# Detection From Above : CS 7643

Derek Griffing
dgriffing3@gatech.edu

Ming Xiang Lim
mlim88@gatech.edu

Tyler Burki
tburki3@gatech.edu

## Abstract

*Drones present an economically sensible approach to several object detection and tracking tasks. We examine the application of transfer learning to object detection and multiple object tracking to establish an accessible foundation for further improvement. Bayesian search methods are leveraged to arrive at optimal hyperparameters quickly and identify which transferred models provide the most benefit. Data augmentation in the form of deblurring images is also explored, but the process was ultimately unsuccessful in improving outcomes in its current state. Optimization is focused on the mAP50 metric for object detection and MOTA for multiple object tracking. We were able to achieve respectable scores approaching state-of-the-art. We believe the framework presented here can serve as a solid bootstrapping mechanism for improvement on VisDrone tasks.*

## 1. Introduction/Background/Motivation

Multiple Object Tracking (MOT) for drones is an important computer vision (CV) task with broad application prospects in urban security, transportation, and surveillance. It involves locating and identifying specific objects in video sequences, assigning fixed IDs, and maintaining stable tracking across frames [3]. However, it possesses a unique set of challenges such as poor illumination, poor image stabilization and fast-moving targets, which might result in low recall and unstable target tracking performance [11]. Blurring is also a common source of noise in images captured from drones because they often shake while capturing images [12].

This project aims to improve the performance of existing state-of-the-art (SOTA) CV algorithms such as YOLOv8 (You Only Look Once) [6] for object detection and ByteTrack for MOT on the VisDrone Dataset through the integration of modern deblurring and hyperparameter tuning techniques into the data pre-processing, tuning and prediction pipeline. The VisDrone Dataset is a large-scale annotated benchmark dataset composed of 288 video clips with 271,908 frames and 10,209 static images captured by drone-mounted camera with varying locations, environment (urban and rural) and object densities (pedestrians, vehicles, bicycles, etc.) under different weather and lightning conditions [17].

When comparisons are made between models, **bold text** denotes the best value in a column or section.

## 2. Approach

This project is split into two phases. In Phase 1, we explore the optimization of object detection for static images within the VisDrone dataset. In Phase 2, we build on the results from Phase 1 and explore MOT on videos from the VisDrone dataset.

### 2.1. Phase 1: Object Detection for Images

We first evaluated the baseline performance of pre-trained YOLOv8 algorithms on static images within the VisDrone dataset, comprising of up to 10 object classes (such as Pedestrian, Bicycle and Bus), before optimizing preprocessing the dataset using deblurring algorithms to improve object detection performance.

**Choice of Model** Existing YOLOv8 algorithms pre-trained on both the COCO (Common Objects in Context) and the Open Images V7 (OIv7) datasets from Ultralytics's publicly available GitHub Repository [6] were adopted as these readily available benchmark datasets comprised of object classes present in the target VisDrone dataset. We leveraged transfer learning for optimizing the number of training epochs required to improve performance on the target object detection task, given the demanding GPU and VRAM requirements for training YOLOv8 from scratch on the large number of VisDrone input images.

**Hyperparameter Tuning** We chose to apply a Bayesian-based approach through use of Optuna's Tree-structured Parzen Estimator (TPE) sampler [2]. The sampler uses two Gaussian Mixture Models (GMMs) to model the behavior of the chosen objective function (i.e. performance metric to be optimized) before automatically selecting the best combination of hyperparameters to trial next by considering results from previous trials. This contrasts with traditional grid search techniques which are inefficient for search in large hyperparameter spaces as they evaluate the performance metric for all possible hyperparameter combinations often without pruning. Carrying out an exhaustive grid search is prohibitive for this task due to time and hardware constraints, given the size and nature of the VisDrone dataset.

We take a two-phase approach to parameter search. First we run small studies on pairings of the models and data (unmodified and deblurred). The information gained is then used to determine which pairing is most promising for indepth study. The result of the extensive study will be used to generate our final model.

