

# New Parallel Decision Fusion Models for Improving Multiclass Object Detection in Drone Images

Allison Nicole Paxton  
Department of Computer Science  
California State University  
Northridge, CA, USA  
allison.paxton.505@my.csun.edu

Abhishek Verma  
Department of Computer Science  
California State University  
Northridge, CA, USA  
abhishek.verma@csun.edu

**Abstract**— Automatic object detection in drone images is an important area of research due to its application such as surveillance and security, reconnaissance, search and rescue mission, land monitoring, self-navigating drones. It is considered a challenging task due to the distance, camera angle, and complex surroundings. Filtering predictions from object detection models and combining the results can help improve the performance of individual models. Selecting the best predictions from predictions of multiple models is called ensemble parallel decision fusion.

We augmented the large-scale grand challenge VisDrone2019-DET image dataset to create VisDrone-split and VisDrone-overlap datasets. We trained seven level-0 large deep learning models on the drone datasets and propose four Parallel Decision Fusion (PDF) deep learning models that improve in terms of the multiclass object detection precision upon their corresponding level-0 models. The seven models were split into two sets of object detection models. The first set of level-0 models includes Faster R-CNN, RetinaNet, and YOLOv5s. The second set consisted of much larger level-0 models namely state of the art YOLOv5x, YOLOv7x, YOLOv8x, and EVA02L. All proposed PDF models improved upon their corresponding state-of-the-art level-0 models on VisDrone-split. The PDF model with the highest metrics was YOLOv8x-s + EVA02L-s, which scored an mAP50 of 0.601 and improved 2.39% upon the state-of-the-art level-0 EVA02L model.

**Keywords**—*deep learning, object detection, decision fusion, VisDrone2019-DET, YOLOv5, YOLOv7, YOLOv8, EVA02*

## I. INTRODUCTION

Object detection is a part of the computer vision field where one or more objects within an image or video are not only classified but also located using a bounding box around them. This research focused on multiclass object detection within images. This research aimed to improve both poor and highly accurate object detection models to achieve the best performing model using ensemble methods and decision fusion.

Decision Fusion combines predictions from multiple machine learning models to create a more refined set of predictions that are more accurate than the individual machine learning models. Parallel Decision Fusion (PDF) fuses the predictions of more than one machine learning model, or base models, in parallel. Base models are referred to as level-0 models in this paper. The decision fusion method filters the predictions of multiple level-0 models. The goal of PDF is to improve upon the level-0 model results by using the results of several level-0 models, grouping predictions of the same objects

together, then only using the best prediction. PDF models are proposed to improve multiclass object detection models that are trained on a drone image dataset. This research implemented 19 total PDF models, and four of the best performing models are proposed in this paper. This research also studied how the metric results differed between level-0 models and affected the metric results of the PDF models on average and between individual classes.

To cover these topics, related works are examined using similar object detection models, dataset, or ensemble methods in section II. Then datasets used to train two sets of object detection models and test the PDF models are explained in section III. Faster R-CNN [1], RetinaNet [2], and YOLOv5s [3] are the first set. YOLOv5x [3], YOLOv7x [4], YOLOv8s [5], and EVA02L [6] are the second set. Section IV explains the level-0 models and the proposed PDF models. Section V and VI explain the experiment setup and the results of the level-0 models and PDF models. Finally, section VII goes over the conclusion of the research and future directions.

## II. RELATED WORK

Researchers have used different ensemble methods to combine more than one object detection model and improve the overall results, such as the Affirmative method, the Unanimous method, the Consensus method, and weighted box fusion as described in [7]. With the Affirmative method, an object is considered detected if at least one model detects an object. If two models incorrectly detect two different objects, then by the Affirmative method, they are both considered detected. With the Unanimous method, an object is considered detected if all the models detect the same object. With the Consensus method, an object is considered detected if more than half of the models detect the same object. Both the Unanimous and the Consensus methods helped decrease the number of false positives because the more models that detect an object, the more likely that detection is a true positive. The problem with the Unanimous method is there might be an increase in false negatives because if one model detects a true positive object that the other models did not catch, then that detection would not be considered and would become a false negative. The increase of false negatives can also occur in the Consensus method but to a lesser degree.

The following book and article documented the research on improving machine learning models. A book published in 2021 explains different techniques to classify pneumonia from chest radiographs. The book describes different types of decision

fusion methods to improve multiple classifiers in Chapter 4 [8]. Some of these decision fusion methods are Serial Decision Fusion (SDF), Parallel Decision Fusion (PDF), and Hybrid Decision Fusion (HDF). In SDF, one classifier takes the original input from the dataset, then the output is the input to the next classifier. The last classifier outputs the final set of outputs. In the PDF model on the other hand, decision fusion is applied to all the predictions of the classifiers at once, or in parallel. Lastly, as the name suggests HDF is a mix of SDF and PDF with a hierarchical setup. Each level in the hierarchy switches between SDF and PDF. The last level applies decision fusion to get the final output. Even though the book mentioned fusing classifier predictions together, our research uses PDF to fuse predictions from object detection models rather than classification models.

