; Sun, C.; Zhu, M.; Korattikara, A.; Fathi, A.; Fischer, I.; Wojna, Z.; Song, Y.; Guadarrama, S.; others. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

[25] Lin, T.; Maire, M.; Belongie, S. J.; Bourdev, L. D.; Girshick, R. B.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C. L. *CoRR* **2014**, *abs/1405.0312*.

[26] Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A. C. *Lecture Notes in Computer Science* **2016**, page 21–37.

[27] Ren, S.; He, K.; Girshick, R. B.; Sun, J. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*(6), 1137–1149.

[28] Ren, S.; He, K.; Girshick, R. B.; Sun, J. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2015**, *39*, 1137–1149.

# Appendix A

# Additional Figures

**Figure A.1:** Example RGB images from Location1.

**Figure A.2:** Example RGB images from Location2.

**Figure A.3:** Example RGB images from Location3.

**Figure A.4:** Example RGB images from Location4.

**Figure A.5:** Example RGB images from Route1.

**Figure A.6:** Example RGB images from Route2.

**Figure A.7:** Example RGB images from FlyToward.

**Figure A.8:** Example RGB images from FlyAway.

**Figure A.9:** Example LWIR images from Location1.

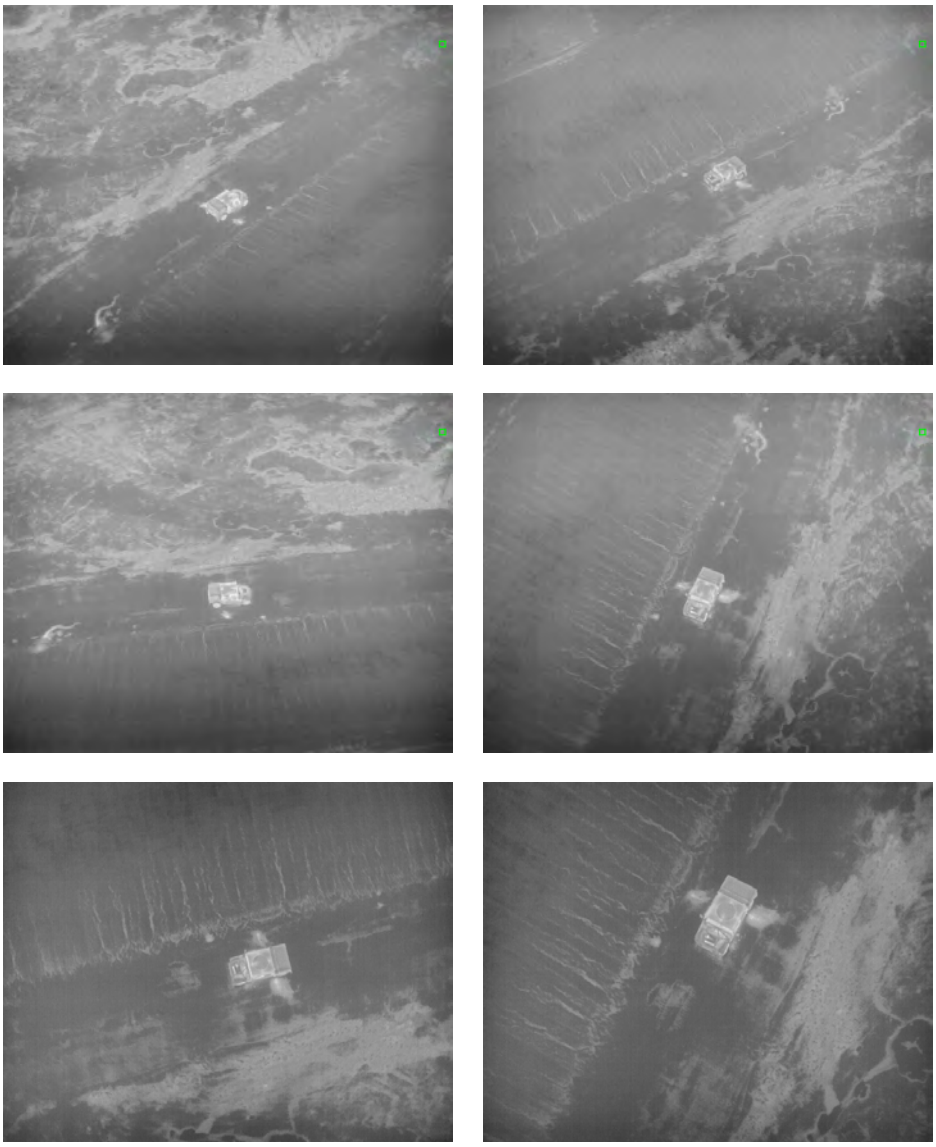