**Choice of Performance Metric** We explored conventional metrics for object detection such as precision, recall and average precision metric (mAPX) calculated across different intersection over union (IoU) ranges denoted by X [19], allowing for fair comparison of performance across the different object classes.

**Areas for Optimization** We explored potential improvements to the baseline performance through deblurring of the Visdrone dataset, borrowing from the ideas of Liu et al., who proposed a method for pre-screening images based on their quality and deblurring necessary images with their custom model to enhance object features [12]. Our approach differs in that we will deblur the entire dataset (train, validation and test) and leverage a pre-trained model of the more available DeblurGAN [8] method from a public code repository [7] and its corresponding pre-trained model [1]. Increasing the availability of object features to the detection algorithm could improve outcomes by capturing features that may have otherwise been overlooked due to blurring. For real-world applications, the images will need to be deblurred as they are captured and before they are passed to the detection module.

### 2.2. Phase 2: Object Tracking for Videos

**Choice of Model** ByteTrack was selected as the tracker for the Phase 2 VisDrone MOT task as it has achieved state of the art performance on MOT20 [5] with a relatively simple association method [16]. Crowded scenes are prone to occlusion, which can significantly reduce tracker performance since trackers only associate detection bounding boxes whose confidence scores are greater than a tunable threshold. The ByteTrack algorithm addresses this challenge by setting two tunable thresholds—a high threshold for detections with high confidence scores that are preferentially associated in the first stage; and a low threshold for detections with low confidence scores that are associated in the second stage [16]. The intuition behind this approach is that occluded objects are more likely to have lower detection confidence scores; discarding these detections may significantly worsen MOT metrics. ByteTrack's primary contribution is its novel approach to association. Outside of this it is similar to other trackers in that it uses a Kalman filter to predict the new location of tracked objects and uses the Hungarian Algorithm to solve the linear sum assignment problem [16]; effectively, it minimizes a cost matrix where the cost is based on similarity (IoU or appearance features). The MOT experiments herein were conducted using the Ultralytics ByteTrack implementation [6] based off the original ByteTrack repository [16].

**Hyperparameter Tuning** Our approach to tuning ByteTrack mirrored how we tuned the detection model using Optuna's TPE sampler. Effectively, we developed a generic tuning pipeline that is agnostic to the tuning task and its hyperparameters. The primary requirements for this pipeline are a configuration file that specifies the hyperparameter, its data type, and a low and high value that define the hyperparameter search space (for data types float and int); a mechanism for returning output from the task (in this case running the tracker), and a mechanism for obtaining an objective score (metric) that is to be optimized. We used the TrackEval repository [14] for metrics as it is the official evaluation code for multiple benchmark MOT tasks, including MOTChallenge [5].

**Choice of Performance Metric** The tracker tuning pipeline optimizes the multiple object tracking accuracy (MOTA) metric. MOTA was selected because it has been a primary MOT scoring metric since around 2006 [14]. MOTA is useful since it captures three key errors in the detection-tracking pipeline: false positives (FP), false negatives (FN), and identity switches (IDSW); moreover, it can be expressed simply as $1 - \frac{|FN|+|FP|+|IDSW|}{|gtDet|}$, where $|gtDet|$ refers to the size of the set of ground truth tracks [14]. Essentially the expression is one minus a fraction whose numerator is a count of tracking errors and whose denominator is the number of ground truth tracks. If a tracker performs perfectly, that fraction evaluates to 0 and the result is the maximum MOTA score (1). Since there is no fixed limit to the number of FP that may occur, MOTA $\in (-\infty, 1]$. [14] describes this as a drawback since it may be difficult to interpret a negative MOTA; however, an intuition for having a negative MOTA is that a tracker that ”does nothing” (MOTA 0) is preferable to a tracker that only produces FP (negative MOTA).