This final article proposed a Fine-grained Target Focusing Network (FiFoNet) to detect small and hard-to-locate objects [9]. FiFoNet has four modules. The first is a CNN-based backbone. The second is a top-down pathway for obtaining features, which uses Global Average Pooling. This module is used to improve feature extractions from small and poorly seen objects. The third is the FiFo module, which works to reduce feature background noise and aggregate the more subtle features, using Fine-grained Feature Aggregation (FIFA). The last module is used to predict the class score and box location. Both VisDrone2019 [10] and UAVDT [11] were used to test the accuracy of FiFoNet. VisDrone2019 was augmented to create a third dataset, called VisDrone\_Foggy, by including what looks like a fog of varying thickness. FiFoNet improved upon the baseline model YOLOv5, on all three datasets used. The article sheds light on the challenges inherent to the highly dense datasets such as VisDrone2019, and the need to further improve upon the state of the art to attain much higher average precision.

### III. DATASET DESCRIPTION

#### A. VisDrone2019-DET (VisDrone)

The object detection models used in this research were trained with a drone image dataset, called Visdrone2019-DET or VisDrone for short, and was created for a computer vision challenge as explained in [10], [12]. The ten classes are pedestrian, people, bicycle, car, van, truck, tricycle, awning-tricycle, bus, and motor for motorcycle. The image dimensions were from 960 by 540 pixels to 2,000 by 1,500 pixels. The dataset has 6,471 training images and 548 images for testing. The images were resized to 1,500 by 1,500 pixels to have all the same dimensions.

#### B. VisDrone-split and VisDrone-overlap

We preprocessed the VisDrone dataset to create two datasets namely VisDrone-split and VisDrone-overlap. To try to increase accuracy and reduce computational limitations, each resized VisDrone image was later cropped to nine 500 by 500 pixels sub-images. The number of images in the new dataset, called VisDrone-split, is 36,264 and the number of testing images in VisDrone-split is 3,293. Annotations from cropped objects were removed, and images without annotations were removed. An example image of VisDrone-split is shown in Fig. 1.

To keep at least the same number of instances, each resized image from VisDrone was cropped with a 25% overlap for each 500 by 500 pixels sub-image. There are far more objects within

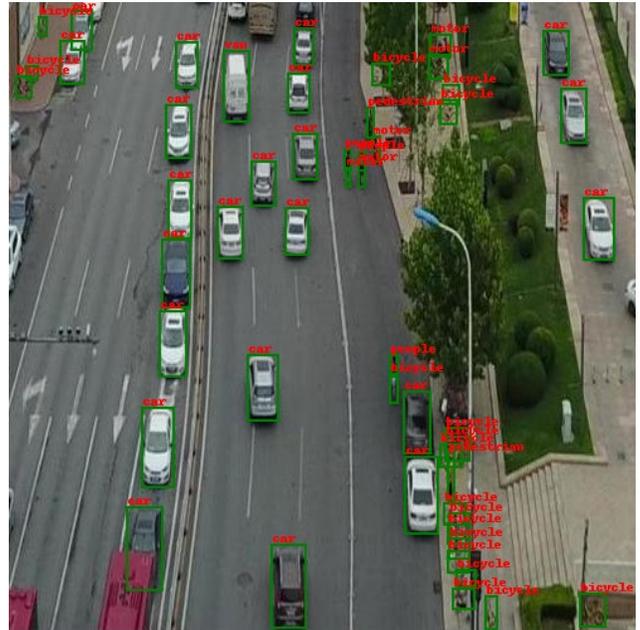


Fig. 1. A 500 by 500 pixels image from the VisDrone-split dataset. The ground truths in the image are shown inside the green bounding boxes and belong to the pedestrian, people, bicycle, car, van, and motor classes.

each class since many of them are within the overlapped part of the images. There are about 67,000 images for training and 6,000 images for testing in this new dataset, called VisDrone-overlap.

### IV. RESEARCH METHODOLOGY

The YOLOv5s, YOLOv5x, YOLOv7x, YOLOv8x, FasterRCNN, RetinaNet, and EVA02L methodology is explained below. The process of how their predictions were combined to apply decision fusion is the PDF method.

#### A. YOLOv5, YOLOv7, and YOLOv8 (Level-0)

There are ten YOLOv5 [3] versions ranging in size. The small YOLOv5s and extra-large YOLOv5x were tested on input images from the MS COCO dataset [13]. Its backbone is the same as YOLOv4 and CSPDarknet53 [14]. CSPDarknet53 comes from YOLOv3's Darknet53 [15] that uses Cross Stage Partial connections (CSP) [16], [17]. The model head is the same as YOLOv3 [15] and has three output heads. The backbone and the head are connected with Spatial Pyramid Pooling Fusion (SPPF) [18], [19] and CSP-Path Aggregation Network (CSP-PAN) [20], [16]. Each output head is for the bounding boxes, objectness, and class. Three loss functions follow the outputs – Complete Intersection over Union (CIoU) for location loss, and Binary Cross Entropy (BCE) for objectness and class loss.