**Figure A.10:** Example LWIR images from Location2.

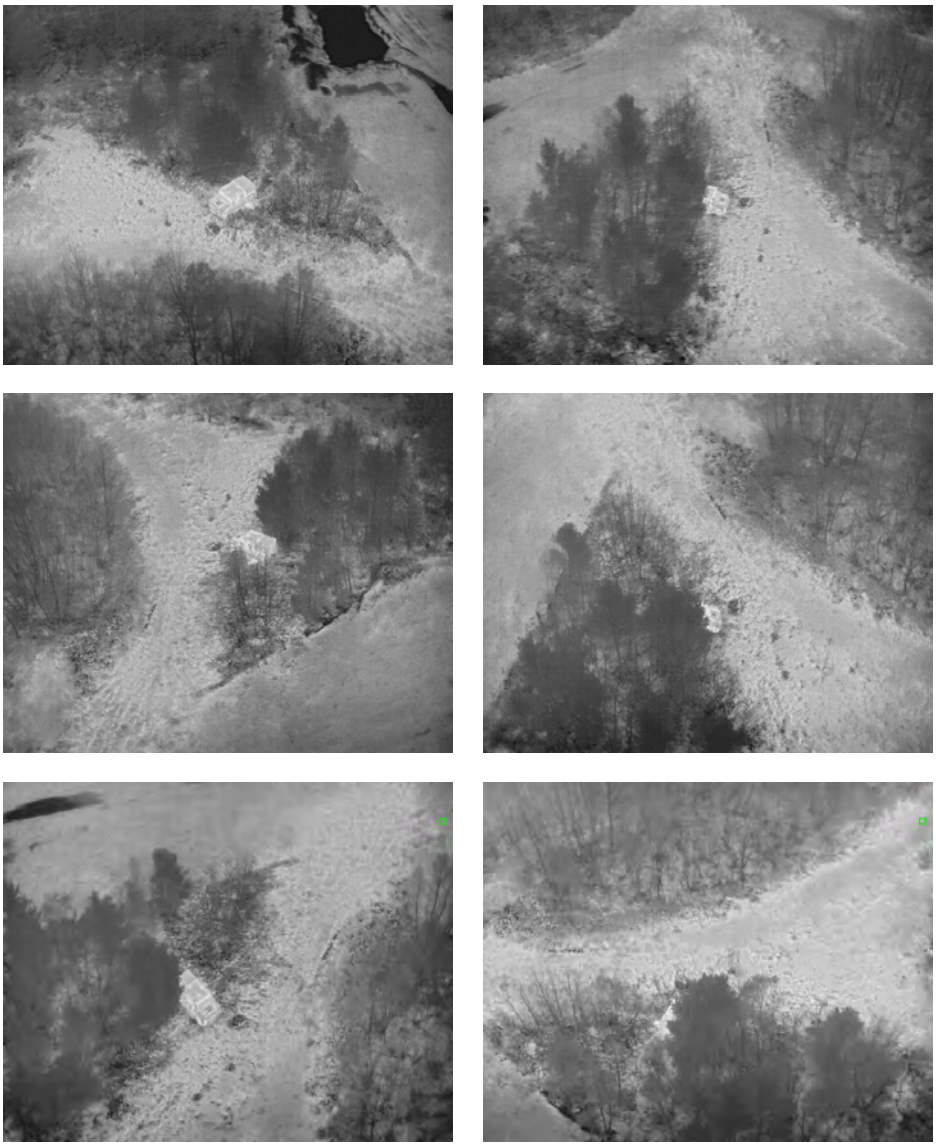


**Figure A.11:** Example LWIR images from Location3.

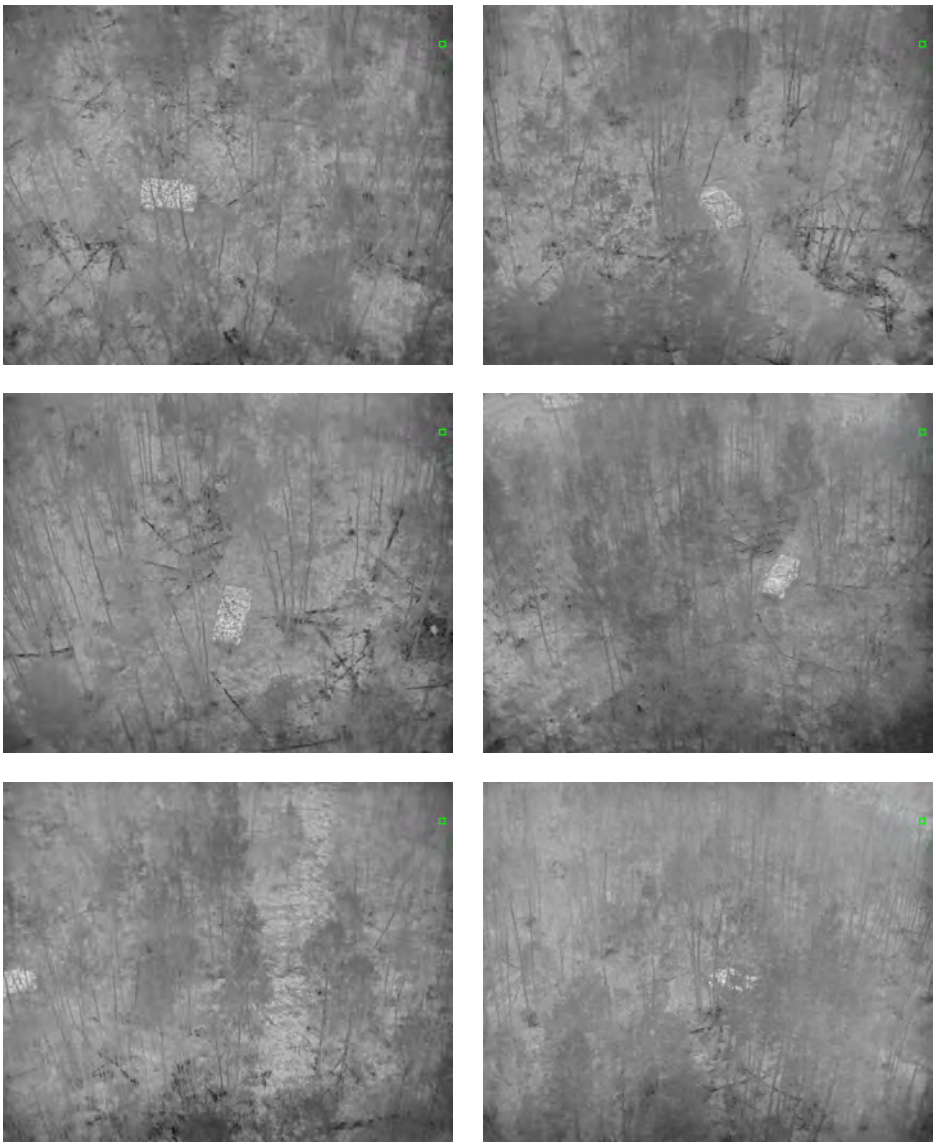


**Figure A.12:** Example LWIR images from Location4.

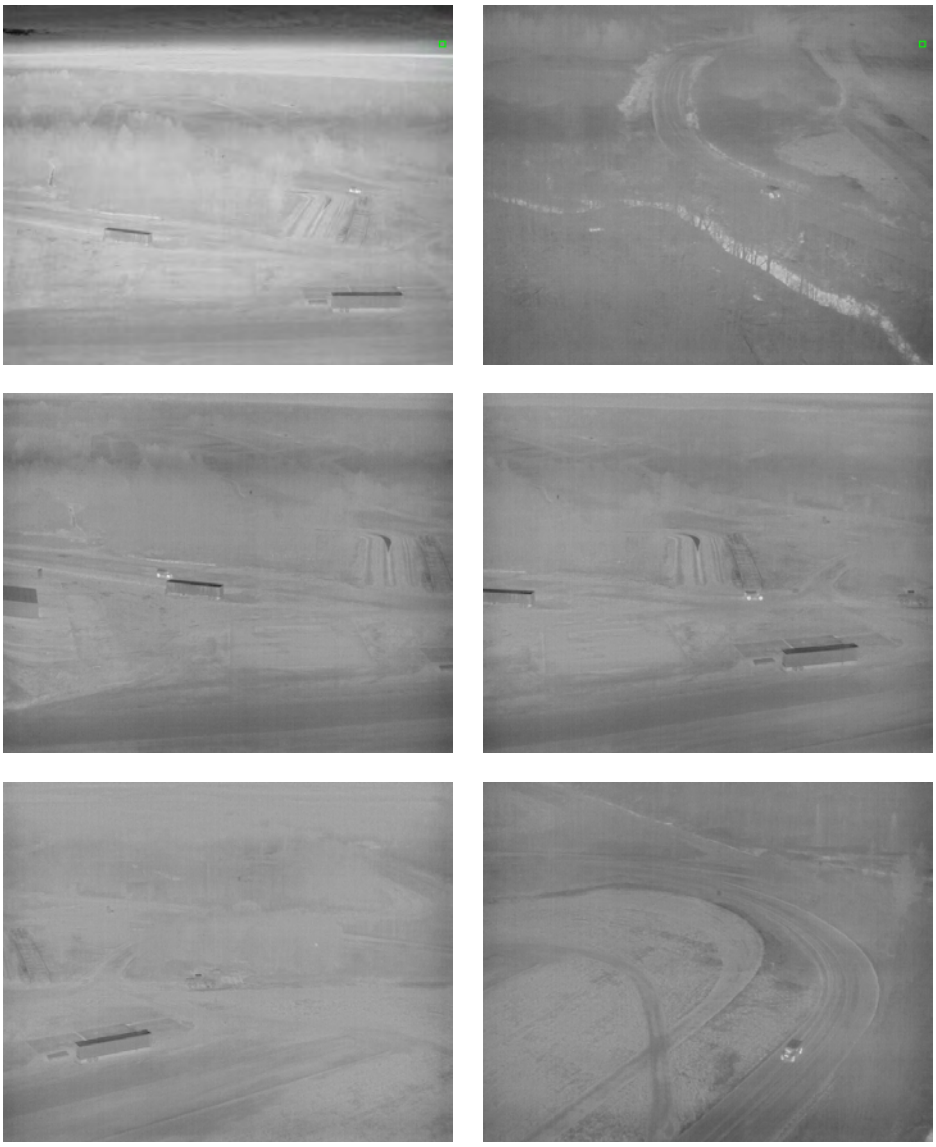


**Figure A.13:** Example LWIR images from Route1.

**Figure A.14:** Example LWIR images from Route2.

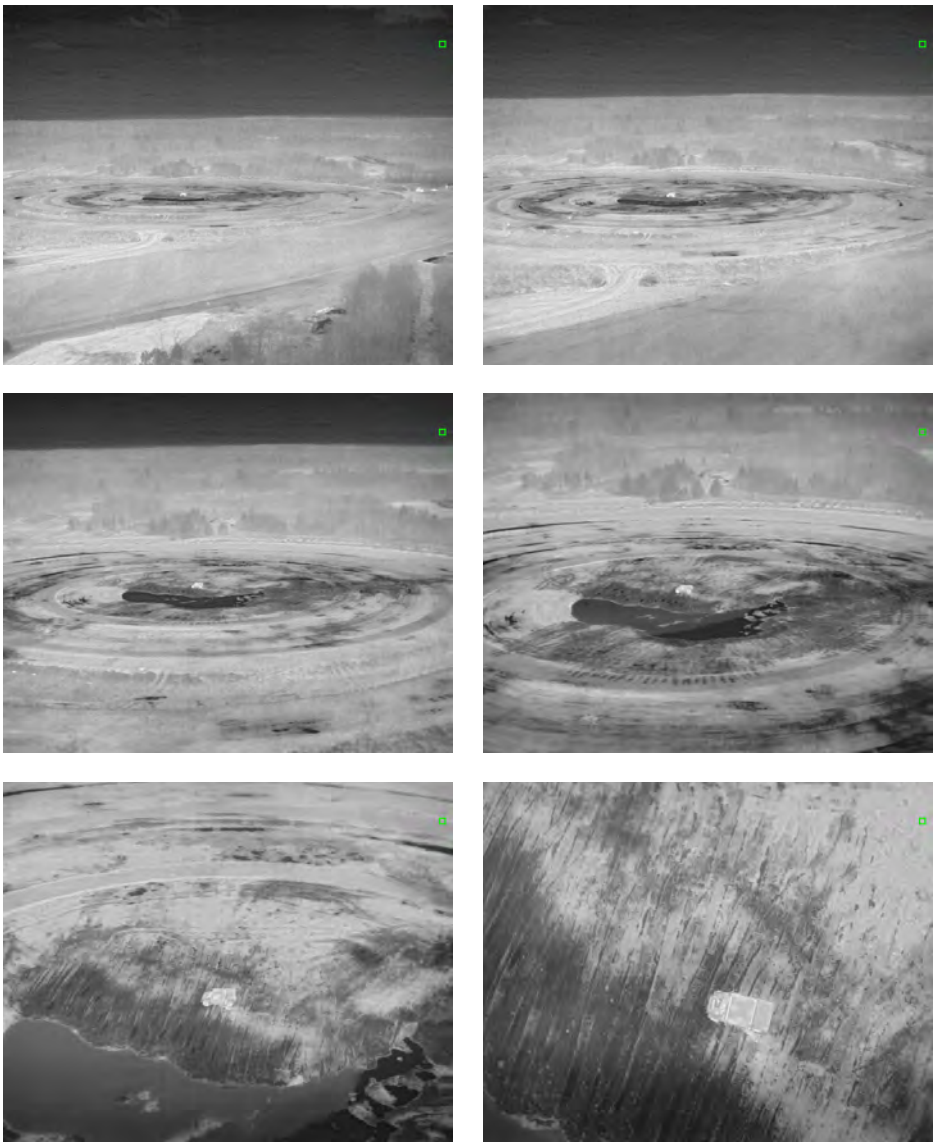


**Figure A.15:** Example LWIR images from FlyToward.

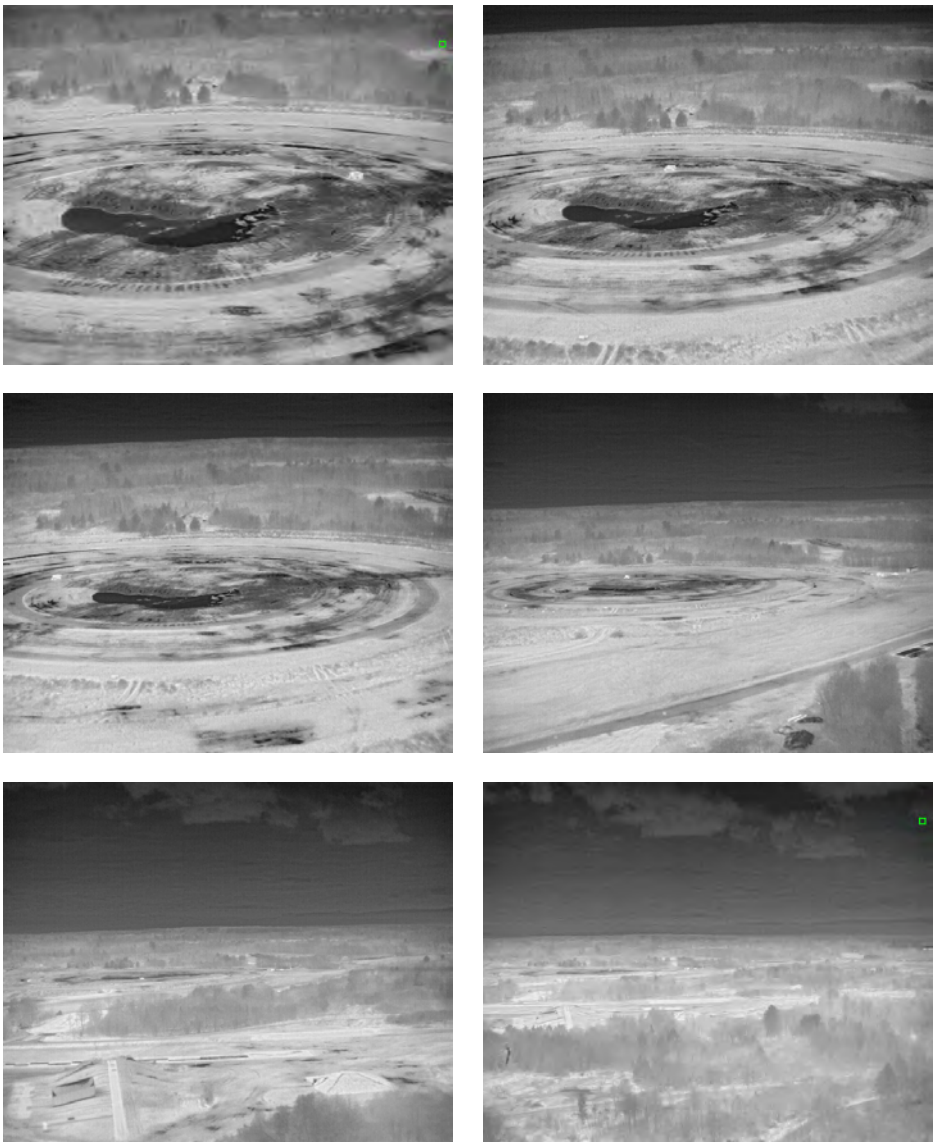


**Figure A.16:** Example LWIR images from FlyAway.

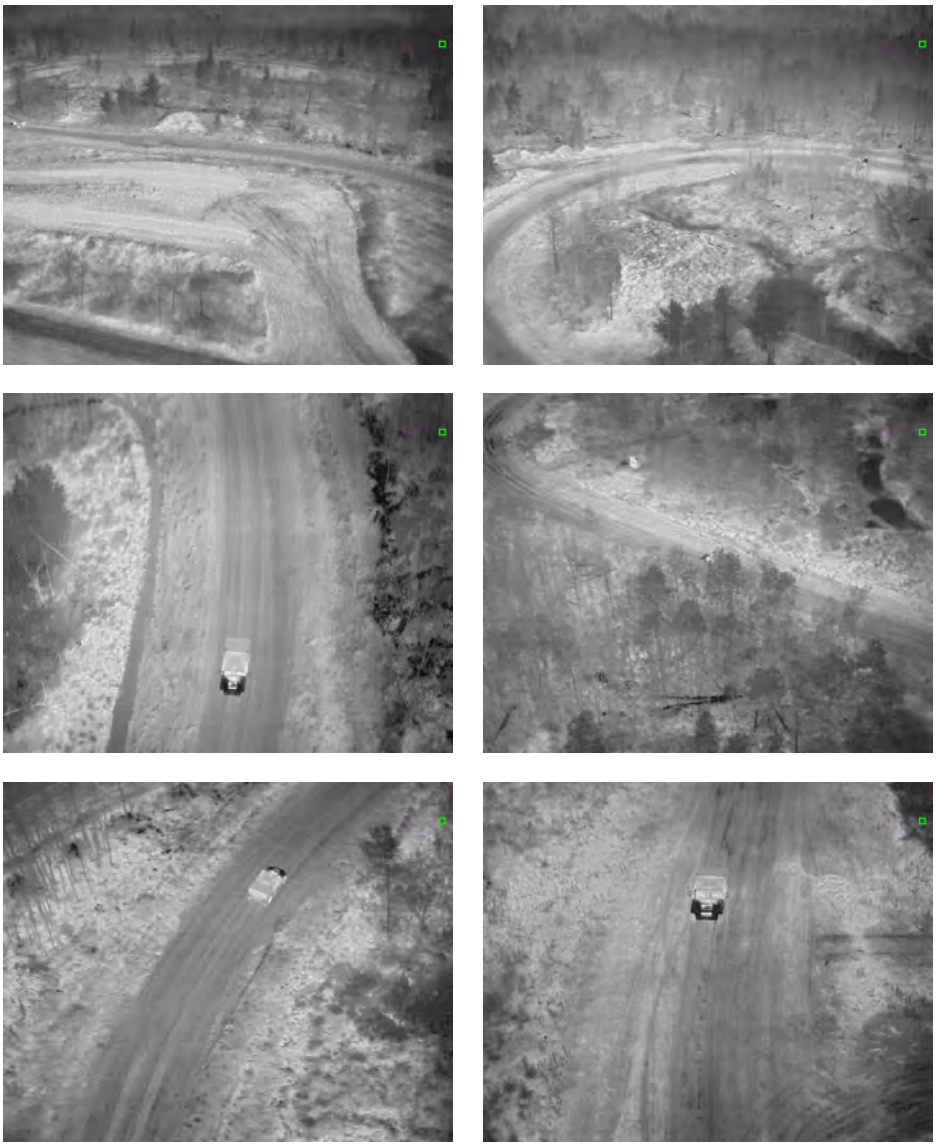