**Areas for Optimization** The tracker tuning pipeline optimized seven parameters that can be grouped as detector-specific parameters (”YOLO parameters”) and tracker-

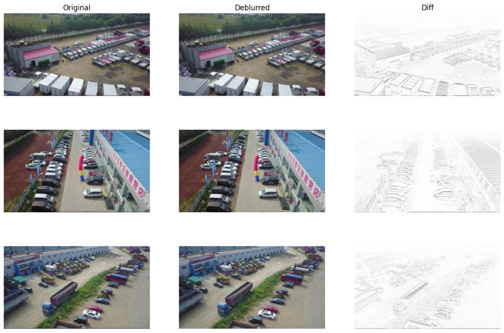

Figure 1. Deblurring the VisDrone dataset with DeblurGAN

specific parameters. The YOLO parameters were detector confidence threshold and non-maximum suppresion (NMS) IoU. Detections where model confidence is less than the detector confidence threshold are discarded; NMS IoU is used to determine the extent to which overlapping bounding boxes should be discarded [6].

The tracker specific parameters were high threshold, low threshold, new track threshold, track buffer, and match threshold. As discussed, high threshold and low threshold are used to separate high and low confidence detections such that high confidence detections may be associated first [16]. New track threshold is the threshold that must be met for a new track to be initialized when a detection does not match existing tracks. Track buffer is the number of frames a track may exist without any new detection, and match threshold is the threshold that must be met in order to associate the same track ID in subsequent frames [16].

## 3. Experiments and Results

### 3.1. Phase 1: Object Detection for Images

**Data Augmentation** Images were run through the DeblurGAN model in inference mode and saved separately. Figure 1 shows examples of the result of the deblurring augmentation. We expected image differences to be concentrated in blurry areas of the original image but the results appear to be more akin to edge detection. As a result, it is unlikely that this method will contribute to the improvement of object identification outcomes, and other approaches such as training DeblurGAN directly on the VisDrone dataset or developing a more task-specific deblurring model like Liu et al.may yield better results [12]. Deblurring a single image takes half a second, so this method would not be suitable for real-world applications in video processing.

| Hyperparameter | Default | Low | High | Categories |
|---|---|---|---|---|
| batch | 16 | | | 8, 16 |
| lrf | 0.01 | 0.01 | 1.0 | |
| weight_decay | 5e-4 | 1e-5 | 1e-1 | |
| box | 7.5 | 6.0 | 9.0 | |
| cls | 0.5 | 0.3 | 0.7 | |

Table 1. Default vs. searched values for hyperparameters

**Initial Parameter Search** To perform an initial comparison between all model-data pairings, we initialize an Optuna study with 10 trials trained on 10 epochs each. All trials search the same parameter space in Table 1. The image size is fixed to 640 for all studies. The YOLO ”small” model was pre-trained on either the COCO or OIv7 datasets. Models were then trained further on either the VisDrone or deblurred datasets.

Losses are similar for all pairings across the training and validation sets, indicating a suitable fit. COCO sees smaller losses in both training and validation than OIv7. OIv7 is trained on 600 class labels—many of which present a semantic shift from VisDrone data [9]. VisDrone has 10 labels consisting of vehicles and people, while OIv7 has classes such as ”Udon” and ”Shamrock”. OIv7 must account for these unrelated (to VisDrone) classes, and has a more difficult time finding a consensus than the COCO model. This can be seen in Table 3 where OIv7 typically has a higher precision but lower recall. It follows that the model is conservative in its decision making when features may be obscured. This could explain why the OIv7 model's metrics increase on deblurred data as it can become more certain of features. See Table 2.

The COCO model on unmodified data saw unquestionably the best performance. The 80-class COCO model is concerned with only people, clothing, vehicles, traffic, animals, recreation and home appliances [10]. The model can therefore transfer to the VisDrone task more easily with higher confidence and subsequently higher mAP scores.

Our deblurring implementation appears to be ineffective on the whole. COCO performs worse on deblurred data while OIv7 performs marginally better—but still worse than COCO on unmodified data. This is the result of the deblurring module not capturing the essence of the perturbations and modifying edges more so than blurry areas.

Optuna found the same best parameters for all pairings, so we can infer that there are no strikingly significant differences in their loss surfaces. Extended tuning trials in subsequent studies should change this, but the search space may have been too large to capture the nuance of the task in only 10 trials apiece.

