



AFRL-RI-RS-TR-2023-053

LEARNING WITH LESS LABELS

EMBEDDED INTELLIGENCE

MARCH 2023

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (<http://www.dtic.mil>).

AFRL-RI-RS-TR-2023-053 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /

PETER A. JEDRYSIK
Work Unit Manager

/ S /

JULIE BRICHACEK
Chief, Information Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

1. REPORT DATE		2. REPORT TYPE		3. DATES COVERED	
MARCH 2023		FINAL TECHNICAL REPORT		START DATE JULY 2019	END DATE DECEMBER 2022
4. TITLE AND SUBTITLE LEARNING WITH LESS LABELS					
5a. CONTRACT NUMBER FA8750-19-C-0510		5b. GRANT NUMBER N/A		5c. PROGRAM ELEMENT NUMBER 61101E	
5d. PROJECT NUMBER		5e. TASK NUMBER		5f. WORK UNIT NUMBER R2TH	
6. AUTHOR(S) Bryant Menn and Yiftach Eisenberg					
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Embedded Intelligence 110 Dale Drive Silver Spring MD 20910				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/RISB 525 Brooks Road Rome NY 13441-4505		DARPA/I2O 675 N. Randolph St. Arlington VA 22203-2114		10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/ RI & DARPA	11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-RI-RS-TR-2023-053
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT This final technical report summarizes Embedded Intelligence's research and development efforts on the DARPA Learning with Less Labeling (LwLL) program. It includes detailed documentation of the major methodologies that were researched and developed, quantitative results from both internal development experiments and independent Government evaluations, as well as key insights gained from analyzing those results.					
15. SUBJECT TERMS Generative Adversarial Networks, Distribution Shift, Domain Adaptation, Consistency Regularization, Pseudo Labeling					
16. SECURITY CLASSIFICATION OF:				17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U	SAR	51	
19a. NAME OF RESPONSIBLE PERSON PETER A. JEDRYSIK				19b. PHONE NUMBER (Include area code) N/A	

Contents

List of Figures	ii
List of Tables	iv
Summary	1
1 Introduction.....	3
2 Methods and Procedures	4
2.1 GAN Turk Pseudo-label.....	4
2.2 GAN Turk Synthetics.....	6
2.2.1 Extensions to objection detection	7
2.2.2 Rareplanes Dataset	8
2.3 Active Learning.....	10
2.4 Self-training & Consistency Regularization for Object Detection Label Complexity.....	10
2.4.1 Synthetic Data Domain Adaptation.....	12
2.4.2 xView Haul Truck	12
3 Results and Discussion	14
3.1 GAN Turk Pseudolabel	14
3.2 GAN Turk Synthetics.....	17
3.2.1 BIRDSAI	20
3.2.2 Rareplanes	22
3.3 Active Learning.....	26
3.4 STAC: Rareplanes Synthetic Domain Adaptation.....	26
3.5 STAC: xView Haul Truck.....	28
3.6 Year 1 Evaluation.....	29
3.7 Year 2 Evaluation.....	32
3.8 Year 3 Evaluation.....	34
4 Software	38
5 Conclusion	41
References.....	42
List of Acronyms	44

List of Figures

Figure 1: GAN Turk Pseudolabel and GAN Turk Synthetics System Diagram.....	5
Figure 2: High level view of networks in GAN Turk Pseudo-label	6
Figure 3: Sample real and synthetic images from the BIRDSAI dataset.....	8
Figure 4: Sample real and synthetic images from the Rareplanes dataset.....	9
Figure 5: STAC Data Augmentation Strategies.....	11
Figure 6: STAC Training Process.....	12
Figure 7: A sample full-size image from the xView dataset	13
Figure 8: Effects of Pseudolabel Thresholding.....	14
Figure 9: Effect of Source Domain Data Size	15
Figure 10: GAN Turk Pseudo-label Development Task Performance	16
Figure 11: Benefit of GAN Turk Synthetic Images.....	17
Figure 12: GAN Turk Synthetics Semantic Preservation Improvement	18
Figure 13: Real Images Effect on GAN Turk Synthetics Training	19
Figure 14: The performance of our object detection approach on the BIRDSAI dataset.....	20
Figure 15: CycleGAN BIRDSAI Example.....	21
Figure 16: CycleGAN BIRDSAI Failure Mode	22
Figure 17: CycleGAN Rareplanes Example	23
Figure 18: CycleGAN Rareplanes Failure Mode.....	25
Figure 19: Coreset Sampling DomainNet Performance	26
Figure 20: STAC Rareplanes Unsupervised Domain Adaptation	27
Figure 21: STAC Rareplanes Semi-supervised Domain Adaptation.....	28
Figure 22: Empty Pseudolabeling Effect on STAC	29
Figure 23: Year 1 Coreset Sampling Evaluation Results.....	30
Figure 24: Year 1 GAN Turk Pseudo-labeling Evaluation Results.....	31
Figure 25: Year 2 Evaluation Result: Image Classification Task 1	33
Figure 26: Year 2 Evaluation Result: Image Classification Task 2.....	34
Figure 27: Year 2 Evaluation Result: Image Classification Task 3.....	34
Figure 28: Year 3 Evaluation: Object Detection Task 1	36
Figure 29: Year 3 Evaluation: Object Detection Task 2.....	36

Figure 30: Year 3 Evaluation: Object Detection Task 3	37
Figure 31: Screenshot of successful system submission for evaluation	39
Figure 32: Screenshot of successful Docker image build on DMC infrastructure for evaluation	40

List of Tables

Table 1: Year 2 Methodology Breakdown.....	32
Table 2: Year 3 Evaluation Methodology Breakdown: Medium Task	35
Table 3: Year 3 Evaluation Methodology Breakdown: Large Task	35

Summary

This final report documents the series of investigations we performed over the course of the DARPA Learning with Less Labels (LwLL) program. We focused on domain adaptation for image classification and geospatial applications for object detection. Generative modeling techniques, including our novel GAN Turk methodology, and active learning were explored to reduce label requirements for image classification. We also investigated using GAN Turk, consistency regularization, and self-training for object detection in geospatial datasets. We found that:

- Using generative models to perform image translation for domain adaptation was sensitive to domain pairing and the instability of generative adversarial network (GAN) model training
- Ideal active learning methods are difficult to select apriori and choices vary from dataset to dataset
- Self-training and consistency regularization are a promising pair of techniques for semi-supervised learning.

We evaluated several generative modeling techniques for both image classification and object detection. We evaluated CycleGAN, CUT, and two iterations of our own GAN Turk methodology, which was inspired by CycleGAN.

As part of the Year 1 Evaluation, we also considered active learning and submitted a coreset active learning component as part of our evaluation. The coreset methodology was selected after comparing with entropy-based selection methods and other sampling methods based on published and internal results in addition to its feature distance motivated design. However, our internal results and evaluation results showed it was not a reliable methodology.

The combination of self-training and consistency regularization has been the most promising method we have researched in this program. We investigated the STAC methodology as our object detection system for the Year 3 Evaluation. Our internal results showed the combination of the two techniques could be utilized for synthetic-to-real domain adaptation and also more generally to geospatial applications. Furthermore, we showed that pseudolabeling empty images for consistency regularization produced a benefit for STAC in a geospatial dataset. The significant benefit of pseudolabeling empty images was not able to be tested as part of the Year 3 Evaluation as those datasets did not contain any images, i.e., all the images contained at least one object of interest.

As part of program requirements, we participated in annual independent, third-party evaluations and have documented these results in Sections 3.6 to 3.8. One of our main challenges during evaluation was computational budget. Our submitted systems for GAN Turk and STAC required significantly more compute than baseline methods. As a result, these methodologies were submitted for only a few checkpoints for each evaluation task.

In Section 4, we document our software deliverable and provide a brief overview of our software architecture. We also provided configurations for Docker images to package our system for evaluation.

Overall, our team contributed to the Government's knowledge base about which methods do and do not show promise for learning with less labeling. This exploration was conducted using sound scientific methods and carefully controlled experiments aimed at providing an unbiased

assessment of various methods. This report summarizes our key findings throughout the LwLL program, and highlights several areas we believe to be promising and others that we would not recommend pursuing based on the evidence we collected on this basic research program.

1 Introduction

This final report summarizes Embedded Intelligence's research and development efforts on the DARPA LwLL program, including the methodologies we developed, their quantitative performance on development tasks and blind evaluation tasks, as well as key insights we've gained from analyzing those results. Section 2 documents the major methodologies we researched and developed with our results from internal development experiments in Section 3. Sections 3.6 to 3.8 summarizes our performance on the independent Government evaluations conducted by NASA Jet Propulsion Laboratory (JPL) and Data Machines Corporation (DMC). Documentation of the software we submitted to the Government is captured in Section 4. Concluding remarks can be found in Section 5.

2 Methods and Procedures

We present two methodologies, GAN Turk pseudolabeling (Section 2.1) and GAN Turk synthetic (Section 2.2), that augment traditional supervised learning approaches to solve the domain adaptation problem from the challenge problem’s base stage domain to the problem’s adaptation stage domain. These systems (Figure 1) were used for both Year 1 and Year 2 evaluations (results presented in Section 3.6 and Section 3.7).

2.1 GAN Turk Pseudo-label

GAN Turk is based on CycleGAN [15], which is a purely unsupervised framework for learning translations of unpaired images between two domains in such a way that they maintain cycle consistency. GAN Turk extends this framework by incorporating knowledge of the labels for some subset of the images, thus making it a semi-supervised domain adaptation method.

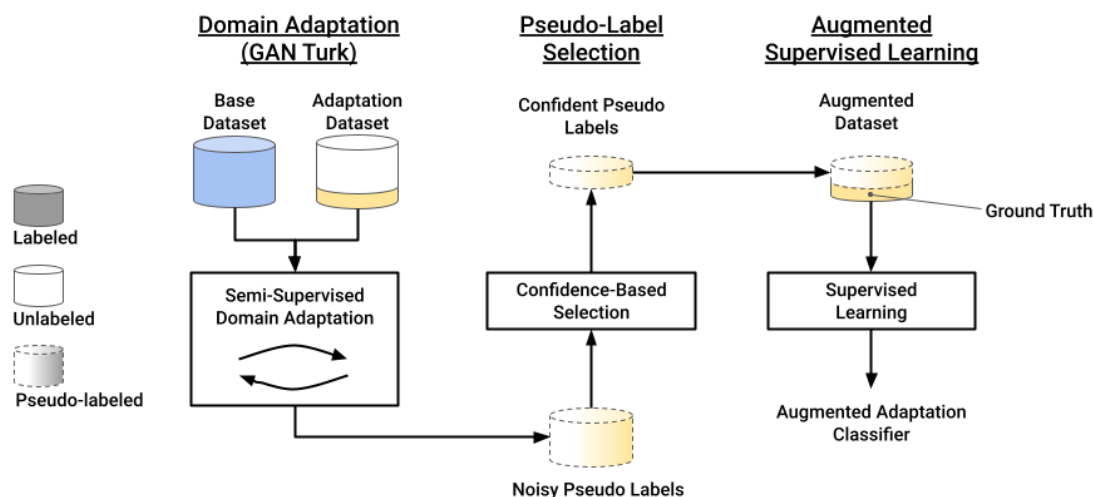
We adapt CycleGAN by adding an additional pair of classifier networks, C_S and C_T as depicted in Figure 2. For each of the two domains, we have one classifier network classifying the real images of that domain (e.g. the base stage domain) and the generated images to that same domain from the other domain (e.g. the adaptation stage domain). Each classifier network is implemented as a PyTorch torchvision ResNet18 network initialized with ImageNet pre-trained weights. During model training, we update the classifier simultaneously with the generator. This results in the losses

$$L_{cls,real} = \text{CrossEntropy}(C_S(x_s), y_s) + \text{CrossEntropy}(C_T(x_t), y_t) \quad (1)$$

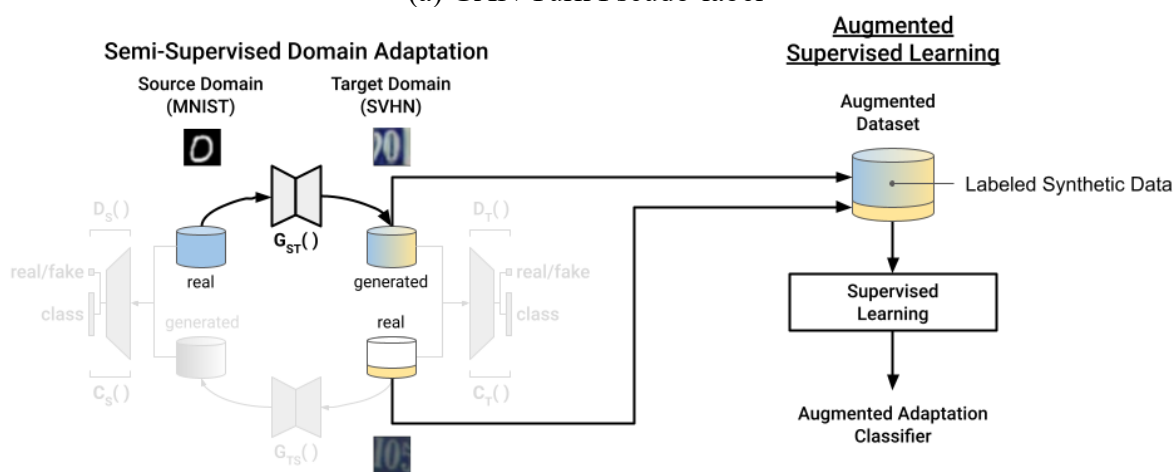
$$L_{cls,fake} = \text{CrossEntropy}(C_S(G_{TS}(x_t)), y_t) + \text{CrossEntropy}(C_T(G_{ST}(x_s)), y_s) \quad (2)$$

$$L_{cls} = L_{cls,real} + L_{cls,fake} \quad (3)$$

$$L = L_{GAN} + \lambda_{cyc}L_{cyc} + \lambda_{idt}L_{idt} + \lambda_{cls}L_{cls}. \quad (4)$$



(a) GAN Turk Pseudo-label



(b) GAN Turk Synthetics

Figure 1: GAN Turk Pseudolabel and GAN Turk Synthetics System Diagram

Based on prior analysis documented in our CLIN 0001AC report, we added a pseudo-labeling step after training GAN Turk to supplement the labeled training data used for supervised learning. The unlabeled training data from the adaptation domain is pseudo-labeled by the adaptation domain classifier C_T . The unlabeled data and pseudo-labeled are filtered by requiring that the classifier C_T must have a minimum confidence p , i.e.

$$(X_{aug}, Y_{aug}) = \{(x_t, y_{pseudo}) : x_t \in X_t, \max \text{softmax}(C_T(x_t)) \geq p, \\ y_{pseudo} = \arg \max \text{softmax}(C_T(x_t))\}. \quad (5)$$

The confidence threshold p was tuned as part of preparation for the Year 1 Annual Evaluation. The results of that tuning are presented in Section 3.1.