There are six YOLOv7 [4] versions ranging in size. YOLOv7 and YOLOv7x model versions were tested on input images from the MS COCO dataset [13]. YOLOv7 proposed the backbone, Extended Efficient Layer Aggregation Network (E-ELAN). Its neck is CSPSPP and PAN, and its head is the same as YOLOR [21], the same creators of YOLOv7. The Sigmoid Linear Units (SiLU) activation function is used and, by default, SGD with momentum 0.937 is used for the optimizer. The image augmentation that is used is translation, scale, horizontal flip,

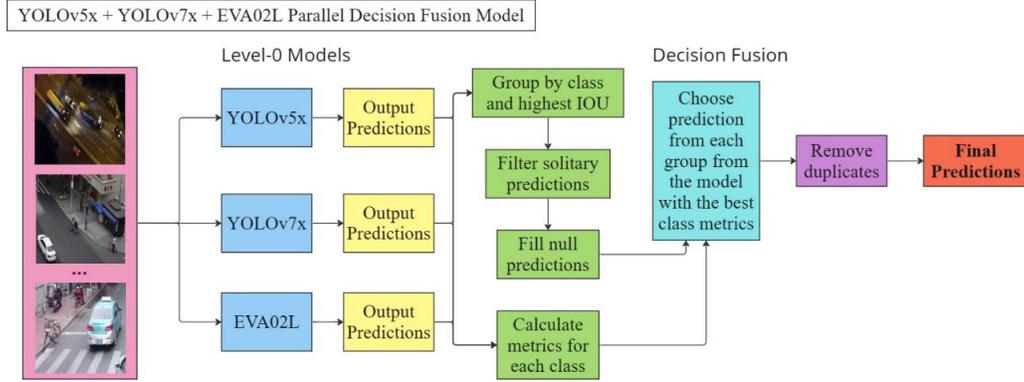


Fig. 2. Shows the pipeline for the implemented YOLOv5x + YOLOv7x + EVA02L Parallel Decision Fusion (PDF) model, which is one of the 19 implemented PDF models. For input images, predictions from each level-0 model are first grouped with the same class and highest IOU, any ungrouped predictions are removed, and incomplete groups are filled. One prediction from each group of predictions with class label “x” of 10 classes is selected from the level-0 model with the highest class “x” metric. The output is the final set of multiclass object detection predictions after removing duplicates.

and mosaic. The same three loss functions as YOLOv5 are used for bounding boxes, objectness, and class.

There are five YOLOv8 [5] versions ranging in size, they were all tested on images from the MS COCO dataset [13]. The backbone is a modified CSPDarkNet53 with a Feature Pyramid Network (FPN) [22]. The CSP connections in CSPDarkNet53 were replaced with c2f modules, or cross-stage partial bottleneck with two convolutions [5] [17]. The PAN is used as the neck, which preserves the localization information in the lower layers. The CSP connections that YOLOv5’s neck used are also replaced with c2f modules in YOLOv8’s neck. The output head is separated into three parts to identify the bounding boxes, objectness, and class. The loss functions are CIOU with Distribution Focal Loss (DFL) [23] for bounding box loss and BCE for class loss. Focal Loss is a general form of DFL, which helps with class imbalance. The dimensions of the prediction boxes are placed into bins to create a probability distribution and compared with the ground truth’s distribution.

### B. Faster R-CNN and RetinaNet

Faster R-CNN (FRCNN) [1] has a ResNet + Region Proposal Network (RPN) backbone. Similarly, RetinaNet [2] has a ResNet backbone followed by an FPN. Our research used a ResNet backbone of 50 layers and included FPN with FRCNN. RPN generates a set of regions using anchor boxes and a neural network to locate potential objects. RetinaNet doesn’t have an RPN but does use the focal loss function to handle class imbalance. The focal loss function is:

$$FL(p_t) = -(1 - p_t)^\gamma \log(p_t), \quad (1)$$

where  $p_t$  is the probability of a true positive prediction and  $\gamma$  is called the focusing parameter to assign more loss to predictions with a smaller probability [23]. The subnetworks of RetinaNet apply anchor box localization and classification.

### C. EVA02

There are two main EVA methods, EVA [24] and EVA02 [6]. EVA02L is used in our research since EVA has over a billion parameters, while EVA02L has 303 million parameters and is almost as accurate. EVA02L uses an improved Vision Transformer (ViT) [25], called Transform Vision (TrV).

Contrastive Language-Image Pre-training (CLIP) model [26] is used to pretrain the TrV using the masked image modeling (MIM) [27] pre-training strategy. MIM randomly masks parts of an image then works to predict the masked portion of the image.

