Efficient hyperparameter optimization for ATR using homotopy parametrization

Sophia Abraham[a], Jeffrey Kinnison[a], Zachary Miksis[a], Domenick Poster[b], Suya You[b], Jonathan D. Hauenstein[a], and Walter Scheirer[a]

[a]University of Notre Dame, Notre Dame, IN 46556, USA
[b]DEVCOM Army Research Laboratory, Adelphi, MD, USA

ABSTRACT

Deep learning has expedited important breakthroughs in research and commercial applications for next-generation technologies across many domains including Automatic Target Recognition (ATR). The success of these models in a specific application is often attributed to optimized hyperparameters: user-configured values controlling the model's ability to learn from data. Tuning hyperparameters however remains a difficult and computationally expensive task contributing to deficient ATR model performance compared to set requirements.

We present the efficacy of applying our developed hyperparameter optimization method to boost the effectiveness and performance of any given optimization method. Specifically, we use a generalized additive model surrogate homotopy hyperparameter optimization strategy to approximate regions of interest and trace minimal points over regions of the hyperparameter space instead of ineffectively evaluating the entire hyperparameter surface. We integrate our approach into SHADHO (Scalable Hardware-Aware Distributed Hyperparameter Optimization) a hyperparameter optimization framework that computes the relative complexity of each search space and then monitors the performance of the learning task over the trials.

We demonstrate how our approach effectively finds optimal hyperparameters for object detection by conducting a model search to optimize multiple object detection algorithms on a subset of the DSIAC ATR Algorithm Development Image Database and finding models that achieve comparable or lower validation loss in fewer iterations than standard techniques and manual tuning practices.

Keywords: automatic target recognition, computer vision, hyperparameter optimization

1. INTRODUCTION

Automated Target Recognition (ATR) is the general term for automated systems designed to detect targets (objects, persons, animals, etc.) based on sensor data. It is a broad term which covers a variety of sensor domains (audio, visual, infrared, radio, etc.) with applications in civilian and military sectors. When dealing with image data, ATR is highly related to the Object Detection and Object Segmentation categories of AI-based Computer Vision algorithms. These algorithms predict what targets are in an image and where in the frame they are. ATR research often contends with additional challenges beyond the non-trivial problem of generic object detection in the form of extreme edge case scenarios (e.g. low resolution and/or long-range targets) and less commonly used sensor suites (such as near infrared, thermal, synthetic aperture radar (SAR), or multi-modal sensor arrays). The Defense Systems Information Analysis Center (DSIAC) ATR dataset,[1] for example, contains images of military and civilian vehicles upwards of 5 km from the thermal camera. The statistics of DSIAC's data distribution differ considerably from typical benchmarks like ImageNet[2] or MSCOCO.[3] Consequently, customized solutions or, in the least, heavily modified off-the-shelf generic object detection algorithms are often required to address specific ATR scenarios. We experimented with different YOLO architectures and compared a newly developed homotopy-based optimization method HOMOPT[4] against random search on the DSIAC dataset. Our findings showed that HOMOPT method was able to effectively find lower validation loss scores over the random search approach on most of the examples. This result has important implications for the ATR field, where optimized hyperparameters play a critical role in meeting specific requirements. Our method can help streamline the hyperparameter optimization process and ultimately lead to improved ATR model performance in challenging scenarios, such as those encountered in the DSIAC dataset.

2. BACKGROUND

2.1 Background

Automatic target recognition (ATR) is a notoriously difficult problem, particularly due to the limited number of research-oriented training data sets for neural network algorithms. The vast majority of previous work relied on hand-tuned hyperparameters in order to produce the best results from the choice of data set. Gregoris *et al.*[5] compare information of inner and outer window sizes to detect targets, where the window sizes, along with a threshold value for comparing ratios, are set as predetermined hyperparameters. Yoon *et al.*[6] utilize localized pixel thresholding, where the thresholds are all set hyperparameters similar to Musicki & Evans.[7] Mahalanobis *et al.*[8] develop a Quadratic Correlation Filter to localize and classify targets that have a preset size of $r \times c$. Zhou and Crawshaw[9] use a thresholder in a single block of their detection algorithm to remove false targets and eliminate redundant target points, which is independent of data and is preset to a low fixed value. They state that the alogirthm does not require human interaction *once a threshold has been set.* There are few neural network architectures in thermal ATR research, particularly demonstrated on the DSIAC.[10] DeepTarget[11] uses a classic Adam optimizer with predefined hyperparameters, using standard β values and where the learning rate is reduced by half after a preset number of iterations. Mahalanobis and McIntosh[12] compare Faster R-CNN[13] and QCF[8] on DSIAC, both of which require preset hyperparamenters in order to obtain the best results.