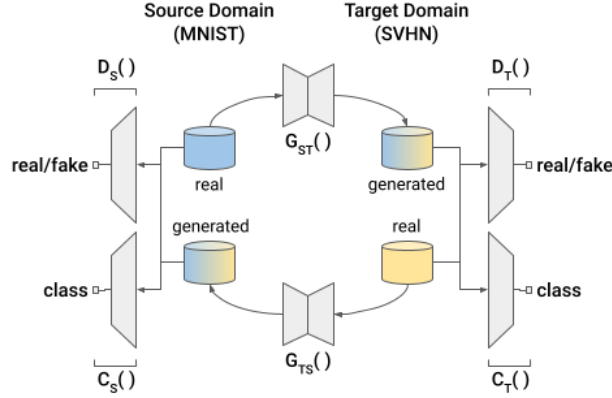


Figure 2: High level view of networks in GAN Turk Pseudo-label

2.2 GAN Turk Synthetics

The GAN Turk Synthetics method provides an alternative way to supplement the small labeled training data in the adaptation domain by generating synthetic data from the base stage domain. After training GAN Turk, the base-to-adaptation generator G_{ST} is used to transform base domain images into the adaptation domain and create an augmented dataset of the synthesized images and the original label

$$(X_{aug}, Y_{aug}) = \{(x, y) : x = G_{ST}(x_s), y = y_s, (x_s, y_s) \in (X_s, Y_s)\}. \quad (6)$$

Similar to GAN Turk Pseudo-label, this new dataset (X_{aug}, Y_{aug}) is combined with the adaptation labeled training data for supervised learning. This method deviates from the training description of GAN Turk Pseudo label in how the classification loss is determined as well as the network architecture. We modify the classification loss used by GAN Turk Pseudo-label to only consider real images, i.e. $L_{cls} = L_{cls,real}$. For the network architecture, the discriminator and classifier network of each domain are combined into a single multi-task network as shown in Figure 1b. As result, the generator and classifier outputs are not optimized in the same training update. Instead, only the generator is updated according to Equation (4), and the multi-task discriminator-classifier network is update via

$$L_{GAN} + \lambda_{cls} L_{cls,real}. \quad (7)$$

2.2.1 Extensions to objection detection

As part of our Phase 2 efforts, we began to consider modifications to GAN Turk to support object detection and other semi-supervised object detection methods. Given the vast amount of geospatial data collected by satellite systems and the effort required to manually annotate imagery at a sufficient scale to enable training of machine learning models, we believe that techniques which leverage synthetic data for model training could greatly reduce the cost and effort required to train reliable models. Since computer generated synthetic data may be automatically annotated during the data generation process, and since large amounts of unlabeled satellite data may be easily obtained, techniques that are able to train effective models with such a combination of data would be highly valuable.

First, we applied our technique to Benchmarking IR Dataset for Surveillance with Aerial Intelligence (BIRDSAI) [2], a dataset which includes real thermal infrared (IR) imagery, captured overhead via unmanned aerial vehicle (UAV), of humans and animals navigating landscapes in Southern Africa, as well as synthetic IR images of similar scenes produced via Microsoft AirSim using a realistic 3D model of an African savanna. Example real and synthetic images from the BIRDSAI dataset are shown in Figure 3.

To prepare the data, we first started with a simplifying assumption of full class overlap between the real and synthetic data. The raw data includes objects from the following set of classes: crocodile, dog, elephant, giraffe, hippo, human, lion, rhino, unknown, and zebra. However, the only object classes which exist in both datasets are: human, elephant, lion, and unknown. Due to the inherent ambiguity of the unknown class, and a very small number of objects belonging to the lion class, we reduced the dataset to only those images which do not contain objects other than humans or elephants. We also removed images from the dataset which are annotated as either containing noise or occlusions. Finally, due to the large scale of the data, we downsample the dataset to a random set of 8,000 real and 8,000 synthetic images for training and 8,000 real images for testing.

To address a potential real-world use case, we designed our experiment to leverage the full set of labeled synthetic data, a small set of labeled real data, and a large set of unlabeled real data to train a model and evaluate the performance on the real test set. Our experiment was setup as follows: 1) we trained a Faster R-CNN object detection model on the synthetic data, 2) we trained a CycleGAN model to translate the synthetic images to the style of real IR images, 3) we then fine-tuned the Faster R-CNN model on a combined dataset of the translated synthetic-to-real images and varying amounts of labeled real data, and finally 4) we fine-tuned the Faster R-CNN model again on only the small set of labeled real data.



(a) Real IR



(b) Synthetic IR

Figure 3: Sample real and synthetic images from the BIRDSAI dataset

2.2.2 Rareplanes Dataset

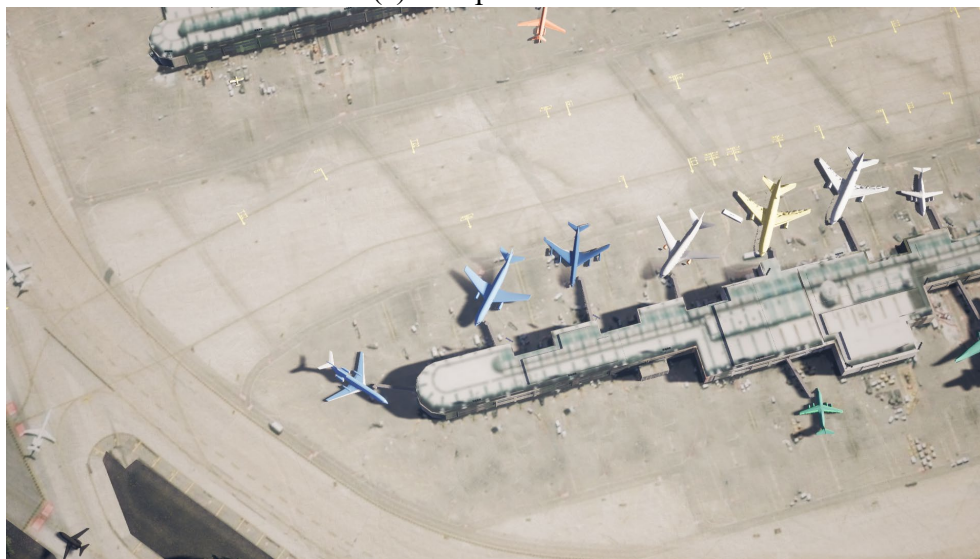
Lastly, we evaluated our approach against Rareplanes [8], a dataset comprising real and computer-generated synthetic satellite images of airport environments with bounding box annotations of various classes of airplanes. Examples of real and synthetic images from the Rareplanes dataset are shown in Figure 4.

Once again assuming full class overlap between real and synthetic datasets, we first prepare the data by removing images containing military planes, which are not present in the synthetic data. This leaves three object classes: Civil Utility/Transport (Large), Civil Utility/Transport (Medium), and Civil Utility/Transport (Small). Due to the large size of the synthetic dataset, we

also downsample the synthetic dataset for tractability, leaving 5,567 images in each of the real and synthetic training datasets, and 2,550 images in the real test set.



(a) Rareplanes Real



(b) Rareplanes Synthetic

Figure 4: Sample real and synthetic images from the Rareplanes dataset

To simulate a real-world use case, we once again assign the synthetic data as the fully annotated source dataset, and the real data as the minimally annotated target dataset and attempt to learn a model that reliably detects objects in the real test set with as few annotations as possible for the real training data.

2.3 Active Learning

In Phase I, we explored using an Active Learning technique to select which images to label in order to assess if label selection strategies may be beneficial to learning with less labeling. In particular, we used coresets sampling [7] to build labeled datasets for a supervised classifier. The premise behind this technique is that given a label budget, the most informative data points to label are the ones that cover all the other unlabeled data points with the smallest radius. By minimizing the covering radius, this algorithm tries to identify the points that most effectively represent the data distribution.

2.4 Self-training & Consistency Regularization for Object Detection Label Complexity

While self-training [6] and consistency regularization [14] are not new ideas, FixMatch [9] was the first to combine both techniques together for semi-supervised image classification to great success. The combined technique applied consistency regularization over pseudolabels instead of just consistency regularization over labeled data or just cross-entropy loss over pseudolabels. Specifically, the training loss function is of the form

$$H(f(x), y) + \lambda_u H(f(A(x^{(u)})), y^{(u)}) \quad (8)$$

where f is the neural network, $H(f(x), y)$ is the cross-entropy of the neural network output of x and label y , $A(x^{(u)})$ is a strong augmentation of unlabeled sample $x^{(u)}$, $y^{(u)}$ is the pseudolabel of $x^{(u)}$, and λ_u is the loss coefficient of the consistency regularization term. This concept was adapted to object detection as STAC [10] and has been the focus of our object detection research.

Self-Training (via pseudolabel) and Augmentation driven Consistency regularization (STAC) [10] adapts FixMatch to object detection tasks with some key differences. First, STAC performs image transformations at two different levels, the global image level and the bounding box level. The transformations at the bounding box level (see Figure 5) allow the object detector to learn localization that is more robust. STAC also does not pseudolabel during a minibatch. Instead, STAC is trained in two stages with pseudolabels being produced after the first stage (referred as the teacher training) and used for training in the second stage (student training, see Figure 6). In addition, STAC does not use exponential moving average.

Visualization of different types of augmentation strategies (Figure 5) from left to right: original image, color transformation, global geometric transformation, box-level geometric transformation, box-level geometric transformation, and Cutout. Figure and caption from Figure 3 of Sohn et al. [10].

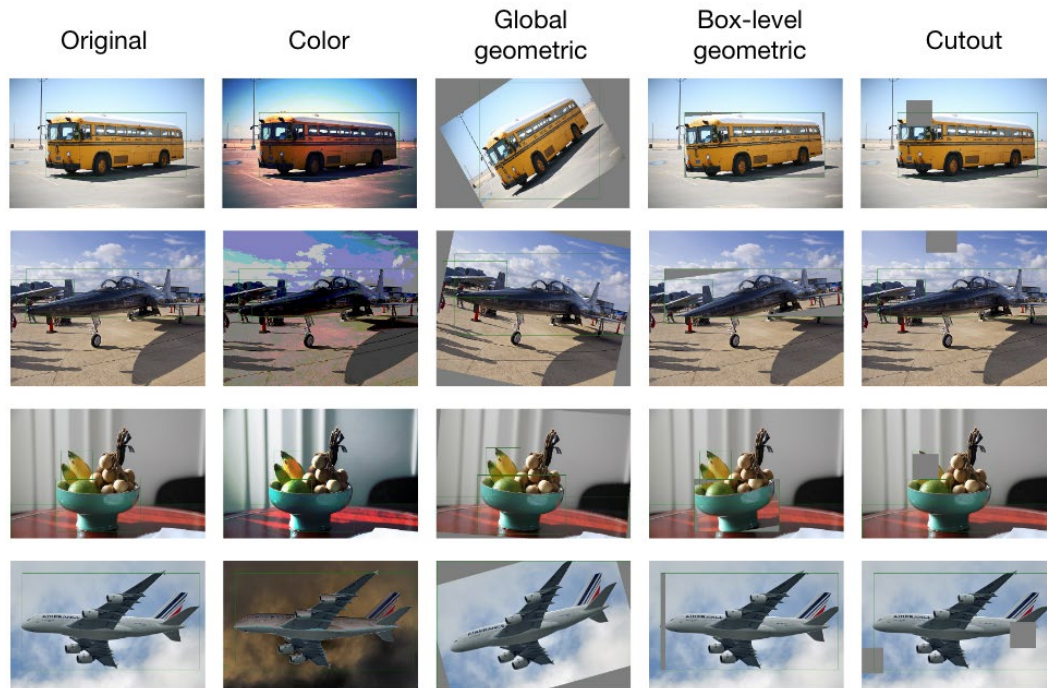


Figure 5: STAC Data Augmentation Strategies

In STAC (Figure 6), teacher models are trained in a traditional supervised manner (left). The teacher model is then used to provide pseudolabels that are filtered by confidence. Confident pseudolabels are then used for consistency regularization during student training (right side). Adapted from Figure 2 of Sohn et al. [10].

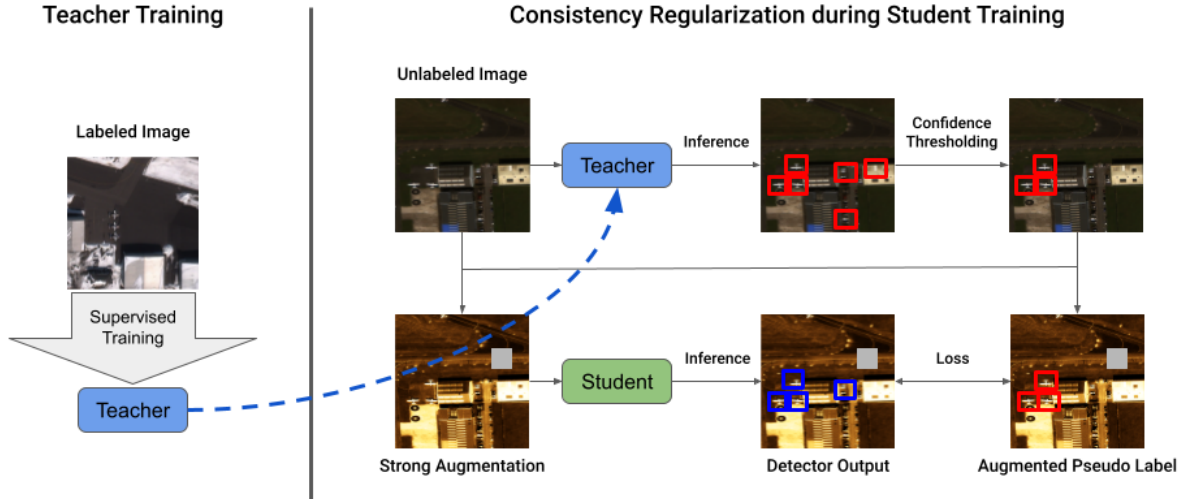


Figure 6: STAC Training Process

2.4.1 Synthetic Data Domain Adaptation

While not all scenarios have synthetic images readily available, using synthetics is an active area of research to reduce labeling requirements for many applications such as self-driving [4] and facial recognition [13]. Much of this research has been focused on closing the “synthetics gap,” i.e. reducing the drop in test performance when training on synthetic data compared to real data. These efforts tend to focus on improving the synthetic image generation process to better match real data [4, 13] or to simply improve downstream model performance via domain randomization [12]. We present what we believe to be a novel application of STAC and illustrate that STAC can better utilize synthetic images than existing standard approaches.