**Figure A.17:** Example LWIR images from Stage4.

**Figure A.18:** Results of single subset method trained with SSD using LWIR spin-set augmentation.

**Figure A.19:** Results of single subset method trained with YOLO using LWIR spin-set augmentation.

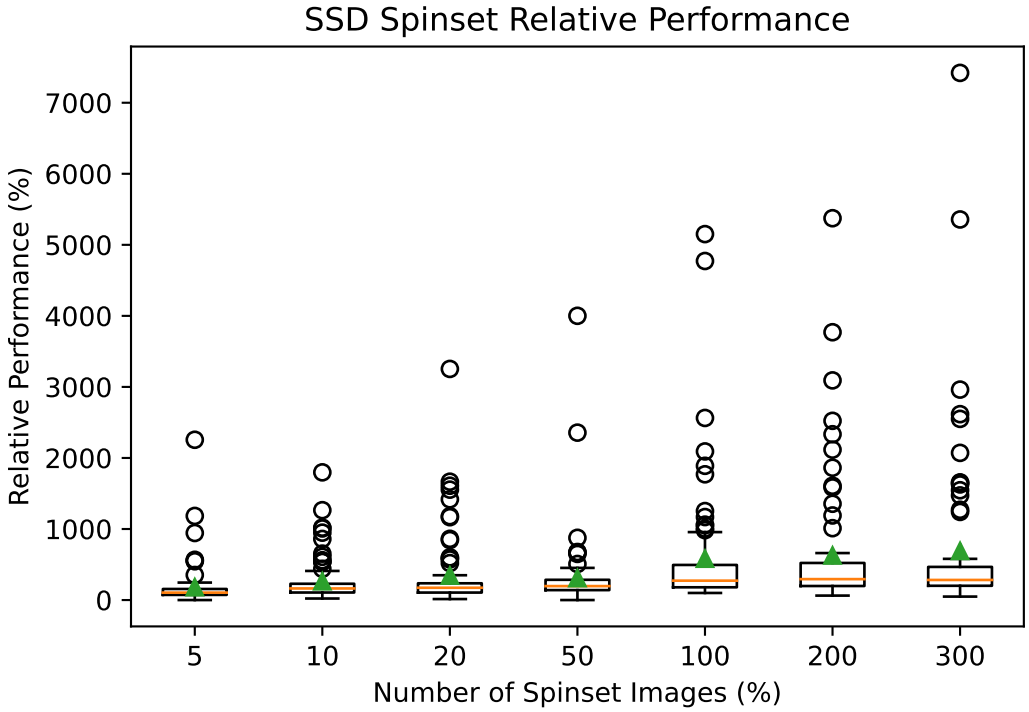



**Figure A.20:** Performance increase of single subset method trained with SSD relative to results without LWIR spin-set augmentation.

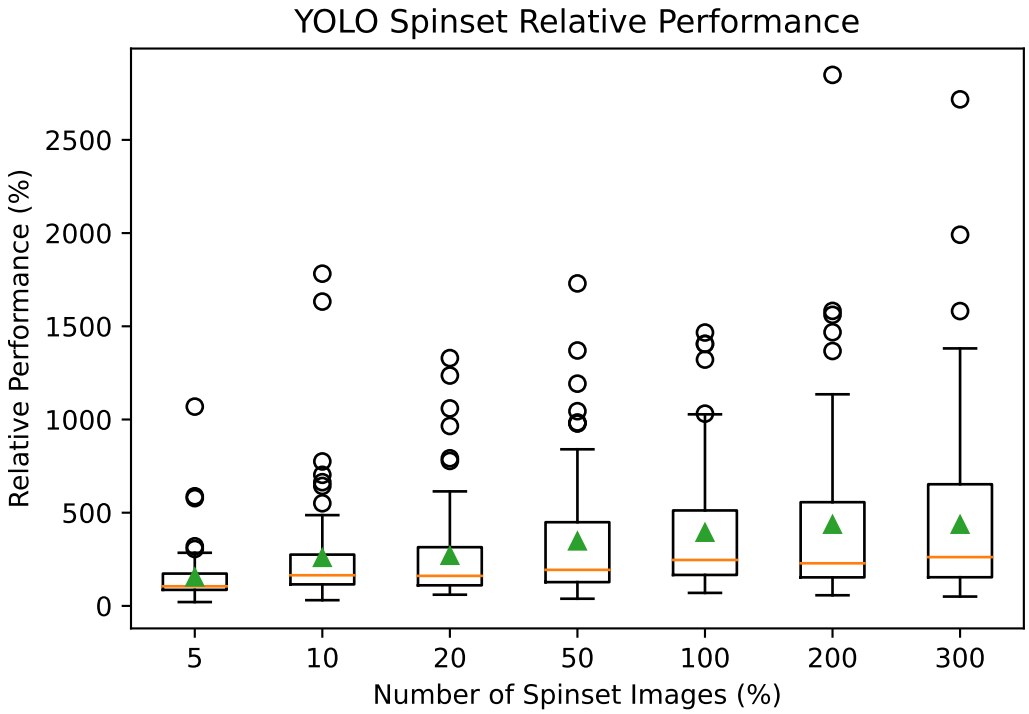



**Figure A.21:** Performance increase of single subset method trained with YOLO relative to results without LWIR spin-set augmentation.