Overall, the COCO model transfers to the task more effectively than OIv7. Deblurring the data shows some promising results with the OIv7 model, but trails COCO on the unmodified data and should not be pursued further. Since COCO performs worse on the deblurred data, the best result should come from COCO on the unmodified set.

| Model | Set | Deblur | Box | Cls | Dfl |
|---|---|---|---|---|---|
| COCO | Train | No | **1.239** | **1.5011** | **1.2221** |
| OIv7 | Train | No | 1.2544 | 1.5306 | 1.2297 |
| COCO | Train | Yes | 1.2419 | 1.5071 | 1.2246 |
| OIv7 | Train | Yes | 1.2602 | 1.5301 | 1.2295 |
| | | | | | |
| COCO | Val | No | **1.2388** | 1.5194 | **1.2065** |
| OIv7 | Val | No | 1.2556 | 1.5412 | 1.2137 |
| COCO | Val | Yes | 1.2513 | **1.5147** | 1.2177 |
| OIv7 | Val | Yes | 1.2742 | 1.5496 | 1.2221 |

Table 2. Best loss on training and validation sets

| Model | Deblur | Precis. | Recall | mAP50 | mAP50-95 |
|---|---|---|---|---|---|
| COCO | No | 0.4256 | **0.3456** | **0.3375** | **0.1946** |
| OIv7 | No | 0.4334 | 0.3336 | 0.3271 | 0.1886 |
| COCO | Yes | 0.4398 | 0.3334 | 0.3341 | 0.1932 |
| OIv7 | Yes | **0.4420** | 0.3374 | 0.3275 | 0.1864 |

Table 3. Metric performance

| Hyperparameter | Initial | Directed |
|---|---|---|
| epochs | 10 | 133 |
| batch | 16 | 8 |
| imgsz | 640 | 960 |
| lrf | 0.08032 | 0.6162 |
| weight_decay | 2.23E-05 | 1.9111 |
| box | 6.0606 | 8.9013 |
| cls | 0.6330 | 0.6054 |

Table 4. Best parameters for COCO on unmodified data

**Tuning for Best Model** After the results of the initial tuning survey, a more pointed tuning study was directed at the COCO pre-trained model on unmodified data. The same search parameters were used as in Table 1 with the addition of 960 as an image size. The number of epochs was also parameterized to between 100 and 300 to capture the best performance possible. The number of trials was increased to 100 with instructions to cease the trials if mAP scores were not improved in the last 10 trials. The best mAP50 score of $0.5228$ was found on the sixth trial. This compares favorably to the initial study's mAP50 of $0.3375$.

It became clear after some manual tuning on the results of the initial study that increasing the image size to 960 had a significant positive effect on scoring. Unfortunately, memory constraints made searching this space prohibitive in the initial phase. We found that adding the larger image size and then increasing the number of epochs altered how Optuna searched the parameter space dramatically. Objects in drone images are often small and far away [11], and it follows that increasing the image size would increase the model's ability to capture better features for each object. This process reduces the receptive field of convolutions and allows the model to make more confident predictions on objects themselves without excessive background noise. The increase in epochs trained enhances scores by allowing the optimizer to traverse the loss surface longer to a better minimum.

A comparison of the best parameters can be found in Table 4. Significantly different areas of the space were searched for the lrf and weight_decay parameters. The improved feature capturing capabilities with larger images allowed the lrf parameter (an annealing factor) to increase almost ten-fold to keep the learning rate high as the gradient has steepened. This was accounted for by a several-order-of-magnitude increase in weight_decay as a regularization factor. The 'box' parameter increased nearly 50% to place more emphasis on correctly defining bounding boxes. This also makes sense given the image size increase, as objects are less likely to be lost in the background. This effect can be seen in Figure 2.

**Best Model Results vs. SOTA** Table 5 summarizes the mAP50-95 metrics obtained for our best performing model, and how it compares with 3 selected SOTA detectors previously reported under the VisDrone-DET2019 challenge [17]. Our model was able to achieve comparable AP50-95 to the best performing DPNet-ensemble model (with AP50-95 of 29.6) on the test dataset. Inspecting the class-specific AP50-95 values, our best performing model had performed well in the prediction of large vehicles such as cars and buses, and out-performed the 3 SOTA models for detection of these object classes.