We explore this scenario using the Rareplanes dataset released by AI.Reverie [8]. We train a teacher model using all synthetic images for the unsupervised domain adaptation scenario. For semi-supervised domain adaptation, we first train the teacher model on all synthetic images and then fine-tune on a small amount of labeled real images (50 images). Unlabeled real images are pseudolabeled and thresholded at a confidence of 0.9 for student training. We only use unlabeled real images for consistency regularization.

2.4.2 xView Haul Truck

In addition to evaluating STAC on the Rareplanes dataset, we also evaluated its performance on a Haul Truck class subset of xView. xView images are moderately wide-area, with each image covering approximately 1 km² of ground area at a ground sampling distance (GSD) of 0.3 meters. Thus, xView images are approximately 3,000 × 3,000 pixels, with large swaths of each image containing no objects at all. Demonstrable performance on xView would then provide confidence that such techniques are likely to extend to operational scenarios due to the lack of pre-filtering to regions of interest (ROIs). A sample full-size image from xView is shown in Figure 7.

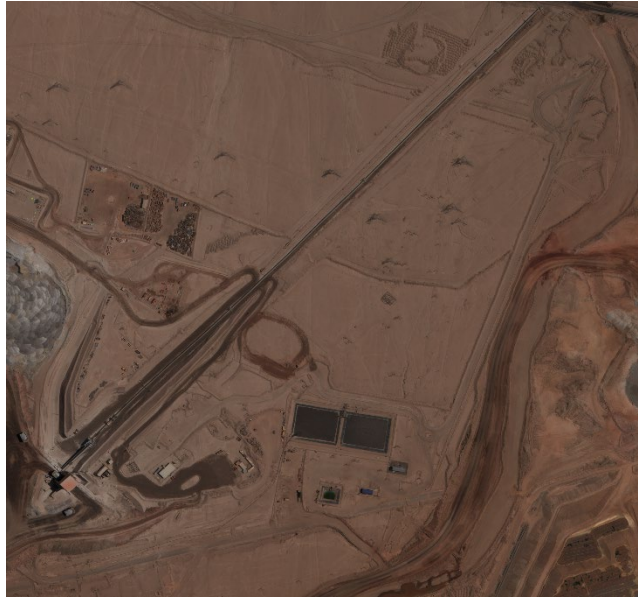


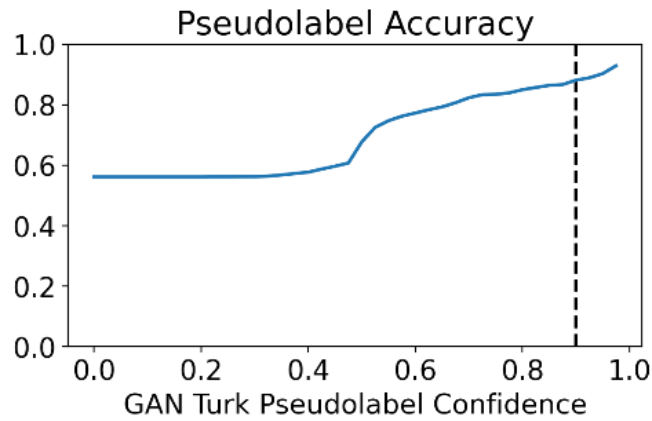
Figure 7: A sample full-size image from the xView dataset

3 Results and Discussion

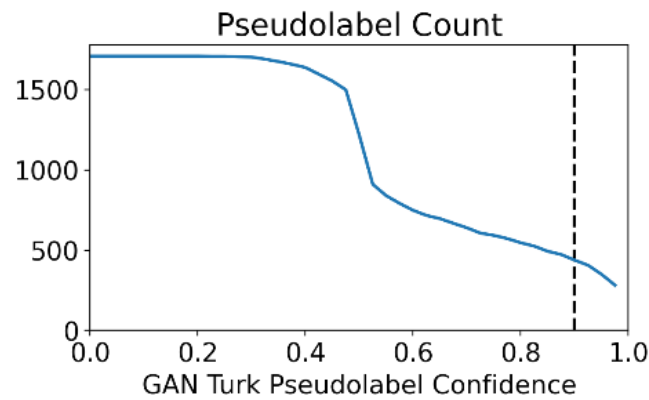
3.1 GAN Turk Pseudolabel

Unsurprisingly, pseudo-label accuracy increased with increasing pseudolabel confidence threshold p , and the number of pseudo-labels decreases as the confidence threshold increased (see Figure 8). We also found that having fewer, higher quality pseudo-labels provided better end-to-end performance than more, lower quality pseudo-labels. Analysis of base domain labeled data size suggests that there may be a non-trivial decrease in image quality as the base domain labeled data size decreases (see Figure 9). Further evaluation using JPL development tasks and Year 1 Evaluation tasks are provided in Figure 10 and Section 3.6, respectively.

Figure 8 shows accuracy and number of pseudo-labels as function of confidence threshold p . Vertical dashed line is $p = 0.9$.



(a)



(b)

Figure 8: Effects of Pseudolabel Thresholding

Reducing the amount of source data decreases image quality. In experiments, reducing the amount of labeled MNIST to 1% decreased image quality metrics (external classifier accuracy and Inception Score (IS) (Figure 9)).

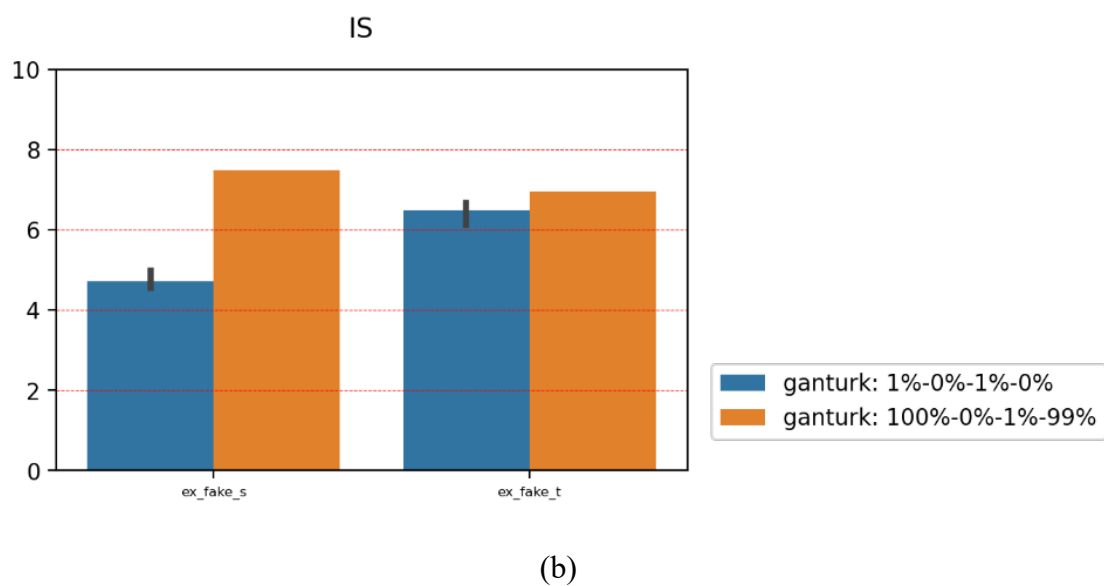
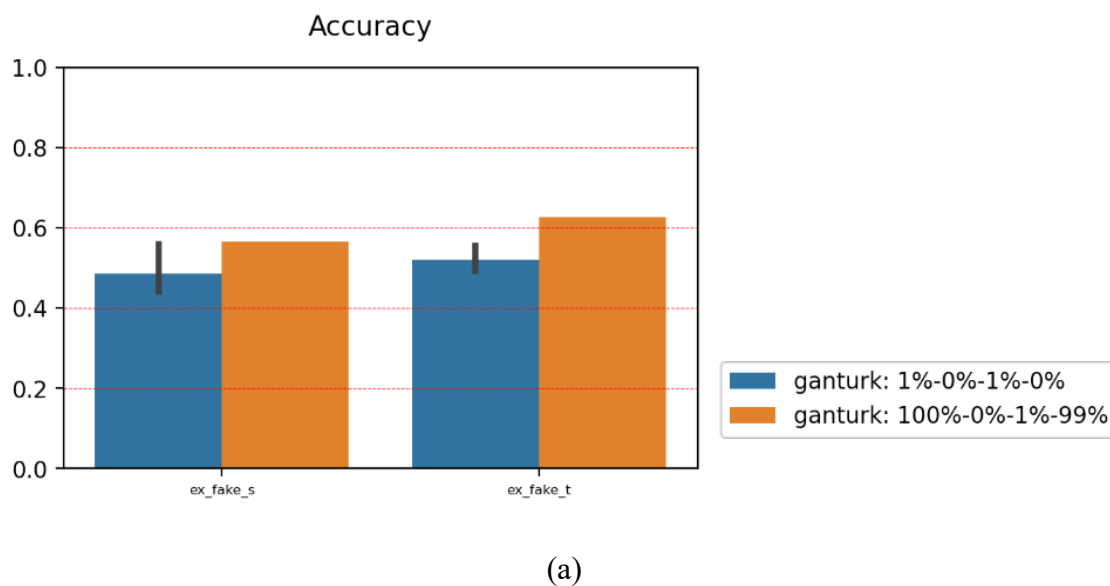


Figure 9: Effect of Source Domain Data Size

Figure 10 shows the JPL development task performance. GAN Turk Pseudo-label reliably performs better than baseline at the seed checkpoint.

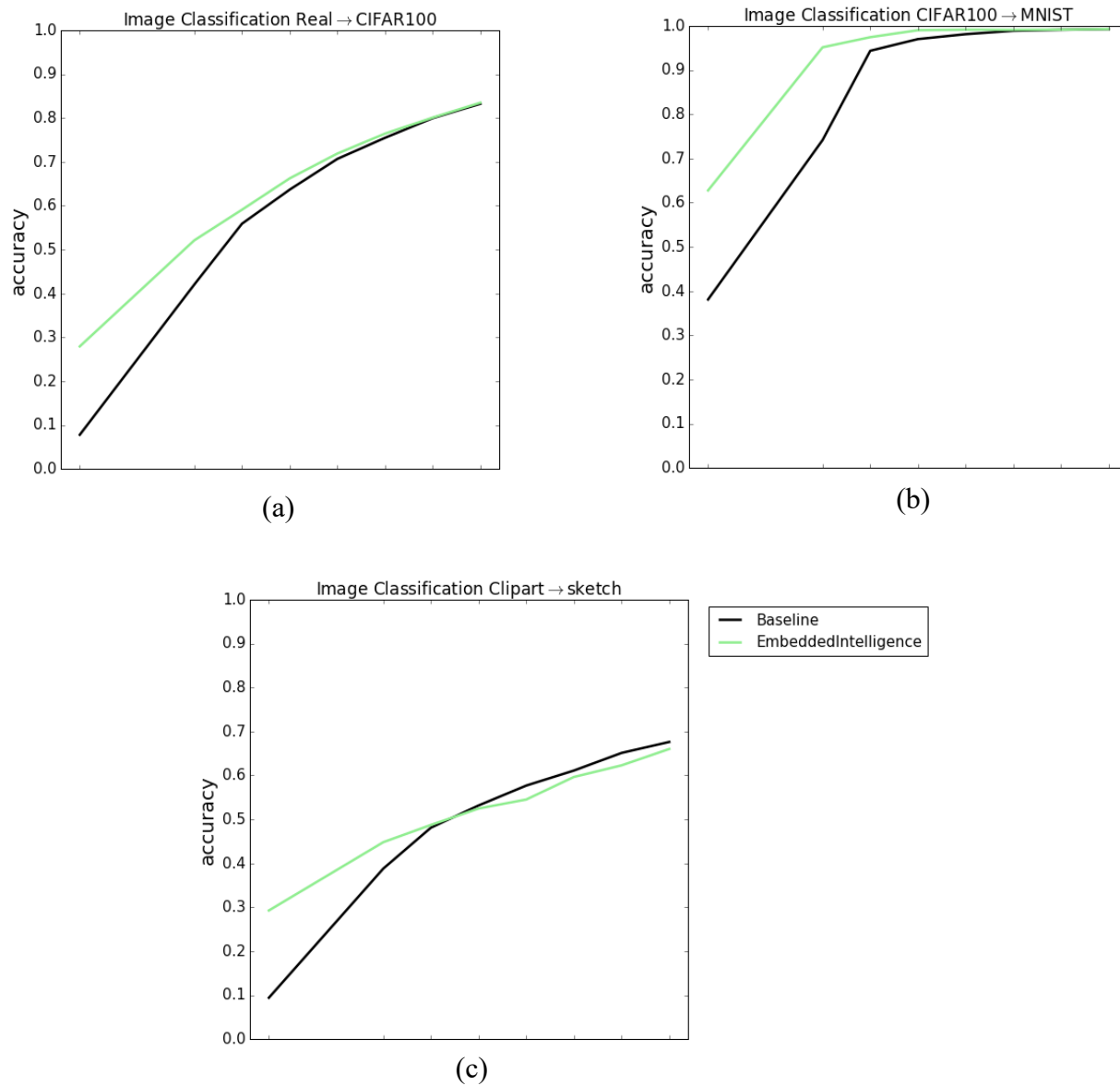


Figure 10: GAN Turk Pseudo-label Development Task Performance

3.2 GAN Turk Synthetics

For the MNIST to SVHN domain adaptation task, the addition of synthetic data for supervised learning showed significant improvement over the baseline supervised classifier at and after 25 labels per class but was approximately equivalent to random guessing until that point (Figure 11).