TrV has multi-head self-attention (MHSA) layers + 2D rotary position embedding (RoPE) [28], a positional embedding that modifies the query and key vectors. TrV also uses random weight initialization in a Swish-Gated Linear Unit (SwiGLU) feedforward network (FFN) [29] with a SiLU activation function and has sub-LayerNorm (sub-LN) [30] for a normalization layer as listed in [6].

### D. Proposed Parallel Decision Fusion Models

We implemented 19 total Parallel Decision Fusion (PDF) models, and four of the best performing models are proposed in this paper. After training seven level-0 models, the PDF methodology was implemented to combine multiple object detectors. Decision fusion combined predictions from the output of different combinations of the level-0 models on the VisDrone datasets. The first eight outputs are all four combinations of Faster R-CNN (FRCNN), RetinaNet, and YOLOv5s trained on VisDrone-split and VisDrone-overlap. The outputs are all 11 combinations of YOLOv5x, YOLOv7x, YOLOv8x, and EVA02L trained on VisDrone-split. Fig. 2 shows one of the 19 implemented PDF models.

The first step in preparing the data was to convert the images and annotations into tables. There is a table for each image in the training dataset. Each instance in the table is a prediction containing the bounding box, confidence score, class label, image number, and model name. Next, each level-0 model prediction was paired with the other level-0 models’ predictions with the highest Intersection over Union (IOU) from each level-0 model. An IOU of two bounding boxes is the ratio of their overlapping area to their union area. If only one level-0 model predicted an object, then the prediction was removed from testing. If more than one model predicted the same object, then any other missing predictions were filled with a prediction from its group. This process was done for each level-0 model, resulting in repeated instances. These duplicate rows of the table were removed. The features for each instance of the resulting

TABLE I. PARALLEL DECISION FUSION MODEL (PDF) RUNTIME IN MINUTES BASED ON THE NUMBER OF LEVEL-0 MODELS

Number of Level-0 Models	2	3	4
PDF Runtime (min)	31	58	111

dataset were the detection box coordinates of each level-0 model, their confidences, and their shared class.

Each prediction from each row was then selected based on the individual class metrics of the level-0 models. If a row of grouped predictions is of class “x” of 10 classes, then the prediction from the level-0 model with the highest class “x” metric was selected and the others were omitted. The motivation for this process was that grouping the level-0 predictions increases the chance that they correctly detected an object and keeping only the prediction from the best level-0 model. The hypothesis for this process is that the metric of each class “x” of 10 classes from a PDF model will be at least as high as highest class “x” metric from its level-0 models.

The time to generate the datasets for each PDF model and to calculate the final results depended on the number of level-0 models, as shown in Table I. The more level-0 models there were, the more combinations of grouped predictions there were and the longer the runtime. The runtime shown for PDFs with two and three level-0 models are the averages of every combination of two and three level-0 models with EVA02L. There is only one PDF with four level-0 models, so no average was taken. Each runtime was rounded to the whole minute.

## V. EXPERIMENT SETUP

This section explains how the datasets were used to train the level-0 models. The hardware and software that were used is also described. A few challenges arose when training the level-0 models on the VisDrone datasets. To understand the results in the next section, the metric description is explained.

### A. Hardware and Software

The three common file types for object detection and object segmentation annotations are text, JSON, and XML. The two common coordinate formats for object detection boxes are “ $X$  min,  $Y$  min,  $X$  max,  $Y$  max,” and “ $X$  center,  $Y$  center, width, height.” These box coordinates can either be in pixels or normalized from the image’s length and width. Two GPUs used were a NVIDIA GeForce RTX 3060 with 6 GB of dedicated memory and a NVIDIA Titan XP with 12 GB of memory. When the first two weren’t enough, Amazon Web Services (AWS) was used to train the larger level-0 models. An instance of Amazon Sagemaker with either one NVIDIA A10G Tensor Core GPU with 24 GB or four of the same GPUs. Sagemaker was used to train YOLOv5x and EVA02L. Lastly, the level-0 predictions were processed using the CPU.

### B. Model Evaluation Metrics

Object detection models calculate average precision (AP), to find the area under the precision-recall curve, detailed in [31]. The mean AP (mAP) is the result of finding the AP for each class separately, then averaging the results. Although some publications make no distinction between AP and mAP, either is intended to be the mean average precision. Our paper uses mAP when discussing mean average precision. The metrics used

in our research were mAP with a 50% Intersection over Union (IOU) threshold (mAP50) and mAP with a 50:5:95% IOU threshold (mAP). The latter metric is calculated by averaging all the mAPs from an IOU of 50% to 95% with increments of 5%. The COCO website [32] states that mAP is the most important metric when determining performance on COCO, so mAP was used to determine the best model in this research.