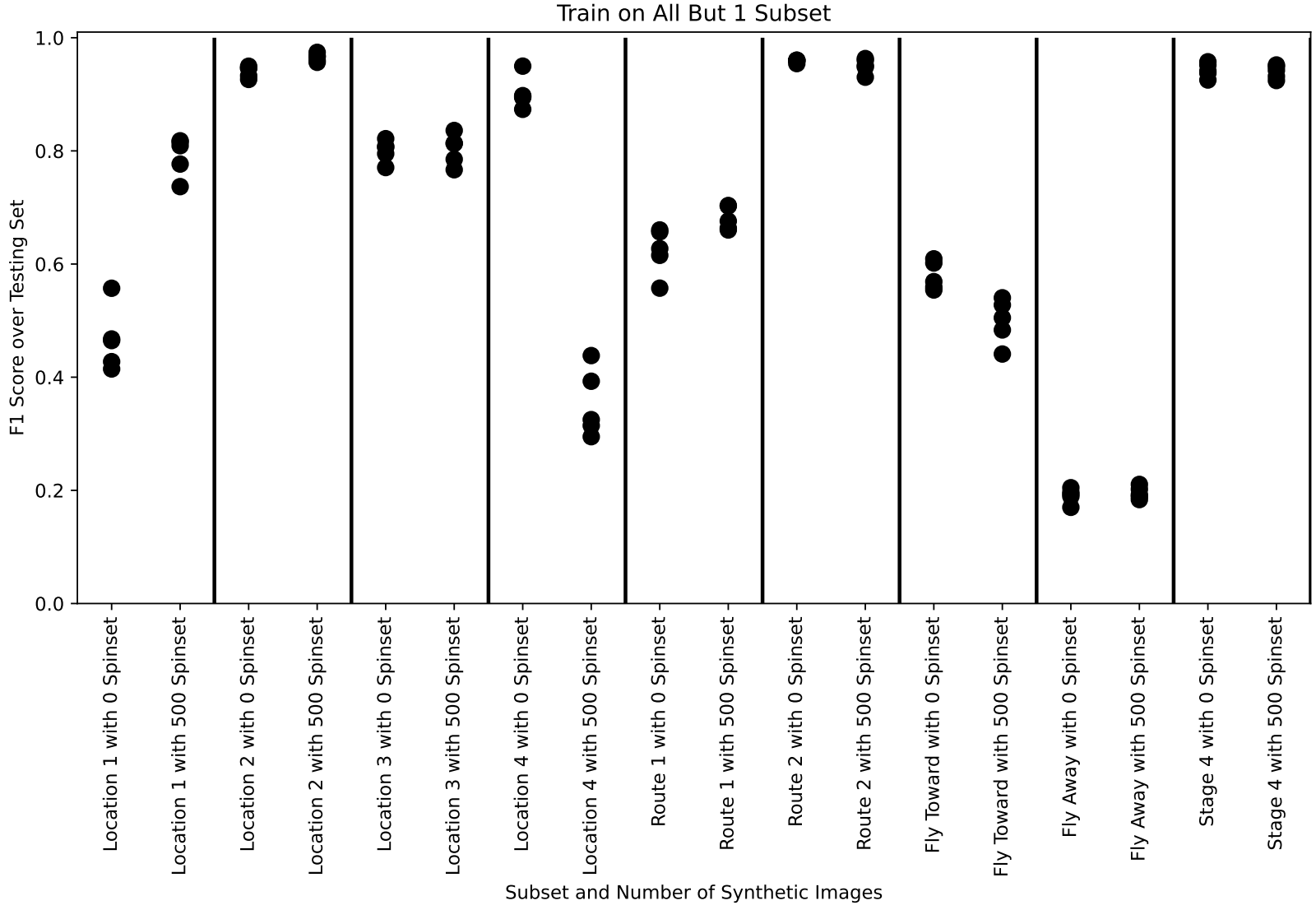



**Figure A.22:** Results of all-but-one method trained with Faster R-CNN using LWIR spin-set augmentation.

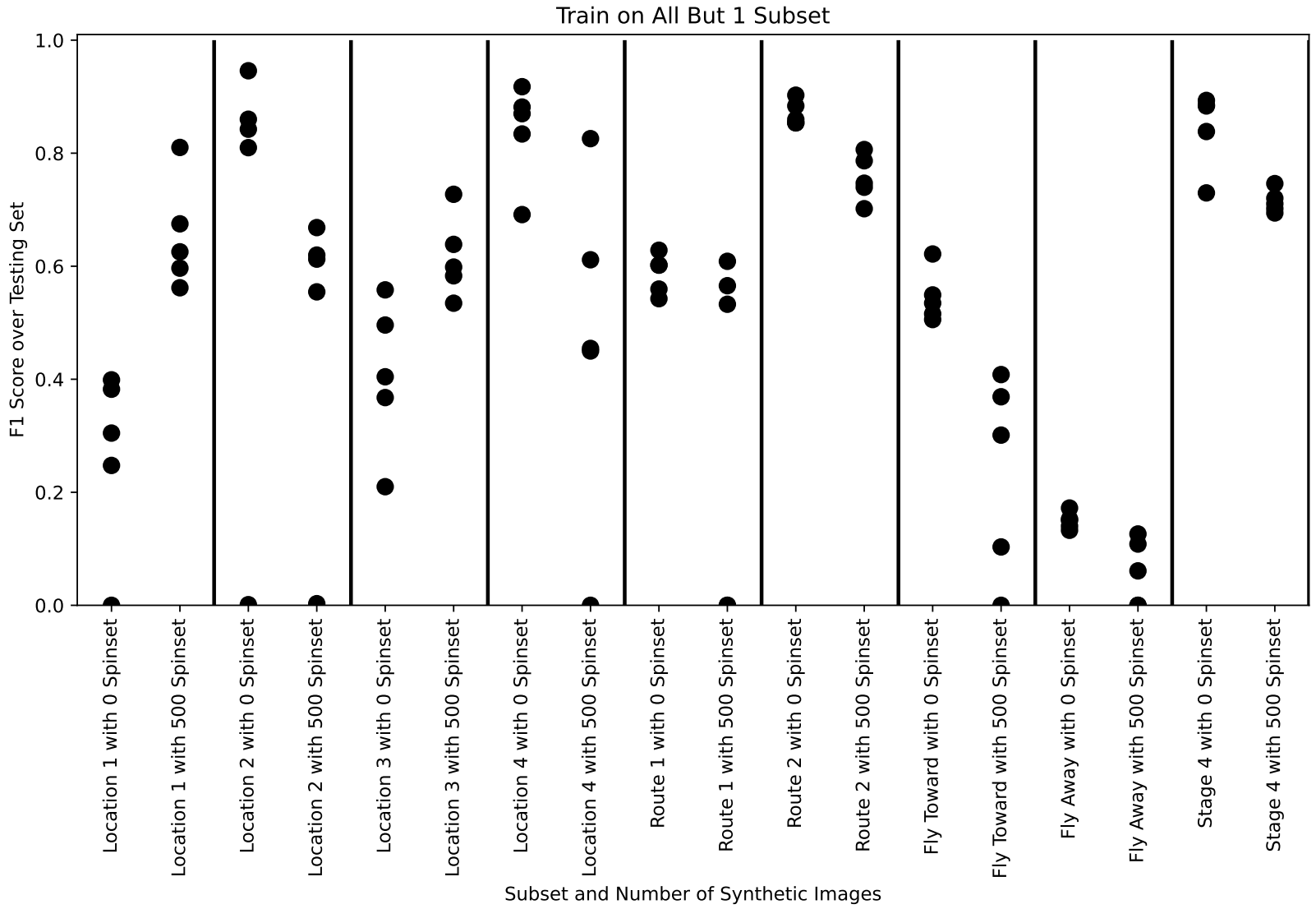
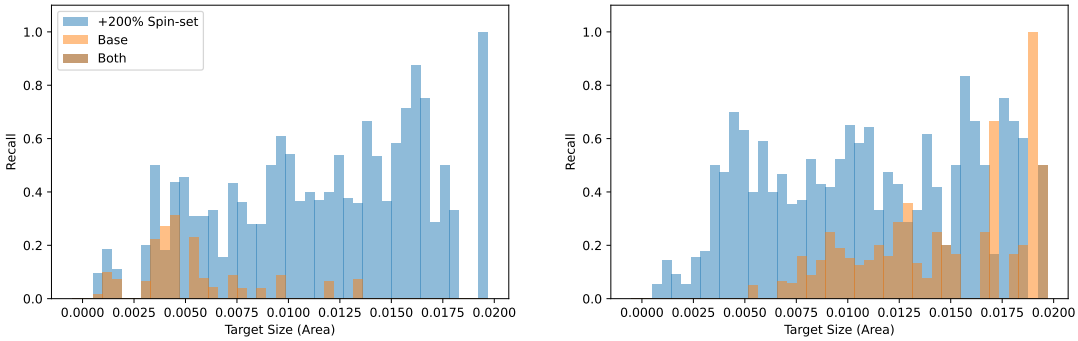