2.2 Motivations

The motivation for this work comes from the need for reliable and efficient ATR systems, which are critical in a variety of applications in both civilian and military sectors. Object detection algorithms, which form the backbone of ATR systems, have advanced significantly in recent years due to the development of deep learning methods. However, ATR research presents unique challenges such as low resolution and/or long-range targets, and less commonly used sensor suites, which require customized solutions or heavily modified generic object detection algorithms. One of the major challenges in developing these systems is the tuning of hyperparameters, which control the model's ability to learn from data. This process can be time-consuming and computationally expensive, leading to suboptimal model performance compared to set requirements. Therefore, the development of efficient and effective hyperparameter optimization techniques is crucial for advancing ATR systems, and this work aims to explore the efficacy of applying more advanced optimization methods to address this challenge. By optimizing hyperparameters without requiring extensive manual tuning, we aim to improve the performance of object detection algorithms in challenging ATR scenarios and contribute to the development of more reliable and efficient ATR systems.

3. METHODS

3.1 Dataset & Processing

The images in the DSIAC dataset are captured by two forward-looking, ground-based sensors: an L3 Cincinnati Electronics Night Conqueror MWIR camera using a 640×480 pixel Indium Antimonide focal plane array with a 28 micron pitch, and an Illunis visible light camera. Images were col- lected from both cameras during daytime and from the MWIR camera at night. For each target/range/time scenario, images were captured at 30 Hz for one minute, yielding 1,680 images per sequence. We use an updated DSIAC protocol provided by Domenick Poster[10] where instead of using the provided annotations, we calculate corrected bounding boxes using positional metadata available for the MWIR imagery. This provided protocol also includes the addition of a training and testing split where the test set contains the same targets in different conditions which is not currently present in the DSIAC dataset itself. We only use data from the following classes: Pickup, SUV, BTR70, BRDM2, BMP2, T72, ZSU23-4, and 2S3 (dismounted civilian and towed artillery scenarios have been omitted). The image frames are from a set of video sequences recording each target individually traveling around a preset, 100 meter diameter circular path at distances ranging from 1000 to 5000 meters in 500 meter intervals. See Table 1 for the sizes of our training, validation, and testing subsets.