### C. Parameter Setups

The seven level-0 models were separated into two separate sets, since metric results between the two sets were significantly different, which will be seen in the next section. The first set of level-0 object detection models are Faster R-CNN (FRCNN), RetinaNet, and YOLOv5s. The second set of level-0 object detection models are YOLOv5x, YOLOv7x, YOLOv8x, and EVA02L.

The preliminary discussion on how to train YOLOv5s on custom datasets could be found in [3]. We performed transfer learning to train the YOLOv5s on the VisDrone datasets and it took two to four hours. Training mainly used the default parameters on each dataset, except for slight variations. Training on the VisDrone-split and VisDrone-overlap used the Adam optimizer, a learning rate of at least 0.0025, and a batch size of 18 with three workers. Training on VisDrone had a smaller batch size, since the images were larger.

FRCNN and RetinaNet were the second and third models in our first set. The training procedure for those two on a generic dataset was described by Detectron2 on GitHub [33]. We performed transfer learning to train FRCNN and RetinaNet on each VisDrone dataset and it took one to two hours. The trained loss seemed to flatten sufficiently, then a confidence threshold of 0.1 during testing gave the best results. Training FRCNN on VisDrone-split and VisDrone-overlap used a learning rate of at least 0.001, a batch size of four, and ran for at least 10,000 iterations. Training on VisDrone took a bit longer with a smaller batch size, since the images were larger. Training RetinaNet on VisDrone-split and VisDrone-overlap used a learning rate of at least 0.001, a batch size of four with four workers, and ran for at least 15,000 iterations.

The YOLO models which were trained in the second set were YOLOv5x, YOLOv7x, and YOLOv8x. The last model that was trained for the second set was EVA02L. Model weights for pretrained YOLO versions 5 and 8 on COCO dataset was given by ultralytics on Github, while weights for version 7 was given by WongKinYiu from [3], [34], [35], respectively. Pretrained weights for EVA02L model on the COCO dataset were provided in [36]. Transfer learning and training YOLOv5x took us about three hours, YOLOv7x took 18 hours, YOLOv8x took 11 hours, and EVA02L took 36 hours. Due to time constraints and computational limitations of these models, these four models were only trained on the VisDrone-split dataset. All four used their default parameters, except the batch size was changed based on the GPU used and the size of the model. The main complication that arose when training these three models was overfitting. To YOLOv8x we added a 20% dropout rate but, adding more image augmentation helped delay overfitting across all models the most. Image rotation, translation, scale, flip across both x and y axes, mosaic, and mixup were applied. The

TABLE II. PROPOSED PDF MODELS THAT IMPROVED UPON THEIR CORRESPONDING LEVEL-0 MODELS FROM THE FIRST SET. MAP PER CLASS AND MAP50 ON THE VISDRONE-SPLIT DATASET IS SHOWN

Model	mAP50	mAP	pedes- trian	people	bicycle	car	van	truck	tricycle	awning- tricycle	bus	motor
Level-0 Models												
FRCNN	0.304	0.168	0.133	0.087	0.023	0.479	0.237	0.173	0.107	0.057	0.257	0.127
FRCNN-s	0.380	0.209	0.245	0.177	0.095	0.513	0.247	0.148	0.135	0.051	0.265	0.216
RetinaNet-s	0.382	0.214	0.215	0.161	0.106	0.534	0.280	0.165	0.144	0.059	0.259	0.221
<b>Proposed PDF model that improved upon its corresponding Level-0 models</b>												
<b>FRCNN-s + RetinaNet-s</b>	<b>0.385</b>	<b>0.217</b>	0.240	0.161	0.107	0.534	0.278	0.164	0.143	0.054	<b>0.268</b>	0.221
Level-0 Models												
FRCNN-o	0.407	0.223	0.291	0.189	0.115	0.521	0.277	0.167	0.142	0.062	0.240	0.231
RetinaNet-o	0.396	0.228	0.229	0.172	0.115	0.558	0.305	0.183	0.149	0.052	0.288	0.226
<b>Proposed PDF model that improved upon its corresponding Level-0 models</b>												
<b>FRCNN-o + RetinaNet-o</b>	<b>0.405</b>	<b>0.231</b>	0.287	0.184	0.114	0.557	0.304	0.180	<b>0.149</b>	0.059	0.251	0.229
Level-0 Models												
YOLOv5s	0.413	0.240	0.259	0.161	0.086	0.624	0.327	0.174	0.152	0.077	0.307	0.236
YOLOv5s-s	0.415	0.233	0.237	0.159	0.094	0.569	0.343	0.167	0.136	0.072	0.324	0.233
<b>YOLOv5s-o</b>	0.460	0.266	0.261	0.178	0.124	0.599	0.378	0.194	0.196	0.086	0.377	0.295

Note: The “-s” and “-o” means which level-0 models were trained on VisDrone-split and VisDrone-overlap, respectively. Bold numeric values are the most improved class per model and the bold model is the best performing model. The mAP values of each class from various models are shown under the individual class columns.

weights were saved after each epoch, and training stopped when the training and testing mAP50s differed more than 0.05.