However, our best performing model performed poorly for the detection of smaller objects such as bicycles and pedestrians, and we observed a similar trend for the 3 SOTA models. We postulate that larger objects are less likely to be obscured by other objects in aerial photography due to their larger dimensions (in contrast to smaller objects such as pedestrians), hence they are likely to be easier to detect and identify.

### 3.2. Phase 2: Object Tracking for Videos

**Parameter Search** Given the lighter hardware requirements for inference with YOLO, and the relatively simple approach of ByteTrack, tracker tuning was conducted locally with 200 trials using the hyperparameter tuning pipeline described in Section 2.2. A large number of trials supported traversing the relatively large search space shown in Table 6. The tuning data was comprised of sequences from the VisDrone MOT validation set. The experiment consisted of tuning ByteTrack (using our tuned detector model) on the validation data and comparing its performance against the default hyperparameters on the VisDrone test dev set. Moreover, we compare our tuned tracker per-

| Model | $AP_{50-95}$ | $AP_{ped}$ | $AP_{per}$ | $AP_{bik}$ | $AP_{car}$ | $AP_{van}$ | $AP_{trk}$ | $AP_{tri}$ | $AP_{awn}$ | $AP_{bus}$ | $AP_{mtr}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Our Best Model | 27.0 | 12.0 | **18.0** | 8.9 | **59.2** | 33.3 | 27.5 | 18.0 | 11.3 | **44.5** | 24.6 |
| DPNet-ensemble | **29.6** | **32.3** | 16.0 | 12.9 | 51.5 | **39.8** | 30.7 | **30.7** | 18.4 | 38.5 | **28.0** |
| RRNet | 29.1 | 30.4 | 14.9 | **13.7** | 51.4 | 36.1 | **35.2** | 28.0 | 19.0 | 44.2 | 25.9 |
| ACM-OD | 29.1 | 30.8 | 15.5 | 10.3 | 52.7 | 38.9 | 33.2 | 27.0 | **21.9** | 41.4 | 24.9 |

Table 5. Comparison of class-specific mAP50-95 between our best performing model and SOTA models [17].

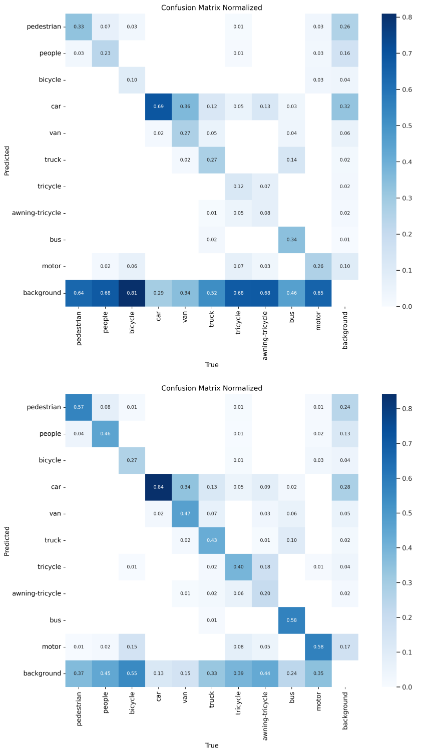

Figure 2. Initial study (top) classifies more objects as background than directed

formance against others described in the literature. While more recent VisDrone MOT papers [4] have focused on AP, the VisDrone 2018 paper [18] reported MOTA and serves as a useful baseline for performance comparisons.

**Results** The tuned tracking parameters are shown in Table 7 alongside the default values. As shown in Table 8, the tuned parameters led to a notable increase in MOTA on both the VisDrone validation and test dev sets when compared to the performance of the default parameters. Compared to theVisDrone2018 MOT submissions described in [18], our tuned tracker MOTA performance on the test dev set ranks 6 out of 12. [18] reported a best MOTA of 0.426.