A limited ablation study using MNIST-SVHN task was done at 73 labels per class to compare the effect of updating GAN Turk Synthetics networks with both real and synthetic images, i.e. $L_{cls} = L_{cls,real} + L_{cls,fake}$ instead of $L_{cls} = L_{cls,real}$, while holding everything else constant.

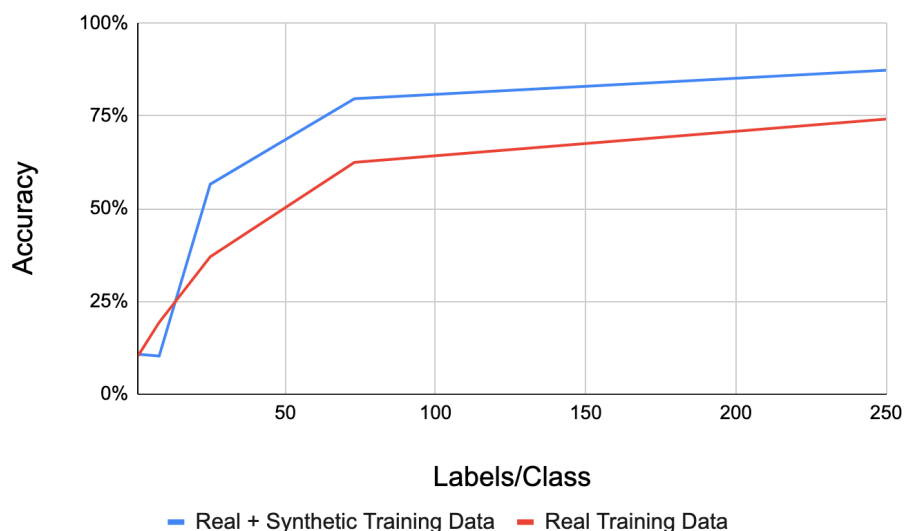
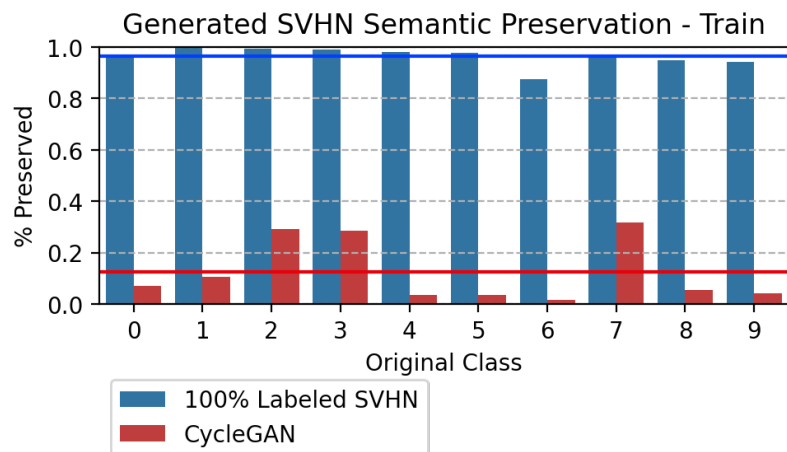
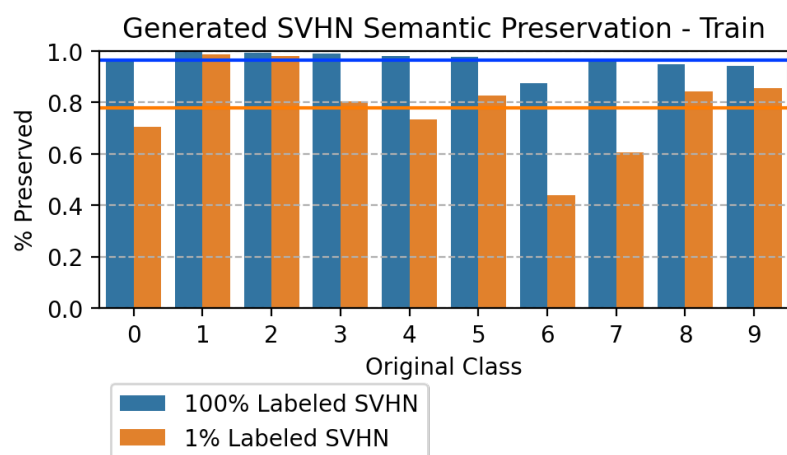


Figure 11: Benefit of GAN Turk Synthetic Images

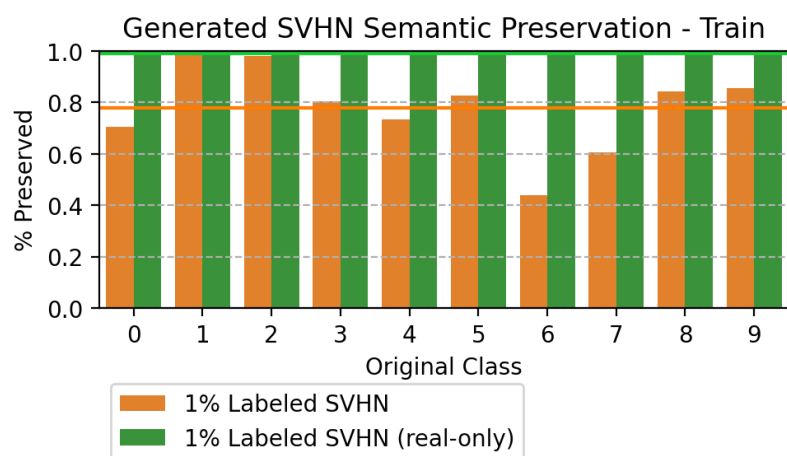
Quantitatively, updating the classifier of GAN Turk Synthetics only with real images improves semantic preservation. Comparison was done on 733 labels randomly sampled from the SVHN distribution (Figure 12). In addition, quantitative image metrics also show improvement when updating the classifier only with real images improves image quality as measured by IS and Fréchet Inception Distance (FID) (Figure 13). Our internal experimentation showed that a threshold of $p = 0.9999$ produced reliable results. This new dataset (X_{aug}, Y_{aug}) is combined with the labeled adaptation training data for supervised learning.



(a)

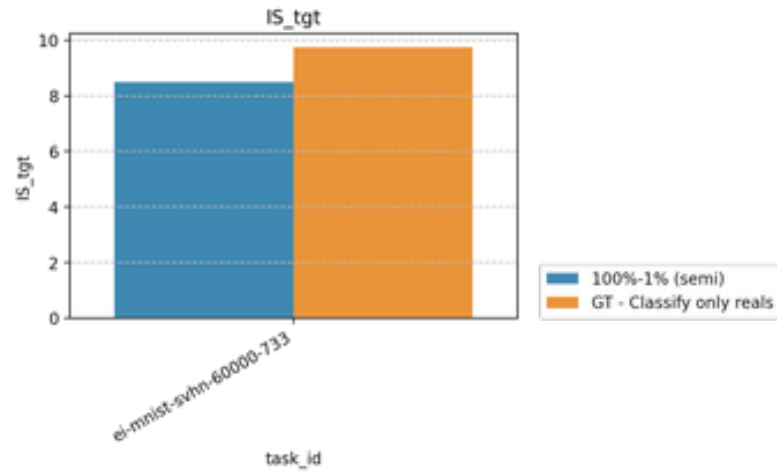


(b)

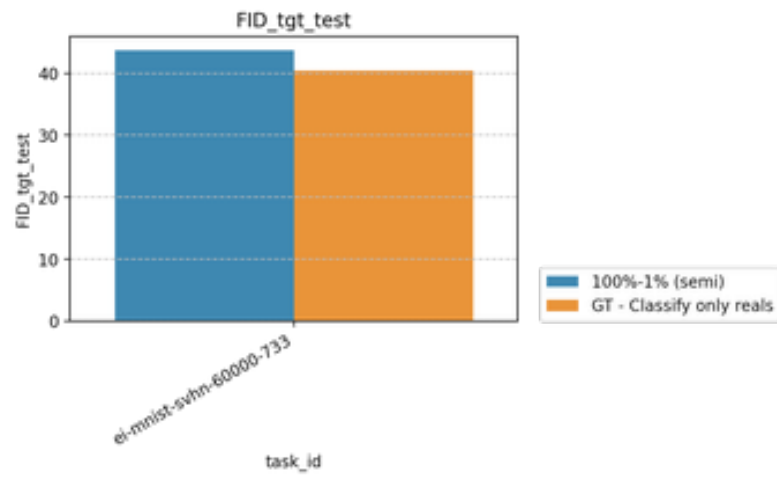


(c)

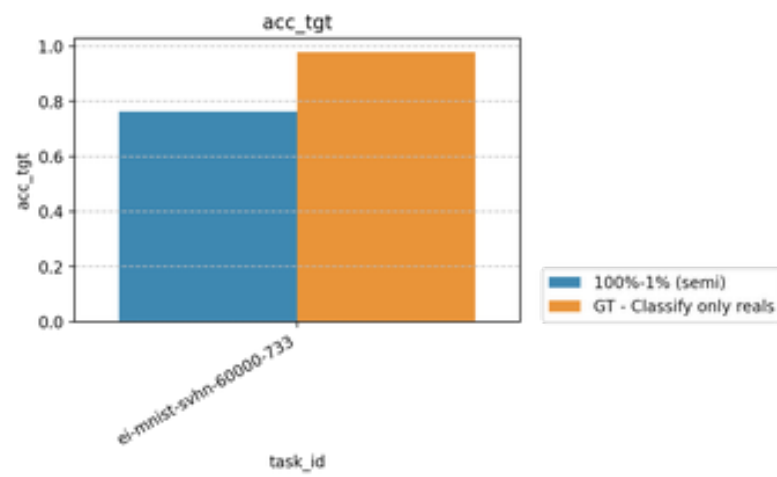
Figure 12: GAN Turk Synthetics Semantic Preservation Improvement



(a)



(b)



(c)

Figure 13: Real Images Effect on GAN Turk Synthetics Training

3.2.1 BIRDSAI

Training a baseline Faster R-CNN model on the full set of BIRDSAI labeled real data produced a 0.44 mean average precision (mAP) on the real test set. We then evaluated the Faster R-CNN model trained exclusively on synthetic data and measured 0.11 mAP when evaluating against the real test set. This significant difference in performance between the two models may be attributed to the sizable domain shift between synthetic and real data and presents an opportunity for our approach to provide benefit.

Results of the above experiment are shown in Figure 14, and indicate that our approach to the object detection task succeeds at improving model performance over baseline in the completely unsupervised case. However, with as few as even ten labeled images available, our approach fails to consistently outperform the baseline model. Consistent with experiments on the Visual Object Classes (VOC) and Clipart1k datasets, the modified approach, which utilizes staged fine-tuning, far outperforms the initial approach, which fine tunes the model a single time on a combined dataset of real labeled images and style transferred images.

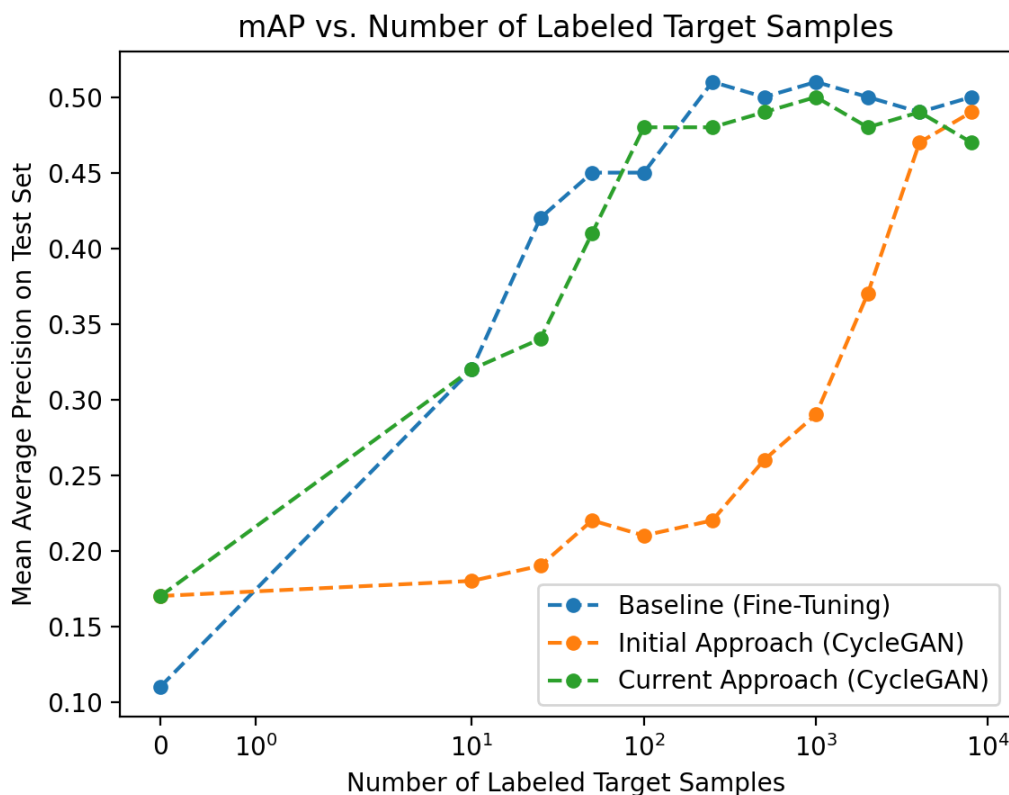


Figure 14: The performance of our object detection approach on the BIRDSAI dataset

To better explain the inability of our approach to outperform the baseline model, we took a deeper look at the images produced by CycleGAN for this task. An example input-output pair from the trained CycleGAN model is shown in Figure 15. In this case, we again note that image

semantics appear to be preserved by the style transfer, and the output image appears to be a better representation of a real IR image than the input. Specifically, the output image appears to use a larger set of gray shades and add textures that are missing in the synthetic domain.

Conversely, for the input-output pair shown in Figure 16, image semantics do not appear to be preserved. Specifically, while the synthetic input image appears to contain two objects aligned horizontally, the output image appears to introduce a third object directly below the other two. This sort of artifact results in a misalignment between the style transferred images and the corresponding annotations, which may negatively impact performance of the resulting Faster R-CNN model. This result highlights the tradeoff between our original GAN Turk model, which explicitly enforces semantic preservation at the cost of image diversity, and an unsupervised style transfer model such as CycleGAN, which demonstrated an ability to preserve image semantics and improve the performance of our object detection model for the VOC and Clipart1k datasets. However, it does not reliably preserve image semantics or consistently yield better object detection models for the BIRDSAI dataset.