## VI. EXPERIMENT RESULTS AND DISCUSSION

The PDF results with the two sets of level-0 models are examined first in this section. Between the two level-0 model sets, the second set provided most improved and best results. Between the two datasets the level-0 models were trained on, VisDrone-split helped provide better results than VisDrone-overlap.

### A. PDF Models With Faster R-CNN, RetinaNet, YOLOv5s

The first set of level-0 models are Faster R-CNN (FRCNN), RetinaNet, and YOLOv5s trained on VisDrone-split and VisDrone-overlap. The mAP and mAP50 results of FRCNN and YOLOv5s trained on VisDrone and VisDrone-split were either similar or the results of the same two models trained on VisDrone-split were significantly better than trained on VisDrone. So, the level-0 models of the PDF models were trained on VisDrone-split and VisDrone-overlap only. Table II shows the level-0 mAPs for FRCNN, RetinaNet, and YOLOv5s trained on different datasets, where “-s” is for VisDrone-split and “-o” is for VisDrone-overlap. The differences in the mAPs from YOLOv5s, FRCNN, and RetinaNet trained on VisDrone-split were 0.024, 0.019, and 0.005. The differences in mAPs from YOLOv5s, FRCNN, and RetinaNet trained on VisDrone-overlap were 0.043, 0.038, 0.005. The mAP model results from these three level-0 models trained on VisDrone-split were closer

than when trained on VisDrone-overlap, so combining the three models improved more on the VisDrone-split.

The first set of three level-0 models trained on VisDrone-split and VisDrone-overlap were combined to create eight combinations of models for PDF, four combinations from each dataset. Table II shows the most and only improved PDF models. These were FRCNN-s + RetinaNet-s and FRCNN-o + RetinaNet-o, whose mAPs improved 1.40% and 1.32% from RetinaNet, respectively. This might be because more true positives than false positives were removed during the preprocessing phase. Furthermore, true positives might have been removed when selecting a prediction from each group of predictions based on the level-0 model with the highest mAP metric because the VisDrone datasets are dense. Some grouped predictions might have been predicting more than one object. The best model according to mAP50 and mAP was YOLOv5s-o.

### B. PDF Models With YOLOv5x, YOLOv7x, YOLOv8x, EVA02L

The second set of level-0 models are YOLOv5x, YOLOv7x, YOLOv8x, and EVA02L. Since they have much larger models in comparison to the previous set of level-0 models, they were trained on the VisDrone-split only. Table III shows the results for the second set of level-0 models trained on VisDrone-split, where “-s” is for VisDrone-split. EVA02L was the most accurate model overall, measuring in both mAP50 and mAP.

TABLE III. PROPOSED PDF MODELS THAT IMPROVED UPON THEIR CORRESPONDING LEVEL-0 MODELS FROM THE SECOND SET. MAP PER CLASS AND mAP50 ON THE VISDRONE-SPLIT DATASET IS SHOWN

Model	mAP50	mAP	pedes -trian	people	bicycle	car	van	truck	tricycle	awning- tricycle	bus	motor
Level-0 Models												
YOLOv5x-s	0.560	0.337	0.340	0.236	0.165	0.631	0.426	0.300	0.242	0.170	0.508	0.326
YOLOv7x-s	0.563	0.341	0.337	0.227	0.206	0.629	0.428	0.316	0.262	0.147	0.528	0.330
Proposed PDF model that improved on its level-0 models												
YOLOv5x-s + YOLOv7x-s	0.567	0.348	0.337	0.234	<b>0.213</b>	0.621	0.430	0.325	0.269	0.172	0.534	0.339
Level-0 Models												
YOLOv8x-s	0.579	0.362	0.367	0.262	0.215	0.656	0.453	0.337	0.285	0.172	0.514	0.357
EVA02L-s	0.587	0.374	0.358	0.281	0.247	0.604	0.437	0.395	0.309	0.213	0.544	0.352
Proposed PDF model that improved on its level-0 models												
<b>YOLOv8x-s + EVA02L-s</b>	<b>0.601</b>	<b>0.381</b>	0.362	0.282	0.248	0.643	0.450	<b>0.397</b>	0.310	0.214	0.543	0.356

Note: The “-s” means which level-0 models were trained on VisDrone-split. Bold numeric values are the most improved class per model and the bold model is the best performing model. The mAP values of each class from various models are shown under the individual class columns.

Out of the second set of level-0 models, EVA02L improves upon other level-0 models on most of the individual class results.