**Figure A.23:** Results of all-but-one method trained with SSD using LWIR spin-set augmentation.




(a) Trained on Location1

(b) Trained on Location2

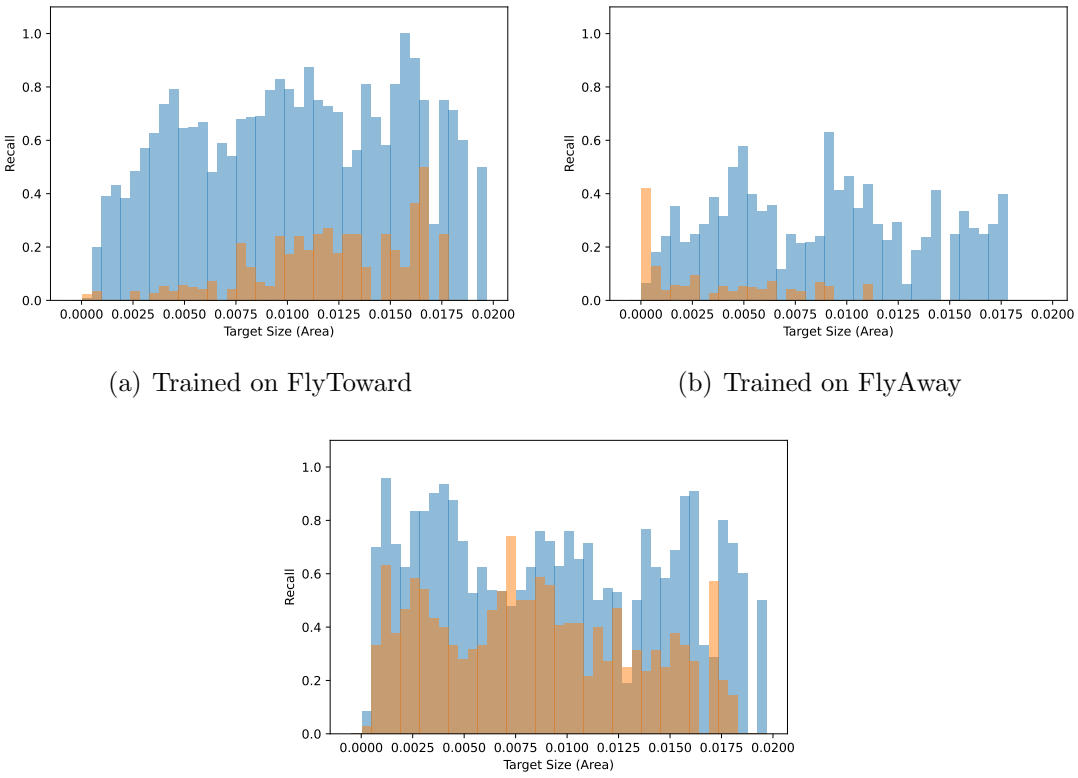
(c) Trained on Location3

(d) Trained on Location4

(e) Trained on Route1

(f) Trained on Route2

**Figure A.24:** Object recall over different target sizes for SSD trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page.

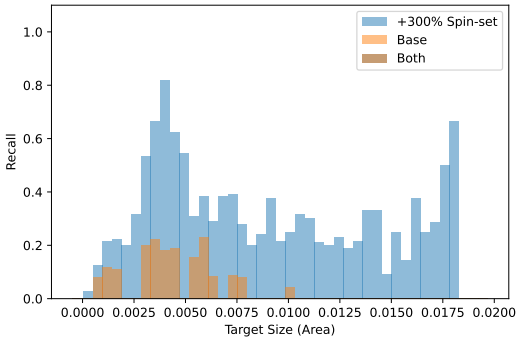



(a) Trained on FlyToward

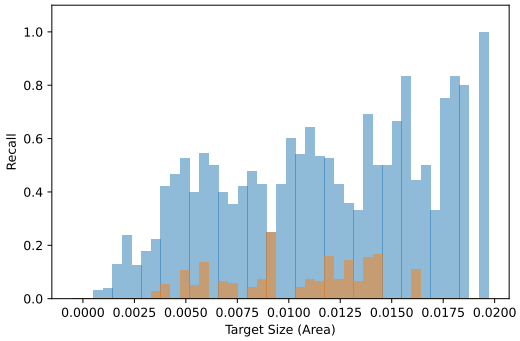
(b) Trained on FlyAway

(c) Trained on Stage4

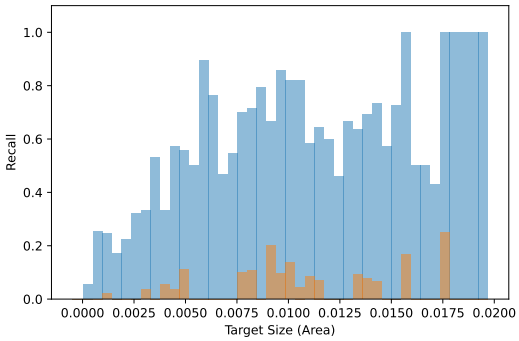
**Figure A.24:** Object recall over different target sizes for SSD trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page.