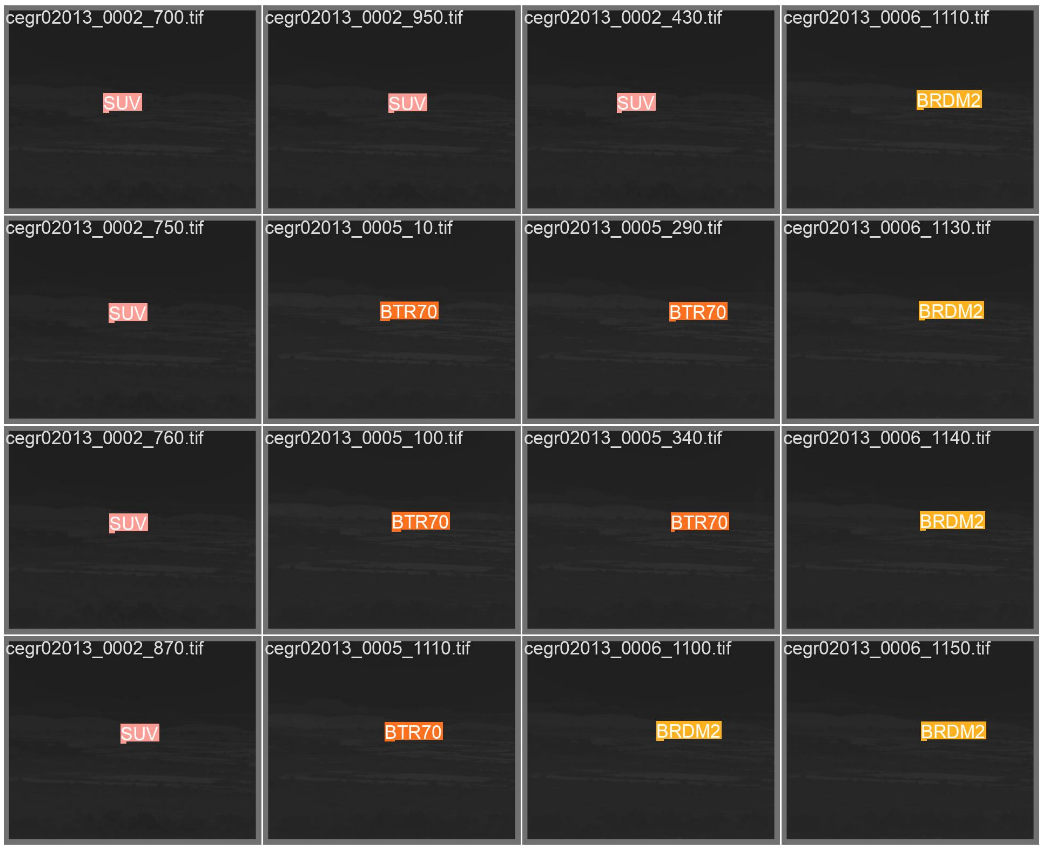

Figure 1. Example subset of images from the DSIAC ATR dataset with labeled vehicles. Note the difficulty of the task as the labeled vehicles are barely visible due to factors such as low resolution and long-range targets.

Range	Train	Val	Test
1000	1843	461	-
1500	-	-	458
2000	1834	458	-
2500	-	-	425
3000	1728	432	-
3500	-	-	425
4000	1709	427	-
4500	-	-	413
5000	1601	400	-

Table 1. Table with Ranges and Training, Validation, and Test sets for DSIAC data.

3.2 Experiments

This study compared two different optimization methods, random optimization and a homotopy-based method (HOMOPT)[4] using the Scalable Hardware-Aware Distributed Hyperparameter Optimization (SHADHO)[14] framework. This optimization framework and our code is available on GitHub[*] for general search problems. Random optimization is a common baseline for hyperparameter optimization. It involves randomly selecting hyperparameters for each trial without any knowledge of the performance of previous trials. This baseline allows for an understanding of how much performance improvement can be gained with a more sophisticated optimization method. HOMOPT in contrast starts with a set of initial values and a function that describes how the optimal point changes as the hyperparameters are adjusted. For our experiments, we use 20 samples utilizing random

[*]https://github.com/jeffkinnison/shadho

search. HOMOPT then gradually adjusts the input hyperparameter values and evaluates the function at each step to find the set of values that produce the best output for the optimization. This is done by repeated optimizations, while gradually reducing a parameter called the homotopy parameter, until the optimal set of hyperparameters is found within the designated set of search trials. The surrogate model in HOMOPT can be thought of as an approximation of the true performance landscape, allowing HOMOPT to be more effective in its exploration of the search space. In contrast, random optimization methods select hyperparameters purely by chance, without considering any underlying structure of the problem. Despite their simplicity, random search methods have been shown to be effective in a variety of settings, making them a natural baseline for comparison. We conducted a hyperparameter search using 100 trials for four different YOLO architectures on two different subsets of the DSIAC dataset. Specifically we tested on the 1000-2000 meter range as well as the 2000-3000 meter range. The models were all trained from scratch with randomly initialized weights and trained for 300 epochs. We monitored the validation box loss during the search, as it is a standard metric for evaluating object detection models, and selected the best observed validation box loss as the performance metric for each model. Validation box loss measures the difference between the predicted bounding boxes of objects in the image and the ground truth bounding boxes provided in the labeled training data. The lower the validation box loss, the better the model's predictions align with the actual objects in the image. In addition, we plot the simple log regret against the number of iterations. The simple log measures the difference between the best observed value found so far and the true optimum, expressed in logarithmic scale. In other words, it represents the number of orders of magnitude that the algorithm is away from the optimal value. A smaller simple log regret value indicates a better performance of the algorithm. A table outlining the search space for the hyperparameters used in the search can also be found in Table 2.

Table 2. Hyperparameters and their domain ranges for ATR experiments. The table on the left lists hyperparameters related to data augmentation, while the right table lists those related to model.