The artifact highlighted above may be attributable to a natural artifact present in the BIRDSAI data; in the real data, elephants are often found in large groups within a single image, while in the synthetic data, single elephants are common. The example images in Figure 3 highlight one other difference between the real and synthetic domains; while the synthetic data assumes an observation angle directly overhead of the scene, much of the real data is recorded from a position lower on the horizon. These inconsistencies between real and synthetic data demonstrates one potential shortcoming of our approach: though the problem at first appears well-suited for a style transfer-based method, subtle differences in the dataset, which are less suited for a style transfer approach, may reduce the effectiveness.

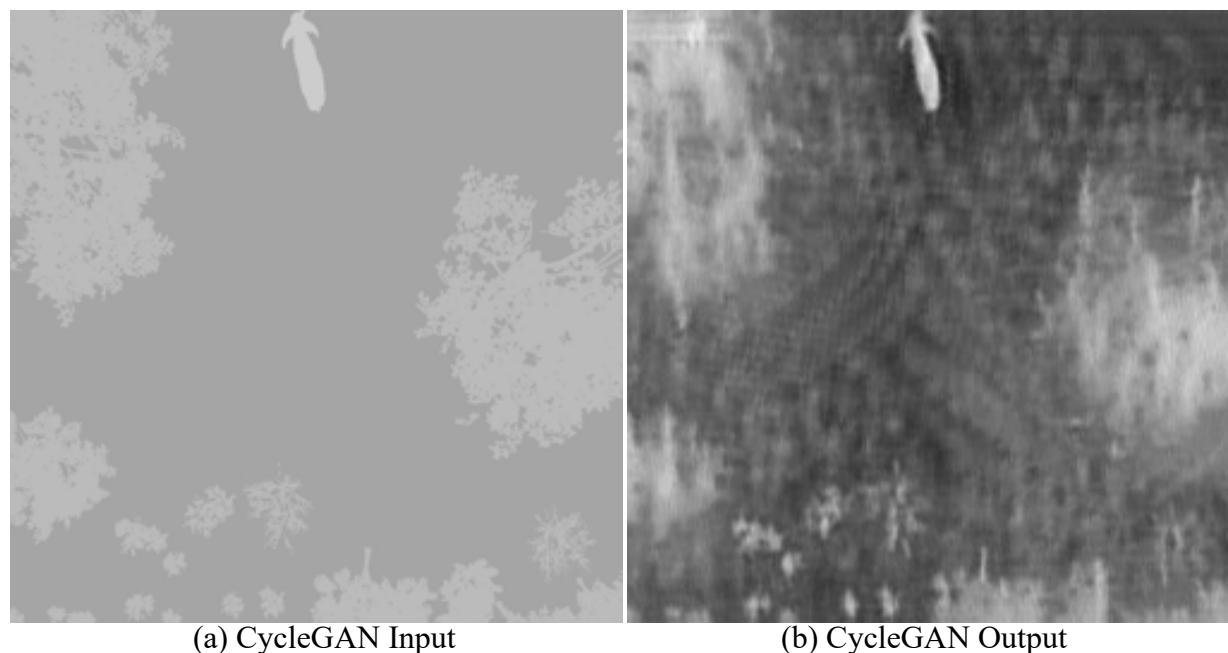


Figure 15: CycleGAN BIRDSAI Example

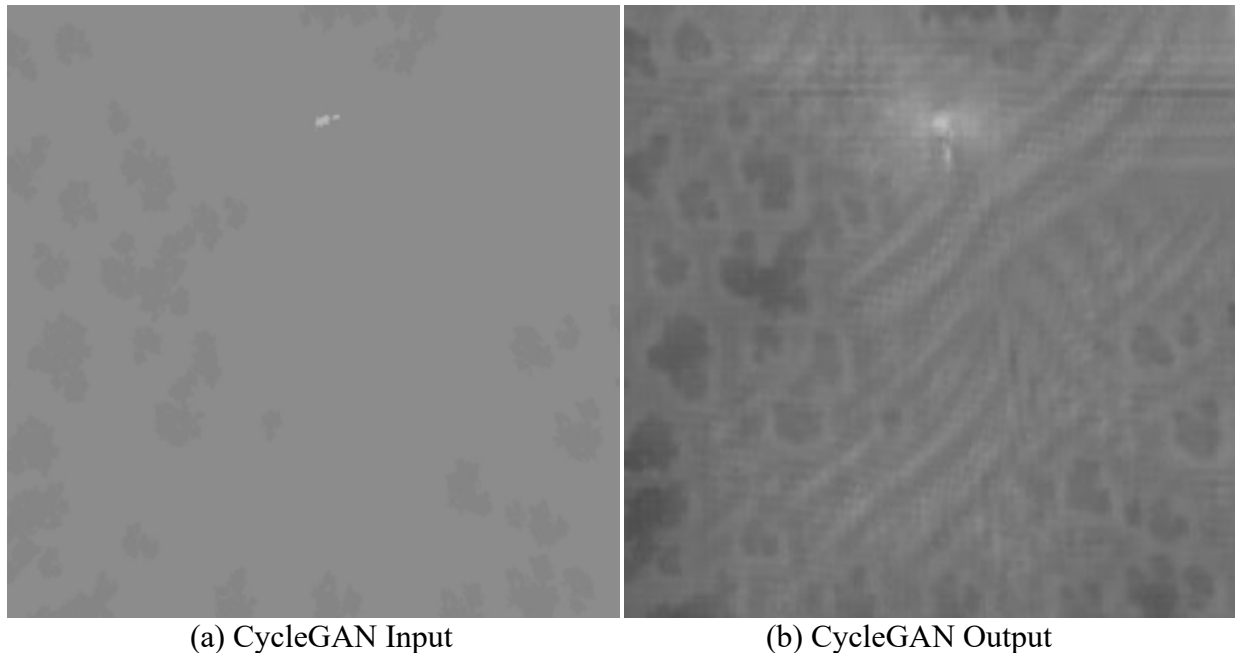
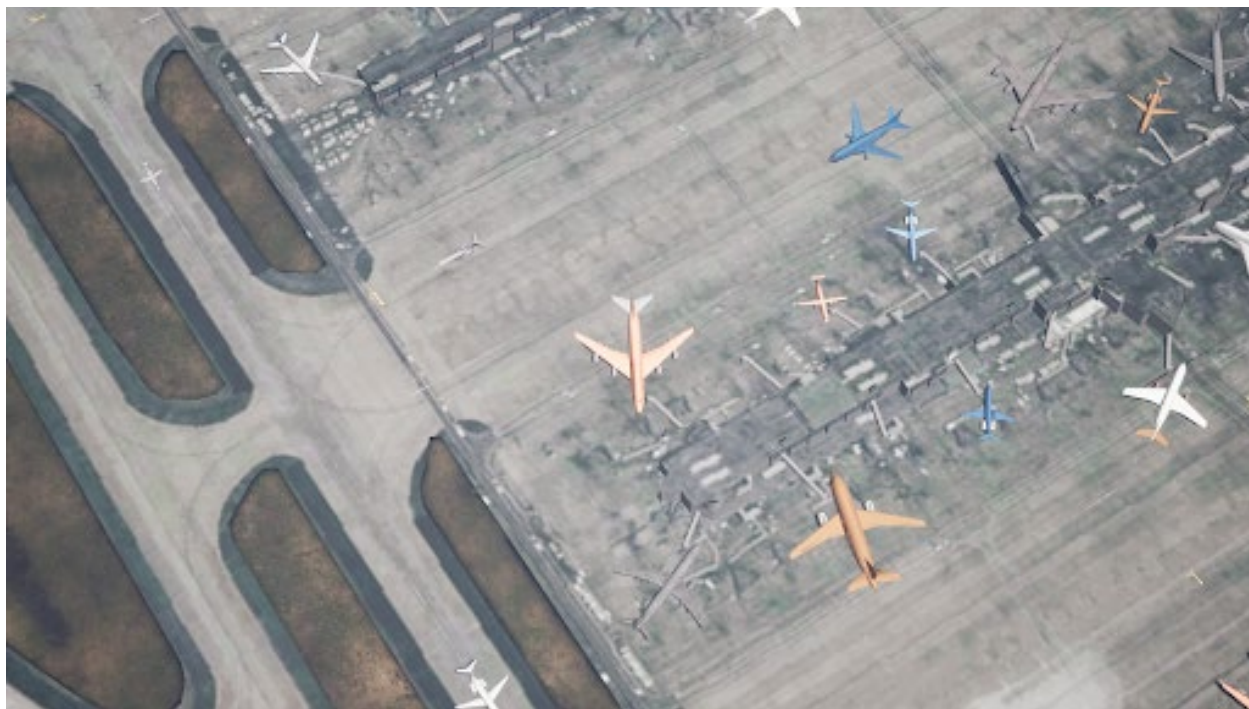


Figure 16: CycleGAN BIRDSAI Failure Mode

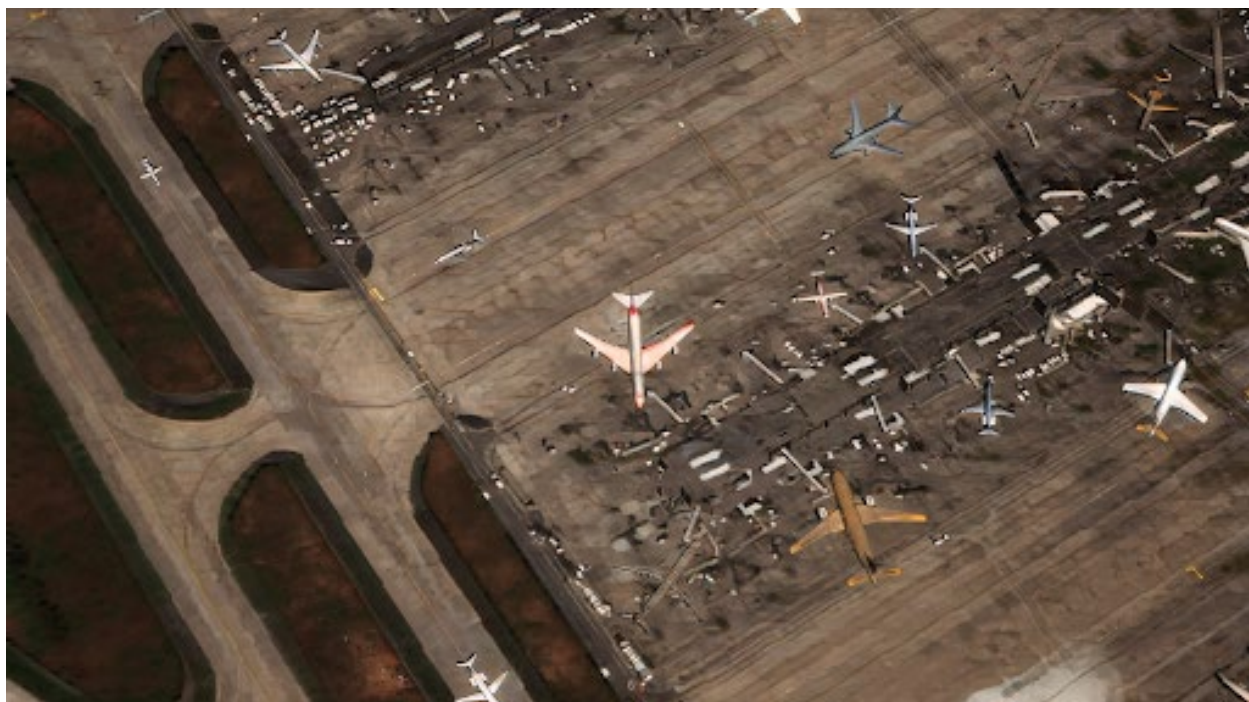
3.2.2 Rareplanes

As an upper bound on expected performance, we train a Faster R-CNN model on the full labeled real training dataset and evaluate performance on the real test set, achieving a 0.938 mAP. To verify the existence of a performance gap between domains, we then trained a Faster R-CNN model on the full labeled synthetic dataset and evaluated performance against the real test set, achieving a 0.615 mAP. All models are trained for 60,000 iterations with a learning rate of 0.001 as recommended in the Rareplanes publication [8].

Next, we trained CycleGAN to translate images from the synthetic domain to real style, then applied this style transfer to the entire Rareplanes synthetic dataset. An example input-output pair produced by CycleGAN is shown in Figure 17. From the example input-output pair, CycleGAN appears to effectively learn to replicate the darker environments present in the real Rareplanes data. However, there are also instances where CycleGAN seems to add unrealistic textures; specifically, gray planes in the input synthetic image appear to be given a similar texture in the output to the concrete runway.



(a) CycleGAN Input



(b) CycleGAN Output

Figure 17: CycleGAN Rareplanes Example

Finally, we fine-tuned the Faster R-CNN model, initially trained on synthetic data, on the style-transferred synthetic data and evaluated model performance against the real Rareplanes test set, resulting in a 0.622 mAP. This is the first pair of datasets in which we did not observe a considerable improvement in model performance in the unsupervised case, motivating further analysis. To better understand the shortcomings of the samples produced by CycleGAN, we measured image quality using Fréchet Inception Distance (FID). First, we measured an FID of 118.63 between Rareplanes synthetic and real datasets, indicating a fairly significant domain shift. Next, we measured an FID of 67.49 between the CycleGAN output and the real Rareplanes dataset, indicating that the CycleGAN output is closer in distribution to the real Rareplanes data than the synthetic Rareplanes data. This result is surprising given the lack of improvement observed in object detection performance when using the CycleGAN-generated output samples.

Additional analysis on the images generated by CycleGAN revealed one possible explanation. As shown in Figure 18, several artifacts are introduced by the translation, despite the enhancement of the color scheme of the image. First, the edges around some of the airplanes are distorted by the translation, which could cause the Faster R-CNN model to overfit unrealistic edge artifacts to detect airplanes. Additionally, yellow airplane markings along the runway in the synthetic image are translated to objects that, in some cases, resemble small airplanes. Again, this unexpected translation may be attributable to differences between the source and target datasets, such as an absence of similar runway markings in the real dataset.

This inability to derive significant benefit from style translated images in the case of the Rareplanes dataset underlines the importance of semantic preservation and high-quality style transfer. While previous experiments showed a lack of image diversity resulting from explicit enforcement of semantic preservation, inconsistencies during the style transfer process for both the BIRDSAI and Rareplanes datasets, both of which attempt to adapt seemingly high-quality synthetic data to a real domain, suggest that additional measures are necessary to ensure that images produced by a generative model meet a minimum quality standard before being used to train an external object detection model.