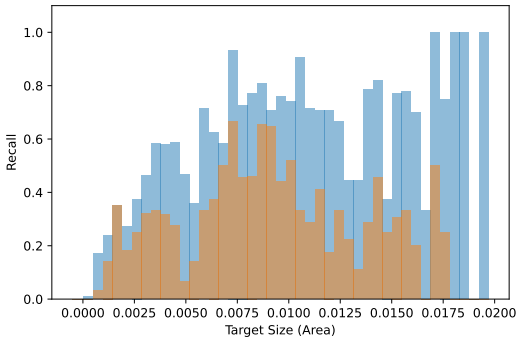

(d) Trained on Location1



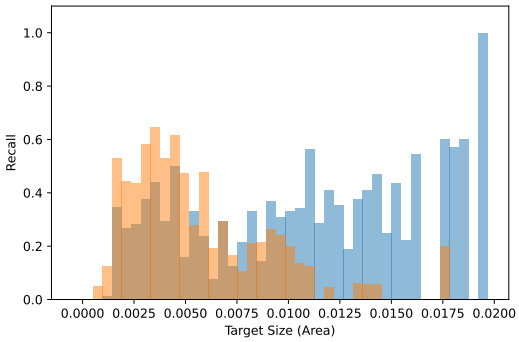

(e) Trained on Location2



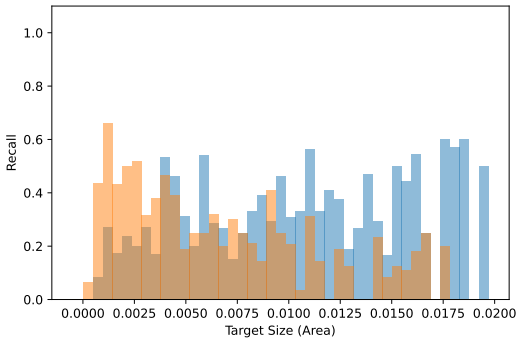

(f) Trained on Location3




(g) Trained on Location4

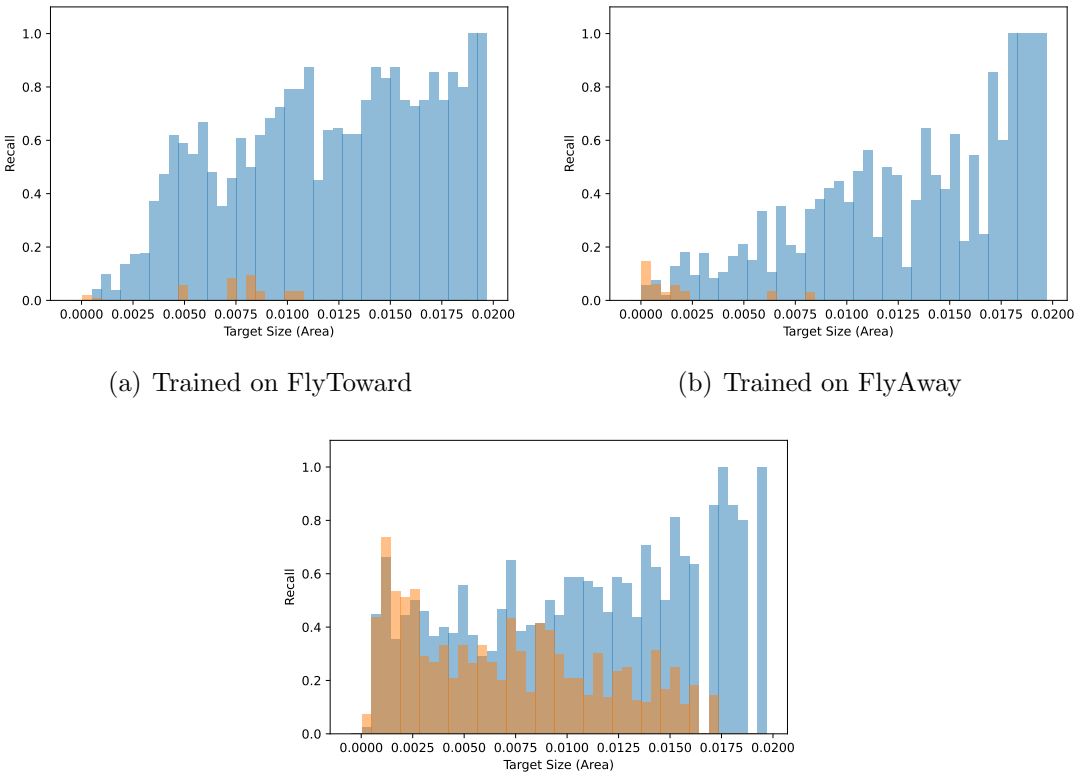



(h) Trained on Route1




(i) Trained on Route2

**Figure A.25:** Object recall over different target sizes for YOLO trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page.

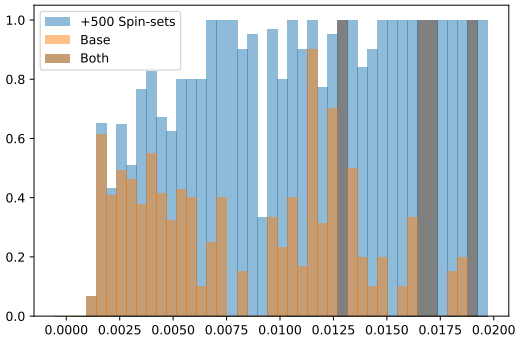



(a) Trained on FlyToward

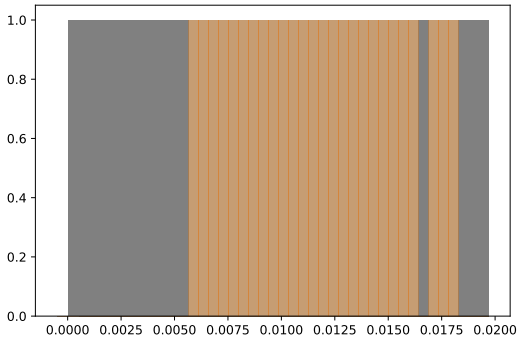
(b) Trained on FlyAway

(c) Trained on Stage4

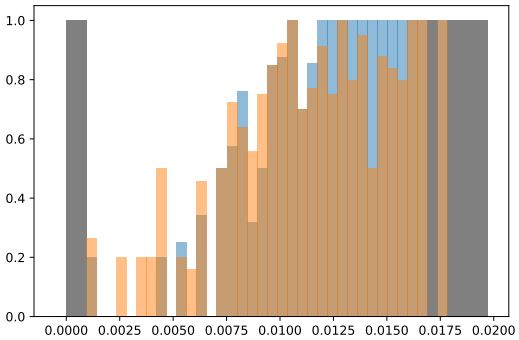
**Figure A.25:** Object recall over different target sizes for YOLO trained with single subset method. Compares base performance to including +10% spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page.



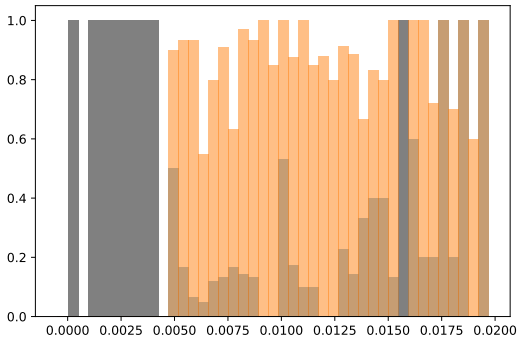

(d) Tested on Location1



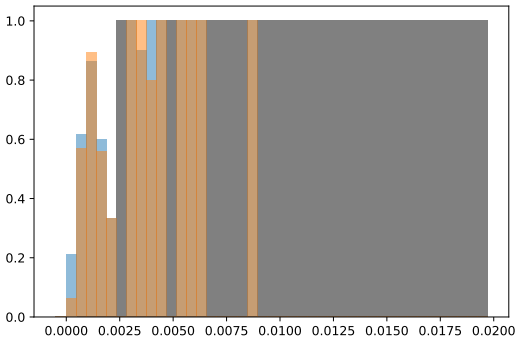

(e) Tested on Location2



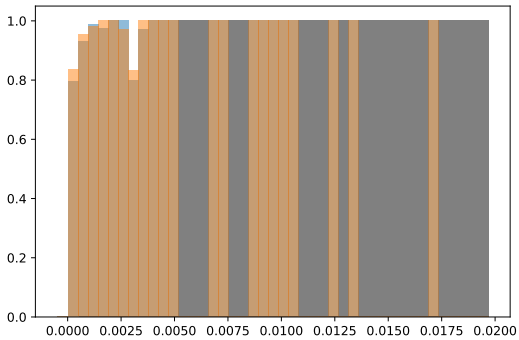

(f) Tested on Location3




(g) Tested on Location4

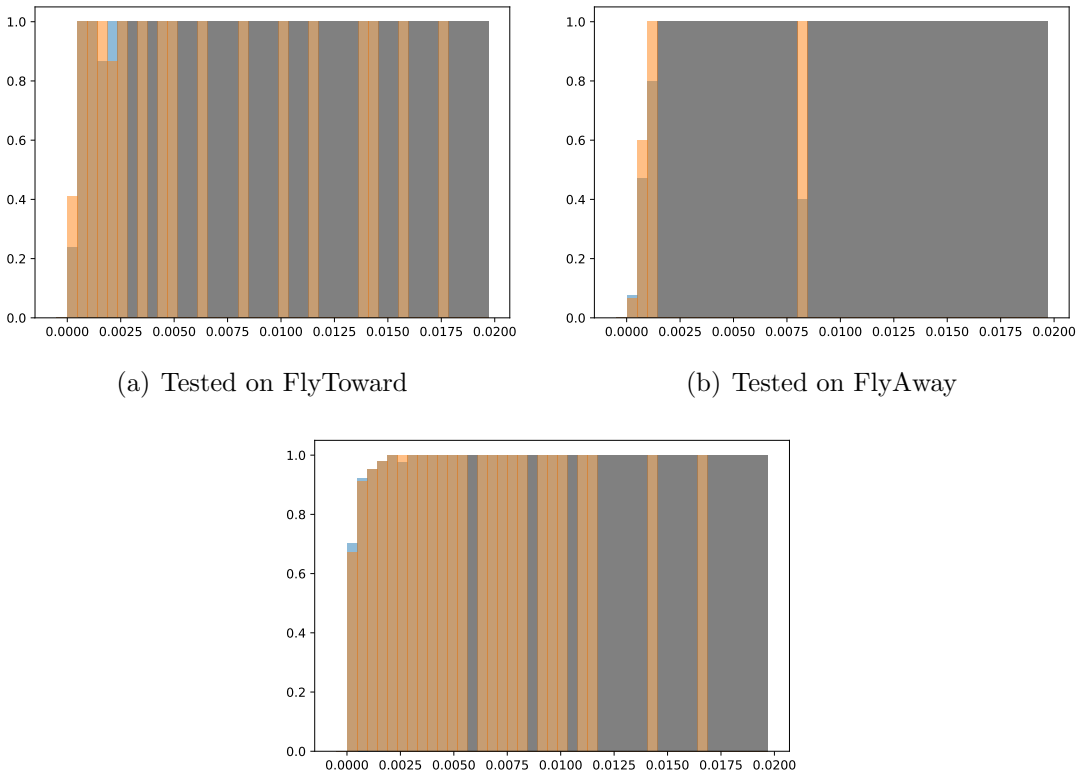



(h) Tested on Route1




(i) Tested on Route2

**Figure A.26:** Object recall over different target sizes for Faster R-CNN trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page.