Hyperparameter	**Domain Range**
hsv_h	$[10^{-4}, 10^{-1}]$
hsv_s	[0, 1]
hsv_v	[0, 1]
degrees	[0, 180]
translate	[0, 0.3]
scale	[0, 1]
shear	[0, 45]
perspective	[0, 0.001]
flipud	[0, 1]
fliplr	[0, 1]
mosaic	[0.5, 1.0]
mixup	[0.5, 1.0]
copy_paste	[0.5, 1.0]

Hyperparameter	**Domain Range**
lr0	[0.0, 0.1]
lrf	[1, 5]
momentum	[0.5, 0.999]
weight_decay	$[10^{-8}, 10^{-5}]$
warmup_epochs	[0, 25]
warmup_momentum	[0.5, 0.9]
warmup_bias_lr	$[10^{-4}, 10^{-1}]$
box	[0, 1]
cls	[0, 1]
cls_pw	[0, 1]
obj	[0, 1]
obj_pw	[0, 1]
iou_t	[0, 0.5]
anchor_t	[1.0, 10.0]
fl_gamma	[0.0, 0.3]
anchors	[1.0, 10.0]

3.3 Model Architectures

We investigate the effectiveness of applying HOMOPT in comparison to random optimization for different object detection models in the context of ATR. The object detection models we have chosen to evaluate are Tiny YOLO, YOLOv5l6, YOLOv5m6, and YOLOv5s6[†]. These models represent a diverse set of architectures, sizes, and complexities, which makes them a suitable choice for exploring the impact of optimization methods on model performance. The selected object detection algorithms offer varying trade-offs between accuracy and computational efficiency, making them suitable for a wide range of applications. By evaluating these models, we can gain a deeper understanding of how optimization techniques such as HOMOPT can influence the performance

[†]The models used for experimentation were accessed from the software developed by Ultralytics[15]

of object detection models with different characteristics. Furthermore, these models are based on the well-established YOLO[15] framework, which is known for its real-time detection capabilities and has been widely adopted in various ATR applications.

Tiny YOLO is a lightweight version of the YOLO object detection algorithm, designed for real-time object detection on devices with limited computational resources. It maintains the core principles of YOLO, such as single-shot detection, while using fewer convolutional layers and having reduced model complexity. YOLOv5l6 is a variant of the YOLOv5 object detection algorithm with a larger architecture and a deeper backbone. The "l" in YOLOv5l6 stands for "large," indicating that the model has more layers and filters, making it suitable for high-resolution images. The "6" in the name refers to the number of output scales used in the model, which helps improve detection performance across different object sizes. YOLOv5m6 is a medium-sized variant of the YOLOv5 object detection algorithm. Like YOLOv5l6, this model uses six output scales to improve performance on objects of various sizes. However, YOLOv5m6 has fewer layers and filters than YOLOv5l6, making it a more balanced choice in terms of accuracy and computational resources. YOLOv5s6 is the smallest variant of the YOLOv5 object detection algorithm, designed for fast inference on devices with limited computational power. Despite its smaller size, YOLOv5s6 maintains six output scales for improved object detection across different sizes.

4. RESULTS

Table 3 shows the best observed validation box loss on the 1000-2000k dataset for four different YOLO models: TINY YOLO, YOLOv5l6, YOLOv5m6, and YOLOv5s6. The models were trained using random search optimization method and HOMOPT. The table shows that all models had a lower validation loss when using HOMOPT, except for YOLOv5m6, which had the same validation loss for both optimization methods. The largest improvement was seen for the TINY YOLO model, which had a 91.2% decrease in validation loss when using HOMOPT method compared to using random search.

Table 3. Best observed validation loss on the 1000-2000 meter data. BASE refers to random search optimization and HOM refers to HOMOPT.

Model	BASE	HOM
TINY YOLO	0.056446	0
YOLOv5l6	0.00070467	0
YOLOv5m6	0	0
YOLOv5s6	0.0021693	0