(a) CycleGAN Input



(b) CycleGAN Output

Figure 18: CycleGAN Rareplanes Failure Mode

3.3 Active Learning

Our internal experiments showed using this sampling technique led to performance that was no worse than our supervised learning baseline, but that potential improvements in performance were either not present or not significant depending on the datasets used during testing.

Coreset sampling did not show any consistent and significant improvement over random sampling for querying a labeled training dataset for supervised learning (Figure 19). This aligns with our previous milestone report “0001AC Internal Assessment - Computer Vision,” where we could not find consistent improvements when using uncertainty sampling. The Year 1 evaluation provided similar results (see Figure 24). Since our results appear to align with several other published results [1], we paused this line of research.

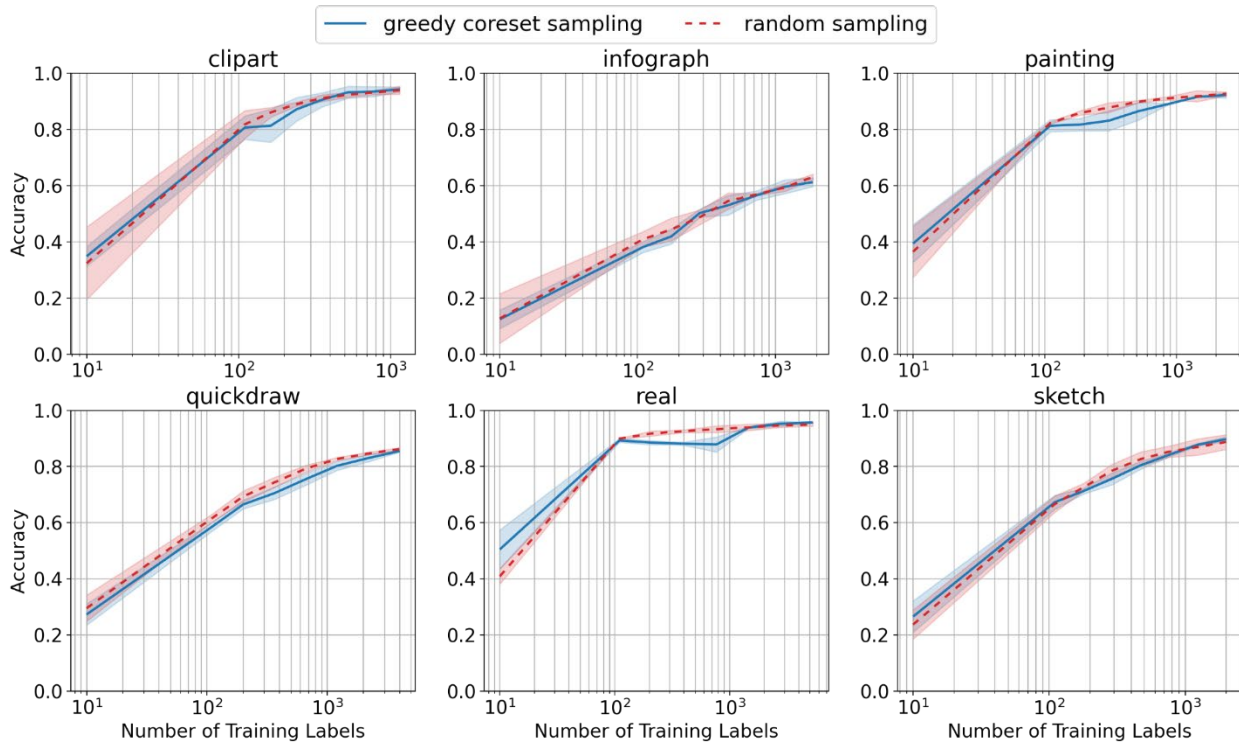


Figure 19: Coreset Sampling DomainNet Performance

3.4 STAC: Rareplanes Synthetic Domain Adaptation

We found that STAC does provide benefits over traditional approaches to unsupervised and semi-supervised domain adaptation for the Rareplanes dataset. STAC improves mAP50 by approximately 10 points for unsupervised domain adaptation (Figure 20) and 3.5 points for semi-supervised domain adaptation (Figure 21). This closes the synthetics gap by 30% and 40% for unsupervised and semi-supervised domain adaptation respectively. We find these preliminary results to be promising indicators of STAC utility in the synthetics domain adaptation setting.

STAC shows benefits in an unsupervised domain adaptation setting. When transferring a model trained only on synthetics, real data test set performance was a mAP50 of 59.09. Supervised training on all available real data had a mAP50 of 92.16. STAC achieved a performance of 69.03 and closed approximately 30% of the synthetics gap (Figure 20).

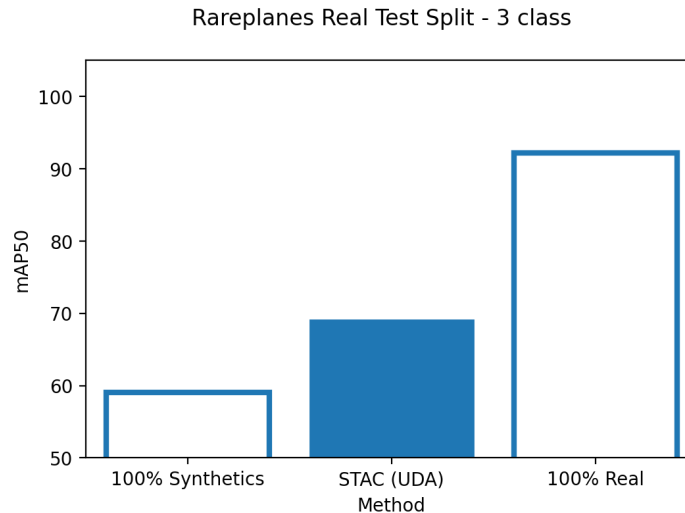


Figure 20: STAC Rareplanes Unsupervised Domain Adaptation

STAC shows benefits in a semi-supervised domain adaptation setting. When fine-tuning a model trained only on synthetics with 50 labeled real images, real data test set performance was a mAP50 of 83.10. Supervised training on all available real data had a mAP50 of 92.16. STAC achieved a performance of 86.76 and closed approximately 40% of the synthetics gap (Figure 21).

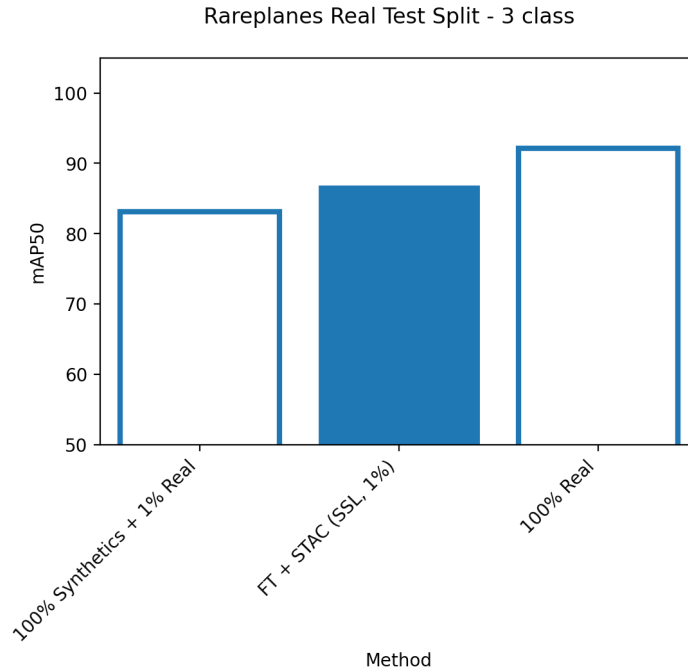


Figure 21: STAC Rareplanes Semi-supervised Domain Adaptation

3.5 STAC: xView Haul Truck

In our April 2022 PI meeting presentation and 0001AK milestone report, we present the negative effects of training on a curated dataset that ensured at least one object of interest in each image and then deploying on a test dataset that contains mostly empty images. This was addressed by including empty training images in the model training process. We hypothesized that the improvements were due to negative images still providing novel information and the additional environments/lighting conditions/etc.

We sought to extend this finding to the more applicable semi-supervised learning setting by also allowing STAC student training on images pseudolabeled as empty which we initially presented in our October 2022 PI meeting presentation. In particular, we trained the teacher models on a small subset of labeled xView training images, totaling 2,500 randomly selected 512×512 tiles. Within these 2,500 tiles, 11 are non-empty for a total of 38 instances of Haul Trucks. Then, assuming the remainder of the training dataset is unlabeled, we trained a student model using self-training and consistency regularization over both images pseudolabeled as empty and containing one or more objects. We observed an improvement in student performance compared to standard student training when using the same teacher model. More importantly, we also saw that the simple addition of empty image pseudolabel show STAC was an effective and beneficial technique even though standard STAC training suggested that the methodology was ineffective and harmful (see Figure 22).

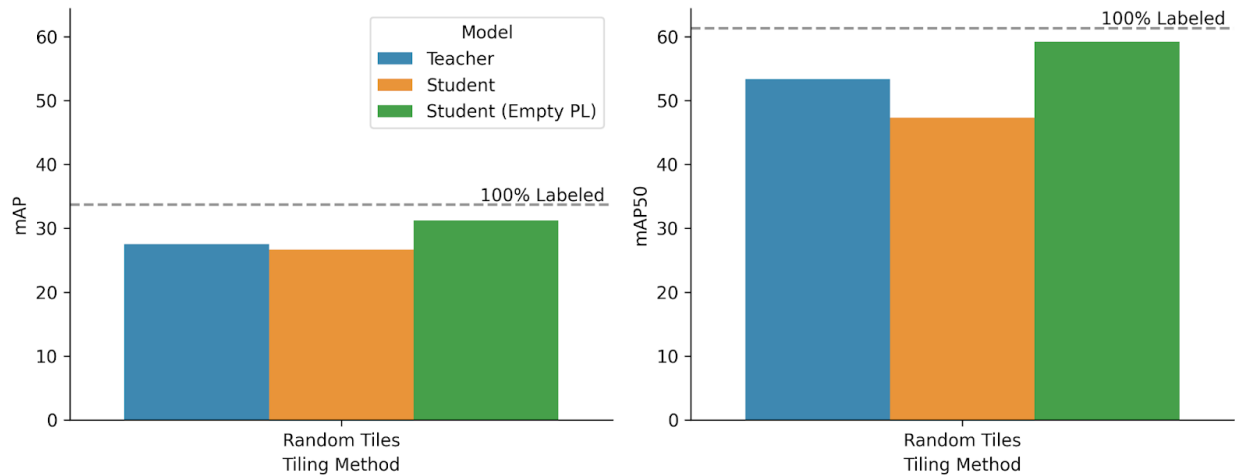


Figure 22: Empty Pseudolabeling Effect on STAC

3.6 Year 1 Evaluation

Our Year 1 Evaluation system utilized coreset active sampling (see Section 2.3) for the base stages and GAN Turk Pseudo-label for adaptation stages of image classification tasks. Coreset sampling was shown to be comparable to random sampling and the provided JPL baseline (see Figure 23). This matched our findings when performing a self-evaluation using JPL development tasks (see Figure 10). The GAN Turk Pseudo-label method was comparable or better than the baseline method for both the JPL Year 1 Evaluation image classification tasks (see Figure 24) and the provided JPL development tasks (see Figure 10). Notably, we showed significant improvement at the seed checkpoint over the baseline. We hypothesize the improvement is caused by the GAN Turk generated pseudo-labels but still require an ablation test to compare the effects of pseudo-labels between traditional pseudo-labels and GAN Turk generated pseudo-labels.

Figure 23 shows the JPL Year 1 evaluation of coresets sampling. The black Baseline curve represents the performance obtained using random sampling. The teal curve uses the Coreset-based sampling strategy.

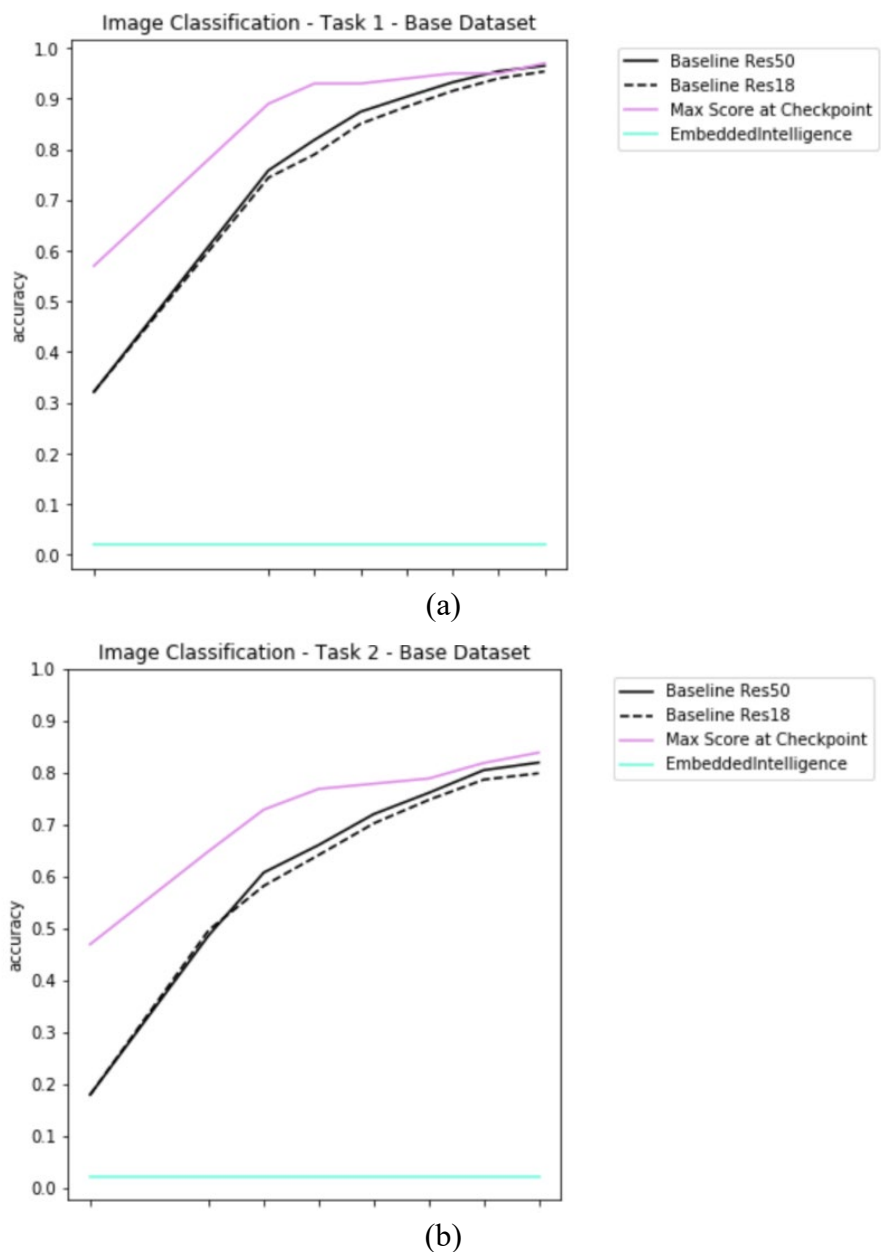


Figure 23: Year 1 Coreset Sampling Evaluation Results

Base stage performance not shown because GAN Turk Pseudo-labeling is designed to reduce label requirements given an initial domain. GAN Turk Pseudo-label performs as well if not better than the baseline method (Figure 24).