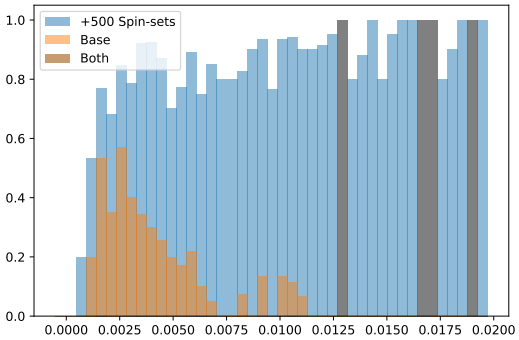



(a) Tested on FlyToward

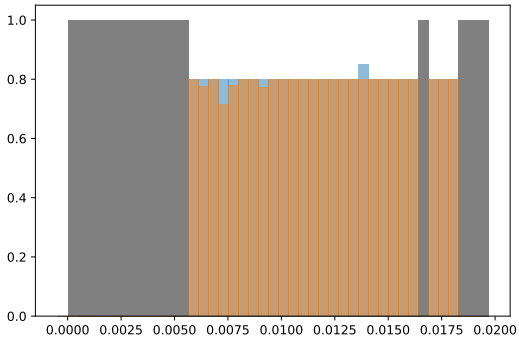
(b) Tested on FlyAway

(c) Tested on Stage4

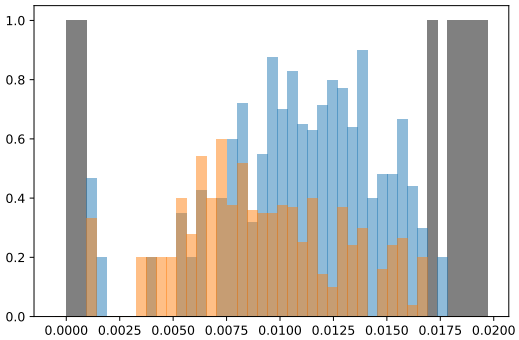
**Figure A.26:** Object recall over different target sizes for Faster R-CNN trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page.



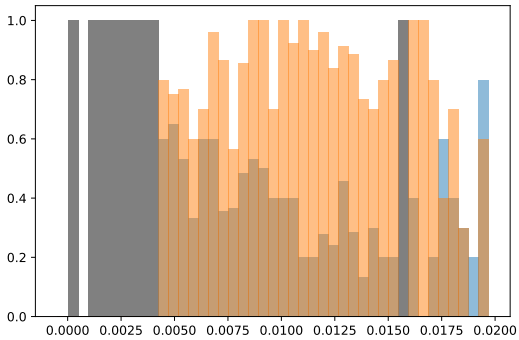

(d) Tested on Location1



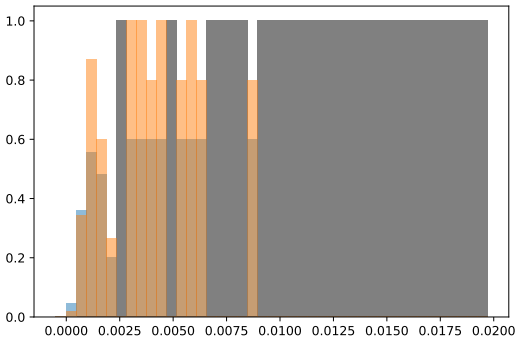

(e) Tested on Location2



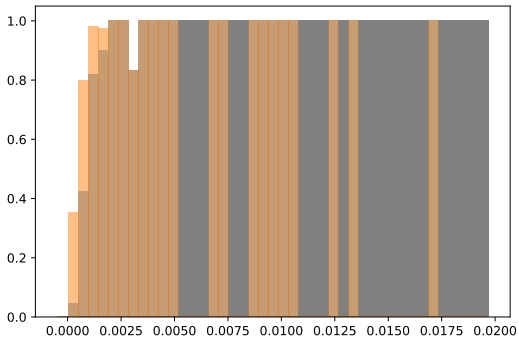

(f) Tested on Location3




(g) Tested on Location4

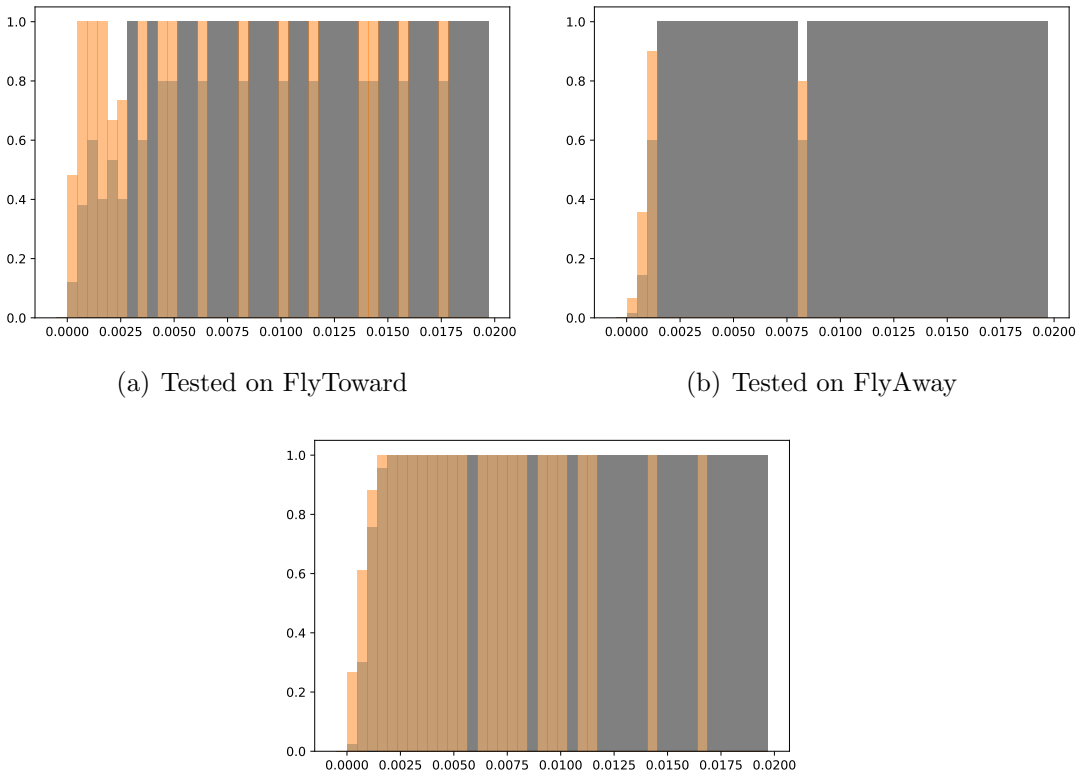



(h) Tested on Route1




(i) Tested on Route2

**Figure A.27:** Object recall over different target sizes for SSD trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued on next page.




(a) Tested on FlyToward

(b) Tested on FlyAway

(c) Tested on Stage4

**Figure A.27:** Object recall over different target sizes for SSD trained with all-but-one method. Compares base performance to including +500 spin-sets. Blue shows improvement due to augmentation and light brown shows reduced performance. Continued from previous page.