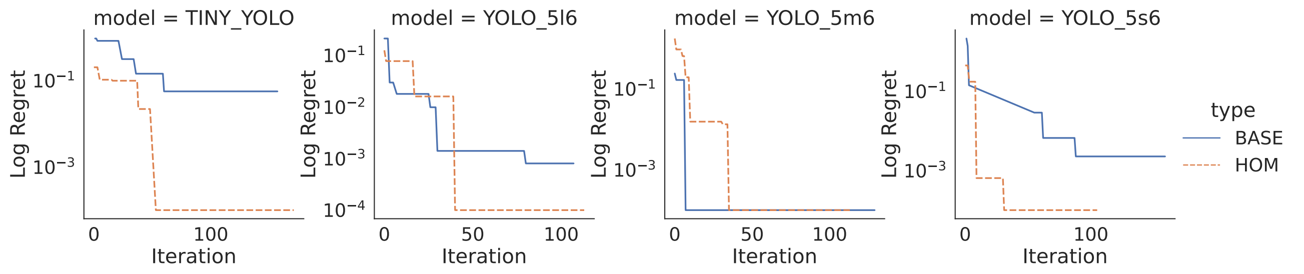

Figure 2. Comparison of random search and HOMOPT on 4 different architectures of YOLO on the 1000-2000 meter data. The log regret at each iteration is displayed.

Table 4 shows the best observed validation box loss on the 2000-3000 meter range data for the same four YOLO models. Similar to Table 3, the models were trained using random search optimization and the HOMOPT method. The table shows that HOMOPT again performed better than random optimization alone for TINY YOLO and YOLOv5s6. For YOLOv5m6 both methods had the same validation loss and random search found a lower validation box loss on the YOLOv5m6 model. The largest improvement was seen for YOLOv5s6, which had a 52.6% decrease in validation loss when using the HOMOPT method compared to random search.

Table 4. Best observed validation loss on the 2000-3000 meter data. BASE refers to random search optimization and HOM refers to HOMOPT.

Model	BASE	HOM
TINY YOLO	0.020519	0.0018149
YOLOv5l6	0	0
YOLOv5m6	0	0.011574
YOLOv5s6	0.00050875	0.00024132

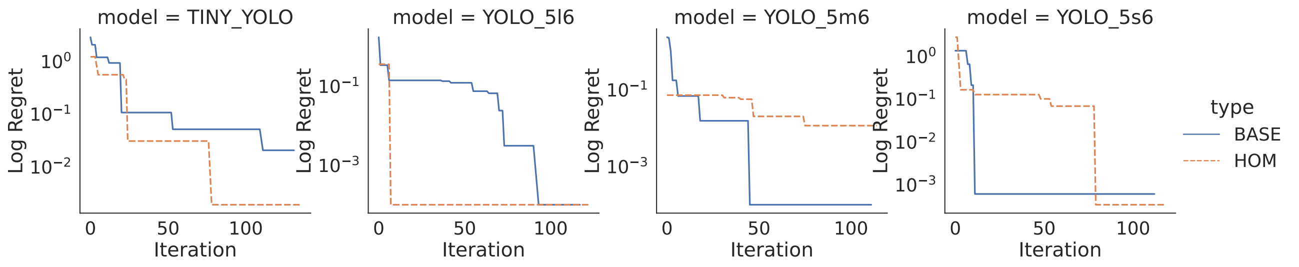

Figure 3. Comparison of random search and HOMOPT on 4 different architectures of YOLO on the 2000-3000 meter data. The log regret at each iteration is displayed.

The simple regret plot (Figure 2) for the 1000 - 2000 meter data range show a clear advantage of using HOMOPT method over random optimization. The TINY YOLO model shows a significantly lower regret for HOMOPT, indicating that it reaches a better optimum faster than the random optimization. This trend is also visible for the YOLOv5l6 and YOLOv5s6 models, where the HOMOPT optimization method again outperforms random optimization. However, for the YOLOv5m6 model, the regret plots are less conclusive. While HOMOPT did not merge to a better optimum sooner than random, it eventually converged to the same best optimum as the random optimization.

The second set of simple regret plots were generated for the 2000-3000 meter range data (Figure 3), and compared the performance of random and HOMOPT method. For the TINY YOLO and YOLOv5l6 architectures, the regret plot for HOMOPT showed consistently lower regret than the plot for random optimization. In the case of YOLOv5s6, random optimization initially converged faster but ultimately reached a suboptimal value, whereas the HOMOPT converged to a better optimal value. However, for YOLOv5m6, HOMOPT did not outperform random optimization in terms of reaching the best optimal value.

We observed that some of the models had a validation box loss of zero for both random search and HOMOPT, indicating perfect performance on the validation set. While this may seem like a desirable outcome, it can also be a sign of overfitting. In particular, a model that is perfectly tuned to the training data may fail to generalize to new data. We also note that a zero validation score may indicate that the model is simply memorizing the training data, rather than learning general patterns that can be applied to new data. We talk about ways to address this in the optimization search in the discussion section.