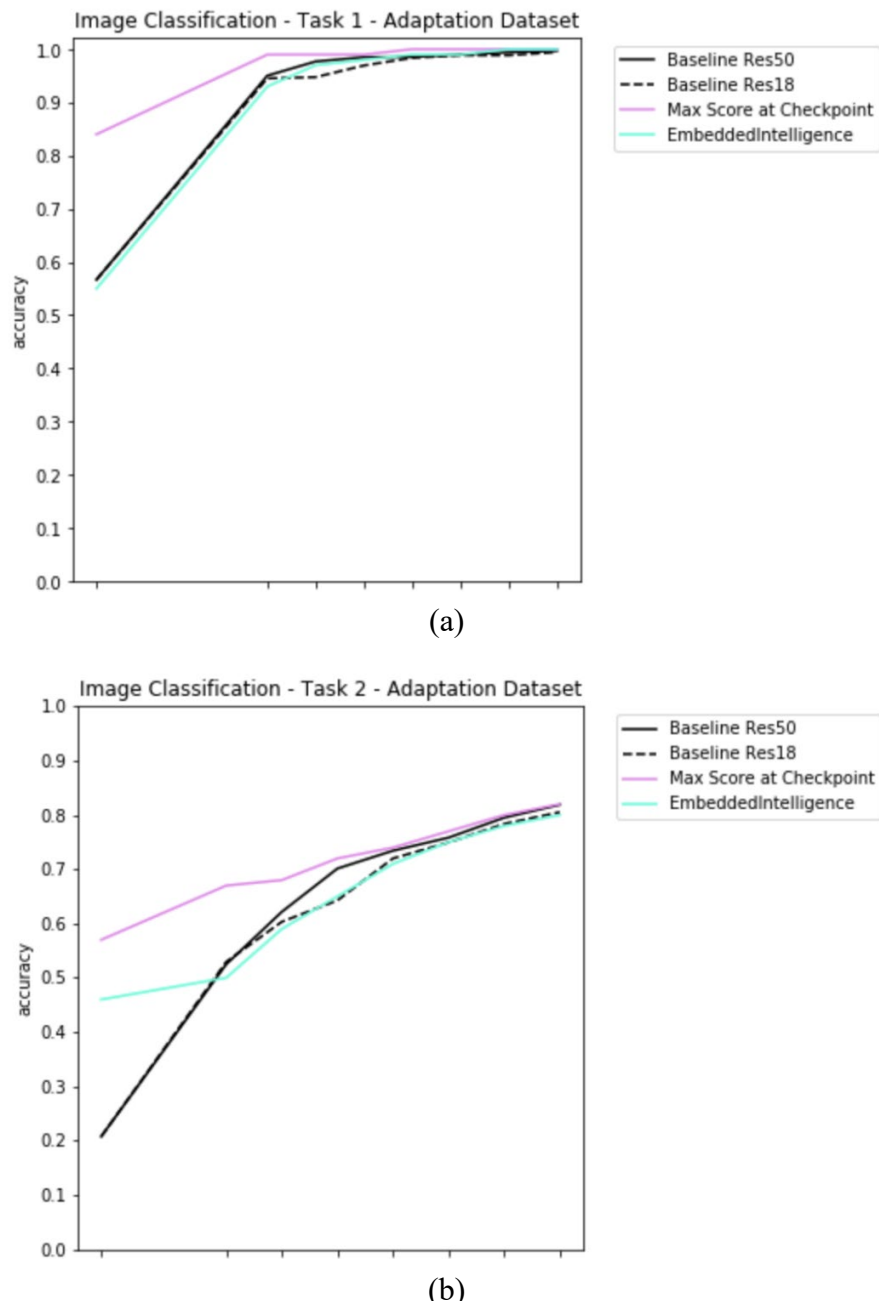


Figure 24: Year 1 GAN Turk Pseudo-labeling Evaluation Results

3.7 Year 2 Evaluation

Our system consists of three different methodologies that are activated at different time points during the evaluation. Our core methodology, GAN Turk Synthetics, is utilized in the second half of the adaptation stage. During the entirety of the base stage, we utilize RandAugment [3] with random sampling to generate predictions. For the first checkpoint in the adaptation stage, we use naive transfer of the source RandAugment classifier trained in the last base stage checkpoint. In the remainder of the first half of the adaptation stage, the combination of labeled base and adaptation domain samples selected by random sampling is used to fine-tune the learned RandAugment classifier from the final base stage checkpoint as a simple form of domain adaptation. In the second half of the adaptation stage, we use GAN Turk Synthetics and randomly sample real target-domain samples. Table 1 summarizes the approaches selected for each checkpoint of the evaluation.

Table 1: Breakdown of methodologies applied during a single Year 2 evaluation task.

Checkpoint	Domain	Methodology
1-8	Base	RandAugment
1	Adaptation	Naive Source Transfer
2-4	Adaptation	Source Fine-Tuning
5-6	Adaptation	GAN Turk Synthetics
7	Adaptation	Checkpoint Skipped
8	Adaptation	GAN Turk Synthetics

This selection of methodologies was motivated by our focus on using GAN Turk Synthetics for domain adaptation. GAN Turk Synthetics attempts to leverage CycleGAN [15] for domain adaptation by using transformed source images as additional training data in the target domain. In order for these transformed source images to be useful, they need to have the same “style” as the target images and still preserve the source class semantics after transformation. The latter requirement motivated the changes to our Year 1 approach that resulted in GAN Turk Synthetics. We use GAN Turk Synthetics only on the later checkpoints of the adaptation stage because the method is semi-supervised and has poor performance when trained on single-digit label count. This mirrors similar results where vanilla generative adversarial network (GAN)s perform poorly when only provided very limited amounts of data.

Because of these current limitations, the other methodologies were selected to complement GAN Turk Synthetics, but note these other methodologies are not our main focus. RandAugment was selected for the base stage because it has been the motivation for many state of the art methods

that utilize data augmentation for learning with less labels, e.g. FixMatch [9]. Similarly, naive source transfer and fine-tuning of a source classifier are common and straightforward domain adaptation techniques.

Evaluation results (Figures 25 to 27) showed that our particular approach does not necessarily provide a means to learn with less labels. This result is not surprising given our internal results immediately prior to submitting for evaluation and also given the fact that we trained on 64×64 images instead of 224×224 images for speed. The speed improvement from using 64×64 images came at the price of reduced information content in the training images and reduced model performance. We believe that the performance gap would be closed if the same image size was used for both our submission and the baseline method.

More specifically, our base stage results showed no significant improvement in accuracy as the number of labels were increased from one to eight labels per class. This could indicate that techniques like RandAugment may have a minimum label requirement in order to function correctly. Given the results that fine-tuning (adaptation checkpoints two through four) generally performed the same or worse than naive source transfer, our fine-tuning approach likely caused overfitting to the small amount of labeled target data. Since the performance gap between our approach and the baseline in adaptation checkpoints 5-8 holds approximately constant or closes significantly, we believe that increasing the resolution of our generated images and training image would boost performance up to the baseline values. We do not believe higher image resolutions would exceed baseline performance because our improvement in semantic preservation may have come at the price of diversity and novelty in our generated images.

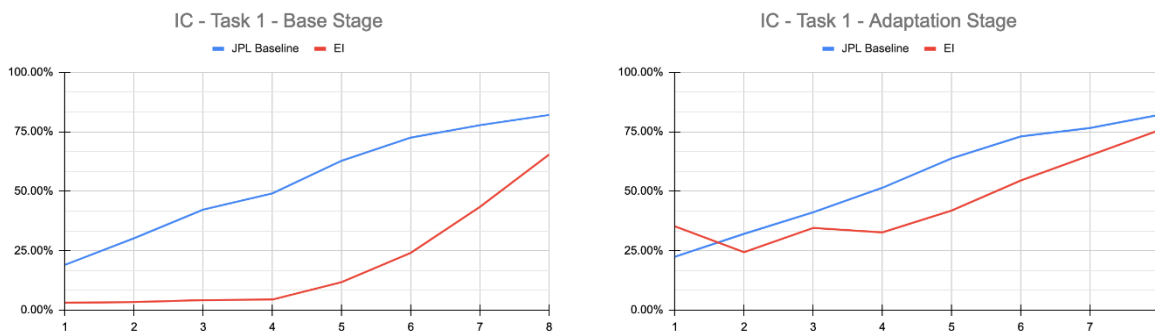


Figure 25: Year 2 Evaluation Result: Image Classification Task 1

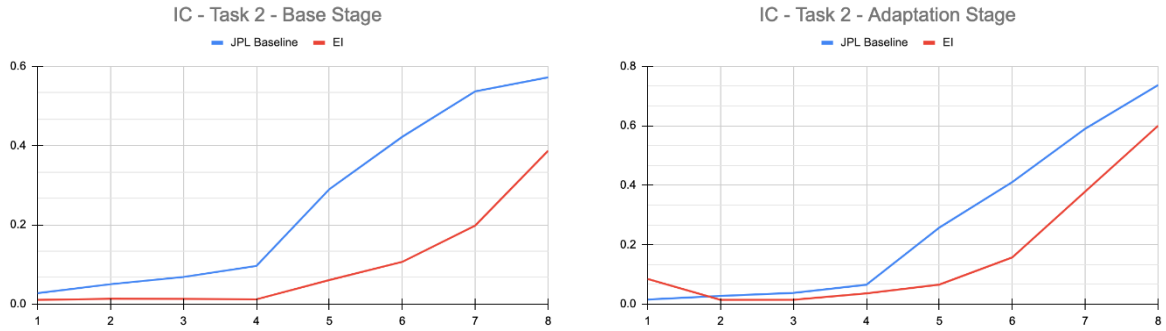


Figure 26: Year 2 Evaluation Result: Image Classification Task 2

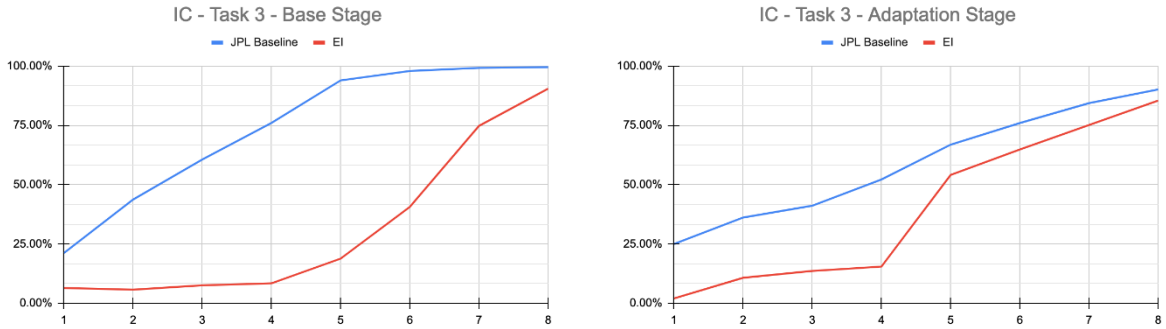


Figure 27: Year 2 Evaluation Result: Image Classification Task 3

Even though our Year 2 evaluation did not show a reduction in label requirements, we still believe GAN-based domain adaptation is a potential path to reducing labeled data requirements for deep learning and believe the evaluation results reflect only on the specific implementation we submitted. This is supported by existing works, e.g. [5, 11], especially in unsupervised domain adaptation, and we provide further evidence in the following section.

3.8 Year 3 Evaluation

Our Year 3 Evaluation submission focused exclusively on applying self-training and consistency regularization to object detection tasks. We used our own implemented version of STAC with the modification of allowing empty image pseudolabeling. In order to satisfy evaluation time constraints, we only used STAC for a subset of checkpoints with a reference Faster R-CNN implementation for other checkpoints (see Tables 2 and 3). The reference Faster R-CNN implementation is particularly important for the medium size task (Task 3) since STAC training was set to what we believe to be a sub-optimal number of iterations for optimization thereby reducing the quality of the model produced. We believe matching the reduction in number of

updates for the Faster R-CNN reference baseline provides a more accurate but perhaps incomplete assessment of the STAC methodology.

The performance from the Year 3 Evaluation for our submitted object detection system is shown in Figures 28-30. Overall, these results are in line with our expectations. We see that there is some natural variation in system performance that likely depends on which samples are randomly selected for labeling as well as natural variations due to stochastic optimization from random batch construction. That is why we believe we see our system and the baseline perform roughly the same, but not exactly the same, despite using the same methodology for checkpoints specified as using Faster R-CNN as the method.

Table 2: Breakdown of methodologies applied during a single Year 3 medium evaluation task.

Checkpoint	Methodology
1-3	Faster R-CNN (w/ 35k reduced iterations)
4-6	STAC (w/ 35k reduced iterations)
7	Skipped
8	Faster R-CNN (w/ 35k iterations)

Table 3: Breakdown of methodologies applied during a single Year 3 large evaluation task.