The second set of four level-0 models were combined to create eleven combinations of models for PDF. The proposed PDF models that used the output from level-0 models were tested on VisDrone-split. Table III shows the best performing and most improved models. Of the eleven PDF models each one with a EVA02L level-0 model improved all their level-0 models. The best performing model in the table came from YOLOv8x-s + EVA02L-s since it has the highest mAP. Half of their classes improved from both level-0 class metric and all except the bus class improved from EVA02L. Similarly with the other PDF models with EVA02L most classes improved from level-0 EVA02L. This was also the case in the most improved PDF model, YOLOv5x-s + YOLOv7x-s, whose mAP improved 2.05% from YOLOv7x. Most classes in the PDF models improved on both level-0 models per class upon the YOLOv7x, its best level-0 model. This trend did not extend to PDF models with YOLOv8x as their best performing level-0 model. Their negative percentage changes were within -1.83% and 0% of

YOLOv8x. Fig. 3 shows how much YOLOv5x-s + YOLOv7x-s and YOLOv8x-s + EVA02L-s increased from their base models. The percent increase is from the level-0 model with the highest metric to the metric of the resulting PDF model.

## VII. CONCLUSION AND FUTURE WORK

The proposed PDF models chose the best predictions from the two sets of level-0 models to improve multiple multiclass object detectors. Improvements were seen in most PDF models from both level-0 model sets, the PDF models with the best performing level-0 model, EVA02L, improved on all their level-0 models. The most improved and best performing PDF both came from the second set. Most improved PDF model was YOLOv5x-s + YOLOv7x-s whose mAP improved 2.05% from YOLOv7x. The model with the highest mAP was YOLOv8x-s + EVA02L-s, which scored an mAP50 of 0.601 and mAP of 0.381. The PDF model’s mAP improved 1.87% and mAP50 improved 2.39% upon state of the art level-0 EVA02L model. Therefore, the YOLOv8x-s + EVA02L-s is the best performing model of all proposed models from both sets.

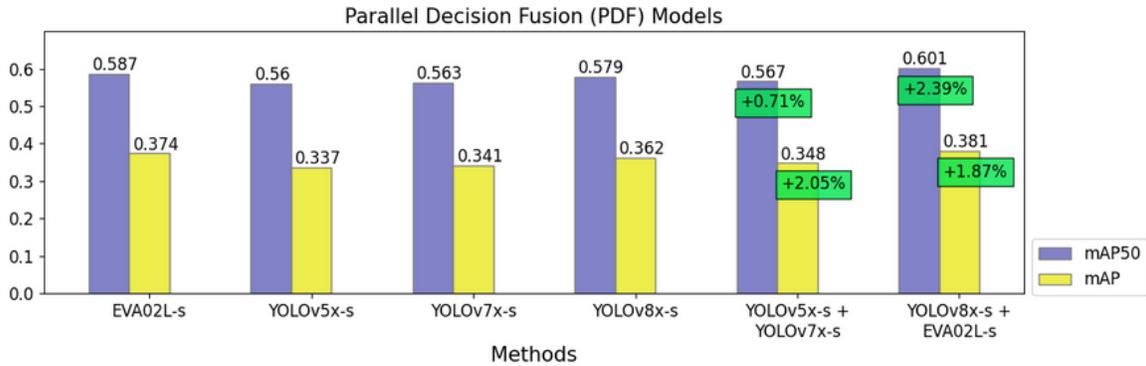


Fig. 3. Metric comparison between the second set of level-0 models and the proposed PDF models. The green boxes show the percent changes between each model compared to their level-0 model with the highest metric. For example, in the case of the PDF model YOLOv8x-s + EVA02L-s the mAP50 value improved by 2.39%, which is from 0.587 of EVA02L-s to 0.601, note that the percentage calculation is based on the higher of the two level-0 models YOLOv8x-s and EVA02L-s.

To further refine these models, future work would include improving the process of the PDF models. Instead of only selecting the best prediction from each group of predictions, every prediction should be taken advantage of to create a refined set of predictions. This can be done by learning each group of predictions. The process of learning from already trained machine learning model predictions is meta-learning. In the future we plan to explore meta-learning approaches.