5. DISCUSSION

In this study, we compared the performance of random optimization and HOMOPT for hyperparameter tuning of YOLO models on the DSIAC ATR Algorithm Development Image Database. Our results indicate that HOMOPT consistently outperformed random optimization for most of the YOLO models and data ranges tested in regards to lowering the validation loss and converging to a better optimum sooner. The fact that the optimization method is able to bring the validation score down to zero is a good indication that it is performing well and able to find the optimal solution for the given problem. However, we also observed that overfitting occurred in the models which we hypothesize is due to the arbitrary choice of training each model for 300 epochs. The effect of this on the predictions from the model can be seen in Figure 7. To address the issue of overfitting during optimization, one can incorporate methods such as early stopping or regularization techniques like dropout in the training process at each search iteration. Early stopping involves monitoring the validation score during training and stopping the optimization process when the validation score no longer improves, thus preventing the model from continuing to overfit. Dropout is a technique that randomly drops out (sets to zero) a fraction of the model's neurons during training, which can help prevent overfitting by encouraging the network to learn more robust representations. Despite this limitation, our results suggest that the HOMOPT could be a promising approach for hyperparameter tuning in computer vision tasks.

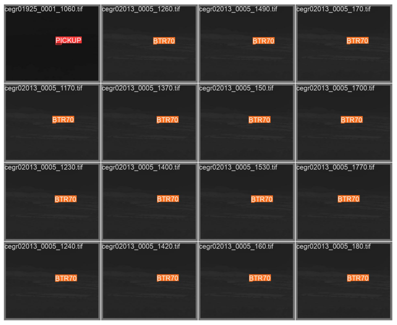

Figure 4. *
(a) True labels

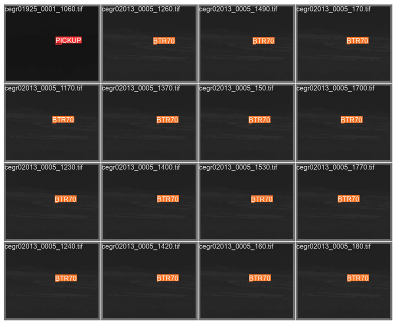

Figure 5. *
(b) Properly trained model

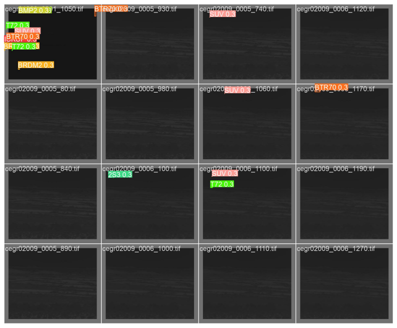

Figure 6. *
(c) Overfit model predictions

Figure 7. Examples of validation images with corresponding model predictions after different sets of training epochs. (a) shows the true labels, (b) shows predictions from a properly trained model for only 60 epochs, and (c) shows predictions from an overfit model trained for 300 epochs.

Furthermore, we realized domain knowledge can play a significant role in improving the results of hyperparameter optimization. In particular, we found that heavy augmentations caused worse performance, indicating that the search space could potentially be improved by constraining the range of augmentations based on domain knowledge. An example of the type of augmentations that resulted in varying performances in the model can be seen in Figure 8. The heavy augmentations illustrated in the mosaics on the left in Figure 8 resulted in a model that was unable to detect the vehicles in the validation samples, whereas the lighter set of augmentations as seen in the mosaic on the right in Figure 8 resulted in a stronger performing model. Heavy augmentations can lead to poor performance in the model because they can introduce too much variation in the data, making it difficult for the model to learn and generalize to new, unseen data. This is particularly true for models trained on small datasets, where the variations introduced by heavy augmentations can dominate the limited amount of training data, leading to overfitting. Some augmentations may also not be appropriate for certain types of data or applications. For example, in object detection tasks, heavy augmentations on the object of interest can make it unrecognizable or difficult to detect, resulting in poor performance. Domain knowledge can thus be incorporated in various ways to improve the search process such as imposing constraints on hyperparameters based on prior knowledge of the problem at hand. Additionally, while the use of multi-objective optimization techniques can further enhance the performance of the search algorithm by considering multiple criteria, such as accuracy and inference time, it can also be used to incorporate more domain knowledge into the optimization process, as multiple objectives can be used to reflect different aspects of the problem. We believe that these findings provide insight into the potential benefits of incorporating domain knowledge into the hyperparameter optimization process, which could lead to more efficient and effective optimization in the future.