Checkpoint	Methodology
1-3	Faster R-CNN
4-6	STAC
7	Skipped
8	Faster R-CNN

The results showing that our STAC based methodology under performs the JPL baseline is also consistent with the results we saw during development and highlighted in Section 2.4. We suspect that part of the reason for the dip in performance for the STAC checkpoints could be due to inappropriate hyper-parameters. Further evidence for this hypothesis comes from looking at checkpoint 6 in the adaptation stage of OD Task 3. Despite using the same methodology (Faster - RCNN) as the JPL baseline, we see that our system is substantially lower for the sixth checkpoint.

As noted above, we had to select different hyper parameters than the standard JPL baseline, in particular the learning rate schedule. Since this was the only difference between our method and the baseline for this sixth checkpoint this indicates that the hyper parameters we selected were not well suited for the datasets used in the year 3 evaluation.

The fact that our STAC methodology was near the baseline (sometimes slightly higher, but also lower at times) for checkpoints 4 and 5 is also consistent with our expectations. Based on our internal results highlighted in Figure 22, we expected that our STAC system would perform at or below the baseline when trained on datasets that are carefully curated to have at least one object of interest present in every image (both training and test data). The benefit of our self-training and consistency regularization showed significant benefits when trained on more operationally relevant datasets that include a significant amount of data that is empty, i.e., does not contain any objects of interest. Unfortunately, the year 3 evaluation datasets did not align with this experimental setup. Therefore, while we were able to show the benefits of this simple but powerful method in our development, we could not independently verify its general applicability as part of the official year 3 evaluation. That said, we fully expect that this methodology would be beneficial for any method that uses self-training and consistency regularization, which is a fair number of teams on the LwLL program. We would be happy if this simple, but powerful modification was incorporated into future systems for learning with less labeling. Figures 28-30 show Object Detection Tasks 1-3 performance of our method versus JPL baseline. mAP vs checkpoint number.

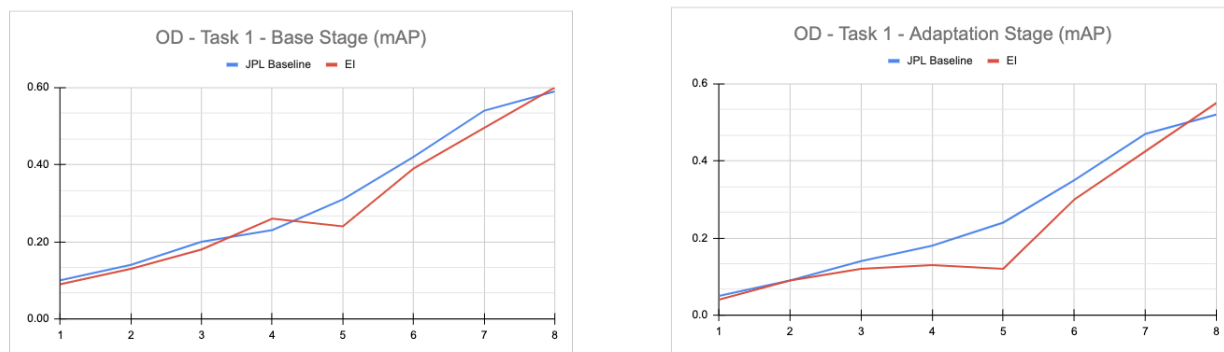


Figure 28: Year 3 Evaluation: Object Detection Task 1

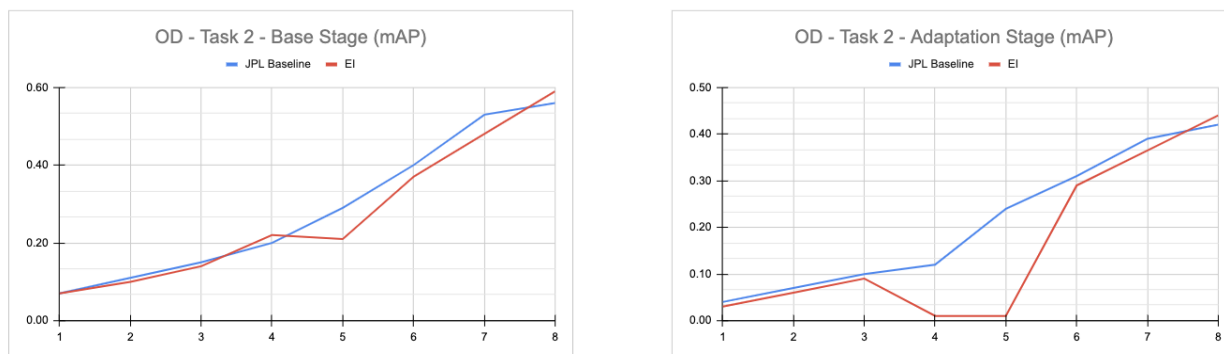


Figure 29: Year 3 Evaluation: Object Detection Task 2

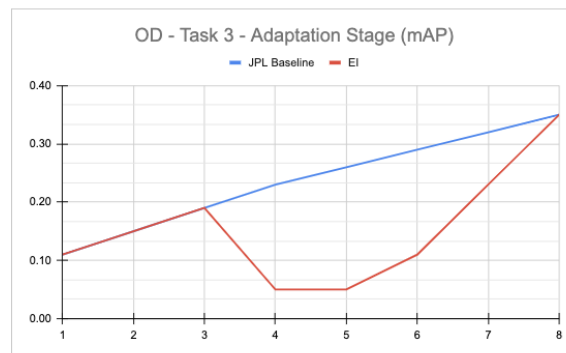
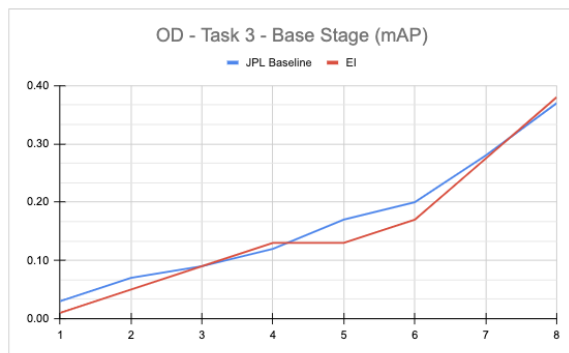


Figure 30: Year 3 Evaluation: Object Detection Task 3

4 Software

We submitted our code for our system to the ei/ei-lwll-pipeline repository (URL: <https://gitlab.lollllz.com/ei/ei-lwll-pipeline>) located in the program wide GitLab instance provided by JPL. Our submission complied with evaluation requirements by generating a Docker image for the evaluation team to run against image classification tasks (Figure 32). An example of how our system may be run is provided in Listing 1. We received confirmation that we completed the final evaluation on December 19, 2022.

Listing 1: Shell command for executing our system.

```
1 docker run --env-file image-classification -env-eval.l i s t  $\mathbb{W}$ 
2   -e LWLL_TA1_PROB_TASK=<TASK_ID>  $\mathbb{W}$ 
3   -v /lwll/development/development:/lwll/development:ro \
4   -v /lwll/development/external:/lwll/external:ro  $\mathbb{W}$ 
5   -v /lwll/evaluation:/lwll/evaluation:ro \
6   -v /lwll/extra/lwll-ei:/lwll/extra  $\mathbb{W}$ 
7   --ipc=host --gpus=8  $\mathbb{W}$ 
8   embedintel/lwll:v1.3.99
```

lwll-ei	10/28/22 6:35PM
Contact Info	
Email Address	Slack Username
bryant@embedintel.com	@ Bryant Menn
Runtime Specifications	
Repository URL	
https://gitlab.lolllz.com/ei/ei-lwll-pipeline:v1.3.99	
Problem Types	GPU Count
["object_detection"]	8
Comments	
Submission Errors (Check Back Day After Submission)	
None	
Submitted by bmenn	
lwll-ei	10/24/22 3:56PM
lwll-ei	9/10/21 1:47PM
lwll-ei	9/1/21 10:56PM
lwll-ei	8/30/21 7:51PM
lwll-ei	8/3/21 12:08AM
lwll-ei	4/3/21 12:32AM
lwll-ei	3/26/21 9:28PM

Figure 31: Screenshot of successful system submission for evaluation

The screenshot shows the GitLab interface for a pipeline. The left sidebar contains navigation links: E, EI LwLL Pipeline, Project information, Repository, Issues (0), Merge requests (0), CI/CD, Pipelines (selected), Editor, Jobs, Schedules, Security & Compliance, Deployments, Monitor, Infrastructure, Packages & Registries, Analytics, Wiki, Snippets, and Settings.

The main content area displays the pipeline details for 'EI LwLL Pipeline > Pipelines > #5906'. A green 'passed' status is shown, indicating the pipeline was triggered 4 days ago by Bryant Menn. The title of the pipeline is 'LWLL-398 Sets up final solvers for evaluation'. Below the title, it states '4 jobs for v1.3.99 in 4 minutes and 33 seconds (queued for 3 seconds)'. A 'latest' tag is visible, and a commit hash '1a2cce5f' is shown. A message indicates 'No related merge requests found.'.

The pipeline progress bar shows four stages: 'Build-packages' (lwl-package), 'Build-images' (dmc_build), 'Test-images' (dmc_test), and 'Tag-images' (dmc_tag). All stages are marked with a green checkmark, indicating they completed successfully.

Figure 32: Screenshot of successful Docker image build on DMC infrastructure for evaluation

5 Conclusion

This final report and deliverable documents Embedded Intelligence's research and development efforts on the LwLL program, as well as our successful integration of our year 3 software system into the Government testbed for the official year 3 evaluation. Our team's final evaluation by the Government (JPL/DMC) was completed on December 19, 2022. This included software delivery, integration, and successful evaluation of our system against all three object detection tasks.

In this report, as with all our deliverables, we documented both what methodologies showed promise as well as those that may be less beneficial to pursue based on sound scientific methods and empirical results analyzed on this program. We are thankful for the opportunity to be part of the LwLL program and to contribute to the Government's and the broader research community's understanding of what methodologies may be most effective for learning with less labeling.

References

- [1] William H. Beluch, Tim Genewein, Andreas Nurnberger, and Jan M. Kohler. The Power of Ensembles for Active Learning in Image Classification. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9368–9377, Salt Lake City, UT, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00976. URL <https://ieeexplore.ieee.org/document/8579074/>.
- [2] Elizabeth Bondi, Raghav Jain, Palash Aggrawal, Saket Anand, Robert Hannaford, Ashish Kapoor, Jim Piavis, Shital Shah, Lucas Joppa, Bistra Dilkina, and Milind Tambe. BIRDSAI: A Dataset for Detection and Tracking in Aerial Thermal Infrared Videos. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1736–1745, Snowmass Village, CO, USA, March 2020. IEEE. ISBN 9781728165530. doi: 10.1109/WACV45572.2020.9093284. URL <https://ieeexplore.ieee.org/document/9093284/>.
- [3] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18613–18624. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/d85b63ef0ccb114d0a3bb7b7d808028f-Paper.pdf>.
- [4] Jeevan Devaranjan, Amlan Kar, and Sanja Fidler. Meta-Sim2: Unsupervised Learning of Scene Structure for Synthetic Data Generation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 715–733, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58520-4. doi: 10.1007/978-3-030-58520-4_42.
- [5] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle Consistent Adversarial Domain Adaptation. In *International Conference on Machine Learning*, pages 1989–1998. PMLR, July 2018. URL <http://proceedings.mlr.press/v80/hoffman18a.html>. ISSN: 2640- 3498.
- [6] Dong-Hyun Lee. Pseudo-Label : The Simple and Efficient Semi Supervised Learning Method for Deep Neural Networks. 2013.
- [7] Ozan Sener and Silvio Savarese. Active Learning for Convolutional Neural Networks: A Core-Set Approach. *arXiv:1708.00489 [cs, stat]*, June 2018. URL <http://arxiv.org/abs/1708.00489>. arXiv: 1708.00489.
- [8] Jacob Shermeyer, Thomas Hossler, Adam Van Etten, Daniel Hogan, Ryan Lewis, and Daeil Kim. RarePlanes: Synthetic Data Takes Flight. *arXiv:2006.02963 [cs]*, November 2020. URL <http://arxiv.org/abs/2006.02963>. arXiv: 2006.02963.
- [9] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A. Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. FixMatch: Simplifying Semi Supervised Learning with Consistency and Confidence. *Advances in Neural Information*

- Processing Systems*, 33:596–608, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/06964dce9addb1c5cb5d6e3d9838f733-Abstract.html>.
- [10] Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A Simple Semi-Supervised Learning Framework for Object Detection. *arXiv:2005.04757 [cs]*, December 2020. URL <http://arxiv.org/abs/2005.04757>. arXiv: 2005.04757.
- [11] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised Cross Domain Image Generation. *ICLR*, 2017. URL <https://openreview.net/forum?id=Sk2Im59ex>.
- [12] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization. *arXiv:1804.06516 [cs]*, April 2018. URL <http://arxiv.org/abs/1804.06516>. arXiv: 1804.06516.
- [13] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J. Cashman, and Jamie Shotton. Fake It Till You Make It: Face Analysis in the Wild Using Synthetic Data Alone. pages 3681–3691, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Wood_Fake_It_Till_You_Make_It_Face_Analysis_in_the_ICCV_2021_paper.html.
- [14] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised Data Augmentation for Consistency Training. In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268. Curran Associates, Inc., 2020. URL <https://proceedings.nips.cc/paper/2020/hash/44feb0096faa8326192570788b38c1d1-Abstract.html>.
- [15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *arXiv:1703.10593 [cs]*, August 2020. URL <http://arxiv.org/abs/1703.10593>. arXiv: 1703.10593.

List of Acronyms

BIRDSAI	Benchmarking IR Dataset for Surveillance with Aerial Intelligence
CyCADA	Cycle-Consistent Adversarial Domain Adaptation
DARPA	Defense Advanced Research Projects Agency
DMC	Data Machines Corporation
DTN	Domain Transfer Network
EMA	Exponential Moving Average
FID	Fréchet Inception Distance
GAN	Generative Adversarial Network
GSD	Ground Sampling Distance
JPL	Jet Propulsion Laboratory
IR	Infrared
IS	Inception Score
mAP	Mean Average Precision
MNIST	Modified National Institute of Standards and Technology
NASA	National Aeronautics and Space Administration
OD	Object Detection
R-CNN	Region-Based Convolutional Neural Networks
ROIs	Regions Of Interest
STAC	Self-Training (via pseudolabel) and Augmentation driven Consistency regularization
SVHN	Street View House Numbers
UAV	Unmanned Aerial Vehicle
VOC	Visual Object Classes