The optuna hyperparameter importances are shown in Figure 3. New Track Threshold was found to be the most important parameter. The tuned value for this parameter was 0.1 higher than the default, suggesting that a stricter threshold for initiating a new track provides better performance. The intuition for this threshold being higher than the other thresholds is that we want to be confident we are detecting a target of interest, whereas once we have established a track, we want to ensure that we do not lose it. This explains the lower values for the other thresholds - in particular track low threshold, which helps to ensure that temporarily occluded targets are not lost.

Match threshold was the second most important parameter. This is not unexpected given its key role in association. TrackEval uses linear sum assignment from the scipy library to minimize a cost matrix made up of negative IoU elements (IoU is negated since a minimizing algorithm is used on a value we seek to maximize). Our experiment demonstrated that the default value (0.8) is effective for the VisDrone MOT task. It avoids the spurious associations that may occur with lower threshold values.

A notable finding of our experiments was the small importance of 'track low threshold'. This parameter is important for ensuring that occluded tracks are not lost. We suspect this finding may relate in part to the imbalanced nature of the VisDrone dataset—individual pedestrians make up 23% of the dataset [15]. Pedestrians are smaller than vehicles thus more likely to be occluded. Perhaps the track low threshold would have a higher importance for a more person-centric dataset such as [5]. The source of video may also play a role. The higher vantage point of drones should provide views with less occlusion in crowded scenes compared to the ground-level cameras seen in [5].

Track high threshold was the third most important parameter. Its tuned value is lower than the default, suggesting that there is some benefit in using a lower confidence threshold in the first association stage. The lower tuned value for track buffer suggests that for this dataset better performance is achieved by requiring tracks to have more frequent detects (every 14 frames instead of 30). The YOLO detector parameters (detector confidence threshold and NMS IoU) were found to have low importances and the tuned values were fairly aligned with the default values.

It is worth highlighting that the importances seem related to the user-defined search space. The search spaces were defined heuristically based on the default parameters with the aim of achieving better results with fewer trials. Future efforts may involve setting search spaces as wide as possible and running sufficient trials to thoroughly explore them.

| Hyperparameter | Default | Low | High |
|---|---|---|---|
| Track High Thresh | 0.5 | 0.3 | 0.7 |
| Track Low Thresh | 0.1 | 0.05 | 0.3 |
| New Track Thresh | 0.6 | 0.2 | 0.8 |
| Track Buffer | 30 | 10 | 50 |
| Match Thresh | 0.8 | 0.3 | 0.9 |
| Detector Conf Thresh | 0.3 | 0.1 | 0.6 |
| NMS IoU | 0.5 | 0.2 | 0.8 |

Table 6. Tracker default vs. searched values for hyperparameters

| Hyperparameter | Default | Tuned |
|---|---|---|
| Track High Thresh | 0.5 | 0.3339 |
| Track Low Thresh | 0.1 | 0.1956 |
| New Track Thresh | 0.6 | 0.7172 |
| Track Buffer | 30 | 14 |
| Match Thresh | 0.8 | 0.7912 |
| Detector Conf Thresh | 0.3 | 0.2036 |
| NMS IoU | 0.5 | 0.4561 |

Table 7. Tracker default vs. tuned hyperparameters

| Dataset | Parameter Set | MOTA |
|---|---|---|
| Validation | Default | 0.3396 |
| Validation | Tuned | **0.4122** |
| | | |
| Test Dev | Default | 0.2827 |
| Test Dev | Tuned | **0.3594** |

Table 8. Tracker experiment results

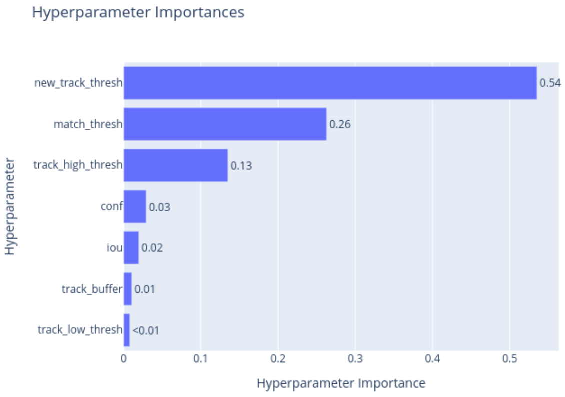

Figure 3. ByteTrack parameter importances

# 4. Conclusion

Data augmentation in the form of deblurring showed some promise with the OIv7 pre-trained model but was overall ineffective at increasing mAP scores. We identified that our use of DeblurGAN was flawed in detecting and eliminating true sources of blurring, and was instead sharpening the edges of objects. This led to only marginal performance improvement in the pre-trained models in the best case, and could not eclipse the best results on unmodified data. The implementation was also too slow for use in video analysis.