Moving forward, one potential avenue for future research is to explore the incorporation of depth cues into object detection algorithms. This could be achieved by jointly optimizing the reconstruction error and object detection performance. Another area worth investigating is the use of multi-objective optimization techniques to simultaneously optimize multiple objectives, such as detection accuracy, computational efficiency, and interpretability. Furthermore, there is a need to investigate the integration of domain knowledge into hyperparameter optimization. Methods like Bayesian optimization and reinforcement learning can help incorporate this knowledge and improve the optimization process.

Hyperparameter optimization is a powerful tool that can improve the accuracy and efficiency of object detection, surveillance, reconnaissance, and other military technologies. By optimizing hyperparameters such as learning rate and momentum, we can improve the accuracy of object detection in challenging conditions like low resolution or noisy images. It can also increase the efficiency of these systems, which is crucial in time-

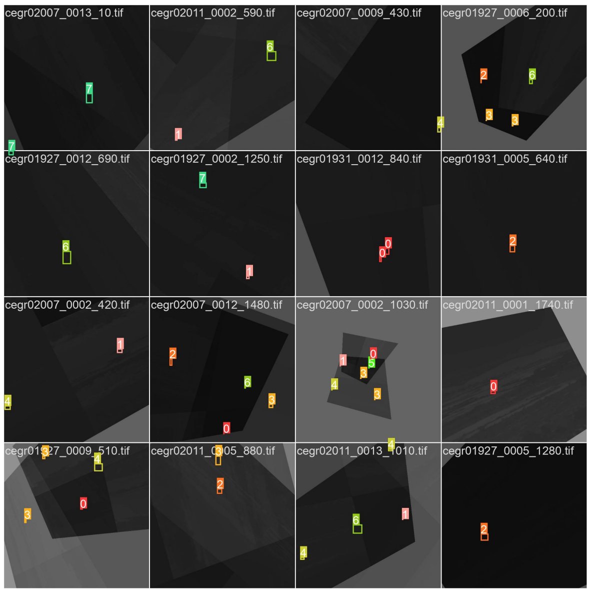
Figure 8. Mosaics with heavy augmentations

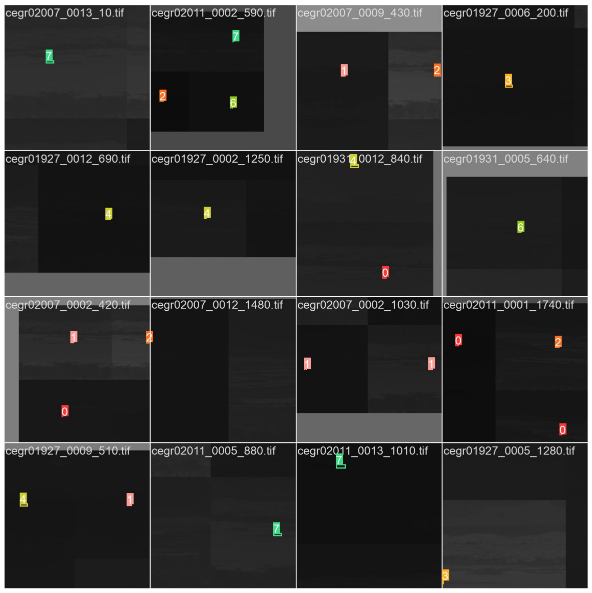
Figure 9. Mosaics with light augmentations

sensitive military applications. Furthermore, hyperparameter optimization can also be applied to optimize other aspects of military technology such as radar systems or unmanned aerial vehicles. The HOMOPT method can efficiently explore the hyperparameter space of these systems and find the optimal set of parameters to maximize their performance. Unmanned aerial vehicles (UAVs) are increasingly being used in military applications for reconnaissance, surveillance, and other missions. Optimizing the hyperparameters of these systems can help improve their accuracy, efficiency, and reliability, which is critical for their success in the field.