## REFERENCES

- [1] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems*, 2015.
- [2] T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318-327, 2020.
- [3] G. Jocher, A. Chaurasia, J. Borovec, A. Stoken, Y. Kwon, J. Fang and e. al, "yolov5," [Online]. Available: <https://github.com/ultralytics/yolov5>. [Accessed Nov 2022].
- [4] C.-Y. Wang, A. Bochkovskiy and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA, 2023.
- [5] D. Reis, J. Kupec, J. Hong and A. Daoudi, *Real-Time Flying Object Detection with YOLOv8*, arXiv:2305.09972, 2023.
- [6] Y. Fang, Q. Sun, X. Wang, T. Huang, X. Wang and Y. Cao, *EVA-02: A Visual Representation for Neon Genesis*, arXiv:2303.11331, 2023.
- [7] A. Casado-García and J. Heras, "Ensemble Methods for Object Detection," in *Frontiers in Artificial Intelligence and Applications (ECAI 2020)*, vol. 325, G. D. Giacomo, A. Catala, B. Dilkina, M. Milano, S. Barro, A. Bugarín and J. Lang, Eds., Logrono, La Rioja: IOS Press Ebooks, 2020, pp. 2688-2695.
- [8] Y. Chandola, J. Virmani, H. Bhaduria and P. Kumar, "Chapter 4 - End-to-end pre-trained CNN-based computer-aided classification system design for chest radiographs," in *Deep Learning for Chest Radiographs*, Y. Chandola, J. Virmani, H. Bhaduria and P. Kumar, Eds., Academic Press, 2021, pp. 117-140.
- [9] Y. Xi, W. Jia, Q. Miao, X. Liu, X. Fan and H. Li, "FiFoNet: Fine-Grained Target Focusing Network for Object Detection in UAV Images," *Remote Sensing*, vol. 14, no. 16, 2022.
- [10] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu and e. al, "Detection and tracking meet drones challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7380-7399, 2022.
- [11] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li and e. al, "The Unmanned Aerial Vehicle Benchmark: Object Detection and Tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [12] D. Du, P. Zhu, L. Wen, X. Bian, H. Lin, Q. Hu and e. al, "VisDrone-DET2019: The Vision Meets Drone Object Detection in Image Challenge Results," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019.
- [13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan and e. al, "Microsoft COCO: Common Objects in Context," in *Computer Vision -- ECCV 2014*, 2014.
- [14] A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, *YOLOv4: Optimal Speed and Accuracy of Object Detection*, arXiv:2004.10934, 2020.
- [15] J. Redmon and A. Farhadi, *YOLOv3: An Incremental Improvement*, arXiv:1804.02767, 2018.
- [16] C.-Y. Wang, L. H.-Y. Mark, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh and I.-H. Yeh, "CSPNet: A New Backbone that can Enhance Learning Capability of CNN," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020.
- [17] J. Terven and D. Cordova-Esparza, *A Comprehensive Review of YOLO: From YOLOv1 and Beyond*, arXiv:2304.00501, 2023.
- [18] "Ultralytics YOLOv5 Architecture," 2023. [Online]. Available: [https://docs.ultralytics.com/yolov5/tutorials/architecture\\_description/](https://docs.ultralytics.com/yolov5/tutorials/architecture_description/). [Accessed Sep 2023].
- [19] K. He, X. Zhang, S. Ren and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," in *Computer Vision -- ECCV 2014*, 2014.
- [20] S. Liu, L. Qi, Q. Haifang, J. Shi and J. Jiaya, "Path Aggregation Network for Instance Segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [21] C.-Y. Wang, I.-H. Yeh and H.-Y. M. Liao, *You Only Learn One Representation: Unified Network for Multiple Tasks*, 2021.
- [22] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [23] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li and e. al, "Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection," in *Advances in Neural Information Processing Systems*, 2020.
- [24] Y. Fang, W. Wang, B. Xie, Q. Sun, L. Wu and X. Wang, "EVA: Exploring the Limits of Masked Visual Representation Learning at Scale," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
- [25] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai and T. Unterthiner, *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, arXiv:2010.11929, 2021.
- [26] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal and e. al, "Learning Transferable Visual Models From Natural Language Supervision," in *Proceedings of the 38th International Conference on Machine Learning*, 2021.
- [27] H. Bao, L. Dong, S. Piao and F. Wei, *BEiT: BERT Pre-Training of Image Transformers*, arXiv:2106.08254, 2022.
- [28] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen and Y. Liu, *RoFormer: Enhanced Transformer with Rotary Position Embedding*, arXiv:2104.09864, 2022, pp. 418-434.
- [29] N. Shazeer, *GLU Variants Improve Transformer*, arXiv:2002.05202, 2020.
- [30] H. Wang, S. Ma, S. Huang, L. Dong, W. Wang, Z. Peng and e. al, "Magneto: A Foundation Transformer," in *Proceedings of the 40th International Conference on Machine Learning*, 2023.
- [31] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto and E. A. B. da Silva, "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," *Electronics*, vol. 10, no. 3, 2021.
- [32] "Detection Evaluation," [Online]. Available: <https://cocodataset.org/#detection-eval>. [Accessed Jan 2023].
- [33] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo and R. Girshick, "Detectron2," 2019. [Online]. Available: <https://github.com/facebookresearch/detectron2>. [Accessed Dec 2022].
- [34] W. Kin-Yiu, A. Bochkovskiy, M. Khoshbin, B. Raymond, A. Khaliq, R. Patankar and e. al, "yolov7," 2022. [Online]. Available: <https://github.com/WongKinYiu/yolov7>. [Accessed May 2023].
- [35] G. Jocher, A. Chaurasia, Y. Kwon, O. Sezer, K. Michael, M. R. Munawar and e. al, "ultralytics," 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>. [Accessed Jun 2023].
- [36] Y. Fang, Q. Sun, Y. Cao, W. Zhang, M. Yousef, W. Wang and e. al, "EVA," [Online]. Available: <https://github.com/baivision/EVA>. [Accessed Jul 2023].