Initial tuning studies on the COCO and OIv7 pre-trained models on both unmodified and deblurred data led us to select the COCO model on unmodified data to explore more extensively. Increasing the image size and training length helped us secure optimal hyperparameters in only six tuning trials. These parameters on the pre-trained COCO model produced a near-SOTA mAP50 score. We feel we have shown the viability of transfer learning for well-known datasets on the VisDrone detection task that can act as a foundation for further exploration.

Our VisDrone MOT results demonstrate success of the detector-tracker training and tuning pipeline. We expect that if MOTA was reported for more recent VisDrone MOT efforts that the benchmark would increase compared to [18]. With that said, we were able to achieve good tracking results and demonstrate the utility of an automatic hyperparameter optimization framework.

## 4.1. Future Work

Deblurring showed some promise, and combinations of training directly on the VisDrone dataset and exploring different deblurring algorithms (e.g. DeepDeblur) could help improve outcomes.

Tuning ByteTrack over a wider search space and more trials could produce new insights into parameter importances and perhaps help to achieve better performance. Repeating the tracking experiments where the optimization is predicated on HOTA [14] instead of MOTA could produce useful quantitative and qualitative insights.

Other tracking approaches like SparseTrack have a simple yet novel approach that uses pseudo-depth information for dealing with occlusion that builds on ByteTrack. Effectively, they assume that objects closer to the bottom of the frame are closer to the camera, whereas objects further from the bottom of the frame are further away and more likely to be occluded [13]. Association occurs at different stages according to pseudo-depth levels, helping reduce ID switches [13]. Benefits may be lost if the drone is looking straight down and position in frame becomes less useful for determining depth position, but SparseTrack may provide performance gains otherwise.

The re-identification (ReID) approach leverages deep learning to learn appearance features that may work in concert with IoU to provide better association. A potential drawback is the extra computational cost, which may prove too expensive in some cases given drone hardware constraints.

| Student Name | Contributed Aspects | Details |
|---|---|---|
| Derek Griffing | Detector and tracker hyperparameter tuning, running trackers, metrics implementation. | Developed task independent (e.g., detection or tracking) hyperparameter tuning pipeline. Investigated tracker metrics pipeline and implemented in the tuning pipeline. Performed tracker research and implementation of running tracker within tuning pipeline. |
| Ming Xiang Lim | Initial parameter search, multi-objective optimization and comparison with SOTA models. | Performed hyperparameter search and analysis, and explored multi-objective optimization to concurrently optimise multiple hyperparameters in Optuna. Compared best-performing models post-tuning with SOTA models. |
| Tyler Burki | Deblurring investigation and analysis, initial parameter search and analysis, performed fine tuning as validation of others' results, explored fine tuning options based on results of initial study | Investigated deblurring methods and selected DeblurGAN. Integrated deblurring process into parameter tuning and analyzed effectiveness. Provided suggestions for fine tuning based on results of initial experiments. Further tuning experimentation for best model (results not used). |

Table 9. Contributions of team members.

## 5. Work Division

See Table 9.