Our work has important implications for the military and defense industries, particularly in surveillance and reconnaissance. Object detection is crucial in these applications, and optimizing hyperparameters in YOLO models can significantly improve their accuracy and efficiency. The HOMOPT algorithm could be used to optimize other aspects of military technology, such as radar systems or UAVs. Moreover, our work could also have significant implications for synthetic aperture radar (SAR) applications in the military and beyond. SAR is a remote sensing technology used for imaging and reconnaissance, and it has a wide range of military and civilian applications. By optimizing hyperparameters in YOLO models, we could potentially improve the accuracy and efficiency of SAR systems. The HOMOPT algorithm could be used to optimize other aspects of SAR technology, such as antenna design or image reconstruction.

6. CONCLUSION

In conclusion, automatic target recognition (ATR) is a challenging problem that has been extensively studied in the military and defense industries. The low resolution and noise in the images make it difficult to accurately detect and recognize objects in the scene. Our study provides valuable insights into the use of hyperparameter optimization to improve the accuracy and efficiency of ATR systems. Specifically, the HOMOPT algorithm provides an efficient way to search the hyperparameter space and identify the optimal set of parameters for YOLO models. Our work has important implications for the military and defense industries, as well as for other applications of deep learning in computer vision. By improving the accuracy and efficiency of ATR systems, we can enhance their performance in challenging conditions and improve their usefulness in real-world scenarios. Furthermore, the HOMOPT algorithm can be applied to other areas of military and civilian technology, such as radar systems and unmanned aerial vehicles, to optimize their performance and capabilities. Our findings pave the way for further research in this area and have the potential to significantly advance the capabilities of military and civilian technologies.

ACKNOWLEDGMENTS

This work was funded by the DEVCOM Army Research Laboratory under the cooperative agreement W911NF-20-2-0218.[‡]

REFERENCES

[1] Defense Systems Information Analysis Center, "ATR Algorithm Development Image Database." https://dsiac.org/databases/atr-algorithm-development-image-database/.

[2] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., "Imagenet: A large-scale hierarchical image database," in [*2009 IEEE conference on computer vision and pattern recognition*], 248–255, Ieee (2009).

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

[4] Abraham, S., Maduranga, K. D., Kinnison, J., Hauenstein, J., and Scheirer, W., "Homotopy-based hyperparameter optimization," in [*Under Review*],

[5] Gregoris, D. J., Yu, S. K., Tritchew, S., and Sevigny, L., "Wavelet transform-based filtering for the enhancement of dim targets in flir images," in [*Wavelet Applications*], **2242**, 573–583, SPIE (1994).

[6] Yoon, S. P., Song, T. L., and Kim, T. H., "Automatic target recognition and tracking in forward-looking infrared image sequences with a complex background," *International Journal of Control, Automation, and Systems* **11**(1), 21–32 (2013).

[7] Musicki, D. and Evans, R., "Clutter map information for data association and track initialization," *IEEE Trans. on Aerospace and Electronic Systems* **40**, 387–398 (April 2004).

[8] Mahalanobis, A., Muise, R. R., and Stanfill, S. R., "Quadratic correlation filter design methodology for target detection and surveillance applications," *Applied Optics* **43**(27), 5198–5205 (2004).

[9] Zhou, Y.-T. and Crawshaw, R. D., "Contrast, size, and orientation-invariant target detection in infrared imagery," in [*Automatic Object Recognition*], **1471**, 404–411, SPIE (1991).

[10] Poster, D., "Defense Systems Information Analysis Center (DSIAC) ATR Algorithm Development Image Database." Online (2023).

[11] Nasrabadi, N. M., "Deeptarget: An automatic target recognition using deep convolutional neural networks," *IEEE Transactions on Aerospace and Electronic Systems* **55**(6), 2687–2697 (2019).

[12] Mahalanobis, A. and McIntosh, B., "A comparison of target detection algorithms using DSIAC ATR algorithm development data set," in [*Automatic Target Recognition XXIX*], **10988**, 47–51, SPIE (2019).

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

[14] Kinnison, J., Kremer-Herman, N., Thain, D., and Scheirer, W., "Shadho: Massively scalable hardware-aware distributed hyperparameter optimization," in [*2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*], 738–747, IEEE (2018).

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

[‡]We also acknowledge the assistance of ChatGPT, a language model trained by OpenAI, for providing helpful suggestions during the writing of this manuscript.