## References

[1] Google drive - pretrained weights. 2

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019. 1

[3] Temitope Ibrahim Amosa, Patrick Sebastian, Lila Iznita Izhar, Oladimeji Ibrahim, Lukman Shehu Ayinla, Abdulrahman Abdullah Bahashwan, Abubakar Bala, and Yau Alhaji Samaila. Multi-camera multi-object tracking: A review of current trends and future advances. *Neurocomputing*, 552:126558, 2023. 1

[4] Guanlin Chen, Wenguan Wang, Zhijian He, Lujia Wang, Yixuan Yuan, Dingwen Zhang, Jinglin Zhang, Pengfei Zhu, Luc Van Gool, Junwei Han, Steven Hoi, Qinghua Hu, Ming Liu, Andrea Sciarrone, Chao Sun, Chiara Garibotto, Duong Nguyen-Ngoc Tran, Fabio Lavagetto, Halar Haleem, Hakkı Motorcu, Hasan F. Ateş, Huy-Hung Nguyen, Hyung-Joon Jeon, Igor Bisio, Jae Wook Jeon, Jiahao Li, Long Hoang Pham, Moongu Jeon, Qianyu Feng, Shengwen Li, Tai Huu-Phuong Tran, Xiao Pan, Young-Min Song, Yuehan Yao, Yunhao Du, Zhenyu Xu, and Zhipeng Luo. Visdrone-mot2021: The vision meets drone multiple object tracking challenge results. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 2839–2846, 2021. 5

[5] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv:2003.09003[cs]*, Mar. 2020. arXiv: 2003.09003. 2, 5

[6] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO, Jan. 2023. 1, 2, 3

[7] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiri Matas. Deblurgan: Blind motion

deblurring using conditional adversarial networks. *ArXiv e-prints*, 2017. 2

[8] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiri Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8183–8192, 2018. 2

[9] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020. 3

[10] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 3

[11] Yan Liu, Jingwen Wang, Tiantian Qiu, and Wenting Qi. An adaptive deblurring vehicle detection method for high-speed moving drones: Resistance to shake. *Entropy*, 23(10), 2021. 1, 4

[12] Yan Liu, Jingwen Wang, Tiantian Qiu, and Wenting Qi. An adaptive deblurring vehicle detection method for high-speed moving drones: Resistance to shake. *Entropy*, 23, 2021. 1, 2, 3

[13] Zelin Liu, Xinggang Wang, Cheng Wang, Wenyu Liu, and Xiang Bai. Sparsetrack: Multi-object tracking by performing scene decomposition based on pseudo-depth, 2023. 6

[14] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip H. S. Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. HOTA: A higher order metric for evaluating multi-object tracking. *CoRR*, abs/2009.07736, 2020. 2, 6

[15] Weiping Yu, Taojiannan Yang, and Chen Chen. Towards resolving the challenge of long-tail distribution in uav images for object detection. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 3257–3266, 2021. 5

[16] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box, 2022. 2, 3

[17] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Heng Fan, Qinghua Hu, and Haibin Ling. Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7380–7399, 2021. 1, 4, 5

[18] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Haibin Ling, Qinghua Hu, Haotian Wu, Qinqin Nie, Hao Cheng, Chenfeng Liu, Xiaoyu Liu, Wenya Ma, Lianjie Wang, Arne Schumann, Dan Wang, Diego Ortego, Elena Luna, Emmanouil Michail, Erik Bochinski, Feng Ni, Filiz Bunyak, Gege Zhang, Guna Seetharaman, Guorong Li, Hongyang Yu, Ioannis Kompatsiaris, Jianfei Zhao, Jie Gao, José M. Martínez, Juan C. San Miguel, Kannappan Palaniappan, Konstantinos Avgerinakis, Lars Sommer, Martin Lauer, Mengkun Liu, Noor M. Al-Shakarji, Oliver Acatay, Panagiotis Giannakeris, Qijie Zhao, Qinghua Ma, Qingming Huang, Stefanos Vrochidis, Thomas Sikora, Tobias Senst, Wei Song, Wei Tian, Wenhua Zhang, Yanyun Zhao, Yidong Bai, Yinan Wu, Yongtao Wang, Yuxuan Li, Zhaoliang Pi, and Zhiming Ma. Visdrone-vdt2018: The vision meets drone video detection and tracking challenge results. In Laura Leal-Taixé and Stefan Roth, editors, *Computer Vision – ECCV 2018 Workshops*, pages 496–518, Cham, 2019. Springer International Publishing. 5, 6

[19] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 111(3):257–276, 2